UNIVERSITY of CALIFORNIA
Santa Barbara

Modeling and Detection of Geospatial Objects Using Texture Motifs

A dissertation submitted in partial satisfaction of the
requirements for the degree

Doctor of Philosophy
in
Electrical and Computer Engineering

by

Sitaram Bhagavathy

Committee in charge:

Professor B. S. Manjunath, Chair
Professor Shivkumar Chandrasekaran
Professor Kenneth Rose
Professor Terence Smith
Professor Michael Goodchild

December 2005

The dissertation of Sitaram Bhagavathy is approved.

Professor Shivkumar Chandrasekaran

Professor Kenneth Rose

Professor Terence Smith

Professor Michael Goodchild

Professor B. S. Manjunath, Committee Chair

September 2005

Modeling and Detection of Geospatial Objects Using Texture Motifs

by

Sitaram Bhagavathy

To my mother, who deserves 99% of the credit for all my successes...

Acknowledgements

I have immensely enjoyed the past six years at UCSB while pursuing the M.S. and Ph.D. degrees. Everything about this part of my life is a plus. The school is well-respected. The faculty is excellent. The location is unbelievably good. The climate could hardly be better. And I have met wonderful people. I thank the forces of nature for landing me here. I now proceed to thank the people who have enriched my life.

First and foremost, I would like to thank Prof. B. S. Manjunath for his constant support and guidance during my study at UCSB. I have been lucky to have him as my advisor. He has always remained accessible to his students in spite of his busy schedule. He is a living lesson in efficient time management. I have also observed him to be genuinely concerned about the welfare of his students.

I am grateful to the members of my doctoral committee for their help and constructive criticism. This includes Prof. Shiv Chandrasekaran, Prof. Ken Rose, Prof. Terry Smith, and Prof. Mike Goodchild. I owe a debt of gratitude to the faculty members of UCSB for enhancing my knowledge through the courses they taught. I thank Prof. Sanjit Mitra for providing me employment references. I appreciate Val DeVeyra and the administrative and computer support staff of the ECE department, for keeping the departmental machinery running smoothly so that students can focus on their studies.

I have seldom been bored at my workplace, the Vision Research Laboratory (VRL), atleast not when my colleagues were present. It has been a pleasure to work with such smart and interesting people. I have greatly benefitted from my research collaborations with Shawn Newsam, Jelena Tešić, Baris Sumengen, and

Marco Zuliani. It has also been an enriching experience to share my workplace with Jiyun Byun, Motaz El-Saban, Dmitry Fedorov, Kaushal Solanki, Ken Sullivan, Xinding Sun, Lei Wang, Gosuke Ohashi, Ching-Wei Chen, Onkar Dabeer, Zhiqiang Bi, and everyone who has been a part of our research group.

I am thankful to Dr. Ajay Divakaran and Dr. Baback Moghaddam for giving me the opportunity to intern at Mitsubishi Electric Research Laboratory (MERL). I learned much working with them and Dr. Fatih Porikli. I appreciate the friendship and help of Dr. Regu Radhakrishnan while I was at MERL and after.

Besides my colleagues, I made many other good friends—especially Jayanth Nayak, Nari Soundarrajan, and Umesh Vaidya—during my time at Santa Barbara. I thank them for all the good times so far and for those in the future. I would like to thank Steve Ota sensei, Clarence Chinn sensei, Koichi Kashiwaya sensei, and my friends at the UCSB Aikido club, for helping me in my quest for learning Aikido. Steve is a very good teacher. What I learned at his Dojo has subtly changed me as a person for the better.

The person I am most grateful to is my mother. There are no words to convey the depth of my gratitude to her. She has worked very hard to ensure that I have the best possible education. She deserves 99% of the credit for all my successes. I am also thankful to my brother, grandmother, and late grandfather, for their love and support.

I wrap up by acknowledging the National Science Foundation for supporting my research and food supply through the following projects: NSF-DLI #IIS-49817432 and NSF IIS #0329267.

Curriculum Vitæ

Sitaram Bhagavathy

July 1999	Bachelor of Engineering Department of Electronics and Communication Engineering National Institute of Technology, Surathkal, Karnataka, India
December 2000	Master of Science Department of Electrical and Computer Engineering University of California, Santa Barbara
December 2005	Doctor of Philosophy Department of Electrical and Computer Engineering University of California, Santa Barbara

Fields of Study

Computer vision, pattern recognition, image processing, image retrieval, statistical learning, object detection

Publications

M. Zuliani, S. Bhagavathy, B. S. Manjunath, and C. S. Kenney. Affine-invariant curve matching. In *Proceedings of the International Conference on Image Processing*, volume 5, pages 3041–3044, Singapore, October 2004.

B. Sumengen, S. Bhagavathy, and B. S. Manjunath. Graph partitioning active contours for knowledge-based geospatial segmentation. In *Proceedings of the CVPR Workshop on Perceptual Organization in Computer Vision (POCV)*, page 54, Washington DC, June 2004.

S. Newsam, L. Wang, S. Bhagavathy, and B. S. Manjunath. Using texture to analyze and manage large collections of remote sensed image and video data. *Journal of Applied Optics: Information Processing*, 43(2):210–217, January 2004.

J. Tešić, S. Bhagavathy, and B. S. Manjunath. Issues concerning dimensionality and similarity search. In *Proceedings of the International Symposium on Image and Signal Processing and Analysis, Special Session on System Perspectives in Information Retrieval*, Rome, September 2003.

S. Newsam, L. Wang, S. Bhagavathy, and B. S. Manjunath. Using texture to annotate remote sensed datasets. In *Proceedings of the International Symposium on Image and Signal Processing and Analysis, Special Session on Texture Analysis and Synthesis*, Rome, September 2003.

S. Newsam, S. Bhagavathy, and B. S. Manjunath. Object localization using texture motifs and Markov random fields. In *Proceedings of the International Conference on Image Processing*, volume 3, pages 1049–1052, Barcelona, September 2003.

S. Bhagavathy, J. Tešić, and B. S. Manjunath. On the Rayleigh nature of Gabor filter outputs. In *Proceedings of the International Conference on Image Processing*, volume 2, pages 745–748, Barcelona, September 2003.

S. Newsam, S. Bhagavathy, and B. S. Manjunath. Modeling object classes in aerial images using hidden Markov models. In *Proceedings of the International Conference on Image Processing*, volume 1, pages 860–863, Rochester, September 2002.

S. Bhagavathy, S. Newsam, and B. S. Manjunath. Modeling object classes in aerial images using texture motifs. In *Proceedings of the International Conference on Pattern Recognition*, volume 2, pages 981–984, Quebec, August 2002.

S. Newsam, S. Bhagavathy, L. Fonseca, C. Kenney, and B. S. Manjunath. Object based representations of spatial images. *Acta Astronautica*, 48(5-12):567–577, June 2001.

S. Newsam, S. Bhagavathy, L. Fonseca, C. Kenney, and B. S. Manjunath. Object based representations of spatial images. In *Proceedings of the 51st International Aeronautical Congress*, Rio de Janeiro, October 2000.

Abstract

Modeling and Detection of Geospatial Objects Using Texture Motifs

by

Sitaram Bhagavathy

The primary goal of this dissertation is the modeling and detection of *compound* objects, such as harbors and golf courses, in remotely sensed geospatial images. Toward this goal, this dissertation makes two important contributions: 1) it demonstrates the potential of frequency-domain texture analysis for model-driven detection of geospatial objects, and 2) it addresses the problem of learning appearance models for objects from their examples. The complexity of geospatial image content present obstacles in using purely spatial (pixel) domain methods for describing objects. In this dissertation, the structure of objects is efficiently described using Gabor filter-based texture analysis, which incorporates information from both the spatial and frequency domains.

The use of *texture motifs*, or characteristic spatially recurrent patterns, is proposed for modeling and detecting geospatial objects. Three approaches are described in this dissertation for learning texture motif representations from object examples and detecting objects based on the learned models. The first approach is a two-layered representation that first learns the constituent "texture elements" of the motif and then the spatial distribution of the elements. In the second approach, the texture elements of a motif are learned as the states of a hidden Markov model (HMM), and the state transitions of the model describe the

spatial arrangement of the elements. The third approach addresses the problem of detecting objects with *quasi-periodic* texture motifs, by analyzing the relations among the characteristic scales and orientations of these patterns. Experimental results demonstrate the effectiveness of the above approaches in detecting compound geospatial objects.

Contents

List of Figures

Chapter 1

Introduction

Aerial and satellite images of the earth (or *geospatial* images) are critical sources of information in diverse fields such as geography, cartography, meteorology, surveillance, city planning, and so on. These images contain visual information about various natural and man-made features on or above the surface of the earth. In each of the aforementioned fields, analysts benefit from knowing the location and extent of different features. For example, roads and buildings are features that help cartographers update maps, and city planners might be interested in knowing the extent of urban areas in order to study their growth.

Advances in sensing and storage technology have resulted in the availability of a high volume of geospatial images covering a large surface area on the earth. Manual annotation of geospatial images covering even a relatively small area of the earth is a tedious task. This has necessitated research into automated annotation of geospatial images. An important component of this research comprises *object detection* methods. These are model-driven methods that seek to identify

probable locations of specified features of interest (*objects*) in geospatial images. For example, detection of buildings and roads is a useful step in cartography. Detection of objects such as harbors, airports, golf courses, trailer parks, and vineyards is useful for updating geographical databases such as the Alexandria Digital Library (ADL) Gazetteer [32] which index the locations of several object types.

Object detection in geospatial images is the primary focus of this dissertation. The detection of geospatial objects with simple geometric or shape models such as buildings, roads, vehicles, etc., has been explored adequately in the literature. This is not the case for compound objects, such as harbors and golf courses, characterized by several "parts" and their spatial layout. For example, harbors contain boats and golf courses contain trees and grass, both with a distinct spatial arrangement (Fig. 1.1). The detection of compound geospatial objects such as harbors, golf courses, and airports from geospatial images is a largely unexplored issue.

There are two domains in which visual structure in images can be analyzed, namely the spatial domain (pixel intensities), and the frequency domain (fourier spectrum). The former has been the preferred domain for describing the structure of compound objects. Spatial analysis methods have been proposed for describing the constituents and layout of compound geospatial objects such as airports [19] and parking lots [63]. However, the problems of detecting such objects in a much larger scene and learning models for these from examples, have not been addressed. In the remainder of this chapter, we shall study the obstacles in addressing these problems using purely spatial analysis, and outline solutions

that combine information from the spatial and frequency domains.

1.1 The Problems

There has been a wide body of work on the spatial analysis of complex geospatial images (Sec. 2.1). Given some amount of contextual knowledge about a scene (e.g. airport, housing development), several systems [55, 19, 59, 63, 51] have been proposed for performing a detailed interpretation of the scene. These methods usually divide an image into spatial units (closed regions, lines, etc.) through image segmentation or edge detection/linking. Spatial relations between units are analyzed using relational models such as production systems [55], semantic networks [59, 63], human-specified constraints or *rules* [19], and evidential reasoning [51]. These frameworks are essentially used for humans to specify spatial constraints among the constituents of a scene or object.

Previously, spatial modeling methods have been applied for describing compound objects such as airports [19] and parking lots [63]. Such methods have also played a major role in exploiting contextual relations [55, 59, 51] for detecting small objects (e.g. houses, driveways etc.) in a large scene (e.g. housing development). However, the problems of detecting a compound object in a much larger scene or learning appearance models for such objects from examples, has not been addressed. The two main problems that we address in this dissertation are the following.

Problem 1: *Detection of compound objects in geospatial images.* Most of the work on object detection has focussed on objects with simple geometric

or shape models such as buildings, roads, vehicles, agricultural crops, etc.

Problem 2: *Learning appearance models for compound geospatial objects from their examples.* Appearance models embody knowledge about object appearance which is required for detecting an object. Appearance models for compound geospatial objects have traditionally been encoded a priori into the system by humans.

There are three major obstacles in tackling the above problems. These are

Obstacle 1: Compound geospatial objects often contain a large number of parts. For example, a harbor may contain hundreds of boats.

Obstacle 2: The structural relations are often loose and vary from one object instance to another. In order to robustly recognize an object, this variation has to be accounted for.

Obstacle 3: Geospatial images are highly detailed, usually on the order of thousands of pixels in each dimension.

Specifically, the above complexities reduce the appeal of methods that perform analysis strictly in the spatial (pixel) domain, namely image segmentation, edge detection/linking, and graph-based models for analyzing spatial layout. All these methods have been applied frequently to the analysis of geospatial images. Obstacle 3 makes it time-consuming to divide an image into spatial units (closed regions, lines, etc.) through image segmentation or edge detection/linking. Most previous methods employ this as the first step in feature extraction from geospatial images. Obstacles 1 and 2 increase the complexity of rigorously describing

(or modeling) the spatial layout of parts. It becomes infeasible to analyze spatial layout using graph-based models where the nodes are the parts and the links indicate their spatial relation. Obstacles 1 and 3 together lead to a high search complexity, i.e. the time required to search for (or detect) the object in a given image.

Let us now regard the second problem mentioned earlier, namely that of learning appearance models for objects. Appearance models embody knowledge about object appearance which is required for detecting an object. For instance, a model for a *harbor* could comprise a model for shapes of boats, and a graphical model for the arrangement of boats. Besides describing its appearance, an object model might also have contextual knowledge about its location with respect to other objects. Appearance models for compound geospatial objects have traditionally been encoded a priori into the system by humans. As a result of the high complexity of purely spatial approaches, there is little work addressing the problem of *learning* appearance models for geospatial objects from examples.

In order to learn models for and detect compound objects, efficient albeit approximate methods are needed to surmount the above obstacles. By performing analysis in both spatial and frequency domains, it is possible to mitigate these obstacles and make object detection and model learning feasible.

1.2 The Proposed Solutions

The main goal of this dissertation is to provide efficient methods for describing and detecting compound objects in geospatial images. Toward this motive,

we incorporate information from both the spatial and the frequency domains. We utilize joint space-frequency methods developed in the framework of texture analysis. Convolution of an image with Gabor filters, which can be efficiently implemented via frequency-domain filtering, provides abundant information about the inherent visual structure in the image. Despite this fact, there is relatively little use of the efficient methods of frequency-domain texture analysis for detecting geospatial objects. Using texture analysis as a foundation, we propose methods for learning appearance models for objects from examples. These methods exploit the ability of Gabor filter-based texture analysis to provide a compact description of visual structure and gracefully handle variations in appearance. In a nutshell, this dissertation focusses on filling two important gaps in the area of geospatial object detection, namely 1) exploiting the potential of frequency-domain texture analysis, and 2) example-based learning of appearance models for objects.

1.2.1 Texture Analysis for Object Detection

Texture analysis gives information about the statistical properties of the spatial arrangements of pixels in a neighborhood. This could translate to information about the arrangement of physical features in the neighborhood, for example the density of trees in an orchard and the quasi-periodicity of boats in a harbor. It is our contention that the potential of texture for the modeling and detection of geospatial objects has not been fully exploited. The role of image texture in geospatial image analysis (Sec. 2.2) is largely limited to the detection and classification of certain types of land-cover such as vegetation, water, and urban settlements. In this dissertation, we attempt to extend the use of texture anal-

ysis to the modeling and detection of more complex geospatial objects such as harbors, golf courses, airports, and so on.

Consider the *harbor* object, which can be detected by detecting boats via their model shapes after segmentation, and finding those that occur in certain regular arrangements which are modeled a priori. This is how most methods discussed earlier would proceed. However, a harbor can also be modeled as a spatially recurrent two-dimensional signal. Such recurrent arrangements of features, quite common in geospatial images, are perfect candidates for description and detection via frequency-domain texture analysis. Texture analysis provides a framework for the efficient analysis of recurrent and possibly regular arrangements of image primitives. At a lower-level, such primitives may be a set of local intensity patterns including edges, bars, and smooth regions. At a higher level, they may be cars, boats, trees, and so on, by whose repetitive spatial arrangements, several geospatial objects are formed.

When such recurrent patterns occur, there are many advantages in using frequency-domain texture analysis to describe them. The major ones are the following.

1. Frequency-domain texture analysis is efficient. Texture extraction methods can be implemented efficiently via frequency-domain filtering. These are less computationally expensive than image segmentation and edge detection/linking, especially for large and highly detailed geospatial images.

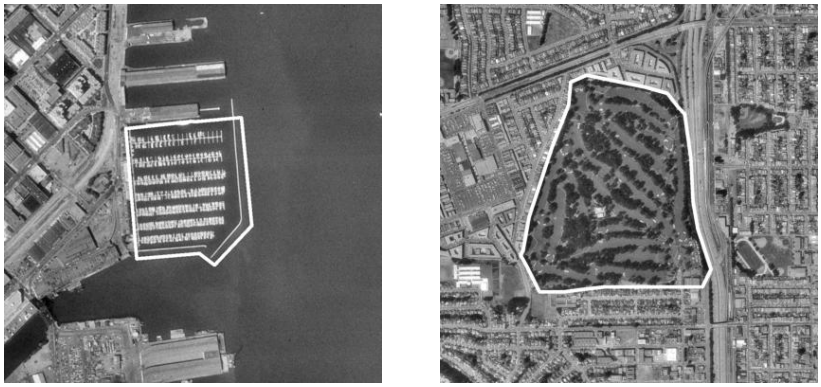
2. Texture analysis using a Gabor filter bank provides a compact description of visual structure present in a neighborhood. Useful information about the

scale(s) and orientation(s) of features in a neighborhood can be encoded as a compact feature vector.

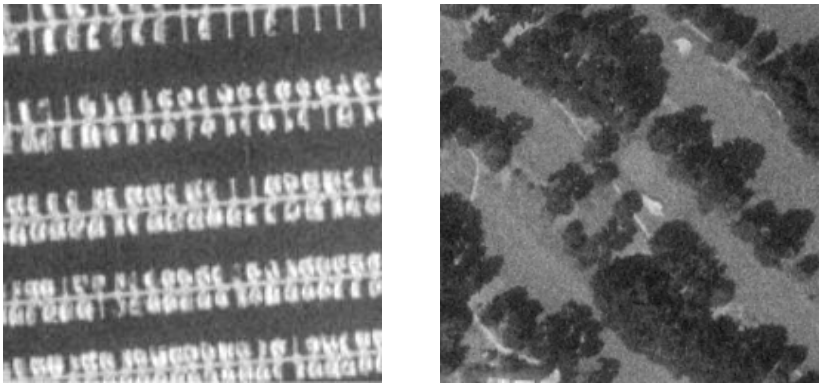
3. Texture can gracefully handle variation in object appearance. Texture analysis can capture regularity in a pattern as well as tolerate a degree of randomness. For example, the boats in a harbor are moored with approximately the same distance to each other but with some variance.

Several geospatial objects contain recurrent spatial patterns with distinct visual appearance. For example, Fig. 1.1 shows the pattern in a *harbor* formed by the arrangement of boats and water, and that formed by the arrangement of trees and grass in a *golf course*. These patterns enable most humans to easily recognize the corresponding object, provided that they have seen it before (even if only briefly). Such spatially recurrent patterns that are characteristic of an object are termed the *texture motifs* of the object. Thus, the pattern formed by boats and water is a texture motif of a harbor, and the arrangement of trees and grass is a texture motif of a golf course.

More importantly from a computational perspective, such recurrent patterns have the property of being distinctive in both their spatial appearance and in their frequency distribution. Thus the spatial appearance of such patterns can be studied via their frequency domain characteristics. By performing texture analysis using Gabor filters at different scales and orientations (Chapter 3), these patterns can be efficiently 1) described in the frequency domain, and 2) localized in the spatial domain. Fig. 1.2 shows the output obtained by convolving two Gabor filters at chosen scales and orientations with the harbor instance in Fig. 1.1(a).

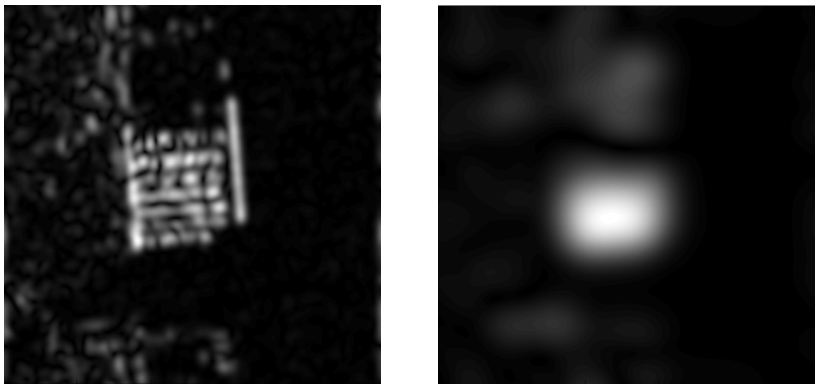


(a) (b)



(c) (d)

Figure 1.1: Examples of geospatial objects and their texture motifs: (a) an instance of a *harbor* object (delineated with a white border); (b) a *golf course* instance; (c) texture motif of harbors, i.e. the arrangement of boats and water; (d) texture motif of golf courses, i.e. the arrangement of trees and grass.



(a) (b)

Figure 1.2: Output obtained by convolving two Gabor filters at chosen scales and orientations with the harbor instance in Fig. 1.1(a). The output in (a) is in response to the frequency of boats parked side by side, and that in (b) is in response to the frequency of the rows of boats.

It can be seen that the first filter responds to the frequency of boats parked side by side, and the second responds to the frequency of the rows of boats. It is thus possible to localize these patterns in the spatial domain without having to perform image segmentation or edge detection/linking. This implies a decrease in the complexity of object description and search.

Gabor filter-based texture analysis thus has the ability to describe higher-order structure in objects. We exploit this ability to address the problem of visually detecting geospatial objects containing texture motifs. We do this by translating the problem of detecting objects to that of detecting their texture motifs. Methods involving a combination of spatial and frequency analysis are used for describing texture motifs. We demonstrate that such an approach is effective in detecting a variety of objects in geospatial images.

1.2.2 Learning Models from Examples

In order to detect geospatial objects, computational models are required for representing their texture motifs. More importantly, such models have to be learned from object examples. Thus, *object modeling* is posed as the problem of learning the texture motifs of an object given a set of examples. In this work, multiple complementary approaches are presented for learning texture motif representations.

In the first approach, a two-layered model is proposed for representing texture motifs. The first layer learns the local intensity variations in the motif that form textural elements such as flat areas, bars, edges, and so on. These can be interpreted as the low-level building blocks of the motif. In the case of boats-and-water motif of harbors, these may correspond to water, boats, and edges between them. We show that these local intensity variations can be effectively captured and described by low-level texture features based on Gabor filters at multiple scales and orientations. Assuming that the texture features generated by different elements populate different volumes of the texture feature space, it is possible to statistically learn the elements of a pattern. In this work, a semi-supervised statistical approach is adopted for this task. The second layer of the representation is the spatial distribution of low-level texture elements in the texture motif, since this influences its distinct visual appearance. A Gaussian mixture model (GMM) for this is learned from examples using features derived from histograms of texture elements in spatial neighborhoods. Confidence measures generated using this model are then used for detecting object presence.

The above approach only describes the likelihood of two elements occurring in

a spatial neighborhood. The second approach adopts a hidden Markov modeling (HMM) framework for learning adjacency relations between elements. Consider a sequence of pixel sites along a line in an object. Each pixel generates an observation, say the texture feature extracted in its neighborhood. Suppose this sequence is modeled as a HMM wherein the *state* of each pixel corresponds to the local texture element, and the *state transitions* from one pixel to the next corresponds to the spatial arrangement of the elements. Then the probability of a transition from a state S_i to a state S_j can be interpreted as the probability of adjacency of the two corresponding texture elements. In this approach, it is possible to combine the two layers of the previous approach into a single stage of learning.

In the third approach, we address the detection of objects with *quasi-periodic* texture motifs. When a texture motif comprises nearly periodic patterns at one or more scales and orientations, it is possible to avoid representing it in a high-dimensional space. In this case, it is not necessary to capture the "amount" of texture at each scale and orientation, but just the *presence* or *absence* of a strong pattern at certain characteristic scales and orientations. Thus texture motifs are just viewed as patterns that respond strongly to Gabor filters at characteristic scales and orientations. For the purpose of detecting objects, it is useful to focus attention on regions that respond at scales corresponding to its texture motifs. Objects are then detected by identifying regions that obey certain "textural rules," which are in the form of relations between their characteristic scales and orientations, and certain proximity criteria. For example, to detect a harbor, we detect the two dominant recurrent patterns as shown in Fig. 1.2, and verify whether they

are perpendicular to each other. These rules can be fully user-specified or learned from examples with user-specified constraints. Note that this paradigm deviates from the probabilistic frameworks discussed earlier, while still preserving the central notion of texture motifs. Furthermore, it has the advantage of avoiding the issues related to clustering and density estimation in a high-dimensional spaces.

1.3 Summary of Contribution

The central goal of this dissertation is to provide methods for the detection of specified objects in geospatial images. Toward this goal, two issues are attended to, namely 1) the use of frequency-domain texture analysis in object detection, and 2) learning texture-based appearance models for objects from examples. The specific technical contributions are as follows.

1. The concept of *texture motifs* for detecting objects in geospatial images [4] (Chapter 4).

2. A two-layered probabilistic modeling framework for learning a representation for texture motifs of objects from their examples (Chapter 4).

3. A hidden Markov modeling (HMM) framework for learning adjacency relations between texture elements that form a motif [58] (Chapter 5).

4. A framework for the detection of objects with quasi-periodic texture motifs that exploits relations of individual scales, orientations, and proximity of the texture motifs of the object. (Chapter 6).

5. A modification [3] of the MPEG-7 homogeneous texture descriptor that nearly halves its dimensionality, while maintaining comparable retrieval performance (Chapter 3).

1.4 Organization of the Dissertation

This dissertation is organized as follows. Chapter 2 provides a detailed overview of previous work on the application of both spatial and texture analysis to geospatial image understanding. There is also a discussion of work related to the application of texture to object detection in general. Chapter 3 introduces the fundamentals of texture analysis using Gabor filters at different spatial scales and orientations. It also proposes a modification of the MPEG-7 homogeneous texture descriptor that nearly halves its dimensionality, while maintaining comparable retrieval performance. Chapter 4 starts by introducing the concept of texture motifs for object detection. It then describes a two-layered model for learning a representation for texture motifs from object examples. The model involves learning the low-level texture elements that form a motif, and the spatial distribution of the elements in a spatial neighborhood. Chapter 5 presents a HMM-based method for learning adjacency relations between texture elements that form a motif. Chapter 6 proposes a framework for the detection of objects with quasi-periodic texture motifs. This framework exploits relations of individual scales, orientations, and proximity of the texture motifs of the object. Chapter 7 concludes with a discussion of some of the outstanding issues and future directions of this work.

Chapter 2

The Background

There has been extensive use of spatial analysis as well as texture analysis in geospatial image understanding. Spatial methods have proved to be successful in interpreting a variety of complex geospatial scenes. These have also been used for the detection of specific objects with "hand-crafted" models. A detailed discussion of these methods is in order to avail a better understanding of the advantages of using texture analysis for detecting compound geospatial objects.

Texture analysis has also long been recognized as a very useful tool for the analysis of remote-sensed imagery. This follows from the ability of texture to quantify visual structure, a virtue that is used for distinguishing distinct entities in geospatial imagery. However, a discussion of its application in the geospatial realm reveals that it has been primarily used for the classification of major terrain types and land use patterns. There has been relatively much less work on the application of texture for detecting more complex geospatial objects. The main goal of this dissertation is to show that texture has far more potential in the

geospatial domain than has been explored in previous work.

This chapter begins by describing spatial analysis methods that have been used for tackling general scene interpretation as well as specific object detection problems. This is followed by previous work on applying texture analysis to the interpretation of geospatial imagery. Finally, we describe the work of a few researchers who have inspired the use of texture in the object detection problem, though not in the geospatial domain.

2.1 Spatial Analysis of Geospatial Images

There has been a wide body of work on the spatial analysis of complex geospatial images. We divide the work into two categories, namely 1) scene interpretation methods, and 2) object-specific methods. Scene interpretation methods are usually complex systems that perform detailed interpretations of geospatial scenes, given some amount of contextual knowledge about the scene (e.g. airport, housing development). The interpretation includes the detection of smaller objects within the scene. Object-specific methods address the detection of specific geospatial objects such as buildings, roads, vehicles, etc. These methods often use contextual information about a particular object to robustly model and detect the object.

It should be noted here that we are focussing on the computer vision problem of analyzing geospatial images based on their visual appearance. Therefore, the class of input data comprises images sensed in the visual spectrum, i.e. *panchromatic* imagery, as opposed to *multispectral* imagery. Multispectral data is obtained by sensing the scene in different spectral (wavelength) bands, some of which may

lie outside the visual spectrum. This provides valuable information regarding the material properties of objects (concrete, vegetation, etc.). However, such data is far more expensive and less easily available than visual images. In this work, we shall use the term *geospatial images* to refer exclusively to panchromatic images.

2.1.1 Scene Interpretation Methods

Given some amount of contextual knowledge about a scene (e.g. airport, housing development), several systems have been proposed for performing a detailed interpretation of the scene. These methods usually divide an image into spatial units (closed regions, lines, etc.) through image segmentation or edge detection/linking. Spatial relations between units are analyzed using relational models such as production systems [55], semantic networks [59, 63], human-specified constraints or *rules* [19], and evidential reasoning [51]. These frameworks are essentially used for humans to specify spatial constraints among the constituents of a scene or object.

As early as 1980, Nagao and Matsuyama [55] developed an aerial image understanding system that was shown to be capable of locating several objects. Images are segmented into elementary regions based on the multispectral properties of each pixel. The elementary regions are classified into characteristic regions based on a combination of spatial and spectral information, for example, large homogeneous regions, elongated regions, shadow region, vegetation, water, etc. Objects such as buildings, roads, and crops are distinguished by identifying characteristic regions that satisfy human-specified "production rules," which are described by `IF (precondition) THEN (action)`. Locational constraints and spatial ar-

rangement rules are utilized to identify context-sensitive objects such as cars on roads and regularly arranged houses.

McKeown et al. [19, 18] proposed *rule-based* systems that used knowledge in the form of user-specified construction rules in order to interpret objects such as airports and housing developments. Examples of rules for airports are "runways are rectangular" and "taxiways are perpendicular to runways." It should be noted that the goal of such a system is not to detect an object in a scene. However, given knowledge about the type of scene (e.g. airport), the system could interpret it in detail by starting with segmented regions and applying spatial constraints to generate hypotheses. Although the above work focussed on specific objects, the system architecture was created to be general. Low-level segmentation is carried out using a number of sources, including edges, depth, multispectral information, and texture. In the above work, texture is mainly used as a coarseness measure, for example, grassy areas are *highly textured*.

Semantic networks are graphical frameworks which are often used for representing spatial models in terms of *entities* and *relations*. These are used by Nicolin and Gabler [59] to realize a knowledge-based system for interpreting aerial images. A system architecture is proposed for model-driven interpretation of suburban scenes. After segmentation, structural analysis is performed to detect regular arrangements between image segments. In doing this, certain principles from Gestalt psychology such as similarity and proximity are utilized. Interpretations are generated in a hierarchical "hypothesize and test" paradigm. Quint [63] developed MOSES, a system for the recognition of objects in aerial images. It utilizes semantic networks to represent models of various objects such as buildings, park-

ing lots, and cars. Compound structured objects such as parking lots can be analyzed by combining image primitives (lines/regions) with a graphical parts model.

Bottom-up (data-driven) and top-down (model-driven) analyses are combined by Matsuyama and Hwang [51] in SIGMA, a system that combines to interpret complex aerial scenes. Their low-level vision expert performs spatial image segmentation to arrive at features such as lines and rectangles. Object recognition is performed within a spatial reasoning framework that accumulates evidence for generating hypotheses of object presence based on spatial relations between objects. The system is shown to be capable of locating cultural structures such as houses, roads, and driveways.

Smyrniotis and Dutta [73] describe a knowledge-based system capable of recognizing man-made objects. The architecture enables goal-oriented image analysis starting from a low-level segmentation of the image into regions of interest. Results are presented on target detection in infra-red imagery and airplane recognition in airport scenes. In [50], Matsuyama discusses combination of AI techniques with geometric models for detecting simple objects such as houses and planes.

2.1.2 Object-Specific Methods

The detection of specific geospatial objects such as buildings, roads, vehicles, etc. has been explored adequately in the literature. In object-specific methods, human knowledge about the concerned object is often exploited to guide their detection. In particular, the detection of *buildings* and *roads* are well-researched topics, presumably due to their importance in cartography and city planning.

The following is a brief overview of notable work that applies spatial analysis for detecting geospatial objects.

Building extraction from aerial imagery is a problem several researchers have focussed on. Most of them make use of properties and relations of lines, corners, rectangles, and shadows. Huertas and Nevatia [35] approach building detection through the detection of lines and corners, and analysis of shadows. Chains of corners connected by edges are analyzed to generate building hypotheses. The rectangularity of buildings is exploited as a simplifying constraint. Irvin and McKeown [36] describe computational techniques for utilizing the relationship between shadows and man-made structures to aid in the automatic extraction of buildings from aerial imagery. Shufelt and McKeown [72] improve building detection by fusing hypotheses across different detection systems. They also merge hypotheses from stereo pairs and multi-temporal images. Subsequent methods [72, 60] also make use of multiple views for improving the robustness of building modeling and detection. Reno and Booth [67] use a two-dimensional viewer-centered reference model for objects to drive their segmentation from aerial images. The model provides a continuous probability density for any deformations about a reference shape. They demonstrated results on objects such as buildings and airplanes. Mueller et al. [52, 53] detect small objects such as houses from high-resolution satellite images with the help of multi-step segmentation and template matching with stored shape models.

Road detection is another topic of interest in the geospatial domain. One common approach relies on a geometric information such as the presence of parallel lines and curves. Some methods also exploit contextual and radiometric informa-

tion. Another common way to extract roads is to track them in high-resolution imagery by extrapolation and matching of profiles after a possibly manual selection of starting points. For a fully automatic extraction, the tracking can be accompanied by an approach to extract roads, which can be used to find starting points. Methods that use snake-based segmentation guided by external constraints are also popular for road detection. A thorough survey of road extraction techniques can be found in [26, 25, 1].

Huertas et al. [34] attack the problem of detecting runways from high-resolution airport scenes. The detection involves the use of image analysis tools guided by the knowledge of runway geometry and markings. Runways are hypothesized based on the width and orientation of detected anti-parallel lines, and verified by detecting markings.

2.2 Texture Analysis in the Geospatial Domain

Land-cover classification has been the primary application of texture analysis in this domain. Remotely sensed images of the earth are generally covered by distinct regions corresponding to many natural and man-made features. Examples include water, forests, agricultural fields, urban settlements, mountainous terrain, and so on. It is of great interest particularly in the areas of geography, cartography, and government, to identify and measure different types of land-cover. A common approach for identifying land-cover types in geospatial imagery is to classify pixels based on the textural properties of their neighborhood. In the process, a variety of texture description methods have been used.

One of the simplest texture descriptors is the variance of image intensity computed in a pixel neighborhood, which is a measure of image heterogeneity therein. This method has been applied as a pre-processing step in a rule-based classification of water in Landsat Multispectral Scanner (MSS) images [79]; as a pre-processing step in mapping land cover as natural vegetation, scattered agricultural, and settlements in Landsat Thematic Mapper (TM) and Shuttle Image Radar (SIR-C) images of East Africa [27]; and as a pre-processing step in classifying forest types in Japanese Earth Resources Satellite-1 (JERS-1) images of Amazonia [23]. Though simple to compute, local statistical measures such as variance have limited scope because they ignore pixel structure.

Gray level co-occurrence matrices (GLCM) are perhaps the geographer's favorite means of computing texture features, due to their simplicity and explicit consideration of structure. In a nutshell, a GLCM tabulates the frequencies with which different gray levels occur in a certain spatial configuration (usually defined by distance and direction). After Haralick [29] proposed a set of 14 texture features based on co-occurrence matrices, such features have been extensively used for remote-sensing applications. Examples include the classification of terrain types in Landsat and aerial photographs [77]; land use classification in the earth resources technology satellite (ERTS) multi-spectral scanner (MSS) imagery [30]; classification of built areas according to their density [38]; classification of forest species composition in high resolution imagery [24]; classification of synthetic aperture radar (SAR) sea-ice imagery [15]; and land cover change detection in moderate resolution imaging spectroradiometer (MODIS) imagery [12]. Extensions have also been proposed, such as a generalized co-occurrence matrix for

multi-spectral analysis [31].

Variograms are being increasingly applied for texture analysis in remote-sensed imagery [70]. The variogram is computed as the expected squared difference between data samples separated by a spatial lag. The semivariogram, or half the variogram, is typically used. Semivariograms are used in [7] to inventory waste-disposal sites in Landsat TM images of Italy. Image regions are classified into the following land cover types: mined land, quarry, dump, landfill, disturbed land, industrial area, urban site, and agricultural area. Semivariograms are used in [2] to classify land cover in Landsat TM images of Turkey into the following types: citrus, two generations of corn, cotton, soil, soya beans, urban, and water. Cross variograms have been prosed for analyzing spatial correlation between different spectral bands. Cross and pseudo-cross variograms are used in [14] to discriminate between rock formations in Landsat TM images of Spain. Cross variograms are also used in [10] to classify five different land cover types in Landsat TM images of Yellowstone National Park.

