Transform Texture Classification

by

Xiaoou Tang

B.S., University of Science and Technology of China (1990)
M.S., University of Rochester (1991)

Submitted in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

and the

WOODS HOLE OCEANOGRAPHIC INSTITUTION

May, 1996

Author .....
Joint Program in Applied Ocean Science and Engineering
Massachusetts Institute of Technology/Woods Hole Oceanographic Institution
May 28, 1996

Certified by .....
W. Kenneth Stewart
Thesis Supervisor

Accepted by .....
Henrik Schmidt
Chairman, Joint Committee for Applied Ocean Science and Engineering
Massachusetts Institute of Technology/Woods Hole Oceanographic Institution

Transform Texture Classification

by
Xiaoou Tang

Submitted to the Department of Ocean Engineering on May 28, 1996, in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Applied Ocean Science and Engineering

Abstract

This thesis addresses the three major components of a texture classification system: texture image transform, feature extraction/selection, and classification. A unique theoretical investigation of texture analysis, drawing on an extensive survey of existing approaches, defines the interrelations among 11 types of texture analysis methods. A novel unification of the different methods defines a framework of transformation and representation in which three major classes of transform matrices capture texture information of increasing coherence length or correlation distance: the spatial domain method (co-occurrence method), the micro-structural method (run-length method), and the frequency multi-channel method (Fourier spectrum method).

A more concise vector representation of a selected transform matrix is then needed for input to a classifier. Unlike traditional methods, which use various special functions to describe the properties of each transform matrix, a new approach directly applies a principle component analysis technique to the transform matrix. The Karhunen-Loeve Transform (KLT) extracts a vector of dominant features, optimally preserving texture information in the matrix. This approach is made possible by the introduction of a novel Multi-level Dominant Eigenvector Estimation (MDEE) algorithm, which reduces the computational complexity of the standard KLT by several orders of magnitude. The statistical Bhattacharyya distance measure is then used to rank dominant features according to their discrimination power.

Experimental results of applying the new algorithm to the three transform matrix classes show a strong increase in performance by texture analysis methods traditionally considered to be least efficient. For example, the power spectrum and run-length methods now rank among the best. Using the same MDEE algorithm, the three extracted feature vectors are then combined into a more complete description of texture images. The same approach is also used for a study of object recognition, where the combined vector also include granulometric, object-boundary, and moment-invariant features.

In most classification experiments, a simple statistical Gaussian classifier is used. The plankton object recognition experiments use a Learning Vector Quantization (LVQ) neural-net classifier to achieve superior performance on the highly non-uniform plankton database. By introducing a new parallel LVQ learning scheme, the speed of network training is dramatically increased. Tests show a 95% classification accuracy on six plankton taxa taken from nearly 2,000 images. This result is comparable with what a trained biologist can accomplish by traditional manual techniques, making possible for the first time a fully automated, at-sea approach to real-time mapping of plankton populations.

Thesis Supervisor: W. Kenneth Stewart, Ph.D.
Title: Associate Scientist, MIT/WHOI Joint Program

Acknowledgments

I would like to thank my advisor, Ken Stewart, for teaching me how to conduct research, for keeping me focusing on the big picture, for showing me all the fascinating well-funded projects, for sending me to those great conferences, and of course, for making this thesis possible. I am also very grateful to all my thesis committee members, Eric Grimson and Dan Dudgeon for their insightful comments, advice, expertise of computer vision and image processing, and continuing encouragement, Rob Fricke and John Leonard for always being there. Thanks to Douglas Carmichael for serving as the chairman of the defense.

Thanks also to:

ONR and the WHOI joint program education office: for supporting this work.

The plankton 'bugs" group:

- Marty Marra: for all the Unix tips and for dealing with those "bugs".
- Luc Vincent: for his command and enthusiasm of mathematical morphology.
- Cabell Davis, Scott Gallager, and Carin Ashjian: for providing the "bugs".

Everyone at the Deep Submergence Lab:

- Hanu Singh, Diane DiMassa, Tad Snow, and Erik Burian: for working as graduate cheap labors together.
- Dan Potter: for many interesting discussions and for proof reading.
- Craig Sayers: for checking the thesis grammar.
- Steve Lerner and Jon Howland: for all the system helps.
- Cindy Sullivan: for re-typing some of the lost files and many other helps.
- Beven Grant and Larry Flick: for all the administrative helps.
- Dezhang Chu: for being a true acoustic expert.
- Everyone at the DSL for a friendly and helpful environment.

Everyone at MIT/WHOI who has made my life here a lot easier:

- Abbie, Julia, Jake, and John at the WHOI education office: for helping me in many ways.
- Mary Jane at the WHOI housing office: for the Oyster Pond, the winding lane, the 264 woods hole road, the 27B challenger drive, the barn, and the 49 school street.
- Jean and Beth at the OE Dept. graduate office of MIT: for all the deadline extensions.

- Ronni at MIT joint program office.
- Huaiyu Fan, Chenyang Fei, Hua He, Brian Tracey, and Qing Wang at MIT 5-007.
- Jubao Zhang: for helping with printing of this thesis.
- Mark, Steve, Dave and Dave: for the great daily basketball game. Also for all the printing and copying help.

My family:

- Dear family friends Ruixin Huang and Liping Zhou: for all the helps and for first introducing the joint program to us.
- A special friend Dan Li: for renewing books, copying papers, helping with defense preparation, washing dishes, and many other things that are beyond the duty of a graduate student.
- My wife and ten year classmate at USTC and MIT, Helen Huang: for revising this thesis including this page, for teaching me neural network, for lending me C program tips, for handling all the bills and tax forms, for keeping the house clean, for giving me all kinds of advice about life, for enjoying my cooking therefore loving me.
- My mother, Xiuzhi Xue, and my father, Guiqi Tang: for being the greatest parents on earth.
- Helen's parents, Huihua He and Guangquan Huang: for having a wonderful daughter.
- My brother Xiaopeng Tang, and my sister Man Tang: for their love.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Advances in remote sensing technology have given us an unparalleled view of the world: satellites routinely provide us with massive amounts of imaging data describing our atmosphere and the surface of our planet; planetary probes give us similar information about our neighbors in space; underwater sensors gather scientific imagery needed to monitor the health of the oceans and the evolution of the seafloor.

These imagery data, if properly interpreted, provide us with a deeper understanding of our world and the ability for better decision-making. However, the sheer amount of information that is becoming available is greatly expanding as surveys cover increasingly finer scales and increasingly larger areas. Consequently, the time required for interpretation of data has increased to a point that automatic image analysis systems must be developed for scientists to process the data efficiently. The attraction of automated processing lies not only in its orders-of-magnitude speed increase but also in the improved accuracy of quantitative measurements when compared with the qualitative estimates of human eyes.

In this thesis, I develop an automatic image classification system utilizing texture properties in the image. Texture is the term used to characterize the surface of a given object or the uniform regions in an image and is undoubtedly one of the main features utilized in image processing and pattern recognition [119]. Images are full of textures such as grass fields, water surfaces, clouds, and linen fibers. It is believed that the perception of texture plays an important role in the human visual system for recognition and interpretation.