Since texture usually results from regular or periodic image patterns, its computation often takes place in the frequency domain. The simplest of such texture features are based on the Fourier power spectrum, given by $|F|^2 = FF^*$, where $F(u, v)$ is the Fourier transform of image $f(x, y)$ (and * denotes the complex conjugate). The radial and angular distribution of values in $|F|^2$ are related to the coarseness and directionality of the texture in f [78]. Texture features based on the Fourier power spectrum have been successfully applied to classify different types of vineyards [76] in high-resolution (.25m/pixel) aerial photographs of France. The features used in [76] also proved useful for providing information

about crop spacing, crop orientation, and how the crop was trained.

Wavelet-based texture analysis methods have a distinct advantage over Fourier-based ones in that they provide localization in both space and frequency. The Fourier transform is only localized in the frequency domain, and thus only contains global information about the image in the spatial domain. Unlike Fourier analysis which decomposes an image using a basis of sinusoids with infinite support, wavelet analysis decomposes it using bases of *wavelets* with compact, sometimes finite, support. This enables the latter to localize both in space and frequency, a virtue that results in powerful descriptors for texture, which is a property bound to both spatial neighborhoods and frequencies.

Wavelet-based texture analysis is used in [65] to detect vineyards in high-resolution infrared aerial images of France by applying an adaptive threshold to the wavelet coefficients that correspond to the spacing between the rows of grapes. Multi-resolution analysis using members of the popular Daubechies wavelet family [16] is used in [39] to classify regions in Landsat images as mountainous or flat by computing the standard deviation of the wavelet coefficients in local windows. In [9], Haar wavelets are used to detect agricultural areas with high weed concentration and Daubechies wavelets are used to detect noxious weeds in Digital Compact Airborne Spectrographic Imager (CASI) images of Mississippi. The energies of the different wavelet subbands are used in the classification. Daubechies wavelets are used in [83] to classify regions in aerial images into natural land cover types such as desert, dune, mountain, and forest. The energies of the subbands are also used for classification via a non-standard "best-resolution" multi-resolution decomposition which successively decomposes the middle-frequency subbands.

Fractal geometry, which describes the self-similarity of a shape across multiple scales, can be applied to describe the complexity of a shape. If an image is viewed as a three-dimensional surface whose height at a pixel location is determined by the pixel value, then the *fractal dimension* [70] of the surface can be used to characterize image texture. The fractal dimension of a surface is a measurement of its complexity, and is thus a scalar measurement of image texture. The higher it is, the rougher the texture. By measuring the fractal dimension of image surfaces, land cover in ATLAS images of Louisiana is classified [54] into six classes: single-family homes with less than 50% tree canopy, single-family homes with more than 50% tree canopy, commercial, woodland, agriculture, and water. Fractal methods are used in [61] to determine which of the 224 spectral bands of Airborne Visible Infra-Red Imaging Spectrometer (AVIRIS) are most suited for classifying land cover types in both urban and rural regions.

Another popular framework for texture analysis is the Markov random field (MRF). In this framework, an image is treated as a realization of a two-dimensional lattice of random variables, or a random field. This is accompanied by the assumption of Markovianity, i.e. a pixel (random variable) is statistically dependent only on a local neighborhood. The equivalence between MRF and Gibbs random field (Hammersley-Clifford theorem [28]) allows a tractable way of computing the joint probability of the pixels in an image through the Gibbs distribution, a global property. The MRF framework only provides local properties, namely a pixel's conditional probability [41]. Texture analysis based on MRF is used in the classification of synthetic aperture radar sea-ice imagery [15]; the labeling of hyperspectral AVIRIS image regions as urban or non-urban [66]; and classification

and similarity retrieval in X-band synthetic aperture radar images using multiple models [71]. MRF framework is often used to segment remote-sensed images in an unsupervised manner. The spatial interaction at the pixel level (or the texture) is modeled using a low-level MRF, and the spatial layout of the processes that generate the different textures is modeled using a separate high-level MRF, sometimes referred to as the smoothing prior. This is the approach taken in [82] and [81] to segment Landsat images into five land-cover classes: water, conifers, deciduous, open fields, and epidemics.

Structural texture analysis refers to a class of spatial domain methods that characterize texture as a set of primitives with a certain spatial arrangement. Many of the spatial analysis methods described in Sec. 2.1.1 could be viewed as some form of structural texture analysis. For example, the primitives may be buildings, which form a "texture" with their spatial arrangement. However, most of the methods in Sec. 2.1.1 do not explicitly view such structure as a texture. One approach that does is that of Lumia et al. [42] in which an image is first segmented into a set of closed regions called units. The units along with their properties constitute the primitives. In the training phase, primitives (clusters of units) and their contextual relationships are learned from representative images for several texture classes. During classification, each unit is assigned to a texture class based in its cluster type and that of its neighbors. Using this method, an aerial image of an urban area image is classified into two texture classes, namely large and small buildings.

2.3 Texture in Object Detection

The importance of texture as a visual primitive has long been acknowledged in computer vision. Of particular importance is the use of texture as a visual cue for object detection. Several researchers have contributed their time to studying this topic. In the following, we describe some prominent past work which address the problem of object detection through the use of texture.

In her dissertation, Mahmood [45] addresses the importance of attentional selection in reducing the combinatorial search that occurs during object detection. She proposes both data-driven and model-driven selection mechanisms using new representations for color, texture, and line group information in images. Texture is stressed as an important visual cue in object detection. Texture is modeled as a short-space stationary process, and is represented using a method called the linear prediction (LP) spectrum. In the data-driven approach, the above method is used for isolating texture regions and each region is ranked by a measure of texture saliency. The LP spectrum is also shown to to be effective for model-driven selection. This representation allows the isolation of a model texture in scenes without requiring a detailed segmentation. Furthermore, it also provides a solution of the pose of the texture region on a model object. Results are demonstrated by using a model texture patch on a cup, and detecting the patch (and hence the cup) in two different scenes with slight changes in illumination and pose. An issue that is not addressed is that of learning the variation of a textural pattern across different instances of an object.

Braithwaite and Bhanu [6] use tuned Gabor filters for detecting objects in

infrared images with strongly oriented and periodic features. It uses a two-stage representation called "hierarchical" Gabor filters. The first stage uses a set of wide-bandwidth Gabor filters to extract local measures from an image, namely the marginal magnitude, peak region of marginal magnitude, and dominant spectral orientation. The second stage makes certain spatial groupings explicit by creating small-bandwidth Gabor filters that are tuned to elongated contours or periodic patterns. They demonstrate the detection a tank by using a filter tuned to the frequency corresponding to the periodic pattern of the rows of wheels. The filters are manually tuned to the patterns of interest.

Jain et al. [37] use a feature-based segmentation method to address the problem of segmenting objects in complex backgrounds. A given image is convolved with a bank of even-symmetric Gabor filters at multiple spatial scales and orientations. A selection of these filtered images is made and each (selected) filtered image is subjected to a nonlinear (sigmoidal like) transformation. Then, "Gabor features" at each transformed image pixel are computed as a measure of texture energy in a window around the pixel. The features and their spatial locations are clustered to obtain a segmentation of the original image. Each pixel is assigned a cluster label that identifies the amount of mean local energy the pixel possesses across different spatial orientations and frequencies. The approach is demonstrated on objects in visual and infrared images such as tanks, cars, and fingerprints. Results show that the region corresponding to the object is usually segmented correctly. For actually identifying which region corresponds to the object, the authors briefly suggest the use of its "unique signature" of Gabor features. However, the issue of modeling the variation of this signature across

different instances of the same object is not addressed. Therefore, this method is mainly data-driven and not model-driven.

Schmid [69] proposed a method to construct models for objects with texture-like visual structure given positive and negative example images. A two-level feature representation was used. Firstly, low-level descriptors are extracted using rotation-insensitive circular Gabor-like filters. These are clustered over the training images to obtain k "generic" descriptors. Secondly, joint probabilities (frequencies) of the k cluster labels in a neighborhood centered at each pixel are computed. These k-dimensional frequency vectors are clustered to get "spatial-frequency" clusters. Given a set of positive and negative examples, the significance of each spatial-frequency cluster for the model is evaluated. Retrieval is done by averaging a score computed for each pixel in a database image, based on the descriptors and joint probabilities in its neighborhood. This score can also be used for localizing the object in each image. Results were demonstrated for objects such as zebras, cheetahs, giraffes, and faces. This work uses rotation-insensitive features which could be a disadvantage in the case of geospatial objects. Features obtained using orientation-selective Gabor filters enable us to capture orientational relationships between spatial patterns which are often important for modeling geospatial objects (see, for example, Fig. 1.2).

2.4 Summary

Judging by previous work, it is clear that the importance of image texture as a visual primitive for scene interpretation has been recognized. Researchers have

also pointed out the use of texture as a visual cue for object detection, in both data-driven and model-driven approaches. However, most of the work employing texture for model-driven object detection does not pertain to geospatial objects. Several objects in geospatial images contain structural patterns that could be modeled and detected using texture.

On the other hand, texture analysis in the geospatial domain has primarily focussed on the classification of major terrain types and land use patterns. Examples include water, forests, agricultural fields, urban settlements, mountainous terrain, and so on. There has been little work on the application of texture for detecting more complex geospatial objects, such as harbors, golf courses, and airports. The main goal of this dissertation is to show that texture has far more potential in this domain than has been explored in the previous work.

Chapter 3

Structural Analysis using Gabor Filters

Image texture has long been acknowledged as an important visual primitive in computer vision. Texture can be applied to image analysis at three levels. At the *low-level*, it can be used to study the statistical properties of the spatial distribution of pixel intensities in a neighborhood. For a given image region, these properties are encoded as a *feature vector*. Distances between feature vectors are then used to quantify visual dissimilarity between image regions. Using low-level features, mid-level representations are constructed. These usually involve mappings from low-level features into classes or attributes. For example, features can be clustered to discover classes of regions with similar visual patterns, or mapped to attributes such as *highly periodic pattern*, *coarse pattern*, and so on.

For object detection, it is necessary to construct a *high-level* representation of texture, wherein we establish the connection between objects of interest and their

visual patterns via a texture-based *model*. A model-driven search then serves to focus attention on regions that have similar patterns as the object of interest. This approach has the potential for detecting complex geospatial objects such as harbors and golf courses.

In this chapter, we motivate the application of low-level texture analysis using Gabor filters to high-level object detection. First, the fundamentals of texture analysis using Gabor filters at multiple spatial scales and orientations are laid out. This includes a discussion of its application to low-level similarity retrieval, and issues related to normalization and dimensionality reduction of texture descriptors. We then illustrate how Gabor filter outputs reveal visual structure in images. The ability of Gabor filter outputs to reveal structural characteristics of complex geospatial objects is the main motivation for its use in object detection.

3.1 Gabor Filters for Texture Analysis

Wavelets filter banks based on Gabor functions have proven quite effective for describing texture. The decision to use scale and orientation selective Gabor filters to describe texture is based on a number of factors, including their interesting mathematical and psycho-visual properties. As mentioned earlier, an advantage of using wavelets is their ability to localize in both space and frequency. Gabor functions have the attractive property of being optimal in the sense of minimizing the joint two-dimensional uncertainty in space and frequency [17]. Thus, they are well suited for texture analysis for which good localization is required in both spatial and frequency domains. Gabor filters are appealing from a psycho-visual

perspective as well. They are known to approximate the characteristics of certain cells in the visual cortex of some mammals [49]. Gabor filters provide good models for certain pre-attentive visual processes of humans.

3.1.1 Two-dimensional Gabor Filters

Texture analysis is performed by applying a bank of scale and orientation selective Gabor filters to an image. These filters are constructed as follows [48]. A two-dimensional Gabor function $g(x, y)$ and its Fourier transform $G(u, v)$ can be written as:

$$g(x,y) = \left(\frac{1}{2\pi\sigma_x\sigma_y}\right) \exp\left[-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}\right) + 2\pi jWx\right] \tag{3.1}$$

and

$$G(u,v) = \exp\left\{-\frac{1}{2}\left[\frac{(u-W)^2}{\sigma_u^2} + \frac{v^2}{\sigma_v^2}\right]\right\} \tag{3.2}$$

where $\sigma_u = 1/2\pi\sigma_x$ and $\sigma_v = 1/2\pi\sigma_y$. A class of self-similar functions referred to as Gabor wavelets is now considered. Let $g(x, y)$ be the mother wavelet. A *Gabor filter bank* can be obtained by appropriate dilations and translations of $g(x, y)$ through the generating function:

$$\begin{aligned} g_{s,k}(x,y) &= a^{-s} g(x', y'), \quad a > 1,\ s \in 0, \ldots, S-1,\ k \in 0, \ldots K-1 \\ x' &= a^{-s}(x\cos\theta + y\sin\theta) \quad \text{and} \\ y' &= a^{-s}(-x\sin\theta + y\cos\theta) \end{aligned} \tag{3.3}$$

where $\theta = k\pi/K$ is the orientation of the filter w.r.t. the vertical. The indices k and s indicate the orientation and scale of the filter respectively. K is the total

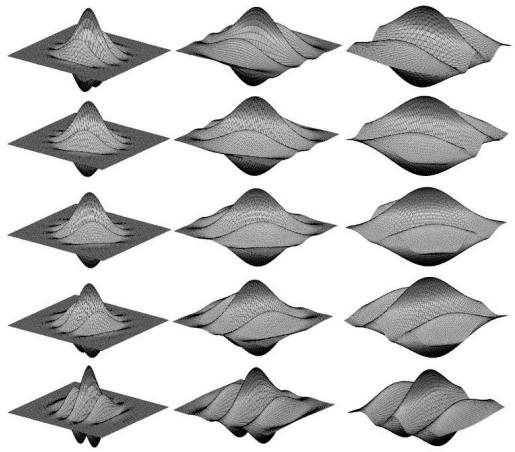


Figure 3.1: A bank of Gabor filters $g_{s,k}(x, y)$ tuned to combinations of three scales ($S = 3$) and five orientations ($K = 5$).

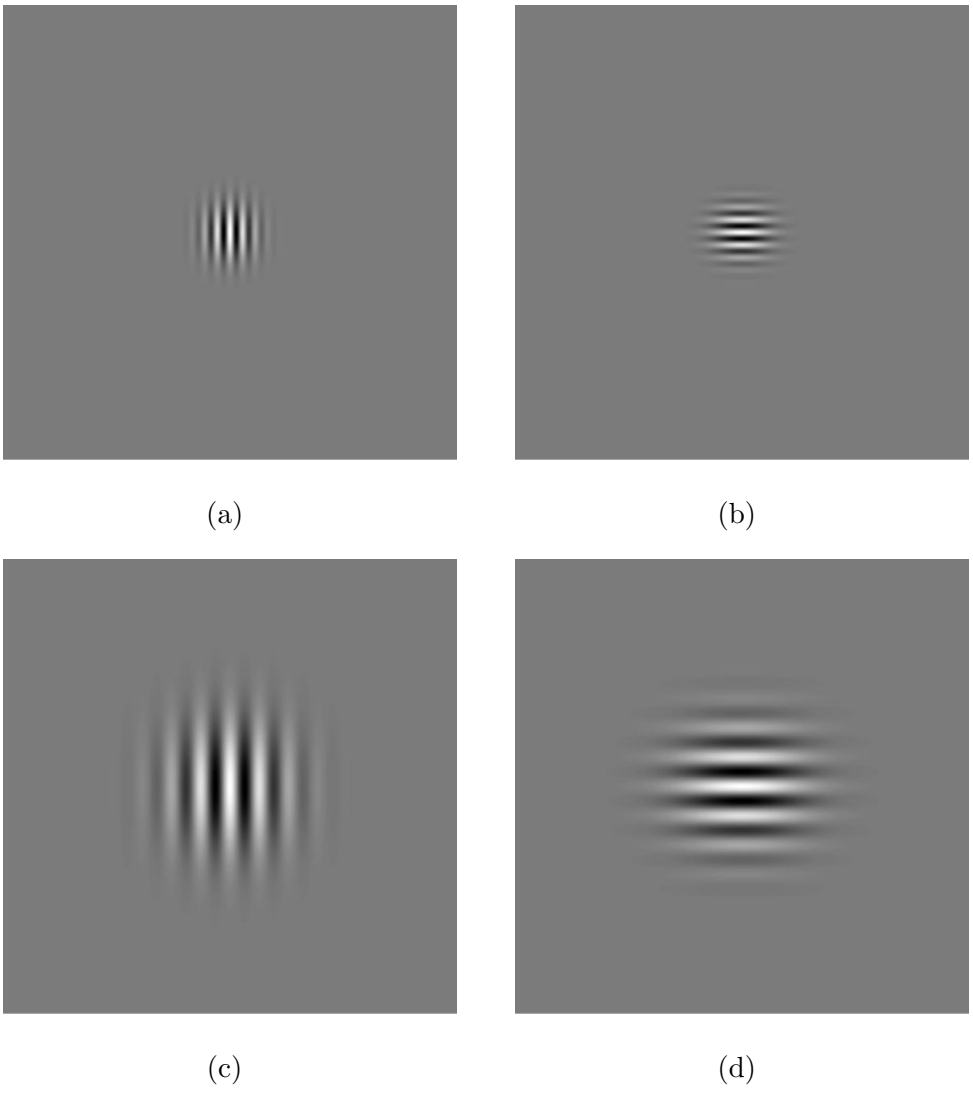
number of orientations and S is the total number of scales in the filter bank. The scale factor a^{-s} in (3.3) is meant to ensure that the energy is independent of s.

The filter bank parameters $\{\sigma_x, \sigma_y, a, W\}$ are computed by the method described in [48]. Given the input specifications S, K, and the upper and lower center frequencies, $W = U_h$ and U_l, of the Gabor filters in the filter bank, the parameters of the filter bank are computed as follows.

$$\begin{aligned} a = \left(\frac{U_h}{U_l}\right)^{\frac{1}{S-1}}, \quad \sigma_u = \frac{(a-1)U_h}{(a+1)\sqrt{(2\ln 2)}}, \text{ and} \\ \sigma_v = \tan\left(\frac{\pi}{2K}\right)\left[U_h - 2\ln\left(\frac{2\sigma_u^2}{U_h}\right)\right]\left[2\ln 2 - \frac{(2\ln 2)^2\sigma_u^2}{U_h^2}\right]^{-\frac{1}{2}}. \end{aligned} \tag{3.4}$$

We compute σ_x and σ_y as $\sigma_x = 1/2\pi\sigma_u$ and $\sigma_y = 1/2\pi\sigma_v$.

Fig. 3.1 displays the real components of a bank of Gabor filters at three scales ($S = 3$) and five orientations ($K = 5$). Fig. 3.2 shows the real parts of example filters at two different scale-orientation pairs. Note that the smaller the scale s, the higher the frequency of the filter and the less its support.



(a) (b)

(c) (d)

Figure 3.2: Real parts of Gabor filters at different scales and orientations: (a) scale $s = s_1$ and orientation $n = 0$ ($\theta = 0°$); (b) $s = s_1$, $n = K/2$ ($\theta = 90°$); (c) $s = s_2$ ($s_2 > s_1$), $n = 0$ ($\theta = 0°$); (d) $s = s_2$, $n = K/2$ ($\theta = 90°$).

3.1.2 Application to Similarity Retrieval

Texture descriptors derived from Gabor filter banks have been widely used for browsing and similarity retrieval in image databases [57, 62, 43, 44]. These are commonly obtained by applying a Gabor filter bank to the texture image and deriving appropriate statistics of the filter outputs. For a texture image $I(x, y)$, the filter output at scale s and orientation k is given by

$$F_{s,k}(x, y) = |g_{s,k}(x, y) * I(x, y)|, \tag{3.5}$$

where $*$ denotes convolution. The filter output at pixel (x, y) can be represented as

$$\mathbf{c}(x, y) = [F_{0,0}(x, y) \;\; F_{0,1}(x, y) \;\; \ldots \;\; F_{S-1,K-2}(x, y) \;\; F_{S-1,K-1}(x, y)]^T. \tag{3.6}$$

A homogeneous texture descriptor based on Gabor filters was proposed in [48]. Since then, the descriptor has been known for its effectiveness and efficiency, and a modified version of it has been adopted by the MPEG-7 Multimedia Content Description Interface standard [47]. The texture descriptor for S scales and K orientations is given by

$$\mathbf{f}_{\mu\sigma} = [\mu_{0,0} \;\; \sigma_{0,0} \;\; \mu_{0,1} \;\; \sigma_{0,1} \;\; \ldots \;\; \mu_{S-1,K-1} \;\; \sigma_{S-1,K-1} \;\; \mu_I \;\; \sigma_I]^T, \tag{3.7}$$

where $\mu_{s,k}$ and $\sigma_{s,k}$ are the mean and standard deviation of the filter outputs $F_{s,k}(x, y)$. μ_I and σ_I are the mean and standard deviation of the pixel intensities of the image. Note that the dimensionality of $\mathbf{f}_{\mu\sigma}$ is $2SK + 2$.

Representing texture as a point in this multi-dimensional feature space is very useful since closeness in the feature space has been demonstrated to correspond

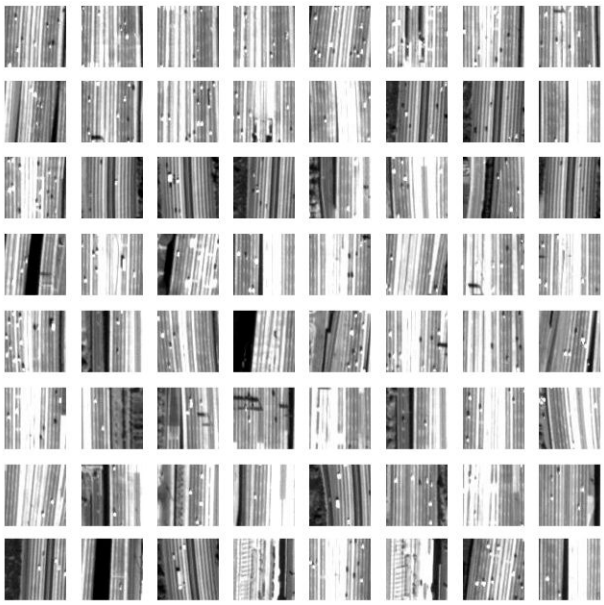


Figure 3.3: The results of a nearest-neighbor query in a dataset of satellite imagery. The top-left tile is the query tile. The other tiles are the results.

well with visual similarity. Visual dissimilarity between two textures is quantified by computing a distance (usually L_1 or L_2) measure between their descriptors. This measure can then be used to perform content-based similarity retrieval within a set of images. Fig. 3.3 shows an example of applying the descriptor to similarity retrieval. The dataset contains 400,000 subimages obtained by dividing large aerial images into tiles. A descriptor is computed for each subimage. The top-left tile in Fig. 3.3 is the query tile, and others are those closest to the query in the feature space. Note that the retrievals are of similar orientation to the query, owing to the dependence of $\mathbf{f}_{\mu\sigma}$ on image orientation.

3.1.3 Normalization Issues

Gabor filter outputs contain implicit information about visual structure in images. In this work, the vector of filter outputs, $\mathbf{c}(x, y)$ in (3.6), is frequently used as a feature vector to describe the texture in the neighborhood of pixel (x, y). However, the convolution of images with Gabor filters at different scales tends to give outputs of different dynamic ranges. It is observed that the higher the scale, the larger the dynamic range of the filter output.

To avoid this problem, the descriptors are normalized over a representative dataset, so that each dimension has approximately the same dynamic range. The normalization method used here forces all the dimensions of $\mathbf{c}(x, y)$ at a particular scale to have unit variance. Accordingly, the normalized feature vector, $\mathbf{c}^{(n)}(x, y)$, becomes

$$\mathbf{c}^{(n)}(x,y) = [F^{(n)}_{0,0}(x,y) \quad F^{(n)}_{0,1}(x,y) \quad \ldots \quad F^{(n)}_{S-1,K-2}(x,y) \quad F^{(n)}_{S-1,K-1}(x,y)]^T, \quad (3.8)$$

where

$$F^{(n)}_{s,k}(x,y) = \frac{F_{s,k}(x,y)}{\sigma\left[F_{s,*}\right]}. \quad (3.9)$$

$\sigma\left[F_{s,*}\right]$ is the standard deviation of the output at any pixel at scale s over all orientations. It is computed over a set of training images, which are chosen to have enough diversity to cover most patterns. Henceforth in this work, it shall be tacitly assumed that the feature vectors $\mathbf{c}(x, y)$ are normalized this way.

3.1.4 Dimensionality Reduction of the MPEG-7 Descriptor

The high dimensionality and computational complexity of the MPEG-7 texture descriptor, $\mathbf{f}_{\mu\sigma}$ in (3.7), adversely affect the performance as well as the storage, computation, and indexing requirements of a content-based retrieval system. By studying the statistical properties of Gabor filter outputs for texture image inputs, it is possible to derive a modified texture descriptor [3] with comparable retrieval performance, nearly half the dimensionality, and less computational expense.

The format of (3.7) for the texture descriptor is driven by the implicit assumption that the filter outputs have Gaussian-like distributions. Therefore, each of these distributions is taken to be described completely by its mean and standard deviation. However, Dunn and Higgins [22] show (for the one-dimensional case) that the Gabor filter outputs for a texture input have a Rice distribution, given by

$$f_R(r) = \frac{r}{\sigma^2} \exp\left(-\frac{r^2 + A_0^2}{2\sigma^2}\right) I_0\left(\frac{A_0 r}{\sigma^2}\right), \tag{3.10}$$

where $I_0(x)$ is the zero-order modified Bessel function of the first kind. Their proof is based on modeling the input texture as a periodic lattice of *texels* with random perturbations. Texels are similar, but not necessarily identical, geometric primitives that constitute a texture. In (3.10), the parameter A_0 indicates the amplitude of periodicity in the texture, and the parameter σ indicates the amount of noise perturbing this periodicity.

The Rice pdf (see Fig. 3.4(a)) in (3.10) can vary from a Rayleigh pdf for small A_0 ($A_0 \approx 0$) to an approximate Gaussian pdf for large A_0 ($A_0 \gg \sigma$). The

latter case occurs when the texture is well-defined and periodic, with a highly peaked frequency component at the center frequency of the Gabor filter. The filter outputs tend to have a Rayleigh pdf when the frequency components of the texture are weak in the vicinity of the center frequency. Figures 3.4(b-d) show the result of filtering a texture image with Gabor filters of two different center frequencies. Fig. 3.4(c) is closer to a Rayleigh pdf, and Fig. 3.4(d) is closer to a Gaussian pdf. In the latter case, the input has strong frequencies in the vicinity of the center frequency of the filter. In the former case, it does not.

The filter bank in (3.4) used for computing the texture descriptor has predefined center frequencies. Over a wide range of textures, the probability that a given texture has a strong component at a specified center frequency, is small. Hence, we claim that the Rayleigh pdf model for filter output distributions is valid with a higher probability than the Gaussian pdf model that inspires the descriptor in (3.7). The Rayleigh pdf, given by $p(z) = \frac{z}{\gamma^2} \exp\left(-\frac{z^2}{2\gamma^2}\right)$, has only one parameter γ. Therefore, instead of (3.7), we propose the following texture descriptor with dimensionality $SK + 2$,

$$\mathbf{f}_\gamma = [\gamma_{0,0} \quad \gamma_{0,1} \quad \dots \quad \gamma_{S-1,K-2} \quad \gamma_{S-1,K-1} \quad \mu_I \quad \sigma_I]^T, \tag{3.11}$$

where $\gamma_{s,k}$ is the Rayleigh parameter computed over the $F_{s,k}$ in (3.5). Given data samples (or filter outputs) $\{z_1, \dots, z_N\}$, the maximum-likelihood estimates of the parameter $\gamma_{s,k}$ is given by

$$\gamma_{s,k}^2 = \frac{1}{2N} \sum_{i=1}^{N} |z_i|^2. \tag{3.12}$$

Besides reduced dimensionality, a merit of the above descriptor is that it is easy to compute using the old descriptor, without having to repeat the computationally

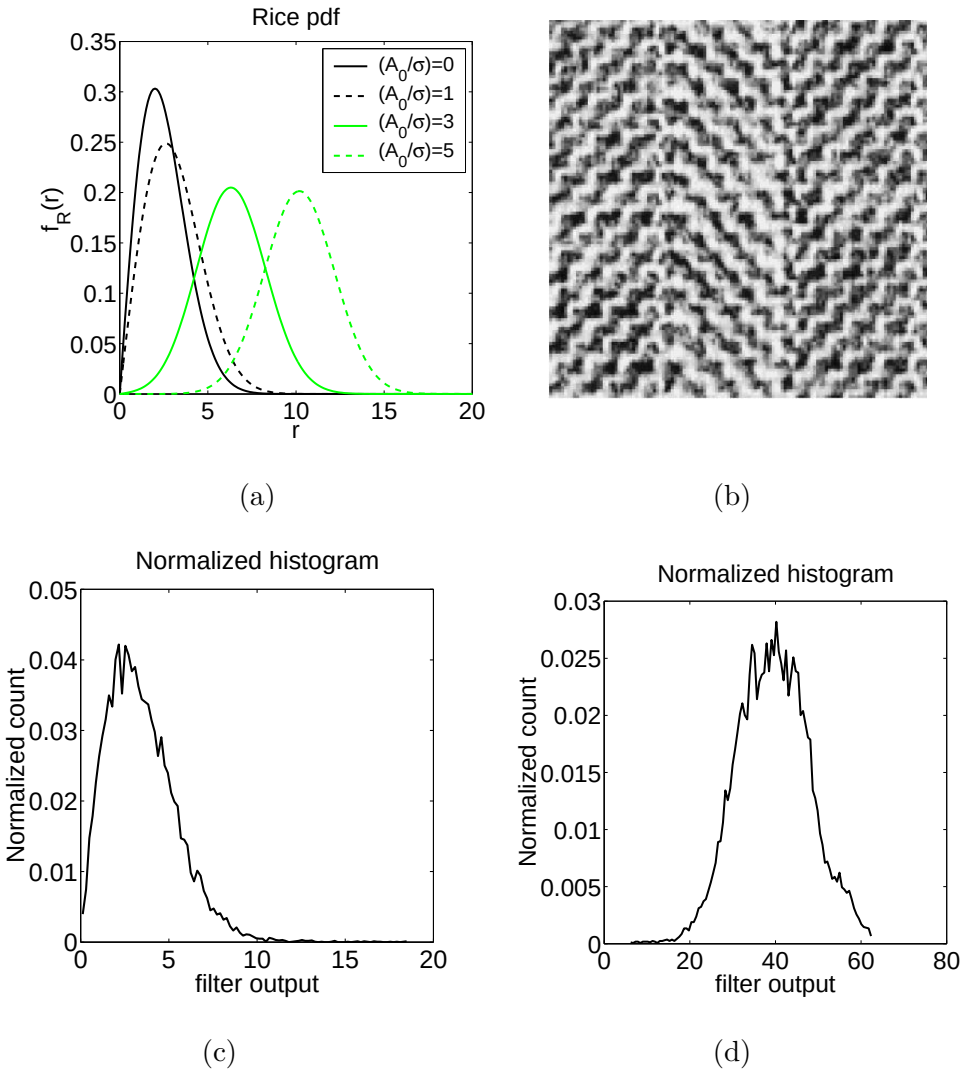


Figure 3.4: (a) The Rice pdf in (3.10); (b) a texture image input; (c) and (d) Histograms (with 100 bins) of Gabor filter outputs for two different center frequencies.

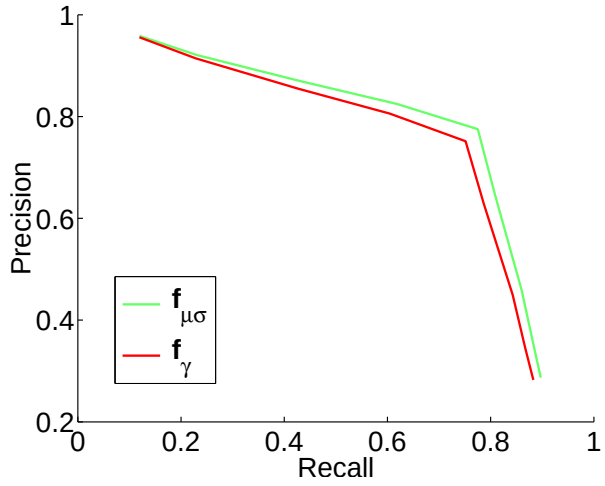


Figure 3.5: Precision vs. Recall curves over the Brodatz dataset.

expensive filtering step. This is useful in case we have precomputed $\mathbf{f}_{\mu\sigma}$ (MPEG-7) features. Since, in (3.7),

$$\mu_{s,k} = \frac{1}{N}\sum_{i=1}^{N}|z_i|, \quad \text{and} \quad \sigma_{s,k}^2 = \frac{1}{N}\sum_{i=1}^{N}|z_i|^2 - \mu_{s,k}^2, \tag{3.13}$$

it can be shown that $\gamma_{s,k}^2 = \frac{1}{2}\left(\mu_{s,k}^2 + \sigma_{s,k}^2\right)$. Also, we can see from (3.12) and (3.13) that, for large values of N, $\mathbf{f}_\gamma$ needs 50% fewer additions in its computation than $\mathbf{f}_{\mu\sigma}$.

Fig. 3.5 shows the precision-recall performance of $\mathbf{f}_{\mu\sigma}$ and $\mathbf{f}_\gamma$ on the Brodatz texture dataset [8], which is widely used for evaluating texture similarity retrieval. The dataset consists of 1856 images (16 from each of 116 texture classes). Since we consider 5 scales and 6 orientations, the dimensionality of $\mathbf{f}_{\mu\sigma}$ is 62 and that of $\mathbf{f}_\gamma$ is 32. Retrieval is done by computing L_1 distances between descriptors. It is observed that the difference in the error-rate between the two curves is less than 3%.

3.2 Texture, Visual Structure, and Object Detection

The previous section described the application of texture for *low-level* similarity retrieval, where the similarity is with respect to a low-level texture description with no explicit meaning of its own. Starting with low-level texture features, researchers have constructed *mid-level* representations which enable easier and more meaningful navigation through image databases. The *texture thesaurus* developed by Ma and Manjunath [44] divide image regions into several visually similar categories by hierarchically clustering their texture features. Each cluster is displayed to the user as the image region closest (in feature space) to its texture "codeword." A user is thus able to navigate the database and look for different types of dominant patterns in the database. Another mid-level representation is the *perceptual browsing component* (PBC) proposed by Wu et al. [80]. Derived using a Gabor filter bank, the PBC provides a quantitative characterization of texture properties such as regularity, directionality, and dominant scale. The PBC allows the user to browse a database and look for textured regions with certain properties.

Thus, at the low-level, texture analysis using Gabor filters enables the retrieval of visually similar regions. At the mid-level, it has the ability to quantify visual structure in images. It enables an explicit characterization of structural attributes such as the density, periodicity, and directionality of arrangement of physical features in images. For example, Fig. 3.6 demonstrates how texture analysis reveals the presence of periodic and directional patterns. Fig. 3.6(a) is an image

tile (128×128 pixels) from a vineyard depicting rows of vines. The image is convolved with a Gabor filter bank at three scales and six orientations, and the means of the filter outputs, $F_{s,k}$ of (3.5), are retained. Let W_s correspond to the period (in pixels) of sinusoidal variation in a filter at scale s ($W_s = a^s/W$, using W from (3.1) and a from (3.4)). Then Fig. 3.6(b) plots the variation in the mean values at each W_s (scale) with the orientation (θ in (3.4)) of the filter. A strong peak is observed at $W_1 = 7.07$ pixels at a 60° orientation. This indicates that the average separation of the vines is 7.07 pixels and the vines are oriented at 60° w.r.t. the vertical. Fig. 3.6(c) shows the real part of the corresponding filter superimposed on the image, corroborating the above conclusions.

Fig. 3.7 illustrates the use of texture for discriminating density of arrangement of physical features. Fig. 3.7(a) and Fig. 3.7(b) show two 128×128 images depicting a higher and lower density of beads, respectively. Fig. 3.7(c) and Fig. 3.7(d) show the corresponding plots of the filter output mean at three orientations ($\theta = 0^\circ, 60^\circ, 120^\circ$) versus the scale s. Note that the former plot peaks at $s = 2$ for all orientations, and the latter peaks at $s = 3$. Smaller scales in general correspond to higher frequencies (Fig. 3.2) and thus denser arrangement of physical features.

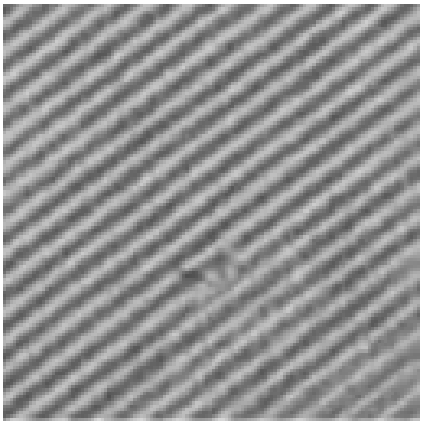
The preceding examples are chosen to be simple and aid an intuitive understanding. In general, however, visual structure in geospatial objects are more complex and contain patterns at several scales and orientations. For example, take the instance of a *harbor* object in Fig. 3.8(a). When the image with the object is convolved with a Gabor filter bank, strong responses are observed inside the harbor at certain scales and orientations, as shown in Fig. 3.8(b) and

Fig. 3.8(c). Through these responses, we can infer the characteristic structural patterns in the harbor. Note that these patterns also related to each other in terms of their relative scales, orientations, and spatial layout. This information is important for describing harbors.

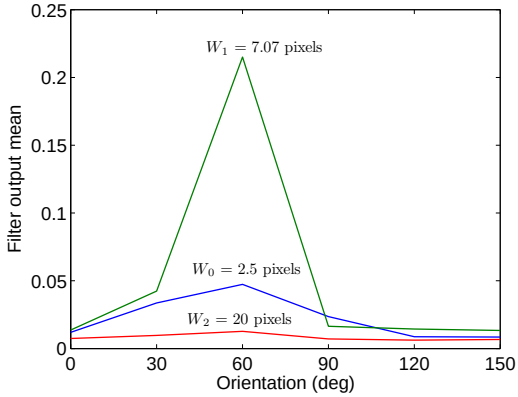
Thus, texture analysis using Gabor filters has the ability to describe interesting structural patterns in geospatial objects. For detecting specified objects, it is necessary to move to a *high-level* representation of texture. Here, the appearance of an object is analyzed from examples using a Gabor filter bank and encoded as a texture-based *model*, which is then used to perform a model-driven search for the object in a scene. This model-driven approach serves to focus attention on regions that have similar patterns as the object of interest. The subsequent chapters describe in detail how appearance models are learned from object examples.

3.3 Summary

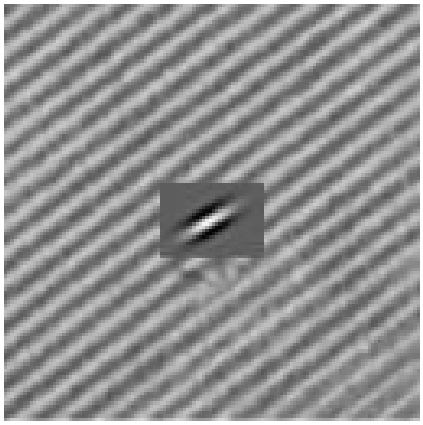
In this chapter, we motivate the extension of texture analysis using Gabor filters from low-level similarity retrieval to high-level object detection. At the *low-level*, Gabor filters are used to study the statistical properties of the spatial distribution of pixel intensities in a neighborhood. Distances between *feature vectors* derived from such analysis quantify visual dissimilarity between image regions. Mid-level texture analysis usually involves mappings from low-level features into visually similar classes or attributes. For object detection, it is necessary to construct a *high-level* representation of texture, wherein we establish the connection between objects of interest and their visual patterns via a texture-based



(a)

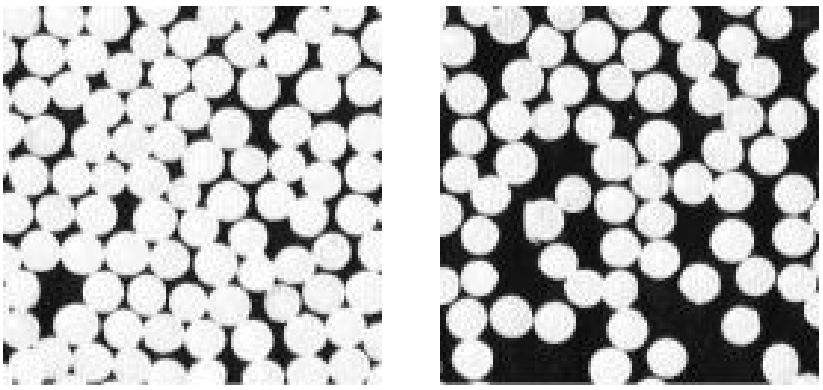


(b)

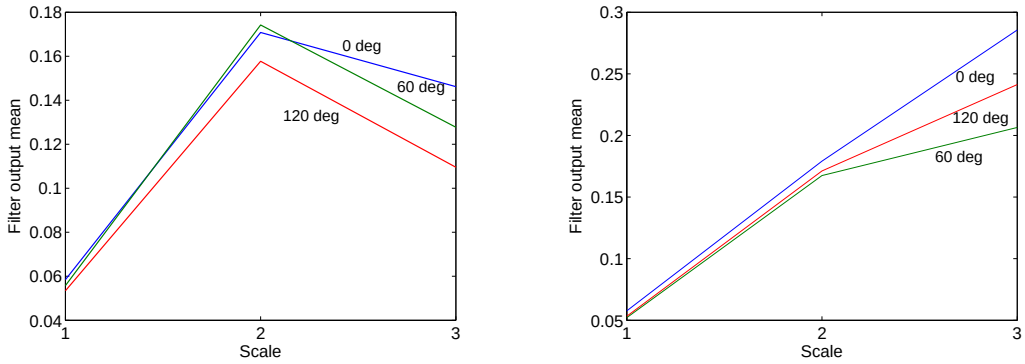


(c)

Figure 3.6: (a) An 128×128 image tile from a vineyard depicting rows of vines; (b) Plot of filter output mean at each scale s (or filter period W_s) versus the orientation (w.r.t. vertical); and (c) The real part of the filter corresponding to the peak in (b) superimposed on the image.

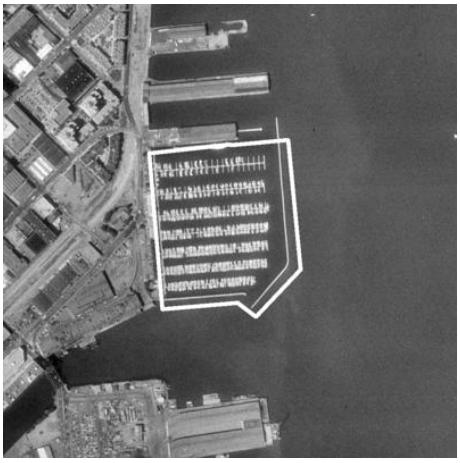


(a) (b)



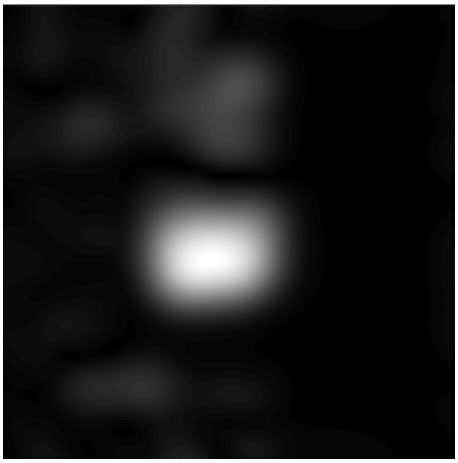
(c) (d)

Figure 3.7: (a) An 128×128 image depicting a high density of beads; (b) An 128×128 image depicting a lower density of beads; (c) Plot for (a) of filter output mean at three orientations ($\theta = 0°, 60°, 120°$) versus the scale; and (d) Plot for (b) of filter output mean at three orientations ($\theta = 0°, 60°, 120°$) versus the scale.



(a)

(b)



(c)

Figure 3.8: (a) An instance of a *harbor* object, with the white line indicating the extent. The ground resolution of the image is 1m/pixel. When it is convolved with a Gabor filter bank, strong responses are observed inside the harbor at two scales. (b) Output of a filter with a orientation (θ in (3.4)) of 0° and scale corresponding to a filter period of 9.3 pixels. This corresponds to an approximate separation of 9.3m between individual boats. (c) Output of a filter with a orientation of 90° and scale corresponding to a filter period of 37.8 pixels. This corresponds to an approximate separation of 37.8m between rows of boats.

model. A model-driven search then serves to focus attention on regions that have similar patterns as the object of interest.

This chapter lays down the fundamentals of texture analysis using Gabor filters at multiple spatial scales and orientations. After illustrating the application of Gabor filter banks in similarity retrieval, some issues related to normalization and dimensionality reduction of texture descriptors are discussed. In particular, a modification of the MPEG-7 homogeneous texture descriptor is proposed, that nearly halves its dimensionality, while maintaining comparable retrieval performance. Moving on from low-level applications of texture analysis, we illustrate how Gabor filter outputs reveal visual structure in images. It is this ability of Gabor filters that motivates its use in object detection. Its efficiency of use further encourages the use of Gabor filters for describing (and detecting) complex geospatial objects such as harbors, golf courses, airports, etc. In the subsequent chapters, we discuss different methodologies of modeling and detecting geospatial objects, all of which make extensive use of Gabor filter banks.

Chapter 4

Texture Motifs for Object Detection

Distinct visual signatures result from many spatial patterns, both natural and human-made, that form objects of interest in geospatial images. Such spatial patterns can be observed in geospatial objects such as golf courses and harbors, instances of which are shown in Fig. 4.1. Notice the recurrent spatial patterns in the harbor formed by the arrangement of boats and water (Fig. 4.2). In the golf course, the recurrent pattern formed by the arrangement of trees and grass is quite distinctive. Such spatially recurrent patterns that are characteristic of an object are termed the *texture motifs* of the object. Thus, the pattern formed by boats and water is a texture motif of a harbor, and the arrangement of trees and grass is a texture motif of a golf course. Of course, there exist objects with multiple texture motifs. These include airports (parked plane pattern, hangar pattern, runway markings) and agricultural areas (different crop patterns). An

example of an airport and its texture motifs are shown in Fig. 4.3.

These recurrent patterns have the property of being distinctive in both their spatial appearance and in their frequency distribution. Thus they are amenable to description via texture analysis using Gabor filters. Of course, not all geospatial objects contain texture motifs. We restrict our treatment to those that do. Examples of objects with texture motifs are golf courses, harbors, trailer parks, vineyards, and airports. The concept of texture motifs leads to texture-based computational models for many objects, which can be applied to object detection. This approach offers a powerful alternative to shape-based and edge-based models, which are prohibitively expensive to compute, due to the level of complexity and detail often found in geospatial objects.

This chapter mainly deals with the problem of learning models for objects given a set of examples.[1] This is posed as a problem of learning a representation for the texture motifs of the objects from low-level texture features extracted from examples. Building on the work done in [4], this chapter presents a probabilistic framework for this learning problem. The quality of the models are evaluated on the basis of their application to object detection.

4.1 Texture Motif Representation

How do we represent the visual appearance of a texture motif, say, the arrangement of boats and water in a harbor?

There are different aspects that constitute this appearance. Firstly, there are

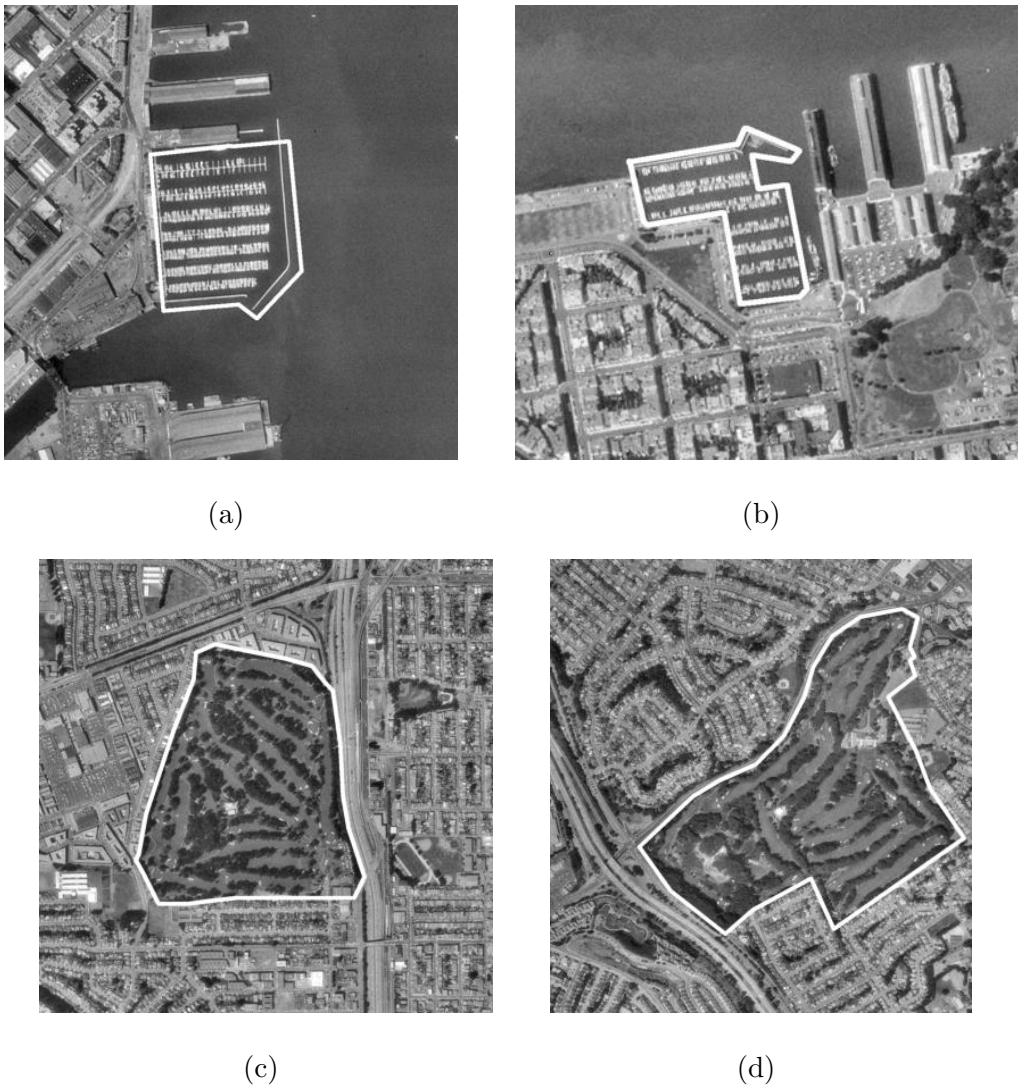
[1] An object "example" here refers to an image containing an instance of the object, along with a binary mask that isolates the object region from the background (see Fig. 4.4).

the local intensity variations that form textural elements such as flat areas, bars, edges, and so on. These can be interpreted as the low-level building blocks of the motif. For example, they may correspond to water, boats, and edges between them. It has been shown (Sec. 3.1.2) that these local intensity variations can be effectively captured and described by low-level texture features based on Gabor filters at multiple scales and orientations. Assuming that the texture features generated by different elements populate different volumes of the texture feature space, it is possible to statistically learn the elements of a pattern. In this work, a semi-supervised statistical approach is adopted for this task. This forms the first layer of the overall representation of the texture motif.

The second layer of the representation is the spatial distribution of low-level texture elements in the texture motif, since this influences its distinct visual appearance. A Gaussian mixture model (GMM) for this is learned from examples using features derived from histograms of texture elements in spatial neighborhoods. Confidence measures generated using this model are then used for detecting object presence.

4.2 Learning the Texture Elements of a Motif

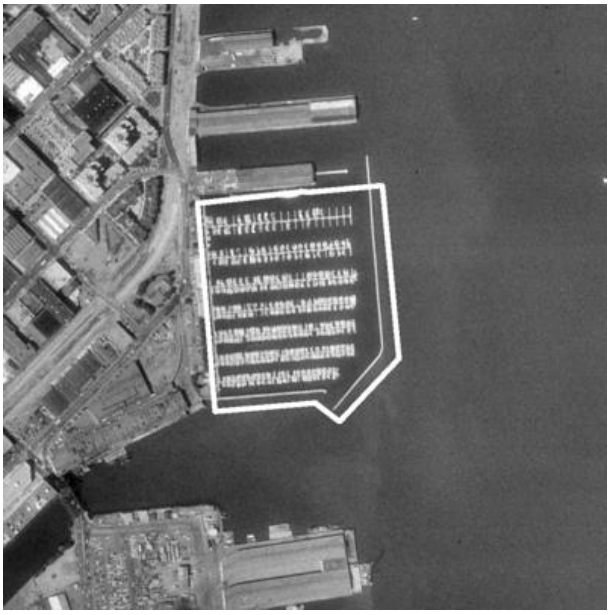
Suppose we are given M examples of an object that contains one or more texture motifs. Let us further assume that all the motifs are formed by a spatial combination of N_t texture elements. Then the N_t elements are learned from low-level texture features extracted from the examples, in order to arrive at the first



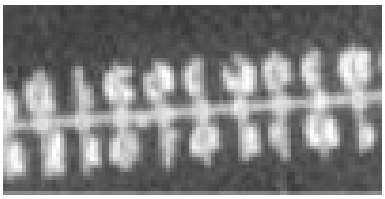
(a) (b)

(c) (d)

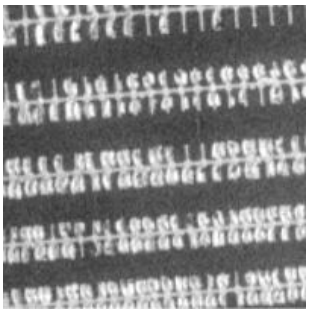
Figure 4.1: Examples of geospatial objects: *harbors* in the top row, and *golf courses* in the bottom row. The white border shows the extent of the object in each image.



(a)



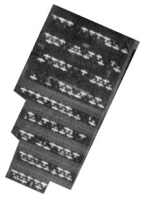
(b)



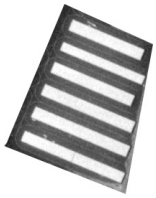
(c)

Figure 4.2: (a) A harbor instance with a white border specifying the object region; The texture motif of harbors includes the patterns formed by (b) boats moored side by side, and (c) periodic rows of boats separated by water (bottom).

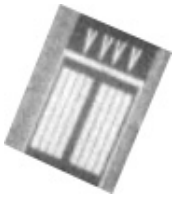
(a)



(b)



(c)



(d)

Figure 4.3: (a) A airport instance with a white border specifying the object region; A few texture motifs of airports are (b) parked planes, (c) hangars, and (d) runway markings.

layer of representation. Let us uniformly sample a number[2] of texture features from each of the M object examples. If the object consists of multiple texture elements, the sampled vectors form several clusters in the texture feature space. Let each cluster be considered to represent a distinct texture element.

It can be argued that as M becomes large, the N_t largest clusters formed by the sampled vectors correspond to the texture elements in the object. The argument is justified thus. The more examples a texture element appears in, the more the evidence in favor of it being an important texture element of the object. If an element occurs in very few examples, it is less likely to be critical to the description of the object. With increasing M, clusters formed by features occurring in a majority of examples are expected to become dominant. On the other hand, clusters formed by features that occur in just a few examples become relatively smaller.

In this work, Gaussian mixture models (GMM) are applied to solve the clustering problem in a semi-supervised approach. Mixtures of Gaussians have been used to model image feature distributions for a variety of research objectives. In [46], texture-based image segmentation is performed by clustering texture feature vectors using mixtures of Gaussians. In the Blobworld system [11], mixtures of Gaussians are used to derive image descriptors for content-based retrieval. The Expectation-Maximization (EM) algorithm is used to discover the feature vector groupings that correspond to the visual blobs in an image. There are several factors that motivated us to use a GMM to cluster texture features instead of the simpler K-means algorithm. Firstly, a GMM accounts for the density of each

[2]This number is usually proportional to the size of the example.

cluster. This is important because the feature vectors from different textures are observed to have different densities of distribution in the feature space. Secondly, GMM has a parametric representation that allows easy model comparison. Finally, the EM framework can be extended to elegantly handle rotations of textural patterns (Sec. 4.3.1).

We model texture features that occur in an object as a GMM with N_t Gaussian components. Each component in the GMM corresponds to one texture element. In other words, the features corresponding to each texture element is assumed to follow a Gaussian distribution. It is also possible to train the GMM with $N_t' > N_t$ components and choose the N_t most probable ones as corresponding to the texture elements. In this work, the choice of N_t is made by the user based on experimental evidence. As will be described in Sec. 4.5.2, the modeling parameters including N_t are chosen to obtain the "best" object detection performance in terms of *precision* and *recall*. The best parameters are chosen from a set of candidate parameters determined by the user based on visual inspection of the object examples.

4.2.1 The GMM Framework

The texture in the neighborhood of a pixel is represented by an SK-dimensional feature vector obtained by convolving the image with a Gabor filter bank at S scales and K orientations. The filter bank (see Sec. 3.1.2) is a set of Gabor-wavelet filters denoted by $\{g_{s,k}(\mathbf{x}); s \in [0, S-1], k \in [0, K-1]\}$. Let $\mathbf{c}(\mathbf{x})$ denote the feature vector extracted from the neighborhood of pixel[3] $\mathbf{x} \in R_o$, where R_o is

[3]Note that we represent a pixel vectorially as $\mathbf{x} = [x\ y]^T$.

the object region. Following (3.6), the feature vector is given by

$$\mathbf{c}(\mathbf{x}) = [F_{0,0}(\mathbf{x}) \;\; F_{0,1}(\mathbf{x}) \;\; \ldots \;\; F_{S-1,K-2}(\mathbf{x}) \;\; F_{S-1,K-1}(\mathbf{x})]^T, \tag{4.1}$$

where $F_{s,k}(\mathbf{x})$ is the filter output at pixel $\mathbf{x}$, obtained by convolving the image $I(\mathbf{x})$ with the filter $g_{s,k}(\mathbf{x})$. In other words, $F_{s,k}(\mathbf{x}) = |g_{s,k}(\mathbf{x}) * I(\mathbf{x})|$.

Assuming that there are N_t texture elements in an object, the probability density function of $\mathbf{c}(\mathbf{x})$ (or simply $\mathbf{c}$) can thus be expressed as a mixture distribution,

$$p_t(\mathbf{c}) = \sum_{j=1}^{N_t} P_t(j) p_t(\mathbf{c}|j), \tag{4.2}$$

where $p_t(\mathbf{c}|j)$ is the conditional pdf of the feature $\mathbf{c}$ given that it is generated by the j^{th} texture element and $P_t(j)$ is the prior probability of the j^{th} element. The subscript t is used to clarify that we are learning the *texture elements*. This subscript is applied to all parameters and probabilities in the first layer of texture motif representation.

The conditional pdf $p_t(\mathbf{c}|j)$ is Gaussian and is given by

$$p_t(\mathbf{c}|j) = \frac{\exp\left[-\frac{1}{2}(\mathbf{c} - \boldsymbol{\mu}_{tj})^T \Sigma_{tj}^{-1} (\mathbf{c} - \boldsymbol{\mu}_{tj})\right]}{(2\pi)^{d/2} |\Sigma_{tj}|^{1/2}}, \tag{4.3}$$

where d is the dimensionality of $\mathbf{c}$. The number of elements N_t along with the distribution means and covariance matrices are the parameters that specify the object model Θ_t. In other words,

$$\Theta_t = \{(P_t(j), \boldsymbol{\mu}_{tj}, \Sigma_{tj}); j = 1 \ldots N_t\}. \tag{4.4}$$

The EM algorithm [21] is used to estimate the GMM parameters from training data, which are obtained from object examples as described in the following section.

Note that the texture element model Θ_t is learnt separately for each object and not over all objects. A high number of texture elements (N_t) is needed to describe texture motifs across all objects. As will be seen later, N_t determines the dimensionality of the second layer of texture motif representation. A small N_t is desirable with regard to the complexity and reliability of the next learning stage. Therefore, we learn texture elements in an object-specific manner.

4.2.2 Feature Sampling for GMM Learning

The training set for an object consists of a set of examples or instances, such as those shown in Fig. 4.4 for harbors. Each instance is provided as an image and an associated mask delineating the object region. The texture samples for training the GMM are drawn from pixels strictly inside the object regions, as depicted in Fig. 4.5(a). Of course, texture is a neighborhood property, not a pixel property. The texture features are generated by convolving the image with square Gabor filter kernels. Let s_f be the *kernel size*, i.e. the length of its side in pixels. We need to make sure that the sampled texture features are not "corrupted" by intensity variations outside the object region. This implies that if the kernel is centered at an object pixel, no part of it should project outside the object (Fig. 4.5(b)). This results in the exclusion of a band of pixels at the borders of the object region. The width of this band is $s_f/\sqrt{2}$ pixels in the worst case when the border is parallel to a diagonal of the kernel, and $s_f/2$ pixels in the best case when it is parallel to a side of the kernel. The object region minus this band is termed the *valid sampling region*. In practice, the valid sampling region is obtained by morphological erosion of the binary mask image with a square structuring element of side s_f pixels.

Let the training set for an object be denoted by

$$\mathcal{O} = \{R_{v,1}, R_{v,2}, \ldots, R_{v,N_o}\}, \tag{4.5}$$

where $R_{v,i}$ is the valid sampling region of the i^{th} example and N_o is the number of training examples of the object. From each example i, $n_i = \beta|R_{v,i}|$ features are sampled uniformly, where $|R_{v,i}|$ is the number of pixels in $R_{v,i}$ and β is chosen according to the acceptable complexity of the GMM learning task.[4] The training data for learning the GMM is obtained from the union of the sampled features from each example in the training set. Thus the GMM is learned from a total of $\sum_i n_i$ sampled features.

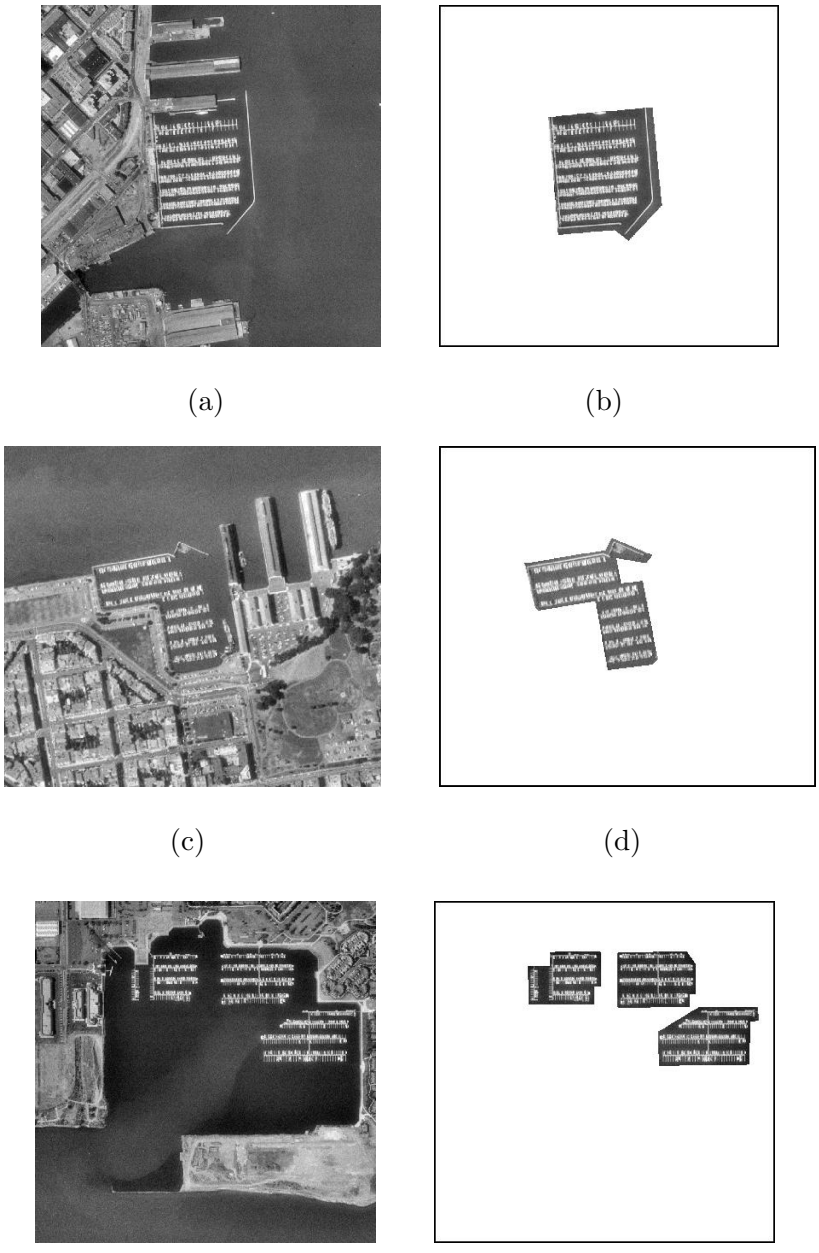
Having obtained this training data, the EM algorithm [21] is used to estimate the parameters of the GMM, which are given by (4.4). A K-means clustering process is applied to bootstrap the EM algorithm. After the learning process, each Gaussian component in the mixture represents one texture element in the object. The prior probability of each component gives information about the relative contribution of that texture element in forming the object.

4.2.3 Texture Element Labeling

After a GMM has been learned for an object, a maximum a posteriori (MAP) classifier is used to label any pixel $\mathbf{x}$ to its generating texture element $i^*(\mathbf{x})$, as follows.

$$i^*(\mathbf{x}) = \arg\max_{1 \le i \le N_t} P_t\left(i|\mathbf{c}(\mathbf{x})\right), \tag{4.6}$$

[4]The acceptable complexity in turn depends on the available resources, such as CPU speed and memory.

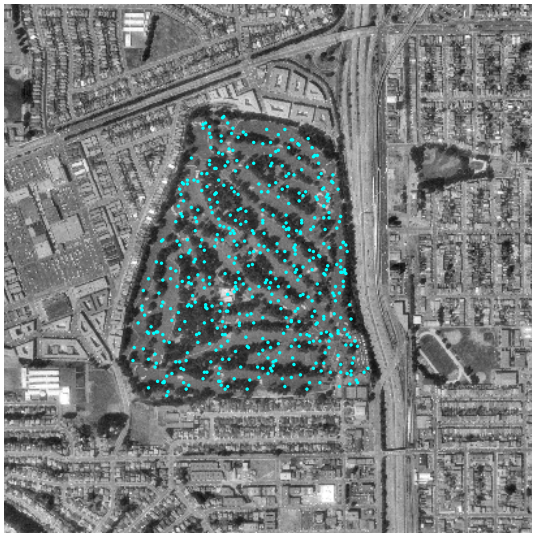


(a) (b)

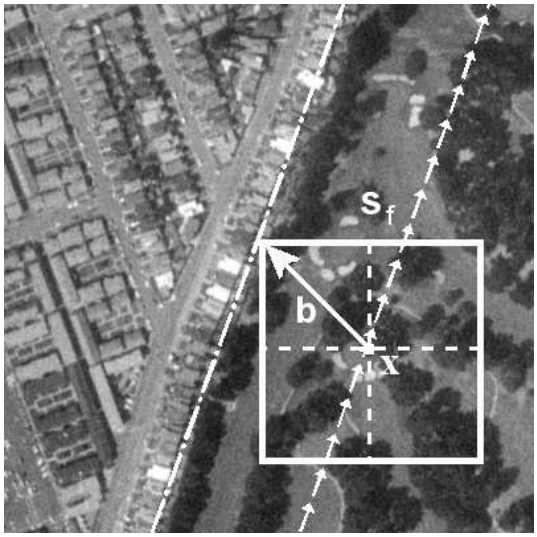
(c) (d)

(e) (f)

Figure 4.4: A small training set for the "harbor" object. The training images are on the left and the associated masks on the right.



(a)



(b)

Figure 4.5: **Sampling methodology:** (a) Uniform random sampling of pixels from the golf course object. The pixels around which the texture features are sampled are marked as dots. (b) Illustration of the valid sampling region. To prevent the square Gabor filter kernel from exceeding the object border (dot-dash line), its center (pixel $\mathbf{x}$) should stay within the short-arrows line.

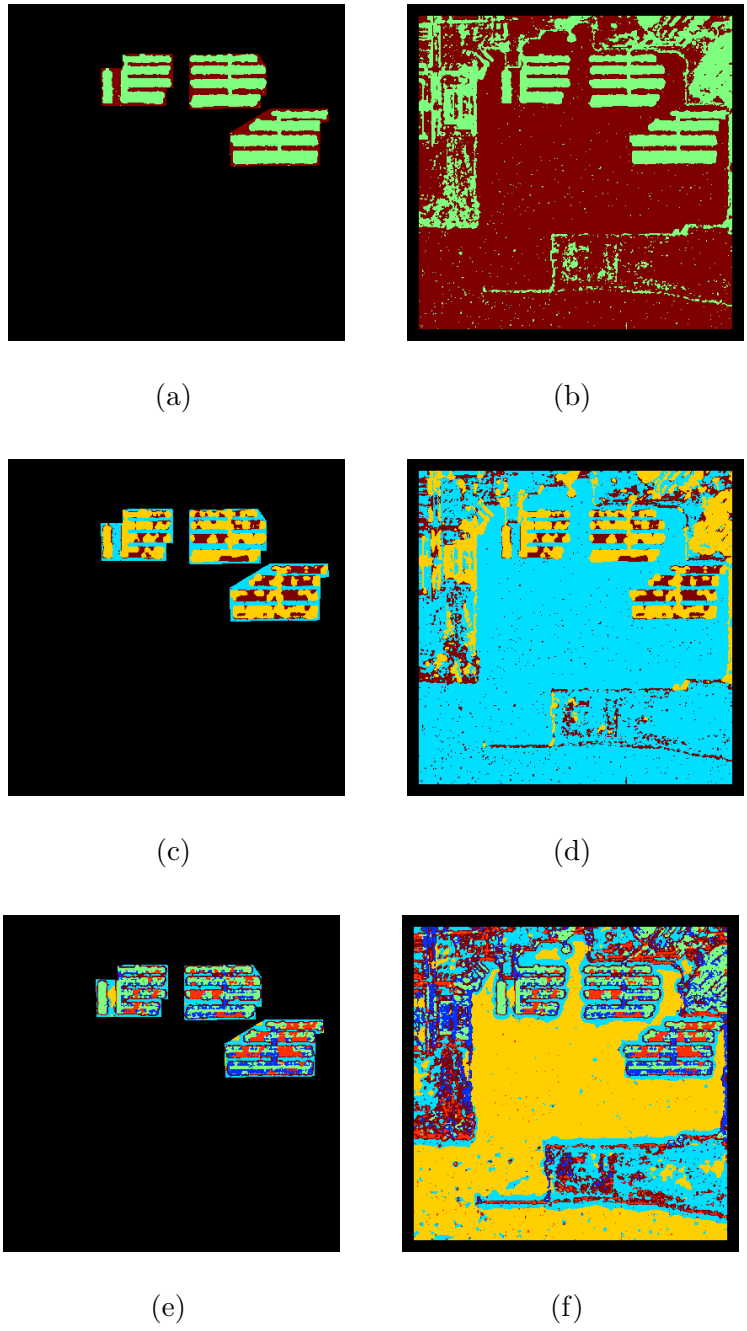
where $P_t(i|\mathbf{c}(\mathbf{x}))$ is the probability that the feature vector $\mathbf{c}(\mathbf{x})$ came from the i^{th} Gaussian component of the GMM. The posterior probabilities $P_t(i|\mathbf{c}(\mathbf{x}))$ are obtained using Bayes' rule as follows,

$$P_t(i|\mathbf{c}) = \frac{p_t(\mathbf{c}|i)P_t(i)}{\sum_i p_t(\mathbf{c}|i)P_t(i)}. \tag{4.7}$$

Fig. 4.6 shows the texture element labels assigned to one of the training images, using GMMs learned from the training set in Fig. 4.4. Different labelings are shown for the same image, obtained by learning GMMs with different N_t (number of components). The function $p_t(\mathbf{c}(\mathbf{x}))$ gives the density in the feature space at the point corresponding to $\mathbf{c}(\mathbf{x})$. Fig. 4.7 displays the scaled log density images corresponding to the same training image, for varying N_t. Brighter pixels indicate higher values of $\log(p_t(\mathbf{c}(\mathbf{x})))$. Note that the high values are not restricted to the object region. Thus $p_t(\mathbf{c}(\mathbf{x}))$ does not directly convey the confidence of a pixel belonging to the object. A reason for this, in addition to the curse of dimensionality, might be that the texture elements of the harbor motif occur in other regions as well. This is clear by observing the left and right columns in Fig. 4.6. Therefore, it is the spatial arrangement of these elements that distinguish harbors from other objects.