Texture exists everywhere. Like color, it is a basic property of the surface of objects. We can see it and feel it, but seem to have a hard time defining it. Many different definitions of texture have been proposed over the years. Not to further complicate the matter, I do not attempt to give another definition here, but only summarize some of the existing ones to get a feel for texture:

"The notion of texture appears to depend upon three ingredients: (i) some local 'order' is repeated over a region which is large in comparison to the order's size, (ii) the order consists in the nonrandom arrangement of elementary parts, and (iii) the parts are roughly uniform entities having approximately the same dimensions everywhere within the textured region." [49]

"A region in an image has a constant texture if a set of local statistics or other local properties of the picture function are constant, slowly varying, or approximately periodic." [99]

"We may regard texture as what constitutes a macroscopic region. Its structure is simply attributed to the repetitive patterns in which elements or primitives are arranged according to a placement rule." [101]

"An image texture is described by the number and types of its primitives and the spatial organization or layout of its primitives. The spatial organization may be random, may have a pairwise dependence.... The dependence may be structural, probabilistic, or functional." [47]

"A fundamental characteristic of texture: it cannot be analyzed without a frame of reference of tonal primitive being stated or implied. For any smooth gray-tone surface, there exists a scale such that when the surface is examined, it has no texture. Then as resolution increases, it takes on a fine texture and then a coarse texture." [47]

"Texons

1. Elongated blobs - e.g., rectangles, ellipses, line segments with specific colors, angular orientations, widths, and lengths.

2. Terminators - ends-of-line segments.

3. Crossings of line segments." [57]

"Texture could be defined as a structure composed of a large number of more or less ordered similar elements or patterns without one of these drawing special attention. So a global unitary impression is offered to the observer." [46]

"When we observe textiles, tree bark, or stones, we perceive that their surfaces are homogeneous, in spite of fluctuations in brightness or color. Such a homogeneous visual pattern is called *texture*." [106]

Human beings can usually easily identify various texture regions in an image. However it is quite a challenge to endow a computer with such an ability. These definitions provide some starting points for establishing an automatic texture classification system. Many texture analysis techniques are developed through modeling or describing certain special texture properties as described above. Different approaches are derived depending on how texture is defined or perceived.

This thesis begins by studying the theoretical interrelations of most existing statistical texture feature extraction methods in the framework of transformation and representation, instead of offering another definition and presenting another texture analysis approach. The study leads to the unification of many different texture representations into a few unique ones, including frequency spectrum, co-occurrence matrix, and run-length matrix. I then develop algorithms to extract and condense as much texture information as possible into a small number of features of high discriminatory power that the classifier can use to label the texture type.

I believe that the most important thing for a texture analysis method is how the method represents the texture. To a large extent, a successful algorithm depends on whether it can transform the spatial representation of the original texture into a new representation best suited for the classification work on hand. The goal that guides the design of the texture classification algorithms in this thesis has always been to extract all the information available and to condense it into a concise representation. As a step toward this goal, I present a general texture classification system in this next section.

1.1 A general texture classification system

Figure 1.1 presents a general processing pipeline for texture classification. We can consider this pipeline as an information condensing process. Starting from the original texture image, we first transform it into a new shift invariant representation, such as the Fourier spectrum shown in the figure. Then, using feature extraction and selection algo-

rithms, we compress most texture information into a small feature vector. Finally, by passing the feature vector through a classifier, the final class label of the texture image is produced. This single class label number contains all the information that we want to know about this texture image sample.

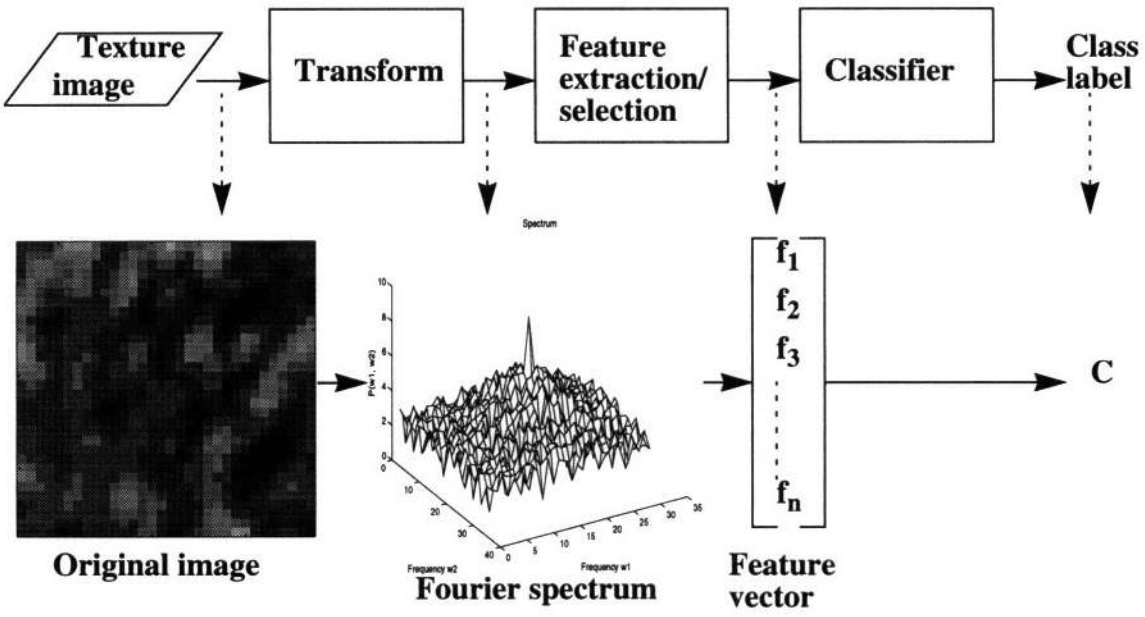


Figure 1.1: General configuration of a texture classification system.

This thesis addresses the three major components of this processing structure—image transform, feature extraction, and feature classification—with emphasis on the first two steps.

I start with a unique theoretical investigation of the interrelations among 11 types of texture analysis methods. A novel unification of the different methods defines a framework of transformation and representation in which three major classes of transform matrices capture texture information of increasing coherence length or correlation distance: the spatial domain method (co-occurrence method), the micro-structural method (run-length method), and the frequency multi-channel method (Fourier spectrum method).

A more concise feature vector representation of a selected transform matrix is then needed for input to a classifier. Unlike traditional methods, which use various special functions to describe the properties of each transform matrix, a new approach directly applies a principle component analysis technique to the transform matrix. The Karhunen-Loeve Transform (KLT) extracts a vector of dominant features, optimally preserving texture information in the matrix. This approach is made possible by the introduction of a novel Multi-level Dominant Eigenvector Estimation (MDEE) algorithm, which reduces the computational complexity of the standard KLT by several orders of magnitude.

Optimal features for image representation are not necessarily the best for image classification. Different criteria must be used for the selection of optimal features for compression and classification. I use the Bhattacharyya distance measure to rank the KLT-extracted dominant features according to their discriminatory power.

I apply this algorithm to the three types of texture transform matrices—the frequency matrix, the run-length matrices, and the co-occurrence matrices—and improve those techniques that have traditionally been considered least efficient, such as the power spectrum method and the run-length method, to being among the best. On a set of eight Brodatz texture images, classification accuracy is improved from about 80%to perfect or near perfect classification.