4.3 A Note on Rotation Invariance

A major obstacle in learning the texture elements with the above GMM formulation is that the texture features $\mathbf{c}(\mathbf{x})$ (given by (4.1)) are derived from orientation-selective Gabor filters and are therefore sensitive to the orientation of the texture element (or the motif/object). Texture elements recurring in several

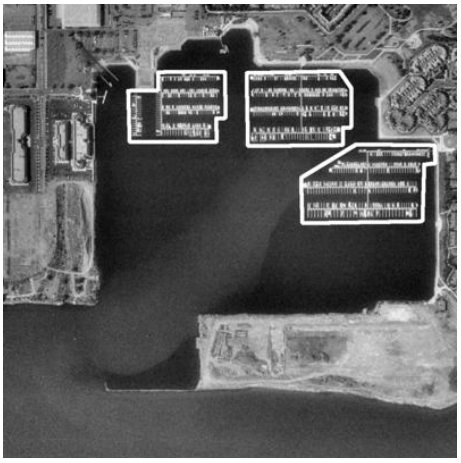


(a) (b)

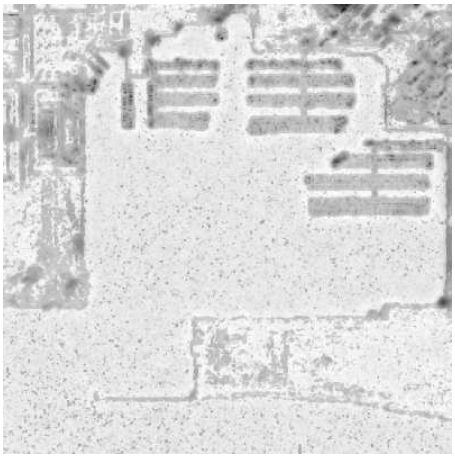
(c) (d)

(e) (f)

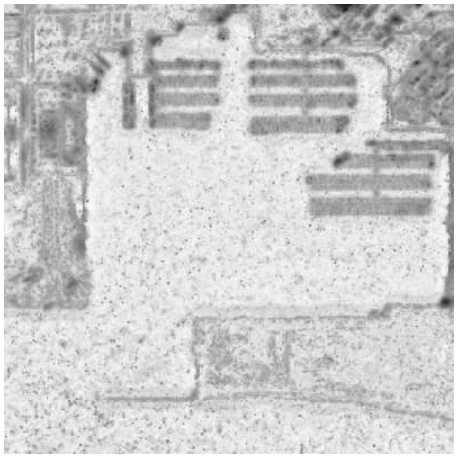
Figure 4.6: The texture element labelings for the third training image in Fig. 4.4, with $N_t = 2$ (top row), $N_t = 3$ (middle row), and $N_t = 6$ (bottom row). Each color corresponds to a label. The left column shows the labels inside the object and the right column shows the overall labelings.



(a)



(b)



(c)

Figure 4.7: Scaled log density images ($\log(p_t(\mathbf{c}(\mathbf{x})))$) corresponding to (a) a training image from Fig. 4.4, with (b) $N_t = 3$, and (c) $N_t = 6$. Brighter pixels indicate higher values of $\log(p_t(\mathbf{c}(\mathbf{x})))$. Note that the high values are not restricted to the object region.

examples can be learned consistently only when the objects in the examples have similar orientations. This is the case with the training set in Fig. 4.4, but often in practice the training examples have arbitrary orientations.

Suppose the texture features are derived from Gabor filters oriented at 30° intervals, i.e. 0°, 30°, 60°, and so on. Then a 30° rotation of the texture is equivalent to a circular shifting of the feature vector components at each scale. Hence, the features sampled from a textural pattern of varying orientation form multiple clusters in the feature space.

In order to handle objects with varying orientations, the number of Gaussian components in the GMM has to be adjusted to take into account the additional clusters formed by variation in orientation. Then the following question arises. Which clusters are associated with the same texture element, i.e. caused by a rotation of the same element? This is a difficult question to answer. Furthermore, to model motif appearance at different orientations, it is necessary to augment the training set by considering all orientations of the training instances. This increases the complexity of the learning process.

Alternatively, Newsam [56] has proposed a variation to the EM algorithm that takes the orientation of a texture into account while training a GMM. By treating the (discretized) orientation of a pattern as a missing variable in the EM framework, the equivalence between rotated patterns is learned automatically. In the resulting GMM, each Gaussian component corresponds to a cluster of "orientation-normalized" texture elements. This variation to the EM algorithm is described below.

4.3.1 The Orientation-Normalized GMM [56]

The conditional probability of a feature vector $\mathbf{c}$, given that it is generated from component j and its orientation index is k, is written as

$$p_t\left(\mathbf{c}|j,k\right) = \frac{\exp\left[-\frac{1}{2}\left(\mathbf{c}_k - \boldsymbol{\mu}_{tj}\right)^T \Sigma_{tj}^{-1}\left(\mathbf{c}_k - \boldsymbol{\mu}_{tj}\right)\right]}{\left(2\pi\right)^{d/2}\left|\Sigma_{tj}\right|^{1/2}}. \tag{4.8}$$

The term $\mathbf{c}_k$ is the vector $\mathbf{c}$ circularly shifted by k orientations where $k \in \{1,..,K\}$. Note that the orientation k is with respect to the normalized orientation of the mixture component. The pdf of the feature distribution in an object class is modeled as a N_t-component GMM,

$$p_t\left(\mathbf{c}\right) = \frac{1}{K}\sum_{k=1}^{K}\sum_{j=1}^{N_t} p_t\left(\mathbf{c}|j,k\right) P_t(j), \tag{4.9}$$

where we have assumed that the orientation k is independent of j and equiprobable (in the absence of a priori information).

Each component represents a single texture element in a manner oblivious to its orientation. This model is completely specified by the parameters $\Theta_t = \left\{\left(P_t\left(j\right), \boldsymbol{\mu}_{tj}, \Sigma_{tj}\right); j = 1 \ldots N_t\right\}$. A modified version of the EM algorithm is used to estimate the parameters of the GMM. Rotation is taken into account by modifying the EM algorithm to include the orientation k of the feature vector as additional missing data.

The procedure in (4.6) for labeling each pixel $\mathbf{x}$ to its texture element $i^*(\mathbf{x})$ has to modified as well. It becomes

$$i^*(\mathbf{x}) = \underset{1\le i\le N_t}{\arg\max}\left[\max_k P_t\left(i|\mathbf{c}_k(\mathbf{x})\right)\right], \tag{4.10}$$

and (4.7) becomes

$$P_t(i|\mathbf{c}_k) = \frac{p_t(\mathbf{c}|i,k)P_t(i)}{\sum_i p_t(\mathbf{c}|i,k)P_t(i)}, \tag{4.11}$$

assuming that the orientations k are equiprobable.

Note: From this point forward, the orientation normalized GMM shall be used to learn the texture elements in the first layer of texture motif representation. This modification does not apply to GMMs for features other than those directly obtained using orientation-selective Gabor filters.

4.4 Spatial Distribution of Texture Elements in a Motif

It can be observed from Fig. 4.6 that the spatial configuration of the labels inside the "boats and water" texture motif of the harbor region is quite different from that outside. Then, the task of the second layer is to describe this spatial configuration of labels, and model its variation within the motif. A simple method of describing the spatial distribution of labels is by the use of a *spatial histogram*, as shown in Fig. 4.8. The descriptor at a pixel $\mathbf{x}$, denoted as $\mathbf{h}(\mathbf{x})$, is the vector of normalized frequencies of texture element labels in a square window centered at $\mathbf{x}$. In other words,

$$\mathbf{h}(\mathbf{x}) = [h_1(\mathbf{x})\ h_2(\mathbf{x})\ \ldots\ h_{N_t}(\mathbf{x})]^T, \tag{4.12}$$

where $h_l(\mathbf{x})$ is the normalized frequency of label l in the window. Obviously, the dimensionality of the above descriptor is N_t, the total number of texture element labels.

The texture element label $i^*(\mathbf{x})$ at a pixel $\mathbf{x}$ is given by (4.10). If we temporarily write $i^*(\mathbf{x})$ as $i^*(x, y)$ (since $\mathbf{x} = [x\ y]^T$), then $i^*(x + x_o, y + y_o)$ is the

label at an offset of (x_o, y_o) from $\mathbf{x}$. Let $I_l(z)$ be an indicator function that is 1 if $z = l$ and 0 otherwise. Then $h_l(\mathbf{x})$ can be computed as

$$h_l(\mathbf{x}) = h_l(x, y) = \frac{1}{s_h^2} \sum_{x_o=-\frac{s_h-1}{2}}^{\frac{s_h-1}{2}} \sum_{y_o=-\frac{s_h-1}{2}}^{\frac{s_h-1}{2}} I_l(i^*(x + x_o, y + y_o)), \tag{4.13}$$

where s_h (usually an odd number) is the length of the side of the square window around pixel $\mathbf{x}$ (Fig. 4.8). The scale of the description depends on the value of s_h. Note that the spatial histogram feature $\mathbf{h}(\mathbf{x})$ is isotropic, i.e. it does not depend on the orientation of the texture motif.

4.4.1 Learning the Second Layer

Once again, the GMM is employed to model the variation of $\mathbf{h}(\mathbf{x})$ in the object region. Let the variation in the spatial configuration be modeled by a GMM with N_s components as follows,

$$p_s(\mathbf{h}) = \sum_{j=1}^{N_s} P_s(j) p_s(\mathbf{h}|j), \tag{4.14}$$

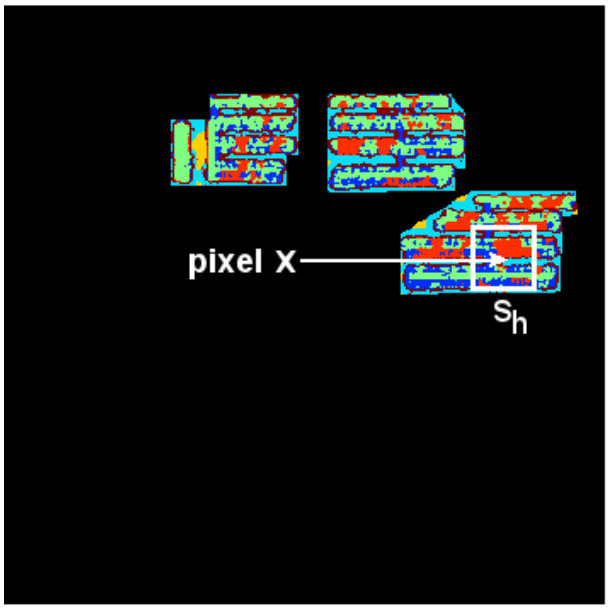
where the conditional pdfs $p_s(\mathbf{h}|j)$ are Gaussian, given by

$$p_s(\mathbf{h}|j) = \frac{\exp\left[-\frac{1}{2}(\mathbf{h} - \boldsymbol{\mu}_{sj})^T \Sigma_{sj}^{-1} (\mathbf{h} - \boldsymbol{\mu}_{sj})\right]}{(2\pi)^{N_t/2} |\Sigma_{sj}|^{1/2}}. \tag{4.15}$$

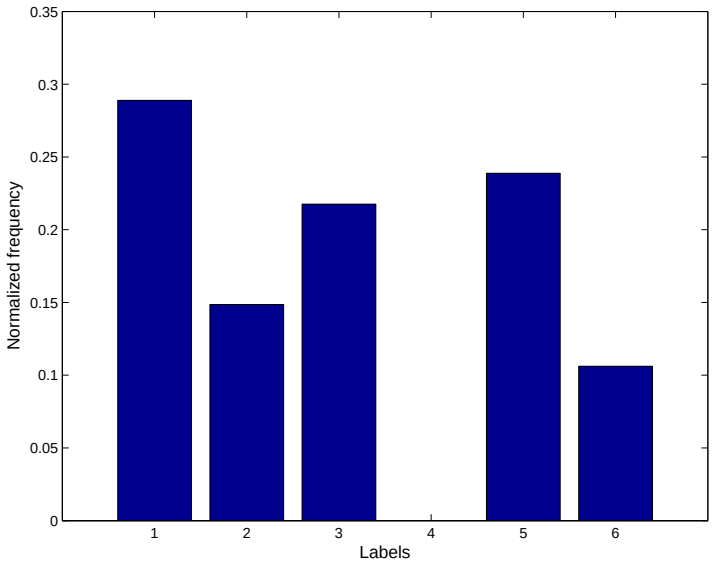
The model for the second layer of representation is then specified by

$$\Theta_s = \{(P_s(j), \boldsymbol{\mu}_{sj}, \Sigma_{sj}); j = 1 \ldots N_s\}. \tag{4.16}$$

The subscript s here is used to clarify that we are learning the *spatial distribution* of the texture elements in the motif.



(a)

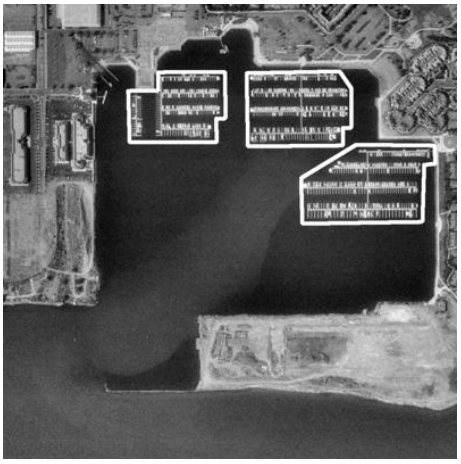


(b)

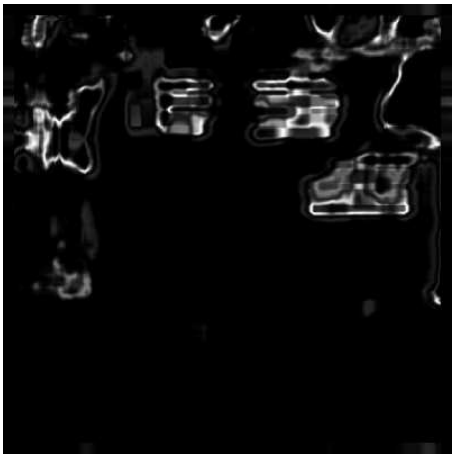
Figure 4.8: (a) The spatial histogram of texture element labels is built by taking a square window of size s_h around pixel $\mathbf{x}$; (b) The normalized frequencies of the $N_t = 6$ labels inside the window, which form the 6-dimensional vector $\mathbf{h}(\mathbf{x})$.

The training data is obtained by sampling spatial histograms $\mathbf{h}(\mathbf{x})$ around several pixels $\mathbf{x}$ inside the object region. The procedure for sampling and creating the training data for the GMM is similar to that in Sec. 4.2.2. The valid sampling region in this case is obtained by morphological erosion of the binary mask image using a square structuring element with a side length of $\max(s_f, s_h)$ pixels, i.e. the larger between the filter kernel size and the spatial neighborhood size. This ensures that neither the texture features nor the histograms at the sampled pixels are influenced by non-object pixels. The sampling parameter β is again chosen appropriately, and need not be the same as the value chosen for the first layer. After creating the training data, the GMM is learned via the EM algorithm. It should be clear that, since the features are non-directional, the conventional GMM formulation is used and not the orientation-normalized version.

Having learned Θ_s, the density function $p_s(\mathbf{h}(\mathbf{x}))$ can be interpreted as the confidence of finding at a pixel $\mathbf{x}$, the spatial configuration corresponding to the learned texture motif. Fig. 4.9 shows the scaled density images corresponding to a training image in Fig. 4.4, for different values of N_s. Note that the object region is found to have relatively high p_s values. Therefore, p_s is a good measure for the confidence of a pixel belonging to a texture motif (or the object containing the motif). The importance of the spatial arrangement of texture elements for describing a motif is evident from this.



(a)



(b)

(c)

Figure 4.9: Scaled density images ($p_s(\mathbf{h}(\mathbf{x}))$) corresponding to (a) a training image from Fig. 4.4, with $N_s = 3$, and (b) $N_t = 3$, and (c) $N_t = 6$. Note that the object region has relatively high values.

4.5 Experimental Results

The primary dataset chosen for our study consists of aerial images drawn from the Digital Orthophoto Quarter-Quadrangle (DOQQ) coverage of California, which is available through the Alexandria Digital Library (ADL). The ADL Gazetteer [32] is a resource that provides georeferencing information for several objects (synonymous with *feature types* in [32]). Several instances of objects such as harbors, golf courses, and airports, can be located through the Gazetteer. The corresponding aerial images are then extracted from the ADL DOQQ coverage. Each object instance used for training is provided in two pieces (see Fig. 4.4): a) a rectangular image region containing the object, and b) a manually created binary mask defining the object region.

4.5.1 Testing Methodology

Suppose, for an object of study, we have a training set and a test set of example images, with their corresponding masks. From the training set, the GMMs Θ_t and Θ_s are learned as described in Sections 4.2–4.4. The specifications of the Gabor filter bank used for extracting texture features, $\mathbf{c}(\mathbf{x})$ in (4.1), are $S = 5$, $K = 6$, $U_l = 0.05$ and $U_h = 0.4$ (see Sec. 3.1.1). In the testing stage, the learned models are applied to each instance of the test set in three steps as follows.

1. Application of Θ_t to obtain the texture element labels $i^*(\mathbf{x})$ as described in Sec. 4.2.3. Note that in practice, the orientation-normalized GMM is used and the labels are obtained using (4.10).

2. Computation of the spatial histogram features $\mathbf{h}(\mathbf{x})$ from the label field

$i^*(\mathbf{x})$, as described in Sec. 4.4.

3. Application of Θ_s to obtain the confidence measure $p_s(\mathbf{h}(\mathbf{x}))$, as described in Sec. 4.4.1.

The main tool used for evaluating the performance of the proposed approach is the *precision-recall graph*. These are obtained by computing precision and recall while varying the threshold t_o on the $p_s(\mathbf{h}(\mathbf{x}))$ measure. Let $I_o(\mathbf{x})$ be an indicator function, which has a value 1 if pixel $\mathbf{x}$ lies inside the object region (defined by the user-provided masks) and 0 if it does not. Let us define another indicator function $I_{t_o}(\mathbf{x})$ thus,

$$I_{t_o}(\mathbf{x}) = \begin{cases} 1, \text{if } p_s(\mathbf{h}(\mathbf{x})) > t_o \\ 0, \text{else.} \end{cases} \tag{4.17}$$

Now, for a given t_o, precision $\mathcal{P}(t_o)$ and recall $\mathcal{R}(t_o)$, are defined as,

$$\mathcal{P}(t_o) = \frac{\sum_i I_o(\mathbf{x_i}) I_{t_o}(\mathbf{x_i})}{\sum_i I_{t_o}(\mathbf{x_i})}, \text{ and} \tag{4.18}$$

$$\mathcal{R}(t_o) = \frac{\sum_i I_o(\mathbf{x_i}) I_{t_o}(\mathbf{x_i})}{\sum_i I_o(\mathbf{x_i})}, \tag{4.19}$$

where $\mathbf{x}_i$ are indexed over all the pixels in the test images, both inside and outside the object region. Therefore, *precision* tells us how many pixels are correctly identified as belonging to the object. The *recall* tells us how many pixels belonging to the object are correctly identified as such. The precision-recall graph plots $\mathcal{P}(t_o)$ against $\mathcal{R}(t_o)$ while varying t_o. It displays the tradeoff between precision and recall at different thresholds.

4.5.2 Results on Geospatial Objects

Two geospatial objects are selected for comprehensive testing of the proposed modeling approach. These are *golf courses* and *harbors*. The dataset for golf courses contains nine instances (Fig. 4.10), and that for harbors contains six (Fig. 4.14). Since the datasets are small, the experimental methodology employs cross-validation techniques. Cross-validation implies that each instance is used in turn for testing, while being excluded from the training set. This technique enables more comprehensive testing on all the instances, which is not possible by rigidly partitioning the dataset into one training set and one test set.

The cross-validation strategy is applied as follows. The nine instances in the golf course dataset are randomly partitioned into three sets of three instances. Each set is used in turn as the test set, while the training set comprises the union of the remaining sets. Similarly, cross-validation for harbors is done by dividing the dataset into two sets of three instances. In the end, we shall have tested and obtained $p_s(\mathbf{h}(\mathbf{x}))$ for all instances in the dataset. The precision-recall graph is then plotted by applying (4.18) and (4.19) to the aggregated test results, for different t_o.

Fig. 4.11(a) shows the precision-recall graph for the golf course dataset, for different modeling parameters (N_t, N_s, and s_h). A plot that lies entirely above another is better, since it implies a higher precision and recall for all thresholds. Therefore, the aim is to attain "higher" plots by choosing modeling parameters wisely. In practice, the plots may intersect one another. When this happens, the model is chosen according to the relative merits of the intersecting plots, e.g. the one that gives higher precision at the required recall rate. It can be observed

from Fig. 4.11(a) that the best overall model (among the ones considered) has parameters $N_t = 6$, $N_s = 3$, and $s_h = 161$.

For object detection, a proper threshold t_o has to be chosen that results in a high confidence of detecting the object (high recall) and a low false-alarm rate (high precision). All pixels $\mathbf{x}$ that have $p_s(\mathbf{h}(\mathbf{x})) > t_o$ shall then be denoted as object pixels. Often, the choice of t_o is based on a trade-off between precision and recall. This process is simplified by means of the *F-measure* [68] which combines precision and recall into one measure that depends on t_o. The F-measure is the harmonic mean of precision and recall, and is defined as,

$$\mathcal{F}_\alpha(t_o) = \frac{1}{\frac{1}{1+\alpha}\left(\frac{\alpha}{\mathcal{P}(t_o)} + \frac{1}{\mathcal{R}(t_o)}\right)} = \frac{(\alpha+1)\mathcal{P}(t_o)\mathcal{R}(t_o)}{\mathcal{P}(t_o) + \alpha\mathcal{R}(t_o)}, \tag{4.20}$$

where $\alpha \in [0, +\infty)$ is the relative weight placed on precision over recall. Fig. 4.11(b) plots $\mathcal{F}_\alpha(t_o)$ against t_o for different α values, choosing $N_t = 6$, $N_s = 3$, and $s_h = 161$. The threshold value t_o^* corresponding to the peak of the plot (with desired α) is chosen for object detection purposes. Figures 4.12 and 4.13 show the detected golf course regions using t_o^*, for $\alpha = 10$ and $\alpha = 2$ respectively. The correctly detected regions are the ones inside the object regions specified by the black borders. Note that with a lower α, recall is higher at the expense of precision resulting in both a higher detection rate and false-alarm rate. Note also that the many of the falsely detected regions contain patterns quite similar to the trees-and-grass texture motif of golf courses.

Fig. 4.15(a) shows the precision-recall graph for the harbor dataset, for different modeling parameters (N_t, N_s, and s_h). The best model parameters (among the ones considered) in this case are $N_t = 3$, $N_s = 1$, and $s_h = 51$. For this

model, Fig. 4.15(b) plots $\mathcal{F}_\alpha(t_o)$ against t_o for different α values. The threshold value t_o^* corresponding to the peak of the plot (with desired α) is chosen for object detection purposes. Figures 4.16 and Fig. 4.17 show the detected harbor regions using t_o^*, for $\alpha = 10$ and $\alpha = 2$ respectively. The correctly detected regions are the ones inside the object regions specified by the black borders. Note a lower α leads to a higher detection rate at the expense of increasing the false-alarm rate.

Fig. 4.18 and Fig. 4.19 demonstrates object detection in larger geospatial images containing several object instances. Fig. 4.18(a) shows a large image containing several golf courses. The object regions are delineated with white borders. Fig. 4.18(b) shows the detected golf course regions following the application of the two-layered texture motif model for golf courses. Similarly, Fig. 4.19(a) shows a large image containing several harbors. Fig. 4.19(b) shows the detected harbor regions using the texture motif model for harbors. It can be observed in both cases that most of the object regions are reliably isolated.

4.5.3 Results on Pruning Large Datasets

Geographic databases such as the Alexandria Digital Library (ADL) Gazetteer [32] index the locations of several object types, including harbors, golf courses, airports, and so on. However, instances of these objects are currently manually located and indexed. The manual labor involved in this process could be greatly reduced by applying model-driven approaches for automatically identifying probable locations of objects. This results in the elimination of many areas that, with high probability, do not contain the object. The resulting *pruned* dataset is much smaller than the original, making it much easier for manual verification of object

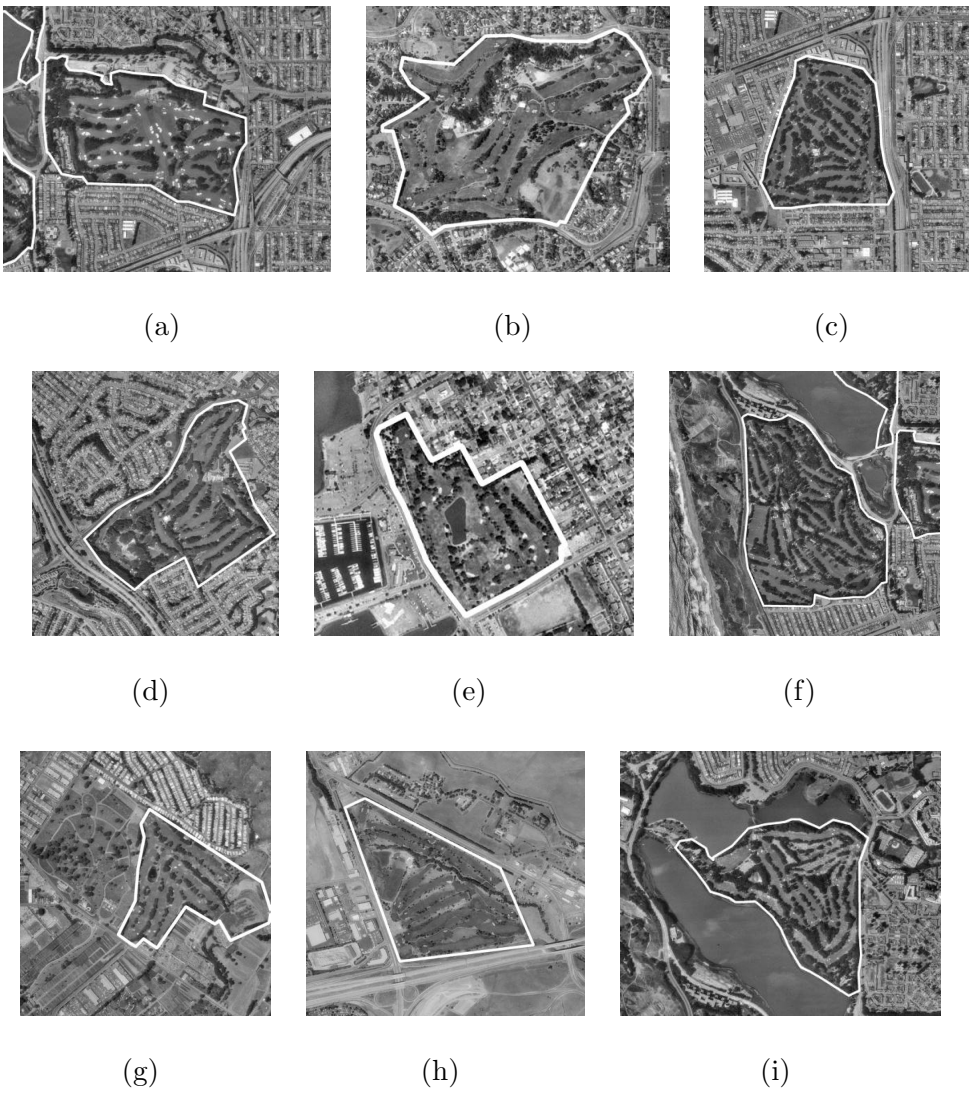
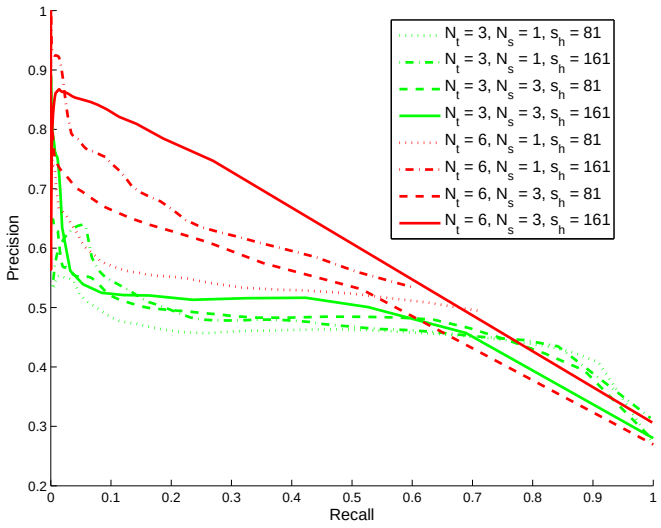
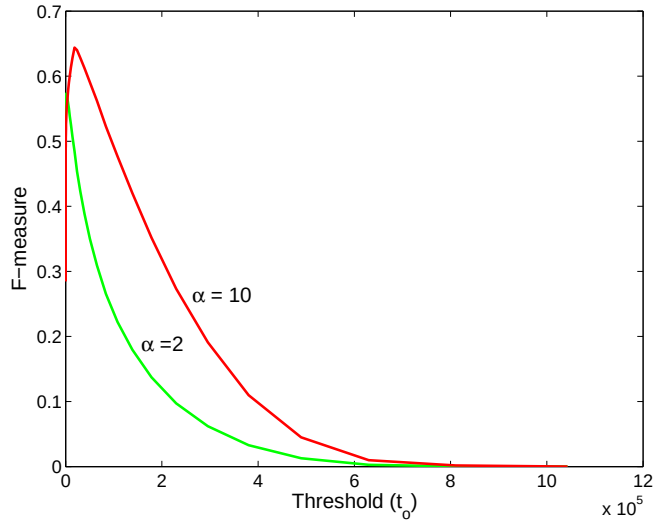


Figure 4.10: The instance dataset for the *golf course* object. The borders of the object regions (masks) are indicated in white.



(a)



(b)

Figure 4.11: (a) Precision-recall curves with different modeling parameters for the golf course dataset, and (b) F-measure $\mathcal{F}_\alpha(t_o)$ for different α computed using the golf dataset with modeling parameters $N_t = 6$, $N_s = 3$, and $s_h = 161$. For $\alpha = 10$, the peak is at threshold value $t_o^* = 1844.6$, and the corresponding $\mathcal{P}(t_o^*) = 73.44\%$ and $\mathcal{R}(t_o^*) = 28.85\%$. For $\alpha = 2$, $t_o^* = 1477.4$, $\mathcal{P}(t_o^*) = 52.7\%$, and $\mathcal{R}(t_o^*) = 69.56\%$.

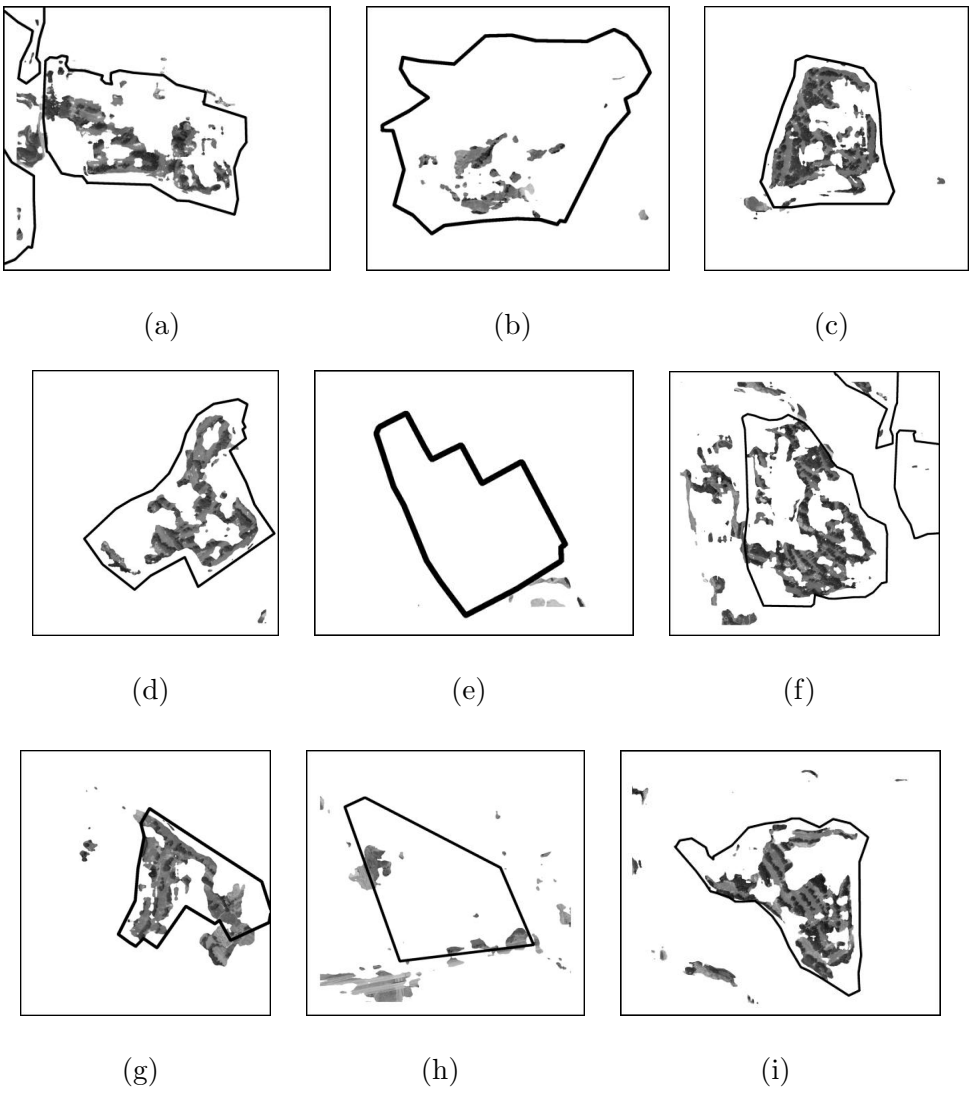


Figure 4.12: The detected golf course regions using the threshold t_o^* chosen from Fig. 4.11(b) for $\alpha = 10$ (with $N_t = 6$, $N_s = 3$, and $s_h = 161$). The borders of the desired object regions are marked in black.

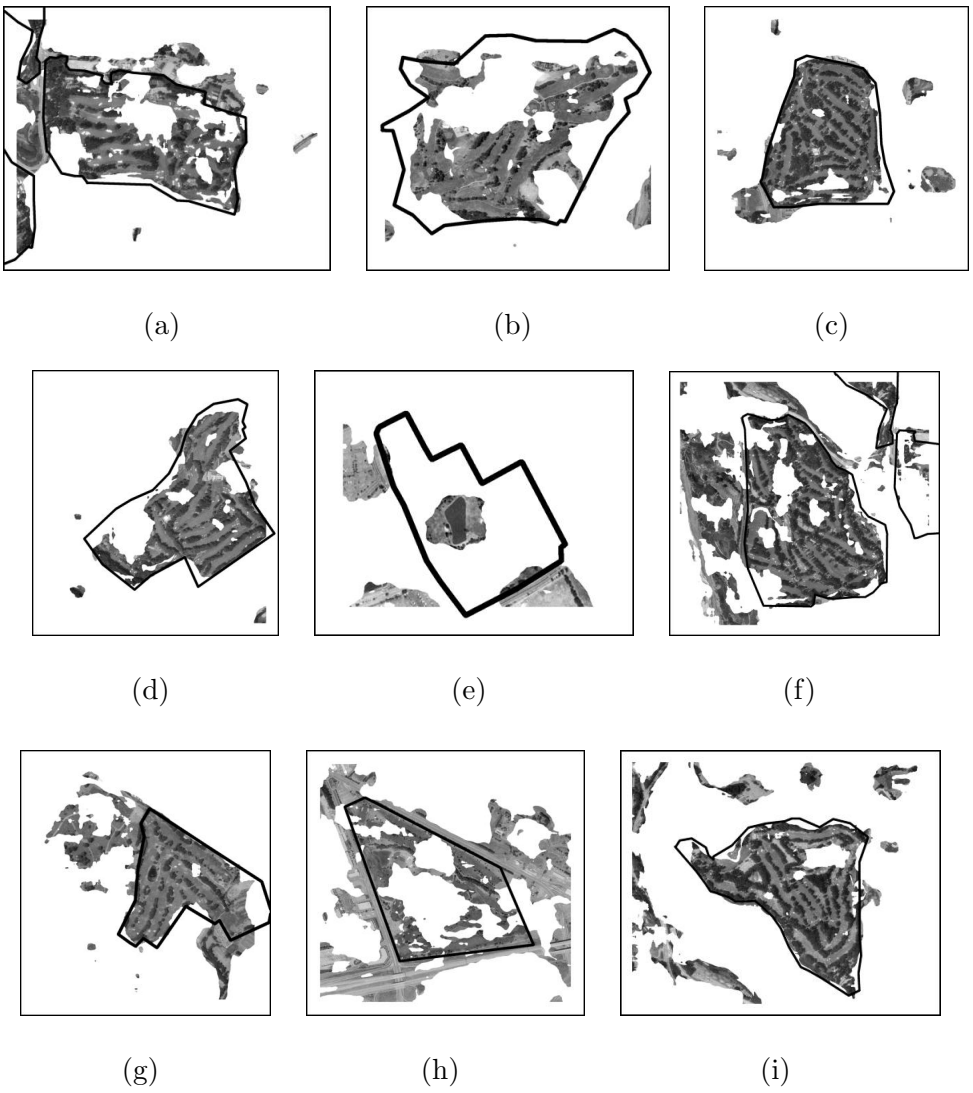
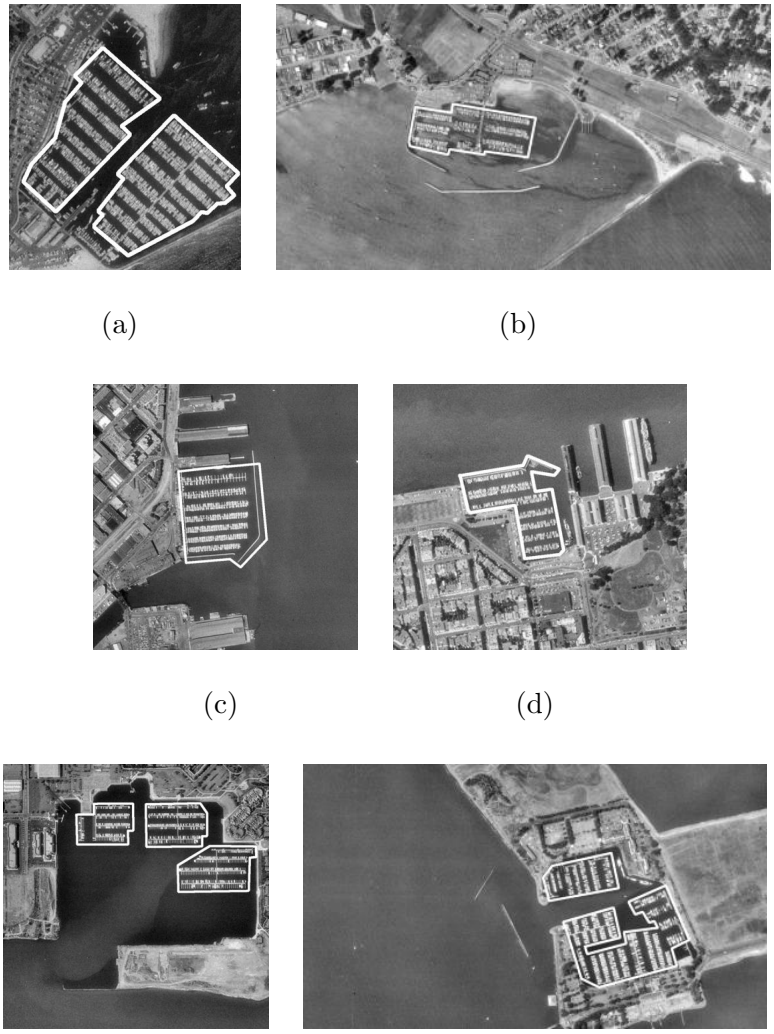


Figure 4.13: The detected golf course regions using the threshold t_o^* chosen from Fig. 4.11(b) for $\alpha = 2$ (with $N_t = 6$, $N_s = 3$, and $s_h = 161$). The borders of the desired object regions are marked in black.

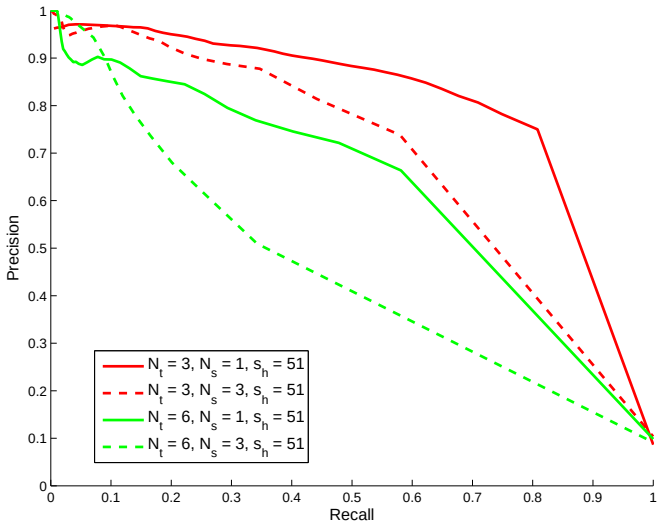


(a) (b)

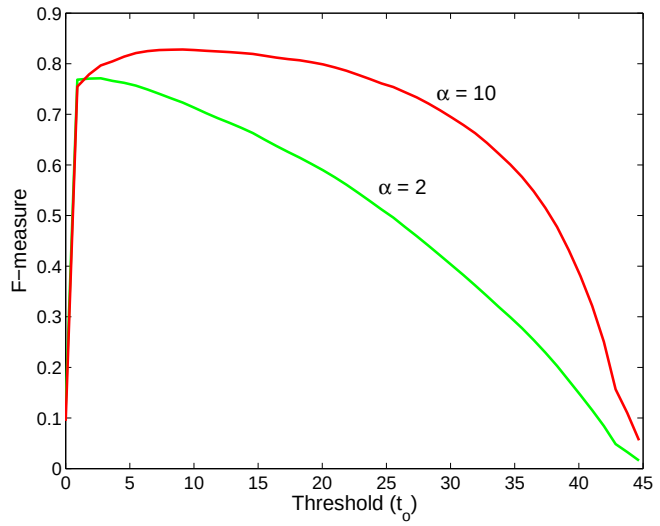
(c) (d)

(e) (f)

Figure 4.14: The instance dataset for the *harbor* object. The borders of the object regions (masks) are indicated in white.



(a)



(b)

Figure 4.15: (a) Precision-recall curves with different modeling parameters for the harbor dataset, and (b) F-measure $\mathcal{F}_\alpha(t_o)$ for different α computed using the harbor dataset with modeling parameters $N_t = 3$, $N_s = 1$, and $s_h = 51$.

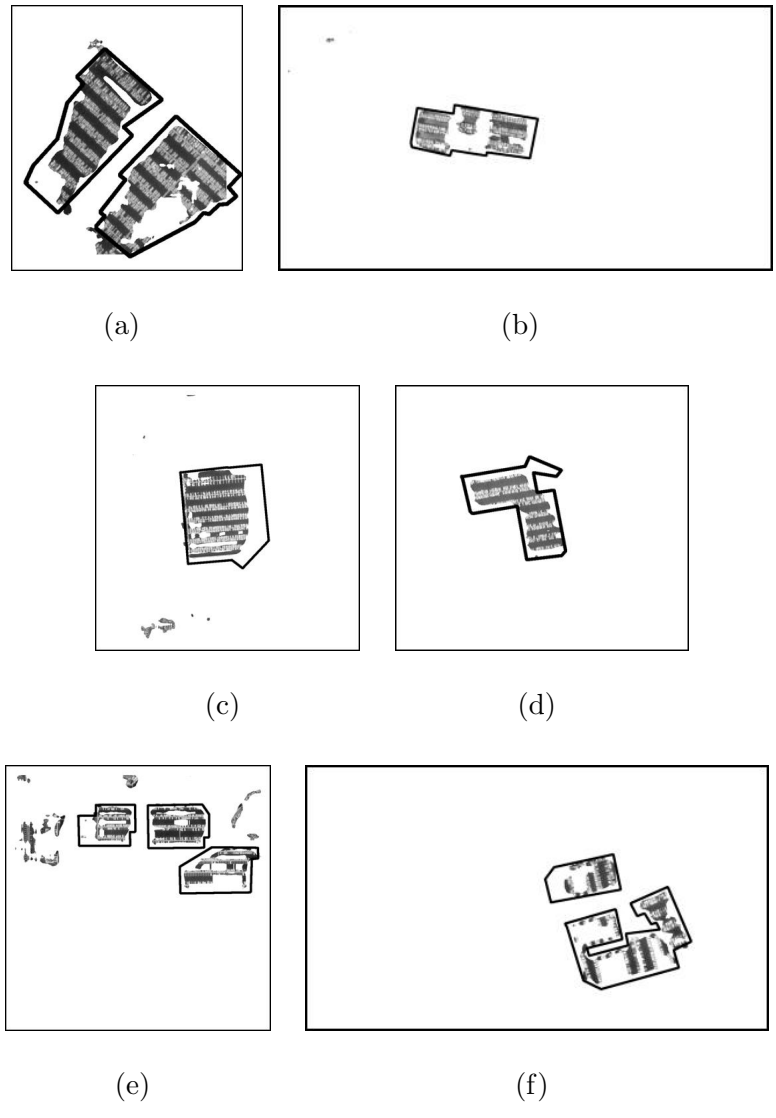


Figure 4.16: The detected harbor regions for $\alpha = 10$ (with $N_t = 3$, $N_s = 1$, and $s_h = 51$). The borders of the desired object regions are marked in black.

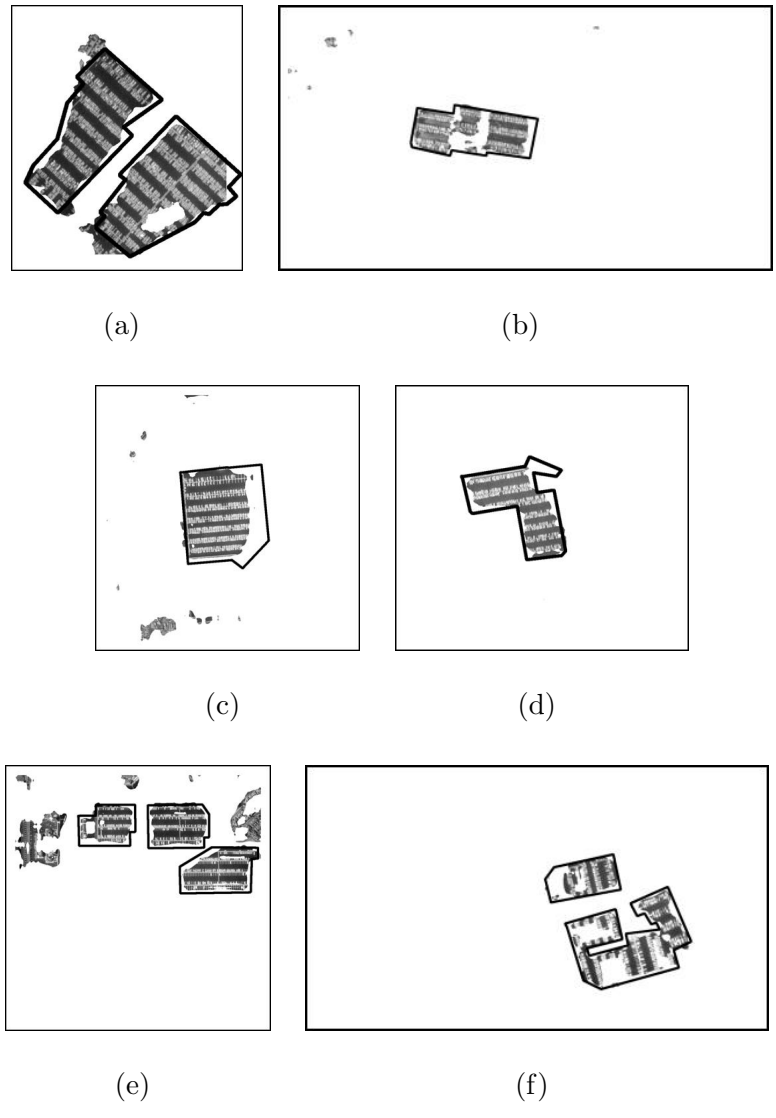
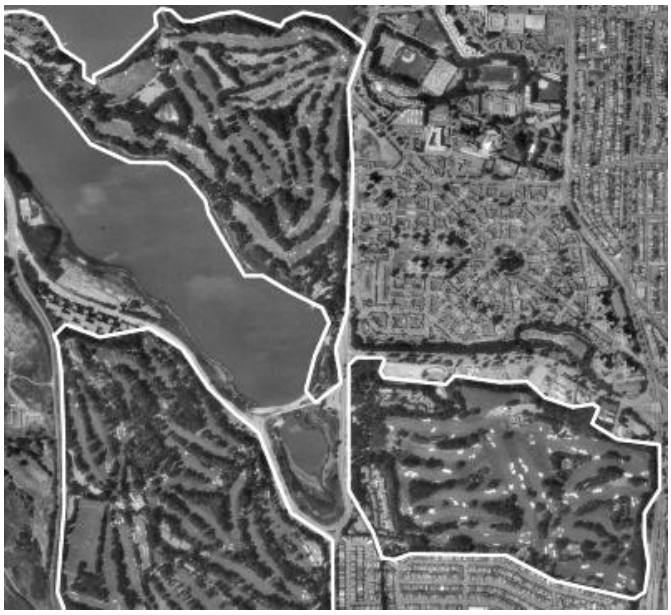
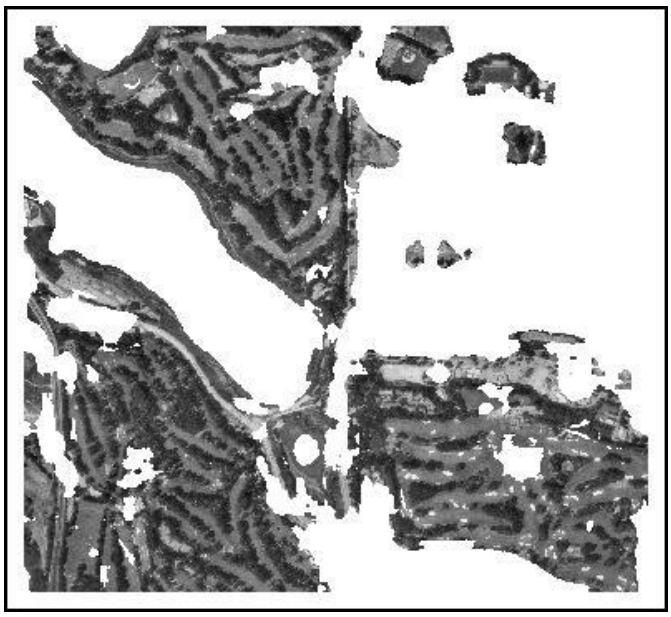


Figure 4.17: The detected harbor regions for $\alpha = 2$ (with $N_t = 3$, $N_s = 1$, and $s_h = 51$). The borders of the desired object regions are marked in black.



(a)



(b)

Figure 4.18: (a) A large geospatial image containing several golf course regions (denoted with white borders); (b) The detected golf course regions with the application of the two-layered texture motif model.

(a)

(b)

Figure 4.19: (a) A large geospatial image containing several harbor regions (denoted with white borders); (b) The detected harbor regions with the application of the two-layered texture motif model.

presence.

The object modeling and detection approach presented in this chapter is very useful in this regard. In the following, we shall demonstrate that our approach is capable of drastically pruning a dataset of images while looking for an object. Once again, we choose the *harbor* and *golf course* objects for our experiments. The dataset consists of large DOQQ images. The typical size of each image is close to 7500×6600 pixels. The images are divided into tiles of size 1024×1024 pixels, with an overlap of 512 pixels between adjacent tiles. For a given object, groundtruth information is created by labeling each of these tiles as 1 or 0 depending on whether the object is present in the tile or not. Let T_i be the i^{th} tile in the dataset, with $G_i \in \{0, 1\}$ denoting its groundtruth label. Suppose we apply an object detection algorithm on tile T_i, and get the "decision" $D_i \in \{0, 1\}$. In other words, $D_i = 1$ if an object is detected by the algorithm in tile T_i, and $D_i = 0$ if not.

We demonstrate the performance of the detection method by plotting the fraction of *false alarm* tiles (false alarm rate) with that of *missed* tiles (miss rate) from the dataset. T_i is a *false alarm* tile if $G_i = 0$ and $D_i = 1$, i.e. an object is detected when in fact it is not present in the tile. T_i is a *missed* tile if $G_i = 1$ and $D_i = 0$, i.e. an object is not detected but it does exist in the tile. Both false alarms and misses are undesirable. Pruning a dataset is often a tradeoff between the two.

Let $fmiss(t_o)$ and $ffa(t_o)$ denote the fraction of missed and false alarm tiles respectively for a threshold t_o on the confidence measure, $p_s(\mathbf{h}(\mathbf{x}))$ in (4.14). These are computed as

$$\begin{aligned} \mathit{fmiss}(t_o) &= \frac{1}{N}\sum_i \left(1 - D_i(t_o)\right) G_i, \text{ and} \\ \mathit{ffa}(t_o) &= \frac{1}{N}\sum_i D_i(t_o)(1 - G_i), \end{aligned} \tag{4.21}$$

where N is the total number of tiles in the dataset. $D_i(t_o) = 1$ if an object is detected at tile T_i given the threshold t_o. In our experiments, we set $D_i(t_o) = 1$ if at least $D_{\min}$ pixels in T_i have a confidence measure greater than t_o. $D_{\min}$ is set at 200 pixels for both golf courses and harbors. In practice, if a tile T_i has $D_i(t_o) = 1$, then we set the detection labels of the neighboring tiles to also be 1. This is done in order to avoid tiles being missed on account of their overlapping a small portion of the object. If a neighboring tile contains a larger portion of the object, its confidence is inherited by the current one. The "false alarm vs. missed" plot for an object is obtained by varying t_o and recording the corresponding values of $\mathit{ffa}(t_o)$ and $\mathit{fmiss}(t_o)$.

Fig. 4.20(a) shows the false alarm vs. missed plot in the case of the golf courses. A total of 157 DOQQs were considered, of which 20 contained one or more golf courses. Among the 22530 resulting tiles, 350 are given a groundtruth label of 1 since they overlap a golf course. The diagonal line connecting the 1's denotes the expected plot for a random detection decision, i.e. the worst possible plot. A plot that dips closest to the origin is considered good since, at certain thresholds, a low rate is obtained for both false alarms and misses. From Fig. 4.20(a), it can be seen that no golf course tiles are missed at a false alarm rate of 43.22%. In other words, 56.78% of the tiles are eliminated without missing any golf course tiles. However, if we relax the acceptable miss rate to 14.57%, then the false alarm rate

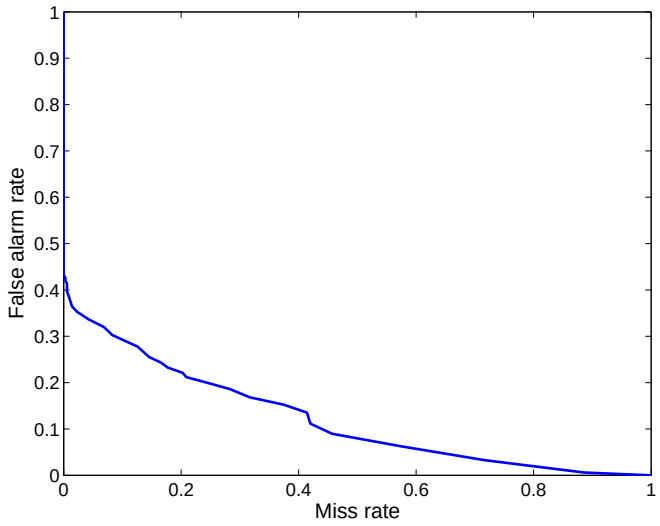
becomes 25.51%, eliminating 74.49% of the tiles.

Fig. 4.20(b) shows the false alarm vs. missed plot in the case of the harbors. A total of 214 DOQQs were considered, of which 24 contained one or more harbors. Among the 30765 resulting tiles, 313 are given a groundtruth label of 1 since they overlap a harbor. From Fig. 4.20(b), it can be seen that no harbor tiles are missed at a false alarm rate of 56.17%. In other words, 43.83% of the tiles are eliminated without missing any harbor tiles. However, if we relax the acceptable miss rate to 9.46%, then the false alarm rate becomes 16.29%, eliminating 83.71% of the tiles. Thus an effective pruning of large datasets is achieved, reducing the manual labor involved in ascertaining the presence and location of objects.

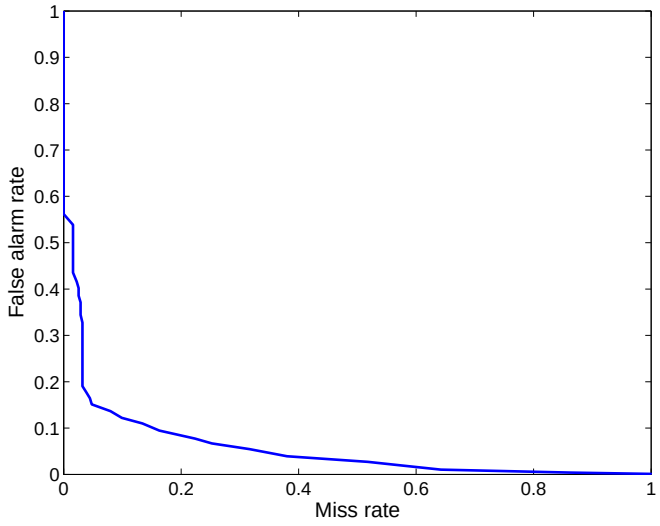
4.6 Summary of Contribution

The main contributions of this chapter are as follows. Firstly, it introduces the concept of texture motifs enabling model-driven detection of geospatial objects. Texture motifs of an object are spatially recurrent patterns that are characteristic to the object. Such spatial patterns can be observed in geospatial objects such as golf courses, harbors, and airports. Detection of an object then reduces to detecting one or more of its texture motifs. This is done by learning an appearance model for texture motifs from object examples. Object models based on texture motifs provide a powerful alternative to shape-based and edge-based models, which are prohibitively expensive to compute, due to the level of complexity and detail often found in geospatial objects.

The second contribution of this chapter is a semi-supervised framework for



(a)



(b)

Figure 4.20: (a) The false alarm vs. missed plots for detecting (a) golf courses and (b) harbors, in large DOQQ datasets.

learning a two-layered model for texture motifs of an object from examples. The first layer learns the local intensity variations in the motif that form textural elements such as flat areas, bars, edges, and so on. These can be interpreted as the low-level building blocks of the motif. This layer is learned by clustering Gabor filter outputs sampled from the object examples in a rotation-invariant manner. The second layer of the representation learns the spatial distribution of low-level texture elements in the texture motif, since this influences its distinct visual appearance. A Gaussian mixture model (GMM) for this is learned from examples using features derived from histograms of texture elements in spatial neighborhoods. Confidence measures generated using this model are then used for detecting object presence.

The quality of the models are evaluated on the basis of their application to object detection. Experimental results demonstrate that such a modeling approach is quite effective in detecting complex geospatial objects. We also illustrate the usefulness of our approach in reducing the manual labor involved in identifying object locations in large DOQQ datasets.

Chapter 5

Modeling Adjacency Using HMM

The approach described in the previous chapter represents a texture motif by learning its constituent texture elements and its spatial distribution in a neighborhood. This only gives information about the likelihood of two elements occurring with a certain proximity. In this chapter, we describe another approach that learns the likelihood of adjacency between the constituent elements of a motif. Originally proposed in [58], this approach adopts a hidden Markov modeling (HMM) framework for learning adjacency relations between elements. In this approach, it is possible to combine the two layers of the previous approach into a single stage of learning.

Consider a sequence of pixel sites along a line in an object. Each pixel generates an observation, say the texture feature extracted in its neighborhood. Suppose this sequence is modeled as a HMM wherein the *state* of each pixel corresponds to the local texture element, and the *state transitions* from one pixel to the next corresponds to the spatial arrangement of the elements. Then the

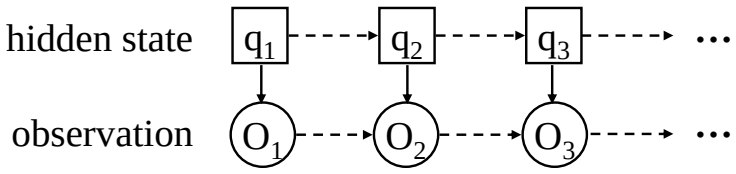


Figure 5.1: A system that is described by a hidden Markov model (HMM). The hidden state of the system at time t is denoted as q_t, and the observation outputted by the system at time t is denoted as O_t.

probability of a transition from one state to another can be interpreted as the probability of adjacency of the two corresponding texture elements. We now proceed to describe this method in detail and demonstrate its application in object detection.

5.1 The Hidden Markov Model

The *hidden Markov model* (HMM) was introduced in the late 1960s, and it has since then been extensively applied to a variety of pattern recognition problems. In particular, the HMM is a popular tool for solving speech recognition problems [64].

Consider a system (Fig. 5.1) that could be described at any time as being in one of M states, $\{S_1, S_2, \ldots, S_M\}$. At regularly spaced discrete times $t = 1, 2, \ldots$, the system undergoes a probabilistic transition of state (possibly back to the same state). The state of the system q_t at a time t is an unobservable or hidden stochastic process. What is observable at each time instant is another stochastic process, namely an *observation* O_t outputted by the system.

The HMM is an effective means of modeling a system such as the above. For

doing this, the HMM makes a simplifying assumption. The sequence of states $\{q_t\}$ is assumed to obey the Markov property, which means that the conditional probability of a state at a time instant only depends on the state at the previous time instant. In mathematical notation, this is written as

$$\begin{aligned} P\left(q_t = S_n | q_{t-1} = S_m, q_{t-2} = S_l, \ldots\right) &= P\left(q_t = S_n | q_{t-1} = S_m\right) \\ &= a_{mn}, \end{aligned} \tag{5.1}$$

where a_{mn} is the probability of transition from state S_m at time $t-1$ to state S_n at time t (as shown in Fig. 5.2). The HMM is completely specified by the following parameters:

1. The set of possible states $\mathcal{S} = \{S_m; m = 1, 2, ..., M\}$,
2. The prior probabilities π_m of each state S_m, i.e. $\pi_m = P(q_1 = S_m)$,
3. The transition probabilities, $a_{mn} = P\left(q_t = S_n | q_{t-1} = S_m\right)$, and
4. The conditional observation densities, $b_m(y) = p(O_t = y | q_t = S_m)$, assuming that the observations are continuous.

Let $\lambda = \{\pi_m, a_{mn}, b_m(y); \forall(S_m, S_n)\}$ denote a HMM. For practical application of this model, there are three basic problems that need to be solved. These are posed as the following questions.

Question 1: Given the observation sequence $O = O_1 O_2 \ldots O_T$, how do we efficiently compute $P(O|\lambda)$, i.e. the probability of an observation sequence given the model?

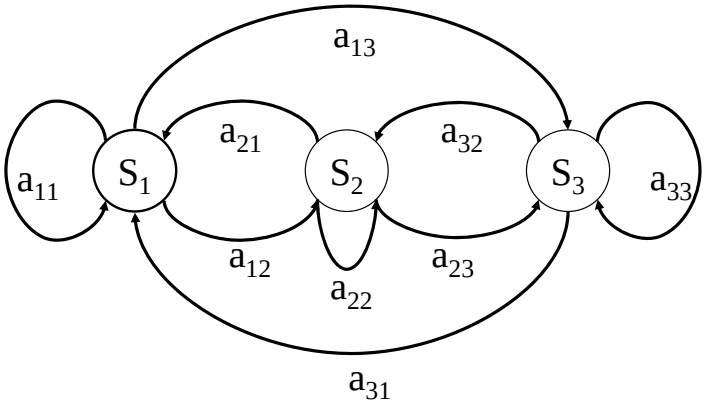


Figure 5.2: Symbolic notation of a hidden Markov model (HMM) with three states ($M = 3$). a_{mn} is the probability of transition from state S_m at time $t-1$ to state S_n at time t.

Question 2: Given the observation sequence $O = O_1 O_2 \ldots O_T$, what is the state sequence $Q = Q_1 Q_2 \ldots Q_T$ that best "explains" the observations?

Question 3: How do we estimate the model parameters to maximize $P(O|\lambda)$?

In his tutorial, Rabiner [64] describes the *forward* algorithm for solving the first problem, the *Viterbi* algorithm for solving the second, and the *Baum-Welch* (EM) algorithm for solving the parameter estimation problem. Having outlined the theoretical foundations of the HMM, we now proceed to demonstrate its application in the modeling and detection of texture motifs.

5.2 A Model for Texture Motifs

Consider a sequence of pixel sites obtained by drawing a line through the object at an angle θ with respect to the vertical, and uniformly sampling the

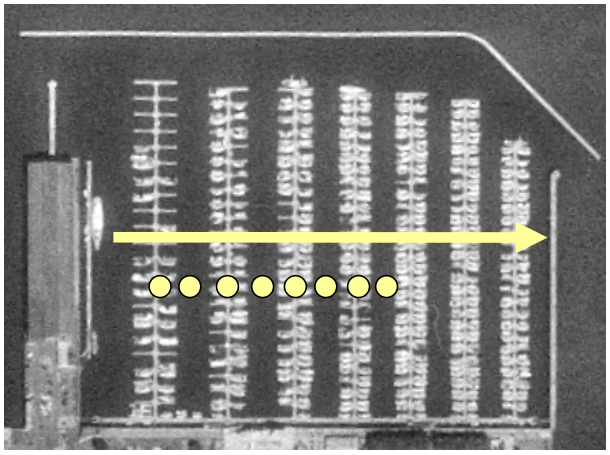


Figure 5.3: Observations (texture features) are sampled along a line (arrow) in an instance of the harbor object. Neighboring samples (circles) are separated by a distance r.

pixels intersecting it. Let the pixels on the line be sampled at a distance of r from each other (Fig. 5.3). Such a sequence from an object is then modeled as a one-dimensional HMM, $\lambda(\theta, r)$. The framework of the one-dimensional HMM is adopted for the sake of computational simplicity. Although two-dimensional HMM frameworks [40, 20] exist and are possibly better suited for image analysis, the available computational solutions are far less efficient than in the case of one-dimensional HMMs. However, the downside of using a one-dimensional HMM is that it becomes necessary to learn models $\lambda(\theta, r)$ for different orientations of the texture motif. This is to enable the detection of the motif in a new image, where its orientation is unknown.

We pose the problem of modeling texture motifs within the HMM paradigm by making the following assumptions.

1. The time instants $t = 1, 2, \ldots$, index the pixel positions along the observation sequence.

2. The states $\{S_1, \ldots, S_M\}$ refer to the texture elements constituting the motif. The terms *state* and *texture element* shall be used interchangeably in this discussion.

3. The transition probabilities a_{mn} denote the probability that two texture elements will be sampled adjacently in the sequence, i.e. S_n will be sampled immediately after S_m.

4. The observation O_t at pixel t is the texture feature vector at that pixel, denoted by $\mathbf{c}(\mathbf{x})$, where $\mathbf{x} = [x\ y]^T$ denotes the spatial coordinates of the pixel indexed by t.

To refresh our memory, the texture in the neighborhood of a pixel $\mathbf{x}$ is represented by an SK-dimensional feature vector $\mathbf{c}(\mathbf{x})$ obtained by convolving the image with a Gabor filter bank at S scales and K orientations. The filter bank (see Sec. 3.1.2) is a set of Gabor-wavelet filters denoted by

$$\{g_{s,k}(\mathbf{x}); s \in [0, S-1], k \in [0, K-1]\}.$$

Following (3.6), the feature vector is given by

$$\mathbf{c}(\mathbf{x}) = [F_{0,0}(\mathbf{x}) \quad F_{0,1}(\mathbf{x}) \quad \ldots \quad F_{S-1,K-2}(\mathbf{x}) \quad F_{S-1,K-1}(\mathbf{x})]^T, \tag{5.2}$$

where $F_{s,k}(\mathbf{x})$ is the filter output at pixel $\mathbf{x}$, obtained by convolving the image $I(\mathbf{x})$ with the filter $g_{s,k}(\mathbf{x})$. In other words, $F_{s,k}(\mathbf{x}) = |g_{s,k}(\mathbf{x}) * I(\mathbf{x})|$.

For computational purposes, we make the additional assumption that the observations $\mathbf{c}$ from a texture element (state) S_m follow a Gaussian distribution,

$$b_m(\mathbf{c}) = \frac{1}{\sqrt{(2\pi)^d\,|\Sigma_m|}} e^{-\frac{1}{2}(\mathbf{c}-\mu_m)^T \Sigma_m^{-1} (\mathbf{c}-\mu_m)} \tag{5.3}$$

where d is the dimensionality of the data, Σ_m is the covariance matrix, and μ_m is the mean vector. The HMM is then fully specified by its parameters as follows,

$$\lambda(\theta, r) = \{\pi_m, a_{mn}, \Sigma_m, \mu_m; \forall(S_m, S_n)\}, \tag{5.4}$$

where π_m is the prior probability of state S_m. Thus, using this framework, it is possible to learn in a single stage both the texture elements and certain spatial relations among them.

5.2.1 Training the Model

Given a training set of object instances, the process of constructing $\lambda(\theta, r)$ is as follows:

1. Construct sequences of pixels from each object instance by sampling along lines at angle θ, w.r.t. the vertical. The pixels are sampled at a distance of r from each other. Obtain the corresponding observation sequences, i.e. texture feature vectors at the pixels.

2. Estimate the model parameters in (5.4) from the above observation sequences, using the Baum-Welch (EM) algorithm described in [64]. The number of states for the model M is manually chosen depending on the visual complexity of the object. Initialization is random.

The first step, i.e. sampling the observation sequences, is illustrated in Fig. 5.4. The arrows indicate different observation sequences sampled (with step r) from an instance of a golf course at an angle 90° w.r.t. the vertical. The training data is formed by the union of the sequences collected from each object instance in the

Figure 5.4: The arrows indicate different observation sequences sampled from an instance of a golf course at an angle 90° w.r.t. the vertical. Sequences collected from several instances are used for estimating the parameters of the model $\lambda(90^\circ, r)$ where r is the sampling step.

training set. This data is then used for estimating the parameters of the model $\lambda(90^\circ, r)$.

Having chosen a sampling step r, the complete model for the object is defined as a collection of HMMs at different θ. In other words,

$$\Lambda_r = \{\lambda(\theta, r); \theta = \theta_1, \ldots, \theta_R\}, \tag{5.5}$$

where the angles θ_i are obtained at appropriate steps in the interval $[0, 180^\circ]$.

5.2.2 Aligning the Training Set

As mentioned before, the issue of detecting a texture motif with unknown orientation is handled by training models for the appearance of the motif at a number of orientations. However, an issue that arises during the training phase

itself is that of aligning the different examples from the training set. This is necessary because each training example may contain the same texture motif at different orientations. This is illustrated in Fig. 5.5. Clearly, without alignment, the observation sequences from the two training examples convey different appearances of the motif. Thus, learning a model from the union of these sequences defeats the purpose. After alignment (Fig. 5.5(c)), the sequences result in a consistent model. The alignment of the examples in the training set is currently done manually.

5.3 Object Detection

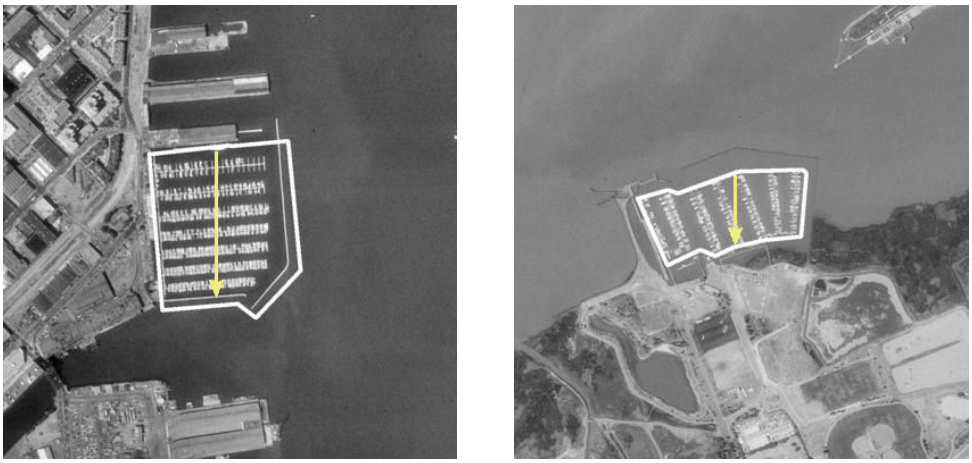
Suppose a HMM-based object model Λ_r has been learned for an object from examples. We now describe a strategy for detecting instances of the object in a given image. Let $\mathbb{O} = \{\mathbf{O}_i\}$ denote a set of observation sequences from a region in the image. The likelihood that $\mathbb{O}$ is generated by the model Λ_r is interpreted as the likelihood that the region belongs to the object. Since the orientation of the object in the image is unknown, this likelihood $L_{\mathbb{O}}$ is computed as,

$$L_{\mathbb{O}} = \max_{\theta} \left[\log P\left(\mathbb{O}|\lambda(\theta, r)\right)\right]. \tag{5.6}$$

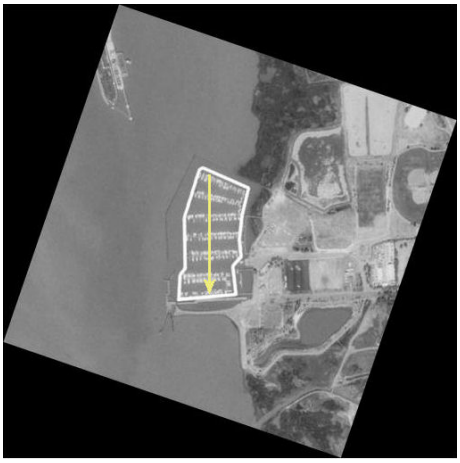
Furthermore, we have

$$P\left(\mathbb{O}|\lambda(\theta, r)\right) = \frac{1}{N_o} \sum_i P\left(\mathbf{O}_i|\lambda(\theta, r)\right), \tag{5.7}$$

where N_o is the number of sequences in $\mathbb{O}$. The forward algorithm described in [64] is used for computing $P\left(\mathbf{O}_i|\lambda(\theta, r)\right)$. It is possible to scale the likelihood such that $L_{\mathbb{O}} \in (-\infty, 0)$ with 0 denoting the highest likelihood.