Next, using the same MDEE algorithm, I combine the above three unique texture representations into a more complete description of texture images. Although similar to many previous studies using combined texture feature vectors, this approach is a logical extension of the MDEE algorithm, which serves to combine new information and to remove overlapped information from different texture matrices. The same approach is also used for an object recognition study, where the combined feature vector includes not only texture features but also the object boundary and moment invariant features.

For the classifier, I use mostly a simple statistical Gaussian classifier for texture classification experiments. The plankton object recognition experiments use a Learning Vector Quantization (LVQ) neural-net classifier to achieve superior performance on the highly non-uniform plankton database. By introducing a new parallel LVQ learning scheme, the speed of network training is dramatically increased.

The system is successfully tested on four data sets: eight Brodatz textures, sixteen classes of Vistex textures, three image classes from a sidescan sonar survey of the Arctic ice canopy, and six types of plankton images. The first two data sets are used for algorithm development and for comparison with existing work in the literature. The last two data sets are direct applications of this work to our continuing oceanography geological and biological research projects.

1.2 Applications in oceanographic research

Texture analysis has found many important applications in such areas as computer vision, medical imaging, remote sensing, and industrial inspection. New applications are still being proposed. One area that increasingly uses texture analysis techniques is oceanographic research. Since it is the main focus of our application study, I give a brief survey of this area.

Sidescan sonar has been an important tool for seafloor survey over the last few decades. The traditional analysis of sidescan sonar images has been done by human photo-interpreters. Human interpretation is slow, expensive, and open to dispute, so automatic processing is important. Due to the highly textured appearance of sonar images, texture analysis techniques become a natural choice for sidescan sonar image analysis.

Texture analysis of sidescan imagery can be applied to various geological feature recognitions. Pace and Dyer applied the co-occurrence features to distinguish the physical properties of sedimentary bottoms using sidescan sonar [84]. Reut et al. conducted an analysis of one dimensional image spectrum for classification of six classes of homogeneous sediment type, including mud, clay, sand, gravel, cobbles, and boulders [93]. Pace and Gao continued this work and developed an alternative approach by extracting 1-D spectral features from sidescan data [85]. Reed and Hussong reported an application of the co-occurrence features to the classification of submarine lava flows, as well as the segmentation of lithified and non-lithified sedimentary formations [89]. Recently, Stewart et al. applied a neural network classifier to several traditional statistical texture features for the classification of terrain types in the midocean-ridge area [100]. Three distinct seafloor types, sediment pond, ridge flank, and axial valley, were classified with near 90% accu-

racy. The new texture analysis techniques developed in this thesis have also been applied to a similar data set [102].

Another important geological application is in Arctic ice canopy survey. As the geophysical and economic importance of the polar seas becomes more widely recognized, the need for intensive study of these regions becomes more apparent. There exists a large body of research on ice types and the roughness of the snow surface of sea ice [10] [37] [69] [88]. Traditional statistical algorithms were used by several researchers to analyze synthetic-aperture radar (SAR) images of the ocean ice surface [52] [78] [108]. Although SAR surveys offer advantages of high speed and large coverage, there are problems that cannot be solved by SAR imagery. For example, the under-ice morphology, which is important to the study the frictional coupling between the ice cover and water, and the scattering of acoustical energy by an ice canopy, can not be characterized with SAR data.

Since the thermal processes tend to level out rough elements much more slowly on the under-ice surface than on the top, the under-ice canopy tends to preserve many more features for the analysis of ice floe, ridge, and crack formation than the upper surface. These issues lead to the considerable amount of work done on the statistical study of the ice draft profile [44] [62] [96] [114] [115] [116] [117]. However, the exclusive concentration on one-dimensional data analysis limits the success of these under-ice morphology studies. In this thesis, I apply a new texture analysis method to the classification of the two-dimensional sidescan sonar images of the Arctic under-ice canopy. The mid-90% classification accuracy on three classes of sonar images is very encouraging.

The second important novel application of the new algorithm is plankton recognition. Because plankton are at the base of the food chain in the ocean and have a large impact on marine ecosystems, it is important to study how changing climate and human activities impact their population dynamics. Such studies require large-scale mapping of plankton distribution and abundance at high spatial and temporal resolutions. However, because of the laborious deployment process, low spatial sampling rate, and the difficult post-processing tasks using traditional equipment—towed nets, pumps, and Niskin bottles—it has been very difficult to conduct such extensive experiments. To overcome the limitations of traditional plankton sampling instruments, a new Video Plankton Recorder (VPR) has been developed [26]. As the VPR is towed through the water, it continuously captures

magnified plankton images, providing a high spatial resolution of plankton distributions. For a large-area survey, the amount of image data can be overwhelming, necessitating an automated approach to plankton recognition if all data are to be processed.

In this thesis, I apply the new image classification algorithms to plankton image classification, making possible for the first time a fully automated plankton population mapping system. The resulting classification accuracy is comparable with what a trained biologist can achieve using traditional manual techniques.

1.3 Thesis overview

The thesis starts with a survey of existing texture analysis work in Chapter 2, categorizing the reviewed approaches into three classes: structural modeling methods, stochastic texture modeling methods, and statistical methods. I briefly summarize the structural modeling methods and the stochastic texture modeling methods first, then concentrate on a more detailed survey of the statistical methods, which are the main focus of the following thesis study.

Chapter 3 contains most of the theoretical development of the texture classification system. I address the three key components of the system: texture image transformation, feature extraction and selection, and classification.

First I examine the analytical interrelations of various existing texture classification methods in the framework of transformations. Many of these relations are studied for the first time. I then unify them into three major categories. Using a novel multi-level dominant eigenvector estimation method and the Bhattacharyya distance measure, I present a new texture feature extraction and selection scheme that optimally obtains texture features from the texture transform matrices. Finally, I describe a simple statistical classifier and an improved neural network classifier.

Chapter 4 studies the frequency transform texture classification methods. I investigate and compare three methods: the wavelet method, the traditional power spectrum method (PSM), and the new dominant spectrum method (DSM).

To fully utilize the power of a wavelet packet transform, I use the feature selection algorithm described in Chapter 3 to combine and select frequency-channel features that give improved classification performance.

I consider the Fourier transform as the highest level of multi-channel decomposition. Instead of painstakingly designing various multi-channel filters, I take the maximum number of filter channels that can be obtained then use the multi-level dominant eigenvector estimation technique to determine which channels to keep and how much of the energy in those channels to keep. The resulting coefficients represent the magnitude of each frequency channel's contribution and form a designed filter.

I also compare the new Fourier features with the traditional power spectrum method and show that by using appropriate feature extraction algorithms the discrimination power of the Fourier transform features can be significantly improved.

Chapter 5 investigates the run-length texture analysis method which, has traditionally been considered a less efficient approach. By using the new MDEE algorithm and the Bhattacharyya distance measure to extract features directly from the run-length matrices, much of the texture information is preserved. Perfect classification is achieved on the eight Brodatz textures compared to the upper 80% accuracy of the traditional run-length method.