(a) (b)



(c)

Figure 5.5: The top row shows two training examples of the harbor object. The arrows in each indicate an observation sequence sampled along the vertical. Clearly, the two observation sequences convey different texture motif appearances. After aligning the second example to the first, as in (c), the sequences convey a consistent motif appearance.

In practice, the method used for object detection is illustrated in Fig. 5.6. The image contains an instance of the golf course object that we desire to detect using a model for the corresponding texture motif learnt from golf course examples. The test image is divided into square tiles. From each tile, a number of observation sequences $\mathbb{O} = \{\mathbf{O}_i\}$ are sampled. Using (5.6) and (5.7), a likelihood of object presence is assigned to each tile. Those tiles that have likelihoods above an appropriately chosen threshold could be detected as belonging to the object. Note that the precision with which the object is detected depends on the size of the tile and the overlap between tiles. However, the tiles should be large enough to capture the properties of the texture motif.

5.3.1 A Confidence Measure for Object Detection

The likelihood in (5.6) provides a confidence measure of finding an object in a region. It is possible to further strengthen this measure as follows. Given a HMM λ for a texture motif, we construct a vector termed the *expected state duration* vector, denoted as

$$\bar{D}_\lambda = \begin{bmatrix} \bar{d}_1 & \bar{d}_2 \dots & \bar{d}_M \end{bmatrix}^T, \tag{5.8}$$

where $\bar{d}_i$ is the *expected duration* of state S_i. The expected duration $\bar{d}_i$ is defined as the expected number of continuous observations in state S_i, given that the

sequence starts in this state. $\bar{d}_i$ is computed as (see [64])

$$\begin{aligned} \bar{d}_i &= \sum_{d=1}^{\infty} d p_i(d) \\ &= \sum_{d=1}^{\infty} d(a_{ii})^{d-1}(1-a_{ii}) \\ &= \frac{1}{(1-a_{ii})} \end{aligned} \tag{5.9}$$

where $p_i(d)$ is the probability of d continuous observations in state S_i, and a_{ii} is the self-transition probability of S_i.

For any region in a test image, a measure of its observed deviation from the complete texture motif model Λ_r (given by (5.5)) is defined as

$$E_{\mathbb{O}} = \min_{\theta} \parallel \bar{D}_{\lambda(\theta,r)} - \hat{D}_{\mathbb{O}} \parallel_1 . \tag{5.10}$$

$\parallel . \parallel_1$ denotes the L_1 norm. We shall call $E_{\mathbb{O}}$ the *state duration distance*. $\hat{D}_{\mathbb{O}}$ is a vector of empirical state durations observed from $\mathbb{O} = \{\mathbf{O}_i\}$ which is the set of observation sequences from the test region. It is given by

$$\hat{D}_{\mathbb{O}} = \left[\hat{d}_1 \;\; \hat{d}_2 \ldots \;\; \hat{d}_M\right]^T \tag{5.11}$$

where $\hat{d}_i$ is the average duration of a continuous run in state S_i and is computed as,

$$\hat{d}_i = \frac{\sum_i \text{duration of the } i^{\text{th}} \text{ continuous run in state } S_i}{\text{number of continuous runs in state } S_i}. \tag{5.12}$$

Of course, for computing $\hat{d}_i$, the optimal state sequences $\mathbf{Q}_j$ corresponding to each $\mathbf{O}_j \in \mathbb{O}$ has to be found. This is done by using the Viterbi algorithm [64] (see Question 2 in Sec. 5.1).

We combine the likelihood $L_{\mathbb{O}}$ in (5.6) with the deviation $E_{\mathbb{O}}$ in (5.10) to get a new confidence measure $C_{\mathbb{O}}$ for a region with observations $\mathbb{O}$. This is done as

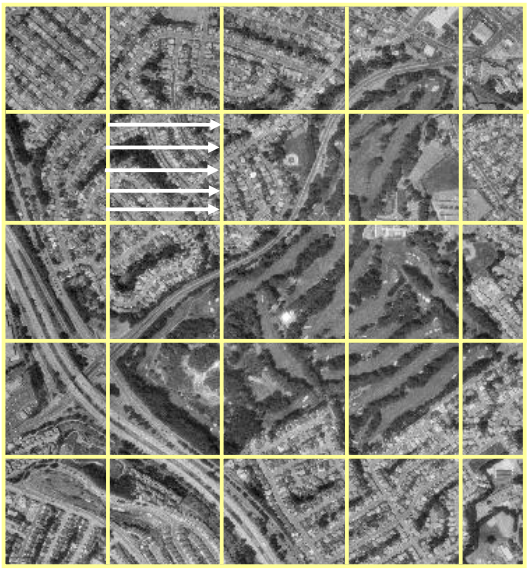


Figure 5.6: Region-based sampling of observation sequences for object detection. The test image is divided into tiles and a number of sequences (arrows) are sampled from each one.

follows.

$$C_{\mathbb{O}} = L_{\mathbb{O}} E_{\mathbb{O}} \tag{5.13}$$

This has the effect of moderating the likelihood $L_{\mathbb{O}}$ with the deviation $E_{\mathbb{O}}$. It is possible to scale the likelihood $L_{\mathbb{O}}$ such that $L_{\mathbb{O}} \in (-\infty, 0)$ with $-\infty$ denoting the least likelihood. We know that $E_{\mathbb{O}} \in (0, \infty)$ with ∞ being the highest deviation. Overall, $C_{\mathbb{O}} \in (-\infty, 0)$ with $-\infty$ denoting the least confidence. Therefore, a high deviation takes the product in (5.13) closer to $-\infty$ diminishing the confidence. Similarly, a high likelihood takes the product closer to 0 increasing the confidence. The highest confidence (closest to 0) is obtained in the case of a high likelihood and low deviation.

5.4 Experimental Results

We now demonstrate the application of the HMM-based texture motif model for object detection in geospatial images. The objects of interest are *golf courses* and *harbors*. The dataset of golf course instances in shown in Fig. 5.7 with the white borders indicating the extent of the golf course regions. Similarly, the harbor dataset is shown in Fig. 5.13. Note that harbors are strongly directional and therefore need to be aligned prior to training, as discussed in Sec. 5.2.2. The aligned version of the harbor dataset used for training purposes is shown in Fig. 5.14. We do not realign the golf courses prior to training since their texture motifs are not strongly directional.

The specifications of the Gabor filter bank used for extracting texture features, $\mathbf{c}(\mathbf{x})$ in (5.2), are $S = 5$, $K = 6$, $U_l = 0.05$ and $U_h = 0.4$ (see Sec. 3.1.1). We fix the spatial direction of the observation sequences to be horizontal (left to right), i.e. $R = 1$ and $\theta_1 = 90^\circ$ in (5.5). The modeling parameters that are allowed to vary are the number of states (M) and sampling step in pixels (r in (5.5)). For experimental evaluation, the dataset is divided into a training set and a test set. For given M and r, a HMM-model for the texture motif is trained as described in Sec. 5.2.1. The training data comprises observation sequences sampled from left to right inside the object (golf course or harbor) regions in the training set.

Object detection is done by dividing the test images into tiles as described in Sec. 5.3. We choose the tiles to be square with a side of length 400 pixels for golf courses and 120 for harbors. The overlap between adjacent tiles are chosen to be half the tile size in both directions. A confidence measure of object presence is

computed for each tile based on the model learned from the training set. Since the tiles overlap, we take the confidence at *each pixel* $\mathbf{x}_i$ to be the maximum of the confidences of all tiles that overlap it. Note that a higher overlap between adjacent tiles gives a higher spatial precision in detecting the object.

Objects are detected by applying a threshold t_o on the confidence measure of detection. In other words, all regions with confidence greater than t_o are detected as object regions. Given M, r, and the confidence measure, the detection performance is evaluated for a range of thresholds by means of a precision-recall graph (Sec. 4.5.1). The *precision* tells us how many pixels are correctly identified as belonging to the object. The *recall* tells us how many pixels belonging to the object are correctly identified as such. The precision-recall graph plots the precision $\mathcal{P}(t_o)$, given by (4.18), against the recall $\mathcal{R}(t_o)$, given by (4.19), while varying t_o. It displays the tradeoff between precision and recall at different thresholds. Note that precision and recall are computed by considering all the pixels $\mathbf{x}_i$ from all the object instances in the test set.

Since the dataset is limited, we employ a cross-validation methodology as explained in Sec. 4.5.2. The nine instances in the golf course dataset are randomly partitioned into three sets of three instances. Each set is used in turn as the test set, while the training set comprises the union of the remaining sets. For each object instance, we compute the tile-based confidence measure based on a model trained from the corresponding training set. The precision-recall graph is obtained by aggregating the test results from all pixels $\mathbf{x}_i$ from all objects in the dataset. The same procedure applies to the harbor dataset, except that we divide this dataset into two sets of three instances. In the case of harbors, the aligned

versions of the instances (Fig. 5.14) are used for training and the original versions (Fig. 5.13) for testing.

Using the golf course dataset, we first evaluate the performance of two confidence measures, namely the likelihood $L_{\mathbb{O}}$ in (5.6) and $C_{\mathbb{O}}$ in (5.13). Fig. 5.8 shows two sets of precision-recall plots for the golf course dataset with different modeling (M, r) and detection ($L_{\mathbb{O}}$, $C_{\mathbb{O}}$) parameters. Fig. 5.8(a) shows plots for different M and r using the likelihood $L_{\mathbb{O}}$ as the confidence measure for object detection. Fig. 5.8(b) shows the corresponding plots when the confidence measure $C_{\mathbb{O}}$ is used. Clearly, for each M and r, the plot in Fig. 5.8(b) has a higher precision over a wide range of recall than the plot in Fig. 5.8(a). This indicates that using $C_{\mathbb{O}}$ as a confidence measure leads to a superior object detection performance. For the rest of our experiments, we choose $C_{\mathbb{O}}$ to be the confidence measure for object detection.

The two best models by a visual inspection of Fig. 5.8(b) are $\{M = 5, r = 8\}$ and $\{M = 5, r = 16\}$. For illustrating results on object detection, let us choose the model $\{M = 5, r = 16\}$. In order to choose the "best" threshold, we plot the F-measure $\mathcal{F}_\alpha(t_o)$ (given by (4.20)) against a range of thresholds t_o, where $\alpha \in [0, +\infty)$ is the relative weight placed on precision over recall. Thus a higher value of α leads to fewer false alarms and more missed object regions. A lower value results in more false alarms but fewer missed regions. Fig. 5.9 shows the F-measure plots for two different α with model parameters set at $\{M = 5, r = 16\}$. For each α, the peak thresholds t_o^* are chosen as the object detection thresholds. In other words, all regions with $C_{\mathbb{O}} > t_o^*$ are detected as object regions.

Figures 5.10 and 5.11 show the detected golf course regions using t_o^*, for $\alpha = 10$

and $\alpha = 2$ respectively. The correctly detected regions are the ones inside the object regions specified by the black borders. Note that with a lower α, recall is higher at the expense of precision resulting in both a higher detection rate and false-alarm rate. Note also that the many of the falsely detected regions contain patterns quite similar to the trees-and-grass texture motif of golf courses. Fig. 5.12(a) demonstrates object detection in a large geospatial image containing several golf course instances. The object regions are delineated with white borders. Fig. 5.12(b) shows the detected golf course regions following the application of the HMM-based texture motif model for golf courses. The detection threshold t_0^* corresponds to the peak of the F-measure $\mathcal{F}_1(t_o)$ (i.e. $\alpha = 1$). It can be observed that most of the golf course regions are reliably isolated.

Similar experiments are performed on the harbor object. In the case of harbors, the use of selected scales in the formation of the feature vector $\mathbf{c}(\mathbf{x})$ (given by (5.2)) is shown to improve the performance significantly. We choose five scales and six orientations in forming the feature vector, i.e. $S = 5$ and $K = 6$ in (5.2). Fig. 5.15(a) shows precision-recall plots for the harbor dataset with different modeling parameters (M, r) using all five scales in forming $\mathbf{c}(\mathbf{x})$. Fig. 5.15(b) shows the corresponding plots using a truncated 12-dimensional feature vector formed from the first two scales only ($s = 0, 1$ in (5.2)). In this case, scale selection clearly leads to superior detection performance. We shall continue to use the truncated feature vector for our experiments on the harbor object.

We choose the best model from Fig. 5.15(b), i.e. $M = 2$ and $r = 2$. Fig. 5.16 shows the corresponding F-measure plots for two different α. For each α, the peak thresholds t_o^* are chosen as the object detection thresholds. In other words,

all regions with $C_{\mathbb{O}} > t_o^*$ are detected as object regions. Figures 5.17 and 5.18 show the detected harbor regions using t_o^*, for $\alpha = 10$ and $\alpha = 2$ respectively. The correctly detected regions are the ones inside the object regions specified by the black borders. Note that with a lower α, recall is higher at the expense of precision resulting in both a higher detection rate and false-alarm rate.

5.5 Summary of Contribution

In this chapter, we presented a framework for modeling texture motifs based on hidden Markov models (HMM). One-dimensional HMMs are used to model the motif appearance along a line in a certain direction by observing the texture features along the line. The *state* of each pixel is taken to correspond to the local texture element, and the *state transitions* from one pixel to the next corresponds to the spatial arrangement of the elements. Using such a modeling scheme enables us to learn higher-order spatial relations among texture elements in a motif. This includes adjacency between states (elements) and expected number of continuous samples from a state (expected state duration). Furthermore, this approach enables us to combine the two layers of the previous approach into a single stage of learning.

We have also demonstrated the application of the HMM-based model for object detection. Using the texture motif model, confidence measure of object presence is computed for any image region. This confidence measure is used in combination with a tile-based search scheme for detecting object regions in a test image. Experimental results demonstrate the effectiveness of the above approach

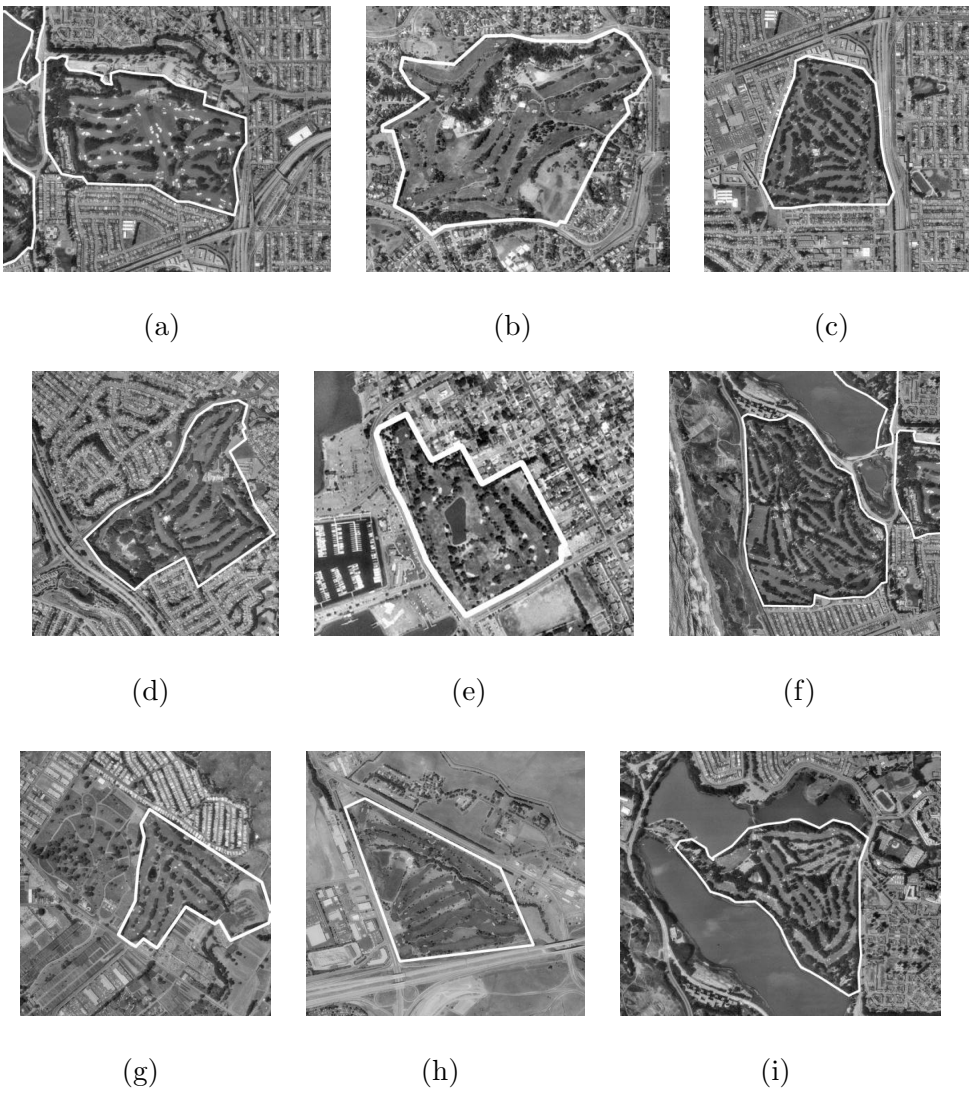
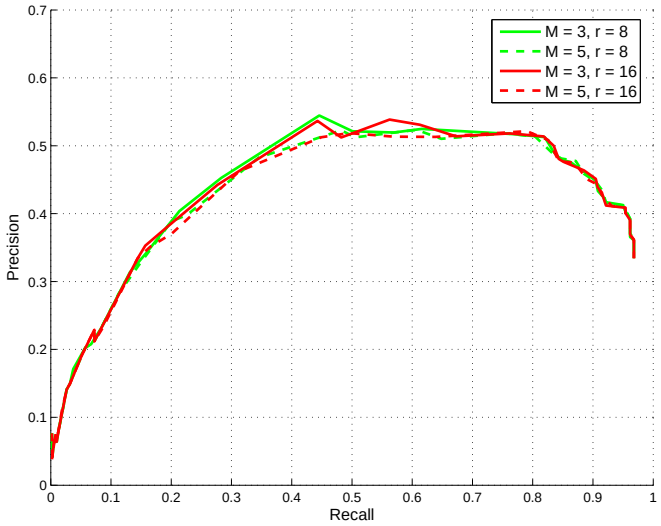
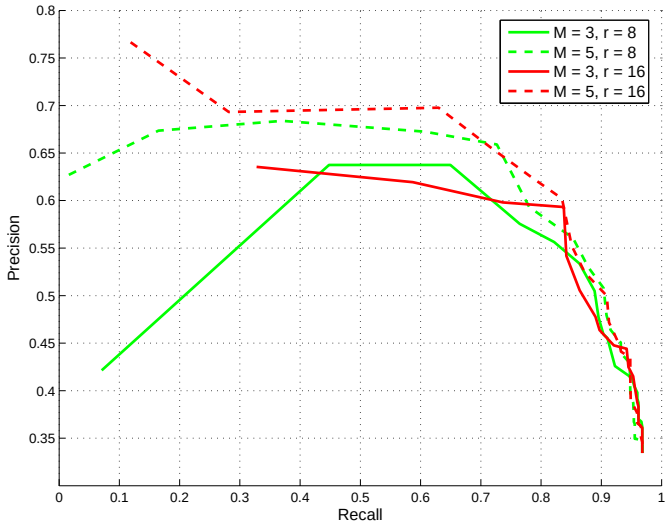


Figure 5.7: The instance dataset for the *golf course* object. The borders of the object regions (masks) are indicated in white.

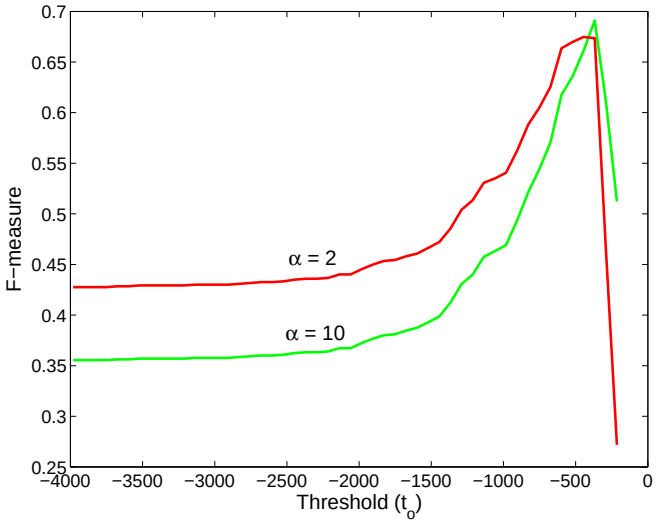


(a)



(b)

Figure 5.8: Precision-recall curves with different modeling and detection parameters for the golf course dataset. The modeling parameters are the number of states (M) and the sampling step in pixels (r). (a) Precision-recall curves with the likelihood $L_{\mathbb{O}}$ in (5.6) as the confidence measure. (b) Precision-recall curves using the confidence measure $C_{\mathbb{O}}$ in (5.13).



(a)

Figure 5.9: F-measure $\mathcal{F}_\alpha(t_o)$, given by (4.20), for different α computed using the golf dataset with modeling parameters $M = 5$ and $r = 16$ and using $C_{\mathbb{O}}$ as the confidence measure. For each α, the peak thresholds t_o^* are chosen as the object detection thresholds.

in detecting complex geospatial objects.

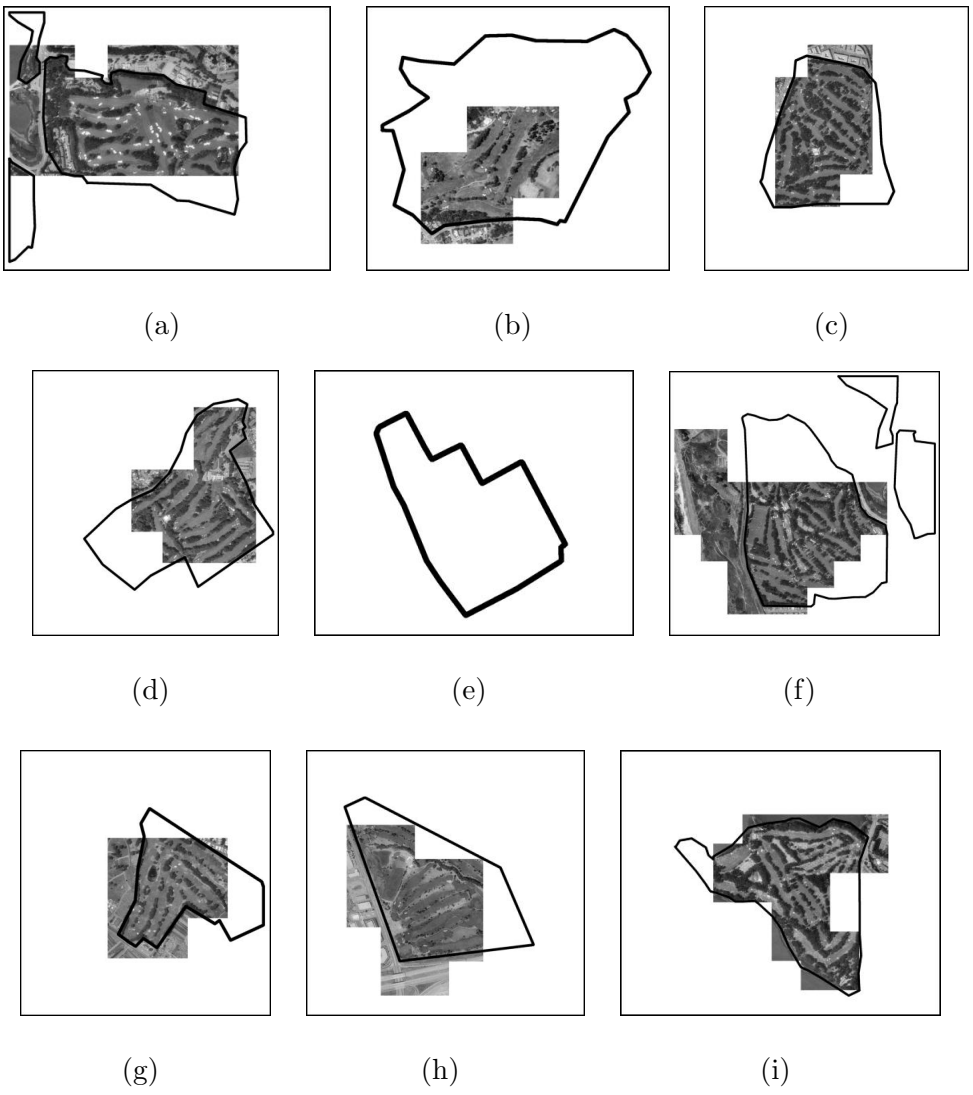


Figure 5.10: The detected golf course regions using the threshold t_o^* chosen from Fig. 5.9 for $\alpha = 10$ (with $M = 5$, $r = 16$, and confidence measure $C_{\mathbb{O}}$). The borders of the true object regions are marked in black.

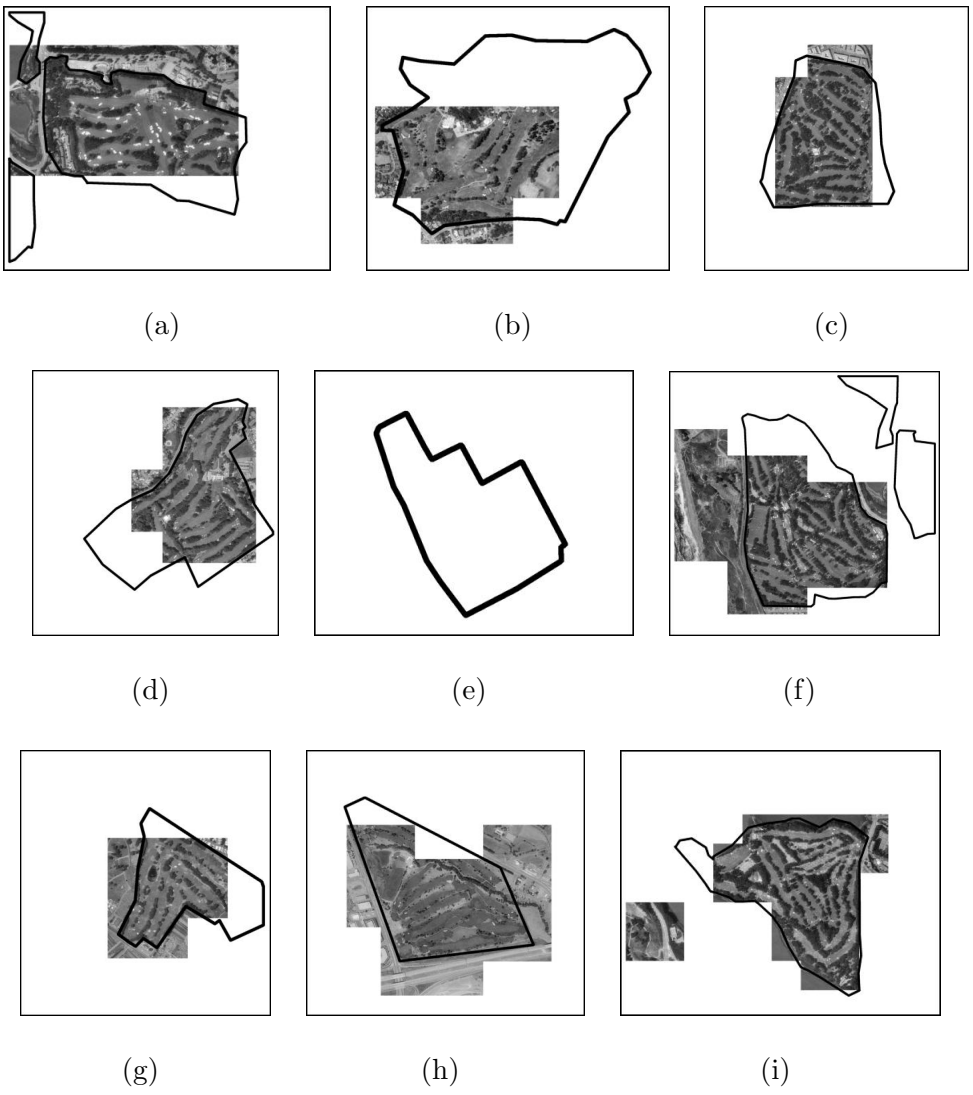
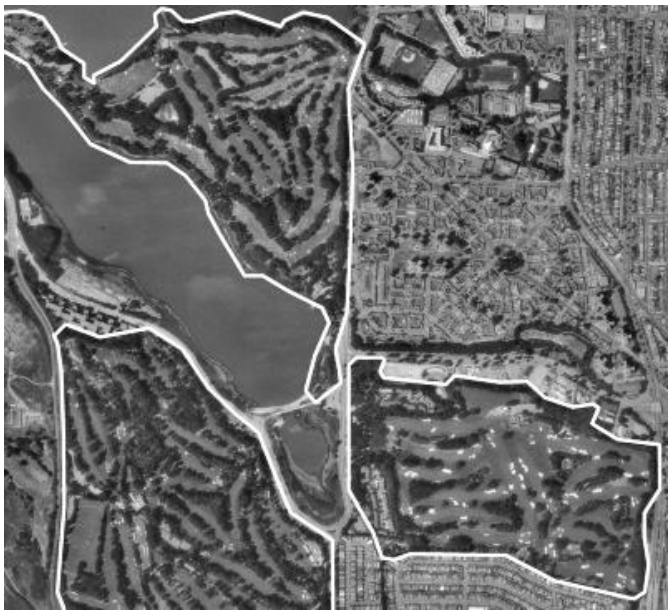
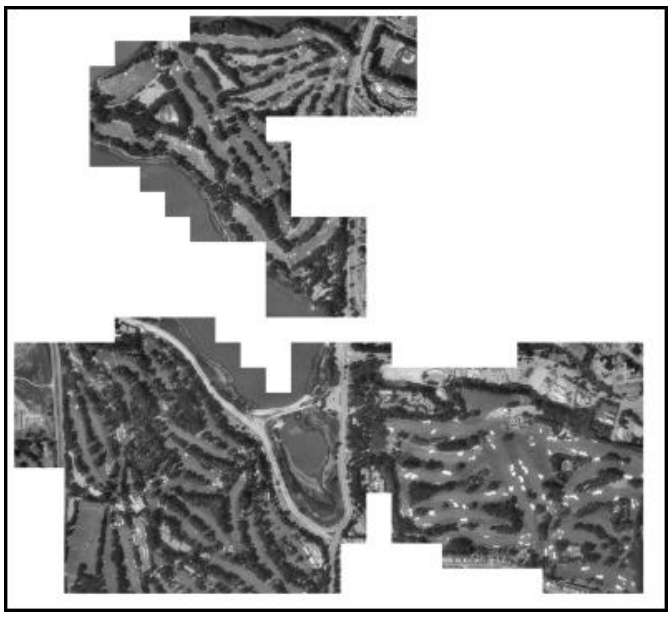


Figure 5.11: The detected golf course regions using the threshold t_o^* chosen from Fig. 5.9 for $\alpha = 2$ (with $M = 5$, $r = 16$, and confidence measure $C_{\mathbb{O}}$). The borders of the true object regions are marked in black.

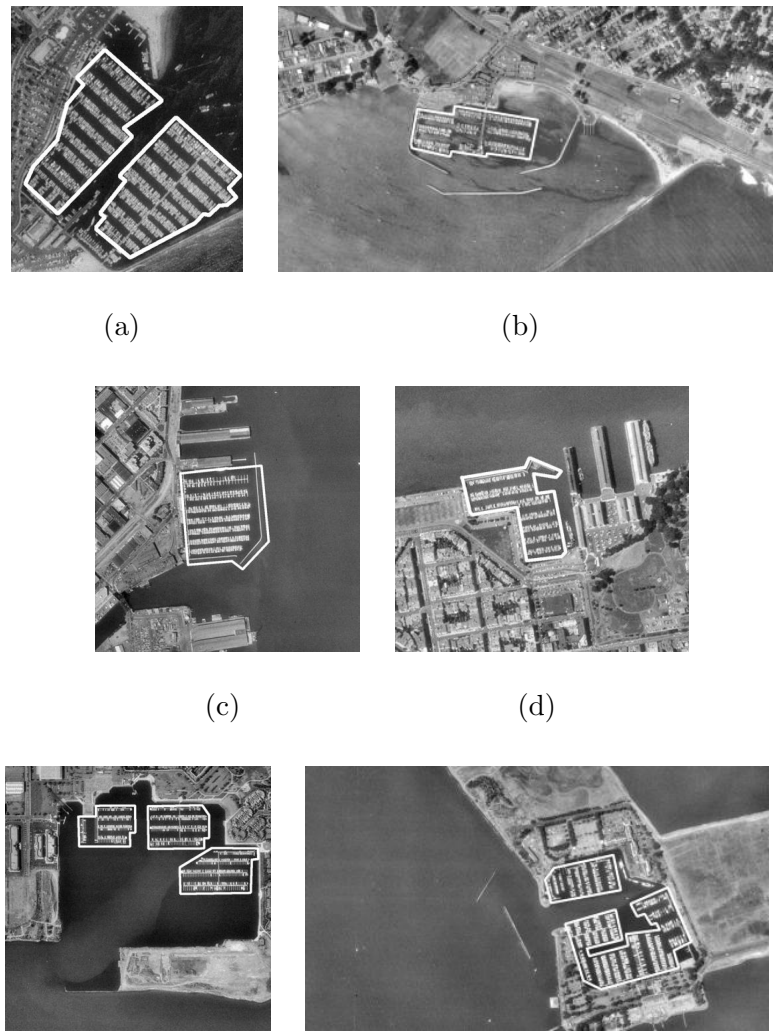


(a)



(b)

Figure 5.12: (a) A large geospatial image containing several golf course regions (denoted with white borders); (b) The detected golf course regions with the application of the HMM-based texture motif model.



(a) (b)

(c) (d)

(e) (f)

Figure 5.13: The instance dataset for the *harbor* object. The borders of the object regions (masks) are indicated in white.

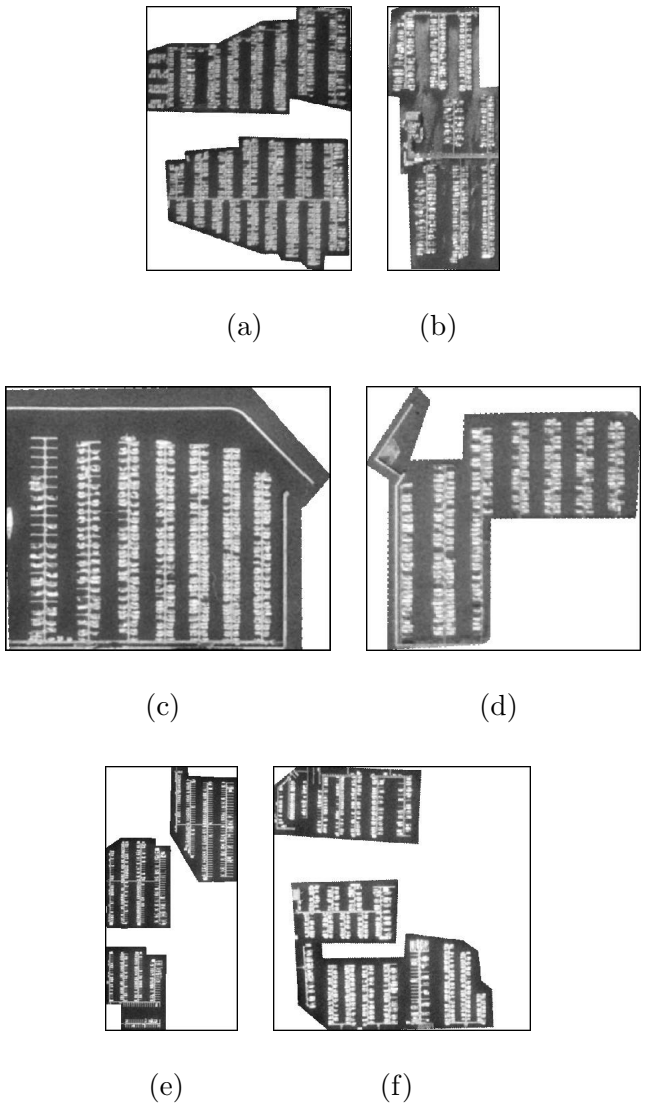
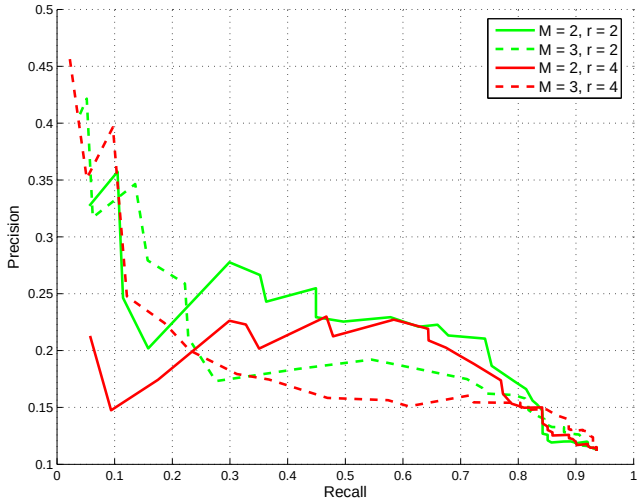
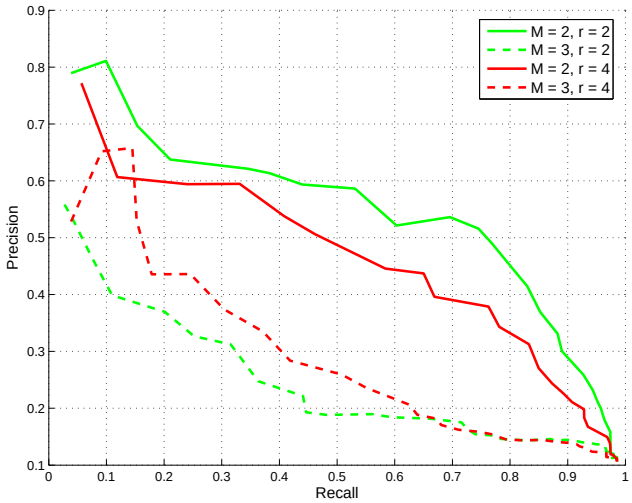


Figure 5.14: The manually aligned version of the harbor dataset. Only the harbor regions are shown here and not the background. The observation sequences forming the training data are horizontal, i.e. $R = 1$ and $\theta_1 = 90^\circ$ in (5.5).

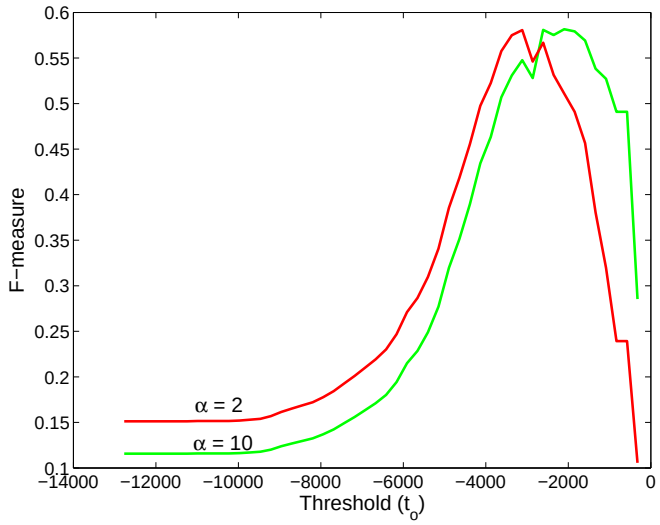


(a)



(b)

Figure 5.15: Precision-recall curves with different modeling and detection parameters for the harbor dataset. The modeling parameters are the number of states (M) and the sampling step in pixels (r). The confidence measure $C_{\mathbb{O}}$ in (5.13) is used for detection. (a) Precision-recall curves using a feature vector $\mathbf{c}(\mathbf{x})$ computed with a filter bank of five scales and six orientations ($S = 5$ and $K = 6$ in (5.2)). (b) Precision-recall curves using a truncated feature vector (first two scales $s = 0, 1$ only in (5.2)).



(a)

Figure 5.16: F-measure $\mathcal{F}_\alpha(t_o)$, given by (4.20), for different α computed using the harbor dataset with modeling parameters $M = 2$ and $r = 2$, and using $C_{\mathbb{O}}$ as the confidence measure. For each α, the peak thresholds t_o^* are chosen as the object detection thresholds.

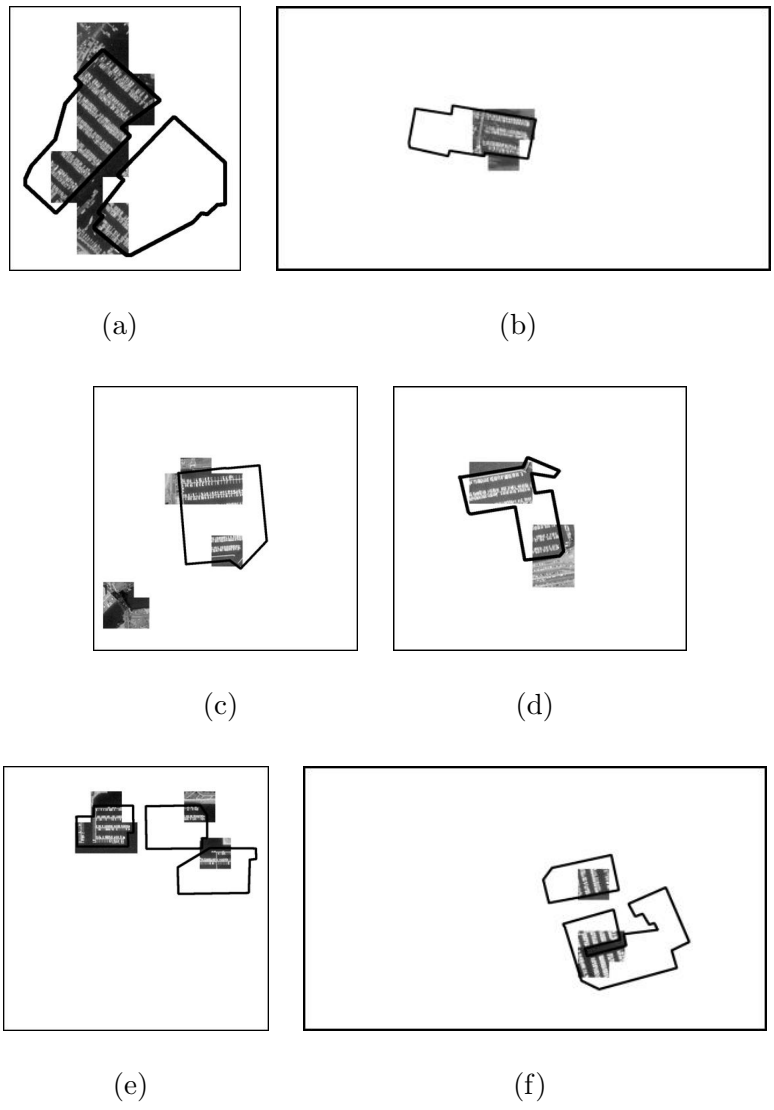


Figure 5.17: The detected harbor regions using the threshold t_o^* chosen from Fig. 5.16 for $\alpha = 10$ (with $M = 2$, $r = 2$, and confidence measure $C_{\mathbb{O}}$). The borders of the true object regions are marked in black.

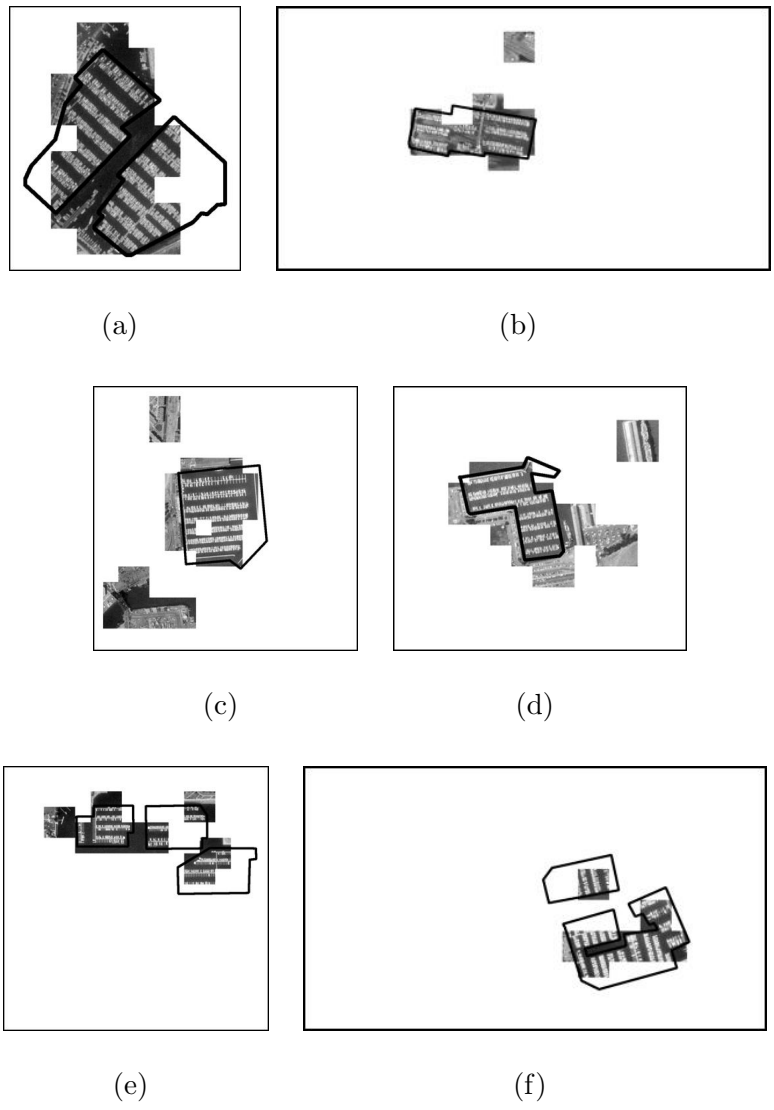


Figure 5.18: The detected harbor regions using the threshold t_o^* chosen from Fig. 5.16 for $\alpha = 2$ (with $M = 2$, $r = 2$, and confidence measure $C_{\mathbb{O}}$). The borders of the true object regions are marked in black.

Chapter 6

Detection of Quasi-Periodic Texture Motifs

The object detection methods discussed in the preceding chapters describe the texture in a neighborhood using a high-dimensional feature *vector*. Each dimension of the vector gives the "amount" of texture at a particular scale and orientation in the neighborhood. The dimensions of the vector are allowed to have values from a continuous interval. While this approach enables the description of a wide range of texture motifs, there are difficulties in working with features in high-dimensional spaces. Particularly, the clustering and density estimation techniques used previously in learning texture motif models prove to be quite unreliable in high-dimensional spaces.

When a texture motif comprises nearly periodic patterns at one or more scales and orientations (Fig. 6.1), it is possible to avoid representing it in a high-dimensional space. We term such patterns *quasi-periodic* texture motifs.

In this case, it is not necessary to capture the "amount" of texture at each scale and orientation, but just the *presence* or *absence* of a strong pattern at certain characteristic scales and orientations. Near-periodic patterns strongly respond to Gabor filters with frequencies close to their own. Thus, when such patterns are analyzed using a filter bank (such as (3.4)), they give rise to strong peaks in the filter output magnitudes at their characteristic scales and orientations.

In this chapter, we discuss the detection of objects with quasi-periodic texture motifs. In the process, we deviate from the high-dimensional frameworks presented in the previous chapters, while still preserving the central notion of texture motifs. Methods are presented for learning relations between the characteristic scales and orientations of a texture motif from object examples. This constitutes the modeling phase, whose output is a set of textural *rules* or *hypotheses* characterizing the motif. The framework also makes it possible for humans to specify rules. Object detection involves searching for regions that obey these textural rules.

6.1 Issues with Density Estimation and Clustering

The learning methods introduced in the previous chapter, although intuitively appealing, depend on clustering and density estimation in high-dimensional L_p spaces. Clustering and density estimation are affected by the so-called *curse of dimensionality*. The cause of the "curse" is the following [5]. The classical L_p

metrics are defined by the equation $d(\mathbf{a}, \mathbf{b}) = \{\sum_i |a_i - b_i|^p\}^{1/p}$. In particular, these metrics are based on sums of terms drawn from independent distributions. As dimensionality increases, the Central Limit Theorem asserts that such sums tend towards a normal distribution. Moreover, the mean of the distances between points increases, while the standard deviation remains roughly unchanged. Hence, *as dimensionality increases, all points begin to appear almost equidistant from one another.*

The curse of dimensionality severely impacts the clustering process in high-dimensional spaces. With increasing dimensionality, clusters have to be increasingly tight in order to be distinct and learned reliably. In practice, there is a significant measurement noise in each dimension and this might prevent reliable clustering.

Density estimation is a type of learning problem wherein the goal is to estimate a function using a finite number of training samples. The finite number of training samples implies that any estimate of an unknown function is inaccurate (biased). Meaningful estimation is possible only for sufficiently smooth functions where the function smoothness is measured with respect to sampling density of the training data. A consequence of the curse of dimensionality is that, for high-dimensional spaces, it becomes difficult to obtain enough samples to attain this high density. This can be better understood from the following properties of high-dimensional spaces (pages 60-65 in [13]).

1. *Sample sizes yielding the same density increase exponentially with dimension.* If a sample containing n points is considered dense in $\mathfrak{R}^1$, n^d points are required in $\mathfrak{R}^d$ to achieve the same density.

2. *A large radius is needed to enclose a fraction of the data points in a high-dimensional space.* Consider points taken from a uniform distribution on a $d-$dimensional unit hypercube. The edge length of another hypercube (inside the first) that encloses a fraction p of samples is given by $e_d(p) = p^{1/d}$. For capturing 10% of samples in a 10-dimensional space, the edge length is $e_{10}(0.1) = 0.80$. This shows that large neighborhoods are required to capture even small portions of the data.

3. *Almost every point is close to an edge of a distribution than to another point.*

4. *Almost every point is an outlier in its own projection.* To someone standing at a sample from a high-dimensional distribution, all other samples appear distant and clumped near the center of the distribution.

Detailed illustrations of the above properties can be found in [13]. Properties 1 and 2 show the difficulty in making local estimates for high-dimensional samples. Properties 3 and 4 indicate the difficulty in predicting a response at a given point, since any point will on average be closer to an edge than to a training data point and thus require extrapolation by the learning machine.

Apart from the above issues, using high-dimensional features implies a higher computational expense than is necessary in many cases. Often the goal is to match a subset of feature dimensions, for example, to compute texture similarity, or the presence of a pattern, at a particular scale. In this case, the other dimensions need not be computed. Moreover, if the entire analysis process takes place in a high-dimensional space, dissimilarities in some dimensions might overshadow similarities in others. If similarity between two textures is always computed as a

distance between high-dimensional points, it is not possible to infer similarities at individual scales. This is important because several textures respond to filters at characteristic scales and give arbitrary responses at other scales.

6.2 Quasi-periodic Texture Motifs

In Chapter 4, we defined the term *texture motifs* of an object as spatially recurrent patterns that are characteristic of the object. If the spatial recurrence of these patterns is nearly periodic, we term these *quasi-periodic texture motifs*. For example, consider a harbor instance in Fig. 6.1. As shown in the figure, the harbor contains two quasi-periodic texture motifs, one corresponding to boats moored side by side and the other corresponding to periodic rows of moored boats separated by water.

A quasi-periodic texture motif has a set of characteristic frequencies and directionalities. These can be detected by using Gabor filters with appropriate frequencies and orientations. Near-periodic patterns strongly respond to Gabor filters with frequencies and orientations close to their own. Thus, when such patterns are analyzed using a filter bank (such as (3.4)), they give rise to strong peaks in the filter output magnitudes at their characteristic scales and orientations. Therefore, information from a few scales and orientations is all that is needed for describing and detecting quasi-periodic texture motifs. It is not necessary to use high-dimensional features such as (3.6), which capture the "amount" of texture at all scales and orientations in the filter bank.

The detection of quasi-periodic texture motifs involves the detection of regions

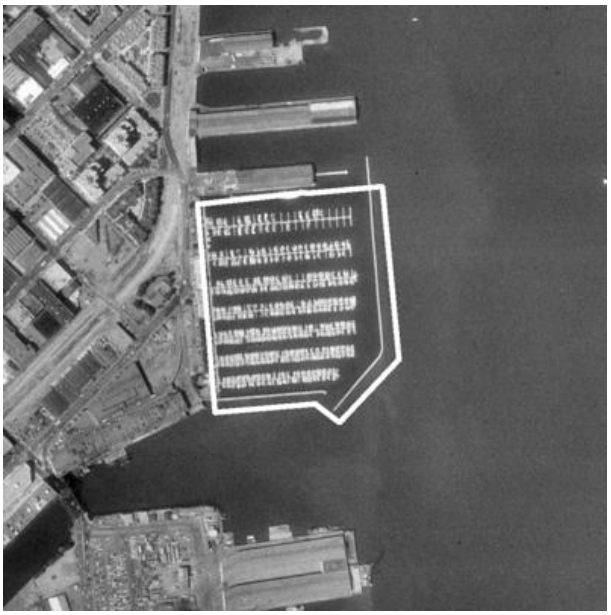
that strongly respond to Gabor filters at appropriate scales and orientations. In other words, we need to *focus attention* on regions that contain patterns at characteristic scales and orientations. For a given scale and orientation, the *focus-of-attention* mechanism is implemented as system with a binary response at each pixel. The response of the system at a pixel is 1 (true) if a "high" filter output is obtained at that pixel. Otherwise, the response is 0 (false). This mechanism ascertains the *presence* or *absence* of a motif at each pixel. The following section describes the implementation of this system.

6.3 Focus-of-Attention Mechanism

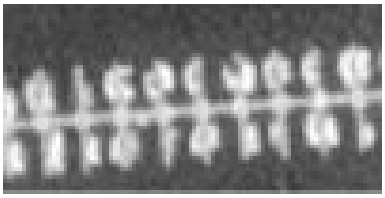
The focus-of-attention mechanism is implemented as a system, $B_{s,k}(x, y)$, with a binary textural response at pixel (x, y). $B_{s,k}(x, y) = 1$ if the neighborhood of the pixel contains a strong periodic pattern at scale s and orientation k. The input to the system is $F_{s,k}(x, y)$ in (3.5), i.e. the convolution of image I with a Gabor filter (from a filter bank) at a particular scale and orientation. Since the input values are usually continuous, a thresholding operation has to be performed for converting them to binary responses.

The general idea behind the thresholding is that $B_{s,k}(x, y) = 1$, if and only if $F_{s,k}(x, y)$ is significantly higher than "normal." Thus, if $F_{s,k}(x, y)$ exceeds a threshold value, it indicates the presence of a strong pattern at pixel (x, y). In other words,

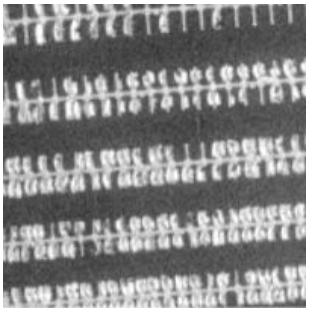
$$B_{s,k}(x, y) = \begin{cases} 1, \text{if } F_{s,k}(x, y) > t \\ 0, \text{else.} \end{cases} \tag{6.1}$$



(a)



(b)



(c)

Figure 6.1: (a) A harbor instance with a white border specifying the object region; The quasi-periodic texture motifs of harbors are (b) boats moored side by side, and (c) periodic rows of boats separated by water (bottom).

While there may be several ways of arriving at a threshold t, we propose the following two methods.

Thresholding Method 1:

In this method, $B_{s,k}(x,y) = 1$ iff $F_{s,k}(x,y) \gg \mathrm{E}\left[F_{s,*}(x,y)\right]$, where $\mathrm{E}\left[F_{s,*}(x,y)\right]$ is the expected output at any pixel at scale s considering all orientations. In other words,

$$B_{s,k}(x,y;\beta) = \begin{cases} 1, \text{if } F_{s,k}(x,y) > \beta \mathrm{E}\left[F_{s,*}(x,y)\right] \\ 0, \text{else.} \end{cases} \tag{6.2}$$

The expected values are computed in practice as means over a set of training images, which are chosen to have enough diversity to cover most patterns. If we assume that the image under analysis is diverse enough, the means can be computed over the image instead of a training set. The β's need not be the same for all scales and can be tuned toward a given task.

Thresholding Method 2:

In this method, $B_{s,k}(x,y;t_p) = 1$ iff $\mathrm{cdf}\left(F_{s,k}(x,y)\right) > t_p$, where cdf stands for the cumulative distribution function, and t_p is a user-specified percentile cut-off. For example, $t_p = 0.95$ if we want to retain only pixels with $F_{s,k}(x,y)$ greater than that of 95% of all pixels. As in the previous case, the cdf can be approximated over a training set or over the image under analysis.

In our experiments, we choose to use the first method because of its relative computational simplicity. The effect of thresholding on attentional focus is illustrated in Fig. 6.3. As shown in Fig. 6.2, two Gabor filters are chosen at frequencies and orientations close to that of the two patterns in Fig. 6.1(b) and Fig. 6.1(c). The outputs obtained by convolving these filters with the image in Fig. 6.1(a)

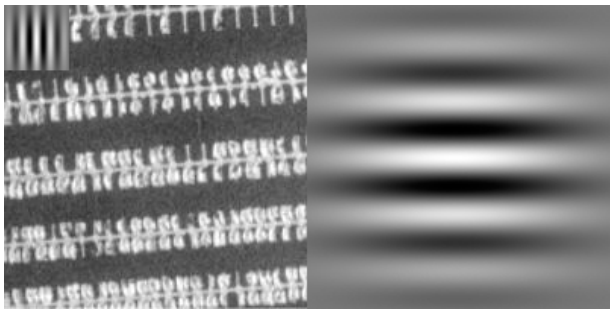


Figure 6.2: Two Gabor filters are shown, one at the top left corner and the other on the right half. These are chosen at frequencies and orientations close to that of the two near-periodic patterns, namely boats parked side by side and rows of boats (Fig. 6.1(b,c)).

are shown in Fig. 6.3(a) and Fig. 6.3(b). Figures 6.3(c-f) show the effect of β in (6.2) on the focussing mechanism. As β increases, higher outputs and therefore patterns with stronger periodicity are isolated.

6.4 The Model Framework

As mentioned earlier, a quasi-periodic texture motif of an object is parametrized by one or more characteristic scales and orientations. Suppose we perform the analysis using a filter bank with S scales and K filter orientations as given by (3.4). Let a row of parked cars in a parking lot form a near-periodic pattern at a certain scale s and orientation[1] k. Then $\mathcal{T}(s,k)$ denotes the "row of cars" texture motif of the "parking lot" object. For detecting parking lots, it helps to focus attention on regions where $B_{s,k}(x,y) = 1$, since these are likely to contain $\mathcal{T}(s,k)$. In general, a texture motif may be parametrized by more than one scale and orientation, and is denoted as $\mathcal{T}(\{(s_i,k_i)\})$, where $\{(s_i,k_i)\}$ are the

[1] k is actually the orientation index of the filter. The actual orientation (in radians) is given by $\theta = k\pi/K$. Also note that lower scales s correspond to higher filter frequencies.

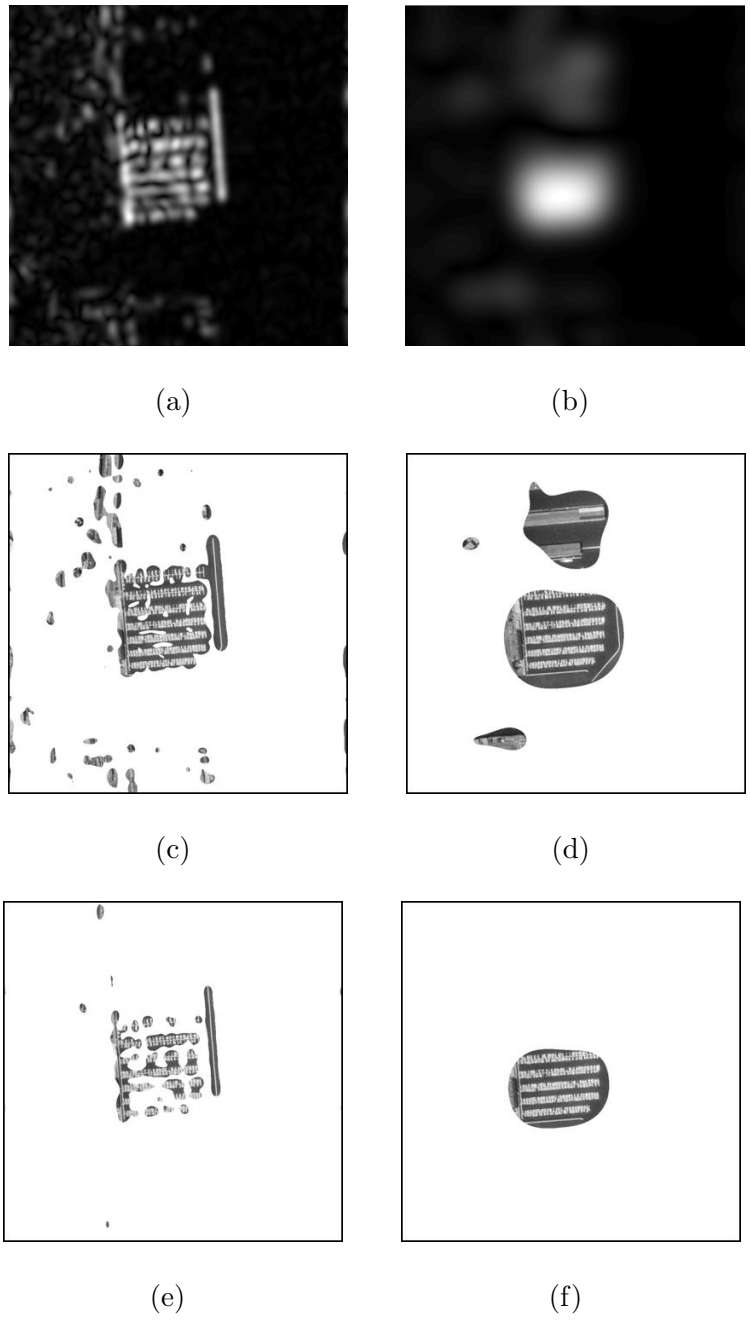


Figure 6.3: The top row shows the outputs obtained by convolving the two filters in Fig. 6.2 with the image in Fig. 6.1(a). The middle row shows the result of thresholding the outputs by the application of (6.2) with $\beta = 3$. The bottom row shows the thresholding result with $\beta = 6$. Note that as β increases, higher outputs and therefore patterns with stronger periodicity are isolated.

set of corresponding scales and orientations. We denote the set of texture motifs corresponding to an object as

$$\mathbb{T} = \{\mathcal{T}_i\left(\{(s_{ij}, k_{ij})\}\right); i = 1, \ldots, N_{\mathcal{T}}\}, \tag{6.3}$$

where $N_{\mathcal{T}}$ is the number of texture motifs. Each texture motif $\mathcal{T}_i$ has a corresponding set $\{(s_{ij}, k_{ij})\}$ of characteristic scales and orientations. To achieve rotation invariance[2], k_{ij} are not specified in an absolute manner. Rather relations between them are specified.

It is best to illustrate the modeling framework with an example object. Consider the harbor example in Fig. 6.1(a). Figures 6.1(b) and 6.1(c) show two quasi-periodic texture motifs of the harbor. The first corresponds to boats moored side by side and the second to periodic rows of boats separated by water. Furthermore, the boats are moored in a direction perpendicular to the rows of boats. In notation, the two motifs are represented as $\mathcal{T}_1\left(s_1, k_1\right)$ and $\mathcal{T}_2\left(s_2, k_2\right)$. Since $\mathcal{T}_1$ and $\mathcal{T}_2$ are perpendicular, we have

$$\left|\frac{k_1\pi}{K} - \frac{k_2\pi}{K}\right| = \frac{\pi}{2} \quad \Rightarrow |k_1 - k_2| = \frac{K}{2}. \tag{6.4}$$

Furthermore, since the separation between rows of boats is much larger than that between individual boats, we have $s_1 < s_2$.

Given the constraints of the problem $|k_1 - k_2| = K/2$ and $s_1 < s_2$, the model learning process comprises the generation of plausible hypotheses for (s_1, k_1) and (s_2, k_2). By allowing humans to specify such constraints, it is possible to greatly reduce the combinatorial learning complexity and obtain more intuitive and accurate results.

[2]The aerial images are assumed to be scale-normalized. This is feasible since the ground resolution of the images is usually known.

6.4.1 Detection Algorithm

We shall first discuss the detection algorithm, since it is made use of in the learning phase as well. Different objects have different constraints and numbers of motifs. Instead of devising a complicated general detection algorithm, we shall describe a simple one applied to harbors alone. Following a similar approach, detection algorithms can devised for objects with different constraints. In the case of harbors, we have two texture motifs $\mathcal{T}_1(s_1, k_1)$ and $\mathcal{T}_2(s_2, k_2)$, with the constraints $|k_1 - k_2| = K/2$ and $s_1 < s_2$.

The idea behind the detection scheme is thus. Let us assume a value for β in (6.2). The harbor object contains regions[3] with $B_{s_1,k_1}(x, y; \beta) = 1$ and $B_{s_2,k_2}(x, y; \beta) = 1$ at some proximity (p) with each other (perhaps overlapping). Therefore, if we find a pair of regions in an image with proximity p, one with $B_{s_1,k_1}(x, y; \beta) = 1$ and the other with $B_{s_2,k_2}(x, y; \beta) = 1$, such that $s_1 < s_2$ and $|k_1 - k_2| = K/2$, then the pair is likely to lie inside a harbor object. Note that as β is decreased and p is increased, more regions are likely to be hypothesized as harbors.

Let R_1 and R_2 be two regions in an image. Let us define their proximity $P(R_1, R_2)$ as the distance between their closest pixels. In order to determine if $P(R_1, R_2) < p$, we employ the following method. Let $D_m(R)$ denote a morphological m-pixel dilation of region R. If the dilated regions $D_{p/2}(R_1)$ and $D_{p/2}(R_2)$ overlap (share pixels), it means that the distance between R_1 and R_2 is less than p, i.e. $P(R_1, R_2) < p$. The morphological m-pixel dilation is a quick way of verifying

[3]In the detection framework, a *region* shall refer to a connected group of pixels. When we refer to "a region R with $B_{s,k}(x, y; \beta) = 1$," we mean that $B_{s,k}(x, y; \beta) = 1\ \forall (x, y) \in R$

closeness between regions as well as patching gaps between them.

Algorithm 6.1 describes a method for detecting harbors. The input is an image $O(x, y)$ and the output is an indicator function $I(x, y; \beta)$ that is 1 if (x, y) is detected as being inside the object, and 0 if it is not (for a given β). We denote by $I(R; \beta) = 1$, the operation $I(x, y; \beta) = 1 \ \forall (x, y) \in R$. Note that our aim here is not to provide a precise segmentation but to detect regions where there is a high confidence of a harbor being present. The extracted regions may be further processed for verification or precise segmentation.

6.4.2 Generating Model Hypotheses

We start with a training set of N_o example images containing the object, given by

$$\mathcal{O} = \{O_1, O_2 \ldots, O_{N_o}\}. \tag{6.5}$$

For each training image, the region corresponding to the object is manually specified in the form of a mask (the interior of the white border in Fig. 6.1(a)). For a training image O_i, the mask M_i is a binary image given by

$$M_i(x, y) = \begin{cases} 1, \text{if } (x, y) \text{ lies inside the object} \\ 0, \text{else.} \end{cases} \tag{6.6}$$

The set of corresponding object masks is given by

$$\mathcal{M} = \{M_1, M_2 \ldots, M_{N_o}\}. \tag{6.7}$$

Suppose we apply a detection algorithm such as Algorithm 6.1 on a training image O_i. Let $I_i(x, y; \beta)$ be the resulting indicator function after detection. Then

Algorithm 6.1 Harbor Detection

Input image $O(x, y)$

Output indicator function $I(x, y; \beta)$

Given parameters s_1, s_2, β (in (6.2)) and proximity p

$D_m(R)$: Morphological m-pixel dilation of a connected region R

$P(R_1, R_2)$: Proximity between regions R_1 and R_2

for $k_1 = 1$ to K **do**

  $k_2 = (k_1 + \frac{K}{2}) \mod K$

  R_{1i}: all connected regions with $B_{s_1,k_1}(x, y) = 1$

  R_{2j}: all connected regions with $B_{s_2,k_2}(x, y) = 1$

  for all i **do**

    for all j **do**

      if $P(R_{1i}, R_{2j}) < p$ **then**

        $I(D_{p/2}(R_{1i}); \beta) = 1$

        $I(D_{p/2}(R_{2j}); \beta) = 1$

      end if

    end for

  end for

end for

the precision $\mathcal{P}_i(\beta)$ and recall $\mathcal{R}_i(\beta)$ of detection is given by

$$\mathcal{P}_i(\beta) = \frac{\sum_j M_i(x_j, y_j) I_i(x_j, y_j; \beta)}{\sum_j I_i(x_j, y_j; \beta)}, \text{ and} \tag{6.8}$$

$$\mathcal{R}_i(\beta) = \frac{\sum_j M_i(x_j, y_j) I_i(x_j, y_j; \beta)}{\sum_j M_i(x_j, y_j)}, \tag{6.9}$$

where (x_j, y_j) are indexed over all the pixels in training image O_i. The precision and recall over the whole training set are given by

$$\mathcal{P}(\beta) = \frac{\sum_i \mathcal{P}_i(\beta)}{N_o}, \text{ and} \tag{6.10}$$

$$\mathcal{R}(\beta) = \frac{\sum_i \mathcal{R}_i(\beta)}{N_o}. \tag{6.11}$$

Given $\mathcal{P}(\beta)$ and $\mathcal{R}(\beta)$, the F-measure $\mathcal{F}_\alpha(\beta)$ over the training set is given by (4.20), where α is the relative weight placed on precision over recall.

The idea behind the model learning process is to learn the hypotheses that maximize $\mathcal{F}_\alpha(\beta)$ for given α. In the case of harbors, a hypothesis includes scale s_1 for the first motif, s_2 for the second motif, and β. Given a training set $\mathcal{O}$ and the set of masks $\mathcal{M}$, Algorithm 6.2 is used to compute the best hypothesis $\mathbb{H}_\alpha = (s_1, s_2, \beta)$. Note that the constraint $s_1 < s_2$ is tightened to $s_1 \leq s_2 + s_{\text{diff}}$, since it is observed that the frequencies of the two motifs have quite a large difference.

6.5 Experiments

Harbors are a suitable object for demonstration of object detection by means of quasi-periodic texture motifs. We use a dataset of ten harbor instances ($N_o = 10$). Fig. 6.4 shows the training set from which we learn the model hypothesis $\mathbb{H}_\alpha$ using Algorithm 6.2.

Algorithm 6.2 Harbor hypotheses generation

Given parameter α of the F-measure $\mathcal{F}_\alpha(\beta)$
Given training set $\mathcal{O}$ and the set of masks $\mathcal{M}$
$\mathbb{H}_\alpha \leftarrow \phi$
$F_{\max} \leftarrow 0$
for $\beta = \beta_{\min} : \beta_{\text{step}} : \beta_{\max}$ **do**
  for $s_1 = 0$ to $S - s_{\text{diff}} - 1$ **do**
    for $s_2 = s_1 + s_{\text{diff}}$ to $S - 1$ **do**
      if $\mathcal{F}_\alpha(\beta) > F_{\max}$ **then**
        $F_{\max} \leftarrow \mathcal{F}_\alpha(\beta)$
        $\mathbb{H}_\alpha \leftarrow (s_1, s_2, \beta)$
      end if
    end for
  end for
end for

The Gabor filter bank (in (3.4)) used for analysis has $S = 10$ scales and $K = 12$ orientations. The frequency range of the filters is 0.02 cycles/pixel at the highest scale ($s = 9$) to 0.25 cycles/pixel at the lowest ($s = 0$). In our experiments, the learning parameters of Algorithm 6.2 are as follows: $\beta_{\min} = 4$, $\beta_{\text{step}} = 1$, $\beta_{\max} = 10$, and $s_{\text{diff}} = 4$. The proximity parameter p is Algorithm 6.1 is set to 80 pixels. Since the Gabor filter kernel is a square with a side of 365 pixels, our analysis excludes a border of half that size (182 pixels) around each test image. This is to ensure that the kernel always lies entirely inside the image.

The model learning process using Algorithm 6.2 with $\alpha = 2$ gives the hypothesis $\mathbb{H}_2 = (1, 8, 5)$. This means that the texture motifs of the harbor are learned to be $\mathcal{T}_1(1, k_1)$ and $\mathcal{T}_2(8, k_2)$, such that $|k_1 - k_2| = K/2$. With $\alpha = 10$, we get the hypothesis $\mathbb{H}_{10} = (0, 9, 7)$. Fig. 6.5 and Fig. 6.6 show the harbor detection results for different α. The test images are shown in the left column with a white border around the desired object. The middle column shows the detection results for $\alpha = 2$ and the right column for $\alpha = 10$. Note that with higher α, there are fewer false alarms but more missed regions.