Based on the observation that most texture information is contained in the first few columns of the run-length matrix, especially the first column of the run-length matrix, I develop a new, fast, parallel run-length matrix computation scheme.

Chapter 6 extends the multi-level dominant eigenfeature analysis approach one level further to a multi-view texture analysis approach. Using the same MDEE algorithm, I combine the feature vectors extracted from several different transform matrices—including the Fourier transform matrix, the run-length matrices, and the co-occurrence matrices—to form a more complete description of texture images. Although similar to many previous studies using combined texture feature vectors, the approach is a logical extension of the MDEE algorithm, which serves to combine new information and to remove overlapping information from different texture matrices.

Before studying the multi-view analysis approach, I first conduct a similar experiment on the co-occurrence matrices using the new feature extraction algorithm to extract maxi-

mum texture information directly from the transform matrices. As discussed in Chapter 3, co-occurrence features offer a unique view of texture images that cannot be totally replaced by other methods, so I incorporate them into the multi-view feature vectors.

Chapter 7 shows a successful application of the multi-view analysis to underwater plankton image recognition. I integrate such popular shape descriptors as moment invariants and Fourier boundary descriptors with granulometric texture features to compose a very effective feature vector for plankton imagery. Then, using the improved Learning Vector Quantization (LVQ) neural network classifier developed in Chapter 3, I classify the plankton images into several taxonomic categories with greater than 90%acuracy.

Chapter 8 summarizes the major contributions of this thesis and suggests future research directions.

Chapter 2

Review of texture analysis methods

The diversity of natural and artificial textures makes it very hard to give a universal definition of texture. This results in a large number of texture analysis techniques. There are many excellent reviews about various texture analysis approaches. For review of earlier methods, refer to the papers by Haralick [47], Wechsler [119] and Gool et al. [46]. Tuceryan and Jain gave a more up to date review in [107].

In general, texture analysis methods can be categorized into three categories: structural modeling methods, stochastic texture modeling methods, and statistical methods. I briefly summarize the structural modeling methods and the stochastic texture modeling methods in the first two sections. Then, in the final section of this chapter, I concentrate on a more detailed survey of the statistical methods, which are the main focus of this thesis study.

2.1 Structural texture modeling methods

Structural methods try to model image texture using a set of texture primitives or elements arranged by certain placement rules [11] [68] [106] [122]. The analysis steps include extraction of texture primitives in the image and estimation of the placement rules for texture elements.

Texture elements can be as simple as points of local extrema or can be complex structures like edges or uniform regions. Tuceryan and Jain [107] use a difference of Gaussian filter to detect connected edges as texture elements. Voorhees and Poggio [113] use Laplacian of Gaussian masks at different scales and combine the output to extract blobs in an image as texture elements. A more complicated procedure for extracting texture primitives developed by Tomita and Tsuji [106] includes segmenting the image into uniform regions, skeletonizing, segmenting complex skeleton axes into simple sub-axes, expanding the sub-axes to the region of original width, and analyzing the shape and intensity properties of the extracted texture primitives in order to group the texture elements into similar groups.

The placement rules of the similar texture elements can be described by various graph theories. The most popular placement rule uses a Voronoi diagram [106] [107]. A Voronoi diagram is a partition of the 2-D image plane into polygonal cells with each cell containing one texture element. Each image point inside the polygon is closer to the center of its texture element than to any other texture element. If two Voronoi polygons have a common edge, the corresponding texture elements are considered as neighbors of each other. Then, features of each Voronoi cell (e.g., moments of the polygon area and the distances of each texture element to its nearest neighbor texture element) are extracted, and elements with similar features are grouped to construct uniform texture regions.

Zucker [122] developed a method treating the real textures as distorted versions of ideal textures. The placement rule for the ideal texture is described by a graph isomorphic to a regular or semiregular tessellation, which is then transformed to generate the real textures. Fu [40] defines the placement rule by a tree grammar. A texture is viewed as a string in the language defined by the grammar whose terminal symbols are the texture elements.

For natural textures, it is difficult to infer the primitive types and the placement rules. Some textures are ambiguous, with more than one choice of primitive. Although structural texture analysis approaches achieve reasonable successes on textures of high regularity, they are very limited in power when applied on real-world gray-scale images. Given our main focus on noisy underwater optical and acoustical image processing, it is impractical to adopt the structural texture analysis approach.

2.2 Stochastic texture modeling methods

Stochastic texture modeling methods treat texture as a stochastic random field process with specific parameters. These parameters are supposed to capture the essential qualities of texture, and thus can be used to synthesize, classify, and segment textured images.

A random field is a joint distribution that imposes statistical dependence on a set of random variables, such as image intensities, in a spatially meaningful way [33]. Abend et al. first proposed random field models in [1]. Following Besag's [6] first mathematically tractable technique for specifying models and estimating parameters for models of gray

values on a lattice, several research groups have contributed to the further development of random field models [7] [30] [33] [43] [59].

Among various random field models, Markov random field (MRF) models are the most frequently used [6] [7] [8] [20] [30] [43]. Markov fields provide a flexible mechanism for modeling spatial dependence and local contextual information in an image. The basic idea of MRF is that a pixel should be statistically more similar to its neighbors than to pixels elsewhere. An excellent review of the MRF models for image analysis can be found in [33].

Although the stochastic approach can provide a scientific basis for understanding the image formation process, and many promising algorithms have been proposed, problems like model selection, parameter estimation, large computational requirement, and the phase transition phenomenon remain to be solved before it becomes a practical method for large natural-image data sets.

2.3 Statistical texture analysis methods

Both the structural modeling methods and the stochastic modeling methods try to model the texture formation process, which is necessary for texture synthesis but not so for image classification and segmentation. The only input a pattern classifier needs is a set of feature vectors that characterizes different traits of the texture images. The three constraints usually imposed on these feature vectors are: homogeneity within each texture class, maximum discrimination among classes, and small dimensionality. The statistical methods are designed to extract such feature vectors that are more suitable and more commonly used for analysis of natural imagery than the structural and stochastic modeling methods.

The first-order statistical measures of texture images do not produce useful texture features, because the statistics are invariant to any rearrangement of the image pixels. To extract the contextual information in the texture, many statistical texture analysis methods have been developed, both in the spatial and in frequency domains. The remainder of this section surveys the commonly used statistical texture features.

2.3.1 Spatial gray-level dependence method (SGLDM)

The spatial gray-level dependence matrix is also called the co-occurrence matrix. Julesz [56] first used gray tone transition matrices in texture discrimination experiments, which are equivalent to nearest horizontal neighbor co-occurrence matrices. Similar texture measures were used by Darling and Joseph [22] for analyzing satellite images of clouds. Rosenfeld and Troy [95] and Haralick et al. [48] first proposed co-occurrence matrices for arbitrary spatial distances and angular directions. Comparative studies [19] [120] show the superiority of this method over several traditional statistical texture measures.