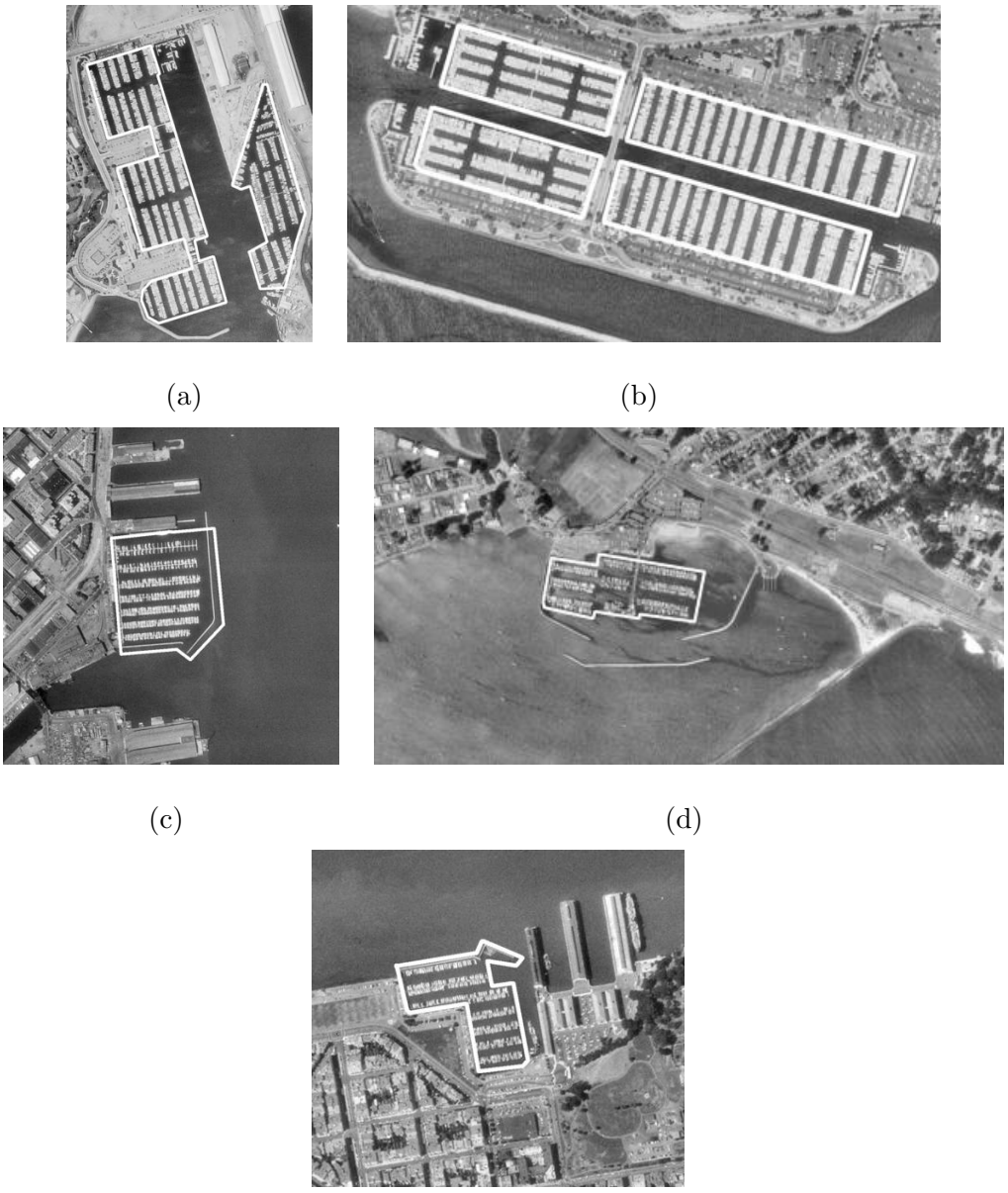
Fig. 6.7 and Fig. 6.8 show detection results from running Algorithm 6.1 on larger geospatial images containing several harbors. Observe that most of the significant harbor regions are correctly extracted. With $\alpha = 2$, a few spurious regions are detected as harbors. The morphological dilation and texture neighborhood effects may account for some of the spurious regions just around the harbors. These could be removed by post-processing. Fig. 6.9 is a geospatial image intended to test whether the algorithm is fooled by an image with no harbor regions. The algorithm performs favorably at $\alpha = 10$ and detects almost no

harbor regions. A small false-alarm region is detected, which may be removed on account of its size.

6.6 Summary of Contribution

In this chapter, we discuss the detection of objects with *quasi-periodic* texture motifs. These are texture motifs that contain nearly periodic patterns at one or more scales and orientations. To detect such objects, it is sufficient to perform analysis using Gabor filters at characteristic scales and orientations. Therefore, it is possible to avoid high-dimensional features that capture the "amount" of texture using a filter bank of several scales and orientations.

The modeling and detection framework proposed in this chapter deviates from the high-dimensional frameworks presented in the previous chapters, while still preserving the central notion of texture motifs. Methods are presented for learning relations between the characteristic scales and orientations of a texture motif from object examples. This constitutes the modeling phase, whose output is a set of textural *rules* or *hypotheses* characterizing the motif. The learning framework also makes it possible for humans to specify constraints that reduce the combinatorial learning complexity. The object detection phase involves searching for regions that obey these textural rules. Experimental results on the harbor object demonstrate the effectiveness of this approach in detecting objects with quasi-periodic texture motifs.



(a) (b)

(c) (d)

(e)

Figure 6.4: The instance dataset for the *harbor* object. The borders of the object regions (masks) are indicated in white.

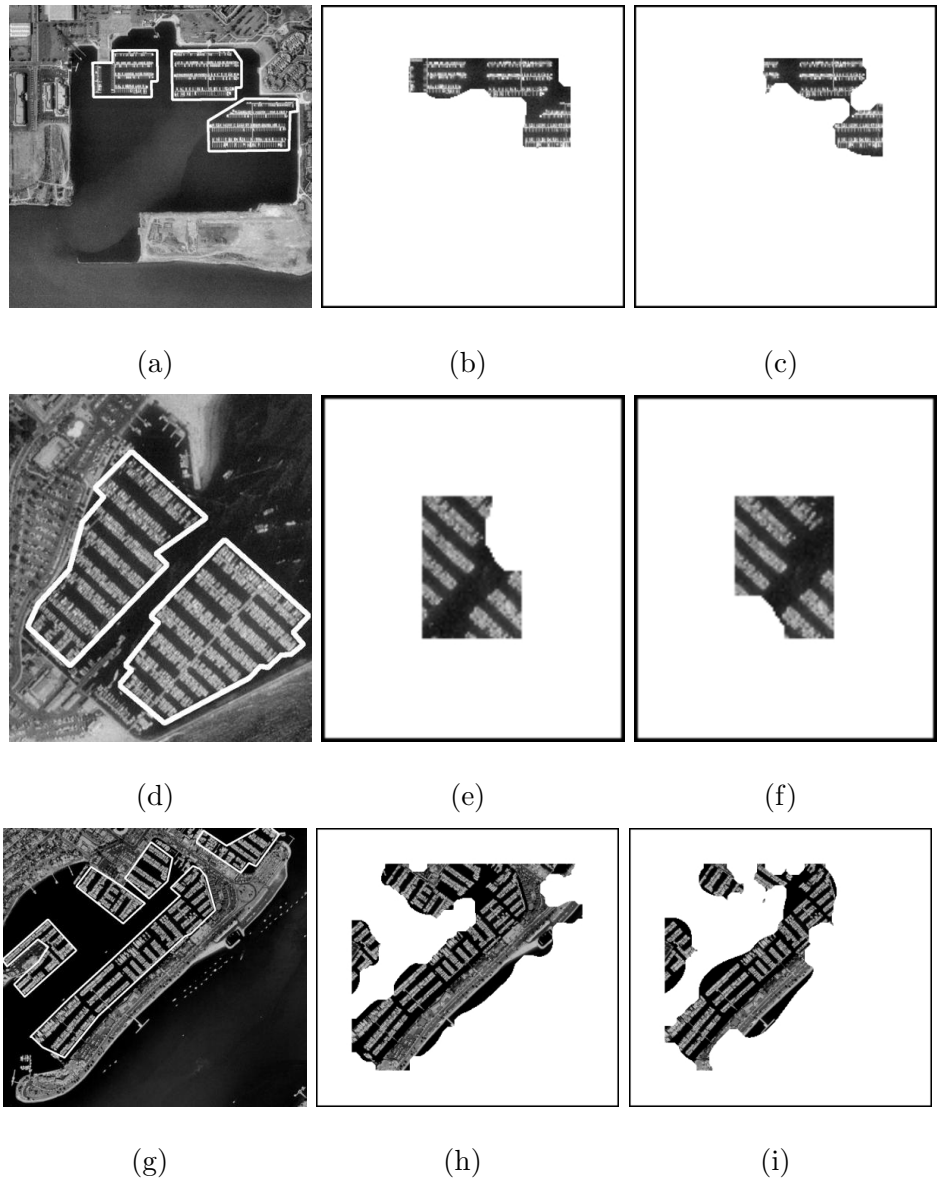
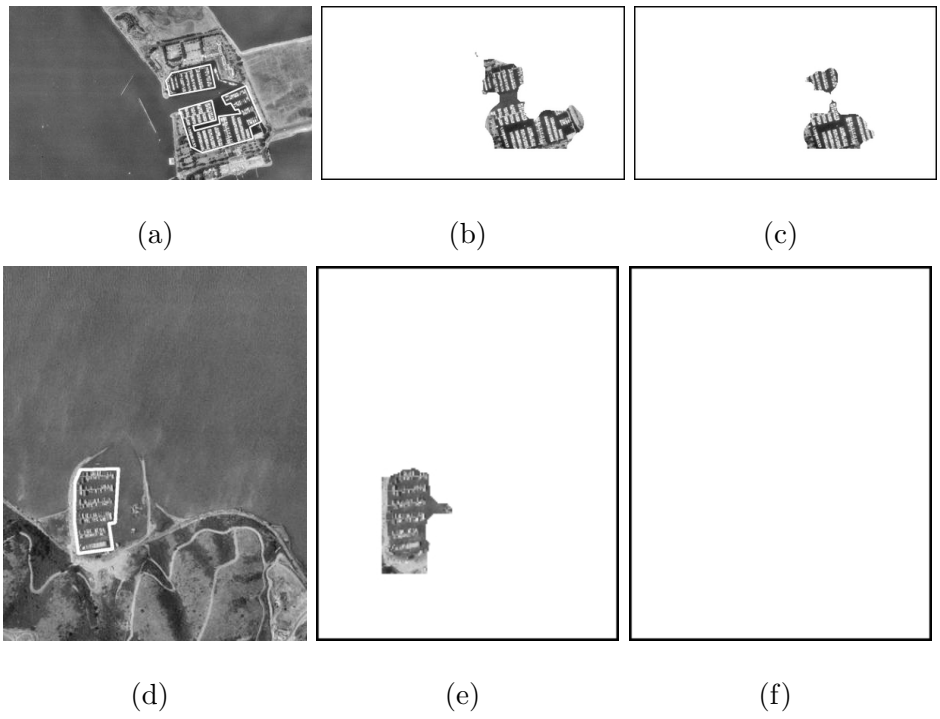


Figure 6.5: Harbor detection using Algorithm 6.1 using the best hypothesis learned from Algorithm 6.2. The left column shows the test image with a white border around the desired object. The middle column shows the detected regions with $\alpha = 2$ ($\mathbb{H}_2 = (1, 8, 5)$). The right column shows the detected regions with $\alpha = 10$ ($\mathbb{H}_{10} = (0, 9, 7)$). Note that with higher α, there are fewer false alarms but more missed regions.

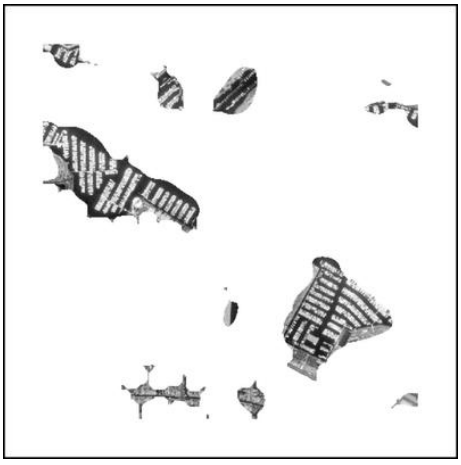


(a) (b) (c)

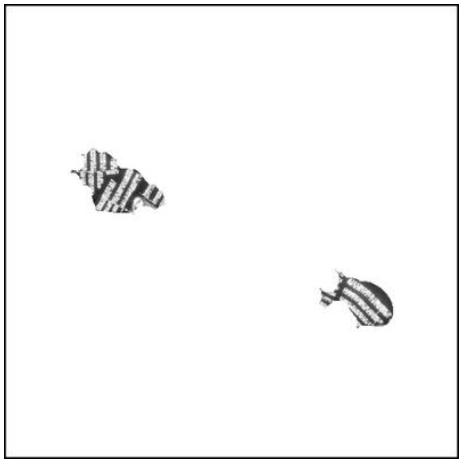
(d) (e) (f)

Figure 6.6: Continuation of Fig. 6.5.

(a)



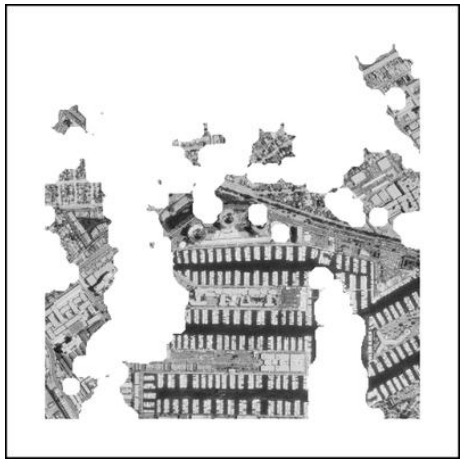
(b)



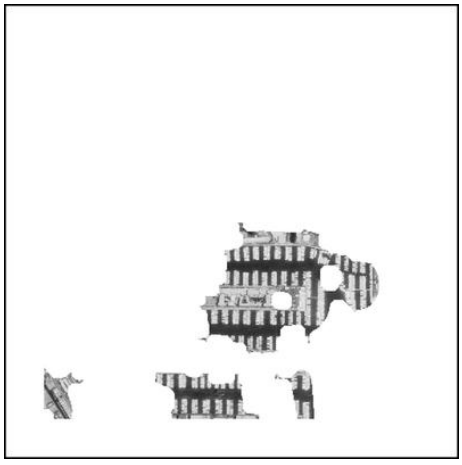
(c)

Figure 6.7: (a) A large geospatial image from which we desire to extract harbors; (b) The harbor regions extracted from (a) with the application of Algorithm 6.1 with $\alpha = 2$ ($\mathbb{H}_2 = (1, 8, 5)$); and (c) The harbor regions extracted with $\alpha = 10$ ($\mathbb{H}_{10} = (0, 9, 7)$).

(a)



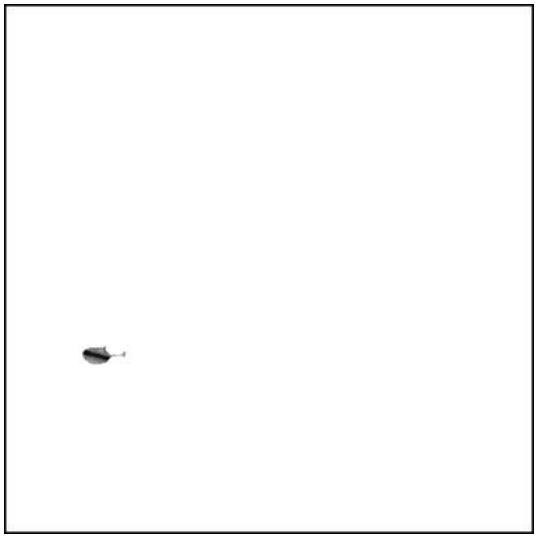
(b)



(c)

Figure 6.8: (a) A large geospatial image from which we desire to extract harbors; (b) The harbor regions extracted from (a) with the application of Algorithm 6.1 with $\alpha = 2$ ($\mathbb{H}_2 = (1, 8, 5)$); and (c) The harbor regions extracted with $\alpha = 10$ ($\mathbb{H}_{10} = (0, 9, 7)$).

(a)



(b)

Figure 6.9: The detection algorithm does not get fooled by this geospatial image that contains no harbor regions. The algorithm performs favorably at $\alpha = 10$ and detects almost no harbor regions. A small false-alarm region is detected, which may be removed on account of its size.

Chapter 7

Conclusion

This dissertation focusses on filling two important gaps in enabling the detection of *compound* objects in geospatial images. These are 1) the exploitation of the potential of frequency-domain texture analysis, and 2) example-based learning of appearance models for objects. The detection of geospatial objects with simple geometric or shape models such as buildings, roads, vehicles, etc., has been explored adequately in the literature. This is not the case for compound objects, such as harbors and golf courses, characterized by several "parts" and their spatial layout. Furthermore, appearance models (prior knowledge about object appearance) for compound geospatial objects have traditionally been encoded a priori into the system by humans and not learned from object examples.

In previous work, compound objects are described using mainly spatial (pixel intensity) domain methods, such as image segmentation, edge detection/linking, and graphical modeling. However, purely spatial methods are crippled by several obstacles, including 1) the large number of parts in compound geospatial objects,

2) the variation in structural relations among parts from one object instance to another, and 3) the large dimensions (in pixels) of geospatial images. By incorporating information from the frequency domain (fourier spectrum), it is possible to mitigate these obstacles and make object detection and model learning feasible.

In this dissertation, methods that perform analysis in both spatial and frequency domains are proposed for modeling and detecting objects in geospatial images. These methods exploit joint space-frequency localization techniques developed in the framework of texture analysis. In particular, Gabor filter-based texture analysis is shown to provide a compact description of the visual structure in objects and gracefully handle variations in their appearance. Despite this fact, there is relatively little use of the efficient methods of frequency-domain texture analysis for detecting geospatial objects.

We propose the use of *texture motifs* for modeling and detecting geospatial objects. A texture motif of an object is a spatially recurrent pattern characteristic of the object. Thus, the pattern formed by boats and water is a texture motif of a harbor, and the arrangement of trees and grass is a texture motif of a golf course. Three approaches are proposed in this dissertation for learning texture motif representations from object examples and detecting objects based on the learned models. Experimental results demonstrate the effectiveness of these approaches in detecting compound geospatial objects.

In the first approach, a two-layered model is proposed for representing texture motifs. Using low-level texture features based on Gabor filters, the first layer learns the local intensity variations in the motif that form "textural elements"

such as flat areas, bars, edges, and so on. This is done using a semi-supervised statistical approach that accounts for the different possible orientations of the texture elements. The second layer of the representation is the spatial distribution of low-level texture elements in the motif, since this influences its distinct visual appearance. A Gaussian mixture model (GMM) for this is learned from examples using features derived from histograms of texture elements in spatial neighborhoods. Confidence measures generated using this model are then used for detecting object presence.

The second approach adopts a hidden Markov modeling (HMM) framework for learning adjacency relations between elements. One-dimensional HMMs are used for modeling sequences of texture features sampled along a line (in a certain direction) in the object. The texture elements of the motif are learned as the states of the HMM, and the state transitions from one pixel to the next describe the spatial arrangement of the elements. In this approach, it is possible to combine the two layers of the previous approach into a single stage of learning.

In the third approach, we address the detection of objects with *quasi-periodic* texture motifs, e.g. harbors. When a texture motif comprises nearly periodic patterns at one or more scales and orientations, it is possible to avoid representing it in a high-dimensional space. Texture motifs are viewed as patterns that respond strongly to Gabor filters at certain scales and orientations. Objects are detected by identifying regions that obey certain "textural rules," which are in the form of relations involving characteristic scales, orientations, and proximity of texture motifs. These rules can be fully user-specified or learned from examples with user-specified constraints. This paradigm has the advantage of avoiding the issues

related to clustering and density estimation in high-dimensional spaces.

7.1 Future Directions

For robust object detection, it is necessary to account for variations in the scale and orientation of the object. The object should be detected irrespective of its orientation and size in the parent image. While the methods proposed in this dissertation do handle variations in orientation, scale invariance is still an outstanding issue. The scale of a geospatial object is determined by the ground resolution[1] of the parent image, which is usually specified. Therefore, the scale invariance problem can be handled by extracting features using Gabor filters whose scale parameters are tied to the ground resolution of the image. Suppose a feature is extracted with a Gabor filter of frequency 0.1 cycles/pixel (period 10 pixels) from an image with ground resolution 1 m/pixel. In order to extract the same feature from an image with ground resolution 2 m/pixel, a Gabor filter of frequency 0.2 cycles/pixel (period 5 pixels) needs to be used. By computing feature vectors with a range of scales proportional to the ground resolution, scale invariant object detection can be achieved.

In Chapter 4, the constituent elements in a texture motif are learned as the first layer of its representation. Ideally, a global set of texture elements should be learned across all objects since elements are often shared among texture motifs. However, in this work the elements are learned separately for each object. This is to ensure that the dimensionality of the spatial histogram feature in (4.12) is

[1]The ground resolution of a geospatial image is the number of meters on the ground covered by one image pixel, e.g. one meter/pixel.

low. A useful project is to build a base of texture elements across all objects. Given a texture motif, a subset of the "most relevant" elements could be chosen for building the second layer of its representation.

The issue of utilizing selected scales of the texture feature vector of (3.6) was brought up in Chapter 5. However, the problem of automatically selecting the optimal set of scales for detecting an object has not been addressed in this work. Various goal-driven feature selection methods such as projection pursuit [33] have been studied in the literature. It is worthwhile to apply some of these results toward boosting the accuracy of object detection.

In Chapter 5, one-dimensional hidden Markov models (HMMs) were used for learning texture motif representations. The framework of the one-dimensional HMM was adopted for the sake of computational simplicity. However, such a model overlooks the two-dimensional nature of texture motifs. With the appropriate computational resources, it should be possible to explore the application of more complex two-dimensional HMM frameworks [40, 20] to the object detection problem.

Finally, it should be observed that though texture is an important feature in object detection, it is by no means the only one. The combination of texture with other features, such as color and shape, could increase the robustness of object detection. Knowledge-guided segmentation schemes [75, 74] could be explored as a means of combining different features and models, with the goal of improving both the reliability and precision of object detection.

Bibliography

[1] E. P. Baltsavias, A. Gruen, and L. V. Gool, editors. *Automatic extraction of man-made objects from aerial and space images (III)*. A. A. Balkema, 2001.

[2] S. Berberoglu, C. Lloyd, P. Atkinson, and P. Curran. The integration of spectral and textural information using neural networks for land cover mapping in the Mediterranean. *Computers & Geosciences*, 26:385–396, 2000.

[3] S. Bhagavathy, J. Tešić, and B. S. Manjunath. On the Rayleigh nature of Gabor filter outputs. In *Proceedings of the International Conference on Image Processing*, volume 2, pages 745–748, September 2003.

[4] S. Bhagavathy, S. Newsam, and B. S. Manjunath. Modeling object classes in aerial images using texture motifs. In *Proceedings of the International Conference on Pattern Recognition*, August 2002.

[5] S. Blott and R. Weber. A simple vector-approximation file for similarity in high-dimensional vector spaces. Technical report, Institute for Information Systems, ETH Zentrum, Zurich, Switzerland, March 1997.

[6] R. N. Braithwaite and B. Bhanu. Hierarchical Gabor filters for object detection in infrared images. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 628–631, 1994.

[7] P. Brivio, I. Doria, and E. Zilioli. Aspects of spatial autocorrelation of Landsat TM data for the inventory of waste-disposal sites in rural environments. *Photogrammetric Engineering & Remote Sensing*, 59(9):1377–1382, 1993.

[8] P. Brodatz. *Textures: A photographic album for artists and designers*. Dover, New York, U.S.A., 1966.

[9] L. Bruce, H. Tamhankar, A. Mathur, and R. King. Multiresolutional texture analysis of multispectral imagery for automated ground cover classification.

In *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium*, volume 1, pages 312–314, 2002.

[10] J. Carr. Spectral and textural classification of single and multiple band digital images. *Computers & Geosciences*, 22:849–865, 1996.

[11] C. Carson, M. Thomas, S. Belongie, J. M. Hellerstein, and J. Malik. Blobworld: A system for region-based image indexing and retrieval. In *Proceedings of the International Conference on Visual Information Systems*, pages 509–516, 1999.

[12] J. C.-W. Chan, R. S. DeFries, X. Zhan, C. Huang, and J. R. G. Townshend. Texture features for change detection at 250m resolution–an application of machine learning feature subset selection. In *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium*, volume VII, pages 3060–3062, 2000.

[13] V. Cherkassky and F. Mulier, editors. *Learning from data: concepts, theory, and methods*. John Wiley & Sons, 1998.

[14] M. Chica-Olmo and F. Abarca-Hernandez. Computing geostatistical image texture for remotely sensed data classification. *Computers & Geosciences*, 26:373–383, 2000.

[15] D. A. Clausi. Comparison and fusion of co-occurrence, Gabor and MRF texture features for classification of SAR sea-ice imagery. *Atmosphere-Ocean*, 39(3):183–194, 2001.

[16] I. Daubechies. The wavelet transform, time-frequency localization and signal analysis. *IEEE Transactions on Information Theory*, 36(5):961–1005, 1990.

[17] J. Daugman. Complete discrete 2D Gabor transform by neural networks for image analysis and compression. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36:1169–1179, 1988.

[18] J. David M. McKeown, W. A. Harvey, and L. E. Wixson. Automating knowledge acquisition for aerial image interpretation. *Computer Vision, Graphics and Image Processing*, 46:37–81, 1989.

[19] J. David M. McKeown, J. Wilson A. Harvey, and J. McDermott. Rule-based interpretation of aerial imagery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 7(5):570–585, September 1985.

[20] D. DeMenthon, D. Doermann, and M. V. Stückelberg. Image distance using hidden Markov models. In *Proceedings of the International Conference on Pattern Recognition*, volume 3, pages 143–146, 2000.

[21] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood estimation from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, Series B, 39(1):1–38, 1977.

[22] D. Dunn and W. E. Higgins. Optimal Gabor filters for texture segmentation. *IEEE Transactions on Image Processing*, 4(7):947–964, July 1995.

[23] L. V. Dutra, R. Huber, and P. Hernandez. Primary forest and land cover contextual classification using JERS-1 data in Amazonia, Brazil. In *Proceedings of the IEEE International Geoscience and Remote Sensing Symposium*, pages 2743–2745, 1998.

[24] S. E. Frankin, A. J. Maudie, and M. B. Lavigne. Using spatial co-occurrence texture to increase forest structure and species composition classification accuracy. *Photogrammetric Engineering & Remote Sensing*, 67(7):849–855, 2001.

[25] A. Gruen, E. P. Baltsavias, and O. Henricsson, editors. *Automatic extraction of man-made objects from aerial and space images (II).* Birkhauser Verlag, 1997.

[26] A. Gruen, O. Kubler, and P. Agouris, editors. *Automatic extraction of man-made objects from aerial and space images (I).* Birkhauser, Basel, 1995.

[27] B. Haack and M. Bechdol. Integrating multisensor data and RADAR texture measures for land cover mapping. *Computers & Geosciences*, 26:411–421, 2000.

[28] J. Hammersely and P. Clifford. Markov fields on finite graph and lattices. Unpublished manuscript, 1971.

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

[30] R. M. Haralick and K. S. Shanmugam. Combined spectral and spatial processing of ERTS imagery data. *Remote Sensing of Environment*, 3:3–13, 1974.

[31] M. Hauta-Kasari, J. Parkkinen, T. Jaaskelainen, and R. Lenz. Generalized co-occurrence matrix for multispectral texture analysis. In *Proceedings of the International Conference on Pattern Recogition*, pages 785–789, 1996.

[32] L. Hill, J. Frew, and Q. Zheng. Geographic names: The implementation of a gazetteer in a georeferenced digital library. *D-Lib Magazine*, January 1999.

[33] P. J. Huber. Projection pursuit. *The Annals of Statistics*, 13(2):435–475, June 1985.

[34] A. Huertas, W. Cole, and R. Nevatia. Detecting runways in complex airport scenes. *Computer Vision, Graphics, and Image Processing*, pages 107–145, 1990.

[35] A. Huertas and R. Nevatia. Detecting buildings in aerial images. *Computer Vision, Graphics and Image Processing*, 41(2):131–152, February 1988.

[36] R. B. Irvin and D. M. McKeown. Methods for exploiting the relationship between buildings and their shadows in aerial imagery. *IEEE Transactions on Systems, Man and Cybernetics*, 19(6):1564–1575, November 1989.

[37] A. K. Jain, N. K. Ratha, and S. Lakshmanan. Object detection using Gabor filters. *Pattern Recognition*, 30(2):295–309, February 1997.

[38] V. Karathanassi, C. Iossifidis, and D. Rokos. A texture-based classification method for classifying built areas according to their density. *International Journal of Remote Sensing*, 21(9):1807–1823, 2000.

[39] S. Leguizamon. Characterization of texture in remotely sensed images by using the wavelet transform. In *Proceedings of the International Symposium on High Mountain Remote Sensing Cartography*, pages 129–139, 1996.

[40] J. Li, A. Najmi, and R. M. Gray. Image classification by a two-dimensional hidden Markov model. *IEEE Transactions on Signal Processing*, 48(2):517–533, February 2000.

[41] S. Li. *Markov random field modeling in image analysis*. Springer-Verlag, New York, 2001.

[42] R. Lumia, R. M. Haralick, O. A. Zuniga, L. G. Shapiro, and T. C. Pong. Texture analysis of aerial photographs. *Pattern Recognition*, 16(1):39–46, 1983.

[43] W. Y. Ma and B. S. Manjunath. NeTra: a toolbox for navigating large image databases. *Multimedia Systems*, 7(3):184–198, May 1999.

[44] W.-Y. Ma and B. S. Manjunath. A texture thesaurus for browsing large aerial photographs. *Journal of the American Society for Information Science*, 49(7):633–48, May 1998.

[45] S. T. F. Mahmood. *Attentional selection in object recognition*. PhD thesis, MIT, Cambridge, 1993.

[46] R. Manduchi. A cluster grouping technique for texture segmentation. In *Proceedings of the International Conference on Pattern Recognition*, pages 1060–1063, 2000.

[47] B. S. Manjunath, P. Salembier, and T. Sikora, editors. *Introduction to MPEG7: Multimedia Content Description Interface*. John Wiley & Sons, 2002.

[48] B. S. Manjunath and W. Y. Ma. Texture features for browsing and retrieval of image data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(8):837–842, August 1996.

[49] S. Marcelja. Mathematical description of the responses of simple cortical cells. *Journal of the Optical Society of America*, 70(11):1297–1300, 1980.

[50] T. Matsuyama. Knowledge-based aerial image understanding systems and expert systems for image processing. *IEEE Transactions on Geoscience and Remote Sensing*, 25:305–316, May 1987.

[51] T. Matsuyama and V. Hwang. *SIGMA: A knowledge-based aerial image understanding system*. Advances in computer vision and machine intelligence, Plenum Press, 1990.

[52] M. Mueller and K. Segl. Object recognition based on high spatial resolution panchromatic satellite imagery. In *Joint workshop of ISPRS on Sensors and Mapping from Space*, 1999.

[53] M. Mueller, K. Segl, and H. Kaufmann. Edge- and region-based segmentation technique for the extraction of large, man-made objects in high-resolution satellite imagery. *Pattern Recognition*, 37:1619–1628, August 2004.

[54] S. Myint. Fractal approaches in texture analysis and classification of remotely sensed data: comparisons with spatial autocorrelation techniques and simple

descriptive statistics. *International Journal of Remote Sensing*, 24(9):1925–1947, 2003.

[55] M. Nagao and T. Matsuyama. *A structural analysis of complex aerial photographs*. Advanced Applications in Pattern Recognition, Plenum Publishing, 1980.

[56] S. Newsam and B. S. Manjunath. Normalized texture motifs and their application to statistical object modeling. In *CVPR Workshop on Perceptual Organization in Computer Vision (POCV)*, June 2004.

[57] S. Newsam, L. Wang, S. Bhagavathy, and B. S. Manjunath. Using texture to analyze and manage large collections of remote sensed image and video data. *Journal of Applied Optics: Information Processing*, 43(2):210–217, January 2004.

[58] S. Newsam, S. Bhagavathy, and B. S. Manjunath. Modeling object classes in aerial images using hidden Markov models. In *Proceedings of the International Conference on Image Processing*, September 2002.

[59] B. Nicolin and R. Gabler. A knowledge-based system for the analysis of aerial images. *IEEE Transactions on Geoscience and Remote Sensing*, 25(3):317–329, 1987.

[60] S. Noronha and R. Nevatia. Detection and modeling of buildings from multiple aerial images. *Pattern Analysis and Machine Intelligence*, pages 501–518, May 2001.

[61] H. Qiu, N. Lam, D. Quattrochi, and J. Gamon. Fractal characterization of hyperspectral imagery. *Photogrammetric Engineering & Remote Sensing*, 65(1):63–71, 1999.

[62] T. Quack, U. Monich, L. Thiele, and B. Manjunath. Cortina: A system for large-scale, content-based web image retrieval. In *ACM Multimedia*, October 2004.

[63] F. Quint. *Recognition of structured objects in monocular aerial images using context*, pages 213–228. Mapping buildings, roads and other man-made structures from images, Ed. F. Leberl. Mnchen, 1997.

[64] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[65] T. Ranchin, B. Naert, M. Albuisson, G. Boyer, and P. Astrand. An automatic method for vine detection in airborne imagery using wavelet transform and multiresolution analysis. *Photogrammetric Engineering & Remote Sensing*, 67(1):91–98, 2001.

[66] G. Rellier, X. Descombes, J. Zerubia, and F. Falzon. A Gauss-Markov model for hyperspectral texture analysis of urban areas. In *Proceedings of the International Conference on Pattern Recogition*, volume 1, pages 692–695, 2002.

[67] A. L. Reno and D. M. Booth. Using models to recognise man-made objects. In *IEEE Workshop on Visual Surveillance*, 1999.

[68] C. J. V. Rijsbergen. *Information retrieval*. Butterworths, second edition, 1979.

[69] C. Schmid. Constructing models for content-based image retrieval. In *Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pages 39–45, 2001.

[70] R. Schowenderdt. *Remote sensing: models and methods for image processing*. Academic Press, Chestnut Hill, MA, USA, second edition, 1997.

[71] M. Schroder, M. Walessa, H. Rehrauer, K. Seidal, and M. Datcu. Gibbs random field models: a toolbox for spatial information extraction. *Computers & Geosciences*, 26:423–432, 2000.

[72] J. Shufelt and D. M. McKeown. Fusion of monocular cues to detect man-made structures in aerial imagery. *Computer Vision, Graphics and Image Processing*, 57(3):307–330, May 1993.

[73] C. Smyrniotis and K. Dutta. A knowledge-based system for recognizing man-made objects in aerial images. In *Proceedings of the Conference on Computer Vision and Pattern Recognition*, pages 111–117, June 1988.

[74] B. Sumengen. *Variational image segmentation and curve evolution on natural images*. PhD thesis, University of California, Santa Barbara, Sep 2004.

[75] B. Sumengen, S. Bhagavathy, and B. S. Manjunath. Graph partitioning active contours for knowledge-based geospatial segmentation. In *Proceedings of the IEEE CVPR Workshop on Perceptual Organization in Computer Vision*, pages 54–54, June 2004.

[76] T. Wassenaar, J. Robbez-Masson, P. Andrieux, and F. Baret. Vineyard identification and description of spatial crop structure by per-field frequency analysis. *International Journal of Remote Sensing*, 23(17):3311–3325, 2002.

[77] J. S. Weska, C. R. Dyer, and A. Rosenfeld. A comparative study of texture measures for terrain classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 6(4):269–285, 1976.

[78] J. Weszka, C. Dyer, and A. Rosenfeld. A comparitive study of texture measures for terrain classification. *IEEE Transactions on Systems, Man, and Cybernetics*, 6(4):269–285, 1976.

[79] P. A. Wilson. Rule-based classification of water in Landsat MSS images using the variance filter. *Photogrammetric Engineering & Remote Sensing*, 63(5):485–491, 1997.

[80] P. Wu, B. Manjunath, S. Newsam, and H. Shin. A texture descriptor for browsing and similarity retrieval. *Journal of Signal Processing: Image Communication*, 16(1-2):33–43, September 2000.

[81] T. Yamazaki and D. Gingras. Image classification using spectral and spatial information based on MRF models. *IEEE Transactions on Image Processing*, 4(9):1333–1339, 1995.

[82] T. Yamazaki and D. Gingras. A contextual classification system for remote sensing using a multivariate Gaussian MRF model. In *Proceedings of the IEEE International Symposium on Circuits and Systems*, volume 2, pages 648–651, 1996.

[83] C. Zhu and X. Yang. Study of remote sensing image texture analysis and classification using wavelet. *International Journal of Remote Sensing*, 19(16):3197–3203, 1998.