For an image with N by N pixels and G gray levels, the co-occurrence matrix [48] for a displacement d in a direction θ is defined to be a G by G matrix whose entry $M(i, j)$ is the number of occurrences of transitions from gray level i to gray level j, given the intersample distance d and the direction θ. Figure 2.1 (b) shows a typical co-occurrence matrix M with $d = 1$ and $\theta = 0$. The matrix gives a measure of the joint probability density of the pairs of gray levels that occur at pairs of points separated by distance d in the direction θ. For a coarse texture, d is relatively small compared to the sizes of the texture elements; the pairs of points at separation d have similar intensity values. This means the matrix M has large values near its main diagonal. Conversely, for a fine texture the values in M are quite uniformly spaced. Thus, a measure of the degree of value spread around the main diagonal of M should provide a good sense of the texture coarseness. Similarly, one can extract other features to measure the directional information, contrast, correlation, etc. Haralick et al. [48] proposed 28 second-order statistic features that can be measured from this co-occurrence matrix.

Davis et al. [29] tried to improve the SGLDM by introducing a generalized co-occurrence method. Instead of looking at the second-order statistics of the pixel gray levels, they studied the statistics of some local texture structures, such as local maxima and edges. The new method performs better than the original SGLDM over a very small set of data. However, the extra complexity of the method outweighs its performance gain.

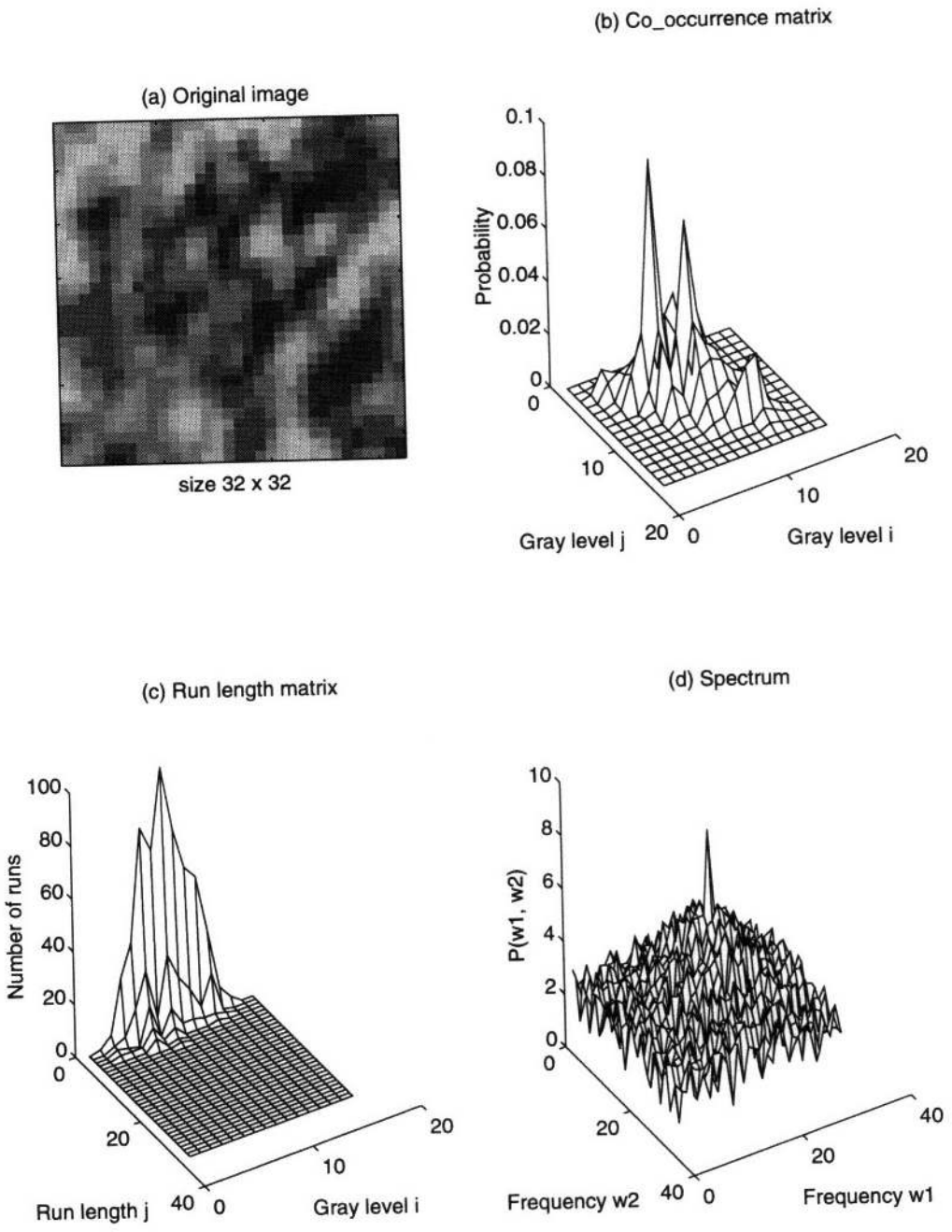


Figure 2.1: Three types of texture matrices for a sample texture image.

2.3.2 Gray level difference method (GLDM)

The gray level difference method uses the probability distribution function of the gray-level difference between two nearby pixels to compute texture features. For an image $f(n_1, n_2)$, let $i = |f(n_1, n_2) - f(n_1 + \Delta n_1, n_2 + \Delta n_2)|$, where $\delta = (\Delta n_1, \Delta n_2)$ is a given row and column displacement. From the probability distribution function $P_\delta(i)$ of i, several statistical features can be computed, including contrast, angular second moment, entropy, and mean. For detailed definitions, refer to [120]. Notice that $P_\delta(i)$ can be computed from the co-occurrence matrix by summing up the elements of the co-occurrence matrix along lines parallel to its main diagonal. Thus the features of GLDM are quite similar to the features of SGLDM.

2.3.3 Autocorrelation method

The autocorrelation function is defined as

$$C(m_1, m_2) = \sum_{n_1 = 1}^{N} \sum_{n_2 = 1}^{N} f(n_1, n_2) f(n_1 + m_1, n_2 + m_2), \tag{2.1}$$

which measures how well a shifted version of an image matches with itself. For a coarse texture, the neighbor pixels are similar to each other and the matching coefficient drops slowly as the shift increases; for a fine texture, neighbor pixels change quickly, so the matching coefficient drops sharply for a small shift. Therefore, the shape of the autocorrelation function reflects the coarseness of a texture. The periodicity of the texture can also be revealed by the energy peaks in the function. An early application study of the autocorrelation function on Arctic aerial photographs can be found in [58] and an application of correlation for image segmentation in [15].

Since most textures are periodic, they can be better described in the frequency domain. The autocorrelation function feature has mostly been supplanted by its frequency-domain counterpart, the power spectrum.

2.3.4 Power spectrum method (PSM)

The power spectrum method uses spectral statistics in the frequency domain [3] [55] [67] [120]. The discrete space Fourier transform of an image $f(n_1, n_2)$ is defined by

$$F(w_1, w_2) = \sum_{n_1 = -\infty}^{\infty} \sum_{n_2 = -\infty}^{\infty} f(n_1, n_2) e^{-jw_1 n_1} e^{-jw_2 n_2}. \tag{2.2}$$

The Fourier power spectrum is

$$P(w_1, w_2) = F(w_1, w_2) \cdot F(w_1, w_2)^*. \tag{2.3}$$

Figure 2.1 (d) shows the Fourier power spectrum of a sample texture image. By measuring the energy distribution in frequency space, various texture properties of an image can be obtained. In smooth images, for example, features of the form

$$PSM_r = \sum_{r_1^2 \le w_1^2 + w_2^2 < r_2^2} P(w_1, w_2) \tag{2.4}$$

have high values for small r because the smooth images have more energy at lower frequencies. For a rough texture, the high-frequency energy dominates, resulting in high PSM_r for large r. For the same reason, features of the form

$$PSM_\theta = \sum_{\theta_1 \le \text{atan}(w_2/w_1) < \theta_2} P(w_1, w_2) \tag{2.5}$$

give a good measurement of directional information [120]. Liu and Jernigan [67] provide an extensive summary of 28 PSM features.

Historically, the texture classification performance of the PSM has been ranked fairly low among most texture analysis techniques [19] [120], resulting in limited applications of the approach. Criticisms of the PSM are of the Fourier transform rather than of the way that texture features are computed from the power spectrum. As one of the contributions of

this thesis, in Chapter 4 I show that using new feature extraction algorithms, the discriminatory power of the Fourier transform features can be significantly improved.

2.3.5 Gray level run length method (GLRLM)

When examining image pixels along a certain direction, we occasionally find runs of consecutive points with the same values. In a rough texture, we expect that relatively long runs would occur more often, whereas a fine texture should contain primarily short runs. For a directional texture, the run-length measurement is also direction dependent, providing a good measure of texture directionality [120].

Galloway first introduced the run-length measures as texture features [42]. To define run-length features within an image, let $p(i, j)$ be the number of runs with pixels of gray level i and run length j, M the number of gray levels, N the number of different run lengths, n_r the total number of runs, and K_i and K_j type controllers. The general run-length gray-level feature (GRLGLF) is defined as [23]

$$GRLGLF = \frac{1}{n_r} \sum_{i=1}^{M} \sum_{j=1}^{N} i^{K_i} j^{K_j} p(i, j). \tag{2.6}$$

Given different type controllers K_i and K_j, eleven run-length measurements are extracted. Figure 2.1 (c) shows a typical run-length matrix. In general, run-length texture features are good indicators for directional linear features.

In several comparison studies, the run-length method is ranked the least efficient [19] [120]. In Chapter 5, I elaborate further on the run-length method and demonstrate a new approach that extracts the maximum information from run-length matrices.

2.3.6 Texture energy filters

Research on multichannel processing for texture analysis started with Laws [66] "texture energy" filter banks. Laws [66] first used a set of small empirical filter masks to filter the texture image, then computed the variances in each channel output as the texture features. Figure 2.2 depicts the general processing pipeline. By experimenting with a wide

variety of filters, Laws chose a set of nine 3x3 (or alternatively 25 5x5) operators with shapes tailored to perform a center-weighted average, directional edge detection, spot detection, and wave detection as the filters F_n in Figure 2.2. After the filtering process, each original image pixel corresponds to nine new pixels, one in each of the nine filtered images. Then a so called macro-window of size 15x15 or larger is drawn around each of the nine pixel points. Various statistical measures are computed within the macro-windows and evaluated according to their discriminatory power. Variance in each channel was found to be the best measure for texture classification. On this basis, Laws uses the nine variances, which he calls the "texture energy measure," as the feature vector for the original image pixel in the center of the macro-window.

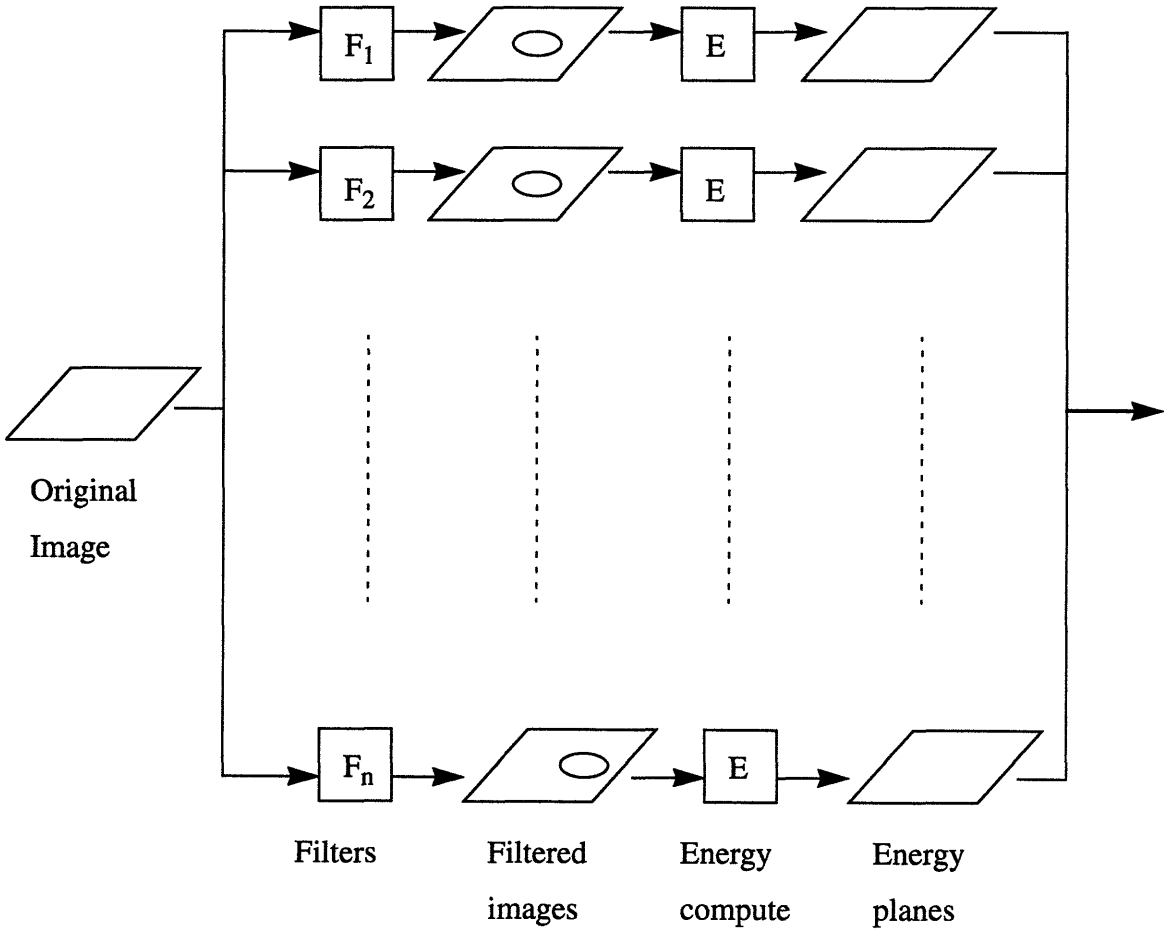


Figure 2.2: Multichannel processing schematics.

2.3.7 Eigenfilters

The filtering operation by Laws filters can also be viewed as a linear rotation of the original nine-dimensional feature vector formed by the 3x3 neighborhood pixels. The texture energy measure is the variance of the projection of the original image vector onto the nine new axes (the filters). These projection axes are chosen empirically based on observations of the texture properties and classification results. If we consider the energy compactness on each axis, the optimal rotation results from the Karhunen-Loeve Transform (KLT). The KLT rotates the original covariance matrix of the neighborhood vectors into a diagonal covariance matrix with the nine energy values on the matrix diagonal. For a Gaussian process, the joint distribution of the original 3x3 neighborhood pixel vector is fully specified by these energy measures. Based on this consideration, Ade [2] introduced the eigenfilter approach, with Laws' empirical filter banks replaced by the eigenvectors of the covariance matrix of the local texture neighborhood vectors. Later Unser [110] conducted a series texture classification experiments, which show the superiority of the eigenfilter method over Laws' filters. The processing pipeline is the same as in Figure 2.2, only with the filters F_n representing the eigenfilters instead of Laws' filters.

2.3.8 Gabor filters

Almost parallel to the development of the eigenfilter theory, the Gabor filter became increasingly used in designing texture analysis algorithms [5] [17] [35] [39] [54]. The approach is inspired by an analogy with the human preattentive vision system, which decomposes the retinal image into a number of filtered images, each containing an intensity variation over a narrow range of frequencies and orientations.

A two-dimensional Gabor filter is defined by a sinusoidal plane wave of a certain frequency and orientation modulated by a Gaussian envelope. Its impulse response is given by,

$$G(x, y) = \exp\left\{-\frac{1}{2}\left(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2}\right)\right\}\sin(\omega(x\cos\theta - y\sin\theta) + \varphi), \tag{2.7}$$

where σ is the Gaussian window width, θ is the filter orientation, ω is the frequency, and φ is the phase shift.

Two major issues with this approach are the design of the individual filters and the combination of the filter banks [35]. Jain et al. [54] and Dunn et al. [35] developed several filter design procedures using Gabor functions. Other types of filters similar to the Gabor filters have also been developed, such as the differences of offset Gaussian (DOOG) filters [70] and the Gaussian derivatives. Addressing the second issue, Malik and Perona [70] derived a filter-bank combination structure mimicking the human early-vision system, which perhaps has provided the most detailed justification for a particular filter-bank structure. The processing structure is also similar to the one in Figure 2.2.

2.3.9 Wavelet and wavelet packet transforms

The filter outputs of the above multichannel approaches are not orthogonal, thus leading to a large overcomplete representation of the original image. Recent advances in wavelet [24] [25] [71] [72] [94] and wavelet packet theory [18] [77] provide a promising solution for this problem. The texture research community is currently devoting considerable effort to wavelet applications in texture analysis [12] [13] [51] [65] [109]. The first study of a wavelet transform for texture analysis was described by Mallat [71]. Later, Henke-Reed and Cheng [51] applied a wavelet transform to texture images, using the energy ratios between frequency channels as the features. Chang and Kuo [12] [13] developed a tree-structured wavelet transform algorithm for texture classification and segmentation, which is similar to the wavelet packet best basis selection algorithm of Coifman and Wickerhauser [18]. Both the standard wavelet and the wavelet packet energy features were used directly as texture features by Laine and Fan [65] in their texture classification work.

These researchers have demonstrated that the wavelet transform is a valuable tool for texture analysis. However, a common problem with these approaches is that they are all direct applications of existing wavelet processing algorithms, which were originally developed for signal representation or compression instead of signal classification. In Chapter 4, I apply a new feature-selection approach to wavelet features and compare them with the new frequency transform features.

2.3.10 Fractal

Another type of texture model receiving attention in the current literature is based on fractals [14] [34] [71] [86]. The theory of fractals was developed by Mandelbrot [73], extended by Barnsley [4] and Feder [38], and applied to texture classification and segmentation by several researchers [14] [61] [86].

The fractal method is developed based on the assumption that textures exhibit some form of similarity over a certain range of scales. By exploiting this self-similarity, a feature called fractal dimension is extracted to model and classify the textures. There are several methods to estimate the fractal dimension, including the power spectrum method [86], the variation method [34], and the box counting method [61].

Pentland [86] successfully applied the fractal feature based on the power spectrum estimation to texture segmentation. The method fits a power-law curve to the power spectrum of the image. The global power spectrum of an isotropic fractal Brownian surface is given by

$$P(w_1, w_2) \propto [(w_1^2 + w_2^2)^{1/2}]^{-\beta} = r^{-\beta} \tag{2.8}$$

where (w_1, w_2) are two-dimensional frequency coordinates and r is the radial frequency. By best fitting the log-log curve of P versus r, we can estimate the slope β. The fractal dimension D can then be computed by

$$D = 4 - \frac{\beta}{2}. \tag{2.9}$$

Figure 2.3 shows a sequence of four fractal images with increasing fractal dimension D. Apparently, the increasing perceptual roughness corresponds quite closely with our intuitive notion of texture roughness [86].

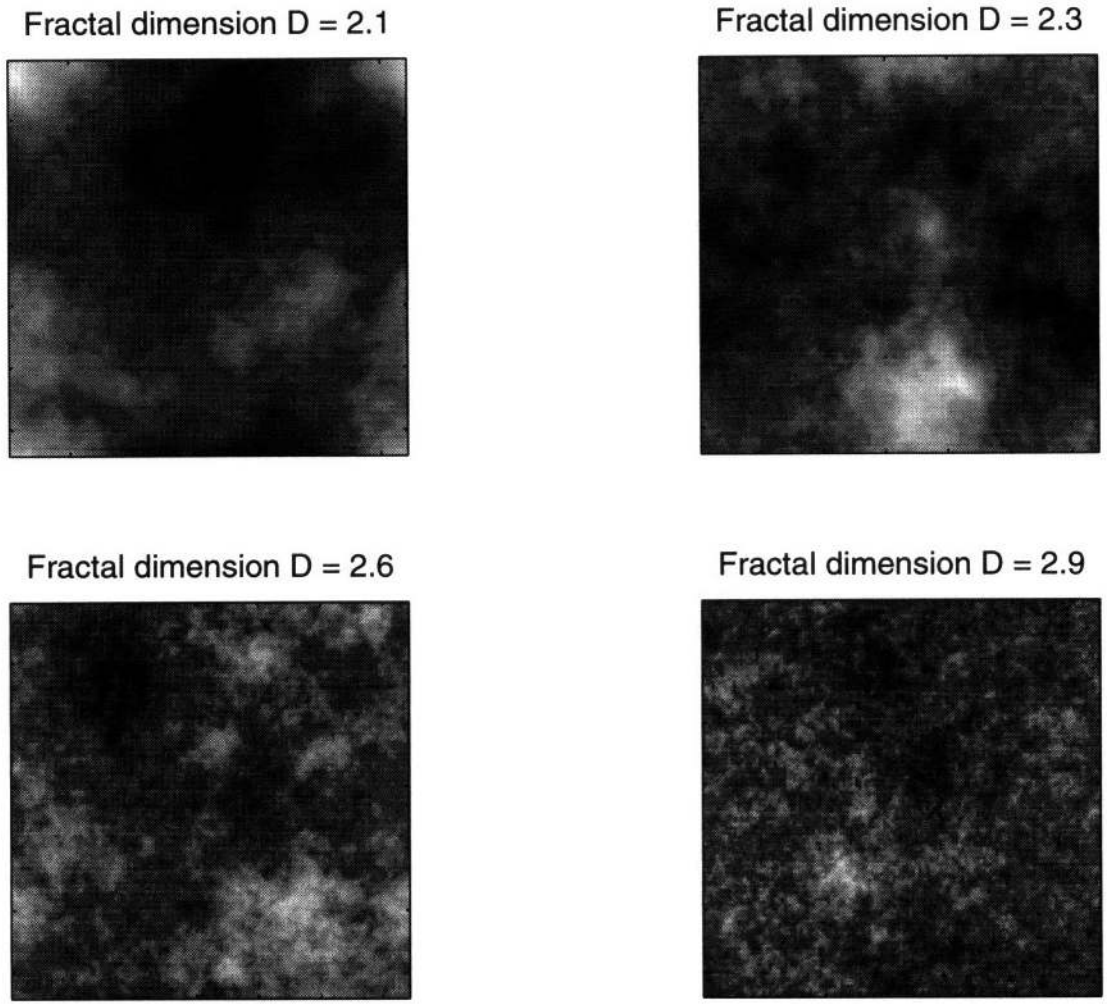


Figure 2.3: Images of increasing fractal dimensions.

2.3.11 Mathematical morphology: Granulometry pattern spectrum

Granulometry is one of the most useful tools of morphological image processing. Since first introduced by Matheron [76], it has been applied to a wide range of areas including shape characterization [111], texture classification, and segmentation [32].

A mathematical granulometry curve is generated by successively opening an image by an increasing sequence of structuring elements and counting the number of pixels elimi-

nated at each stage of the opening to generate a size distribution as a signature of the image. Dougherty et al. [32] used a binary version of granulometry segmentation of thresholded images.

However, because of the slow processing algorithm, granulometry has not been widely used by the image analysis community [111]. In Chapter 6, I use a new fast algorithm recently developed by Vincent [111] [112] in a new feature selection algorithm for textured object classification and demonstrate the exceptional power of gray-scale granulometry for classification of textured objects.

2.4 Summary

This chapter reviews the three major types of texture analysis methods. The focus is on statistical texture analysis methods, because they are more suitable for our natural image classification applications.

The many texture analysis methods provide researchers much freedom of choice, but a persistent problem is how to choose the right method for the application at hand. From the survey, we can see that many of the techniques offer much overlap texture information. In fact, some methods reveal texture information that is only a subset of other methods. In the next chapter, I conduct an in-depth investigation of the theoretical relations of all 11 methods reviewed here and unify them into three categories. Within each category, I identify a method that extracts texture features nearly a superset of others in that category.

Chapter 3

Transform texture classification algorithms

This chapter addresses the three major components of a texture classification system: texture image transformation, feature extraction, and classification. First, I examine the interrelations among the 11 texture transforms and unify them into three major categories. Then, using a novel multi-level dominant eigenvector estimation method, I present a new texture feature extraction and selection scheme that optimally obtains texture features from the texture transform matrices. Finally, a simple statistical classifier and an improved neural network classifier are described at the end of this chapter. The experiments conducted in the following chapters all use the same feature extraction and classification algorithms developed here.

3.1 Texture image transformations

An original image cannot be input directly into a classifier, not only because the feature vector dimension is too large, but also because those seemingly similar texture images can differ in every pixel. The spatial representation of the image must be transformed into a representation more suited to the classifier. Chapter 2 reviews many types of transformations proposed over the years. Some formal transforms, such as the Fourier and wavelet transforms, are invertible and information preserving. Some ad hoc transforms, such as run-length and co-occurrence methods, are not. Note that the term 'transform' is not used in a rigorous sense here. It simply means a process that converts one form of representation into another.

To obtain a suitable representation of textures for the classifier is crucial for the success of the classification. We cannot better stress the importance of a proper representation than Marr [75] did:

> A representation, therefore, is not a foreign idea at all - we all use representations all the time. However, the notion that one can capture some

> aspect of reality by making a description of it using a symbol and that to do so can be useful seems to me a fascinating and powerful idea. But even the simple examples we have discussed introduce some rather general and important issues that arise whenever one chooses to use one particular representation. For example, if one chooses the Arabic numeral representation, it is easy to discover whether a number is a power of 10 but difficult to discover whether it is a power of 2. If one chooses the binary representation, the situation is reversed. Thus, there is a trade-off; any particular representation makes certain information explicit at the expense of information that is pushed into the background and may be quite hard to recover.
>
> The issue is important, because how information is represented can greatly affect how easy it is to do different things with it. This is evident even from our numbers example: It is easy to add, to subtract, and even to multiply if the Arabic or binary representations are used, but it is not at all easy to do these things - especially multiplication - with Roman numerals. This is a key reason why the Roman culture failed to develop mathematics in the way the earlier Arabic cultures had.

It is the same situation in the case of texture classification. One type of transformation may produce an excellent representation of certain information in the texture but may lose others. For example, the Fourier transform generates a frequency-domain representation that reflects the periodical property of the texture but loses local structural information because of averaging over the whole image.

The choice of a good representation should be based on ultimate goals. For numerical representation, the goal is mathematical computation; for our study, it is texture classification. We need to develop a representation that will make explicit all information with discriminatory ability and suppress noise or other signals with no discrimination power. There are two types of information in texture images: one describes the common properties of all sample images in a particular texture class, which is different between different classes and therefore useful for discrimination; the other represents the deterministic prop-

erties of every individual image sample, which is different for each image sample regardless of the class, thus offering no discriminating information.

As a typical example of the two types of information, consider periodic textures. The energy at major frequencies contained in all image samples will be similar within one class and thus can be a useful property for the classification of that class of image. However, the phase values representing the shift positions of the periodic signals are different for each individual texture sample and therefore have no useful information for classification. If the original image matrix is used directly to form feature vectors, the input to the feature selection algorithm will be a combination of the above two kinds of information. The output performance will depend on the ratio of the two. Using the shift-invariant property of the power-spectrum representation, we can remove the special absolute-position information in the texture sample while preserving the frequency energy information.

The methods described in Chapter 2 are all quite successful in suppressing the texture shift information but have great differences in the degree of success in making explicit other useful texture information. In the rest of this section, we study the theoretical relations among different methods to determine whether different methods reflect different aspects of texture or simply offer a subset representation of other methods. Figure 3.1 summarizes all the relations:

(1). The run-length and granulometry methods both use structural elements to extract statistical texture features. In fact, they are both related to the frequency domain method. The long runs and large morphological structure elements are like low-pass filters, while the short runs and small morphological structure elements correspond to high-pass filters. Instead of getting frequency information by averaging over the whole image, like the Fourier transform, the two methods fit local structural elements of increasing size to the image to preserve the local pixel interactions.

(2). The relationship between the autocorrelation function and the power spectrum is defined by a Fourier transform,

$$P(w_1, w_2) = \sum_{n_1 = 1}^{N} \sum_{n_2 = 1}^{N} C(n_1, n_2) e^{-jw_1 n_1} e^{-jw_2 n_2}, \tag{3.1}$$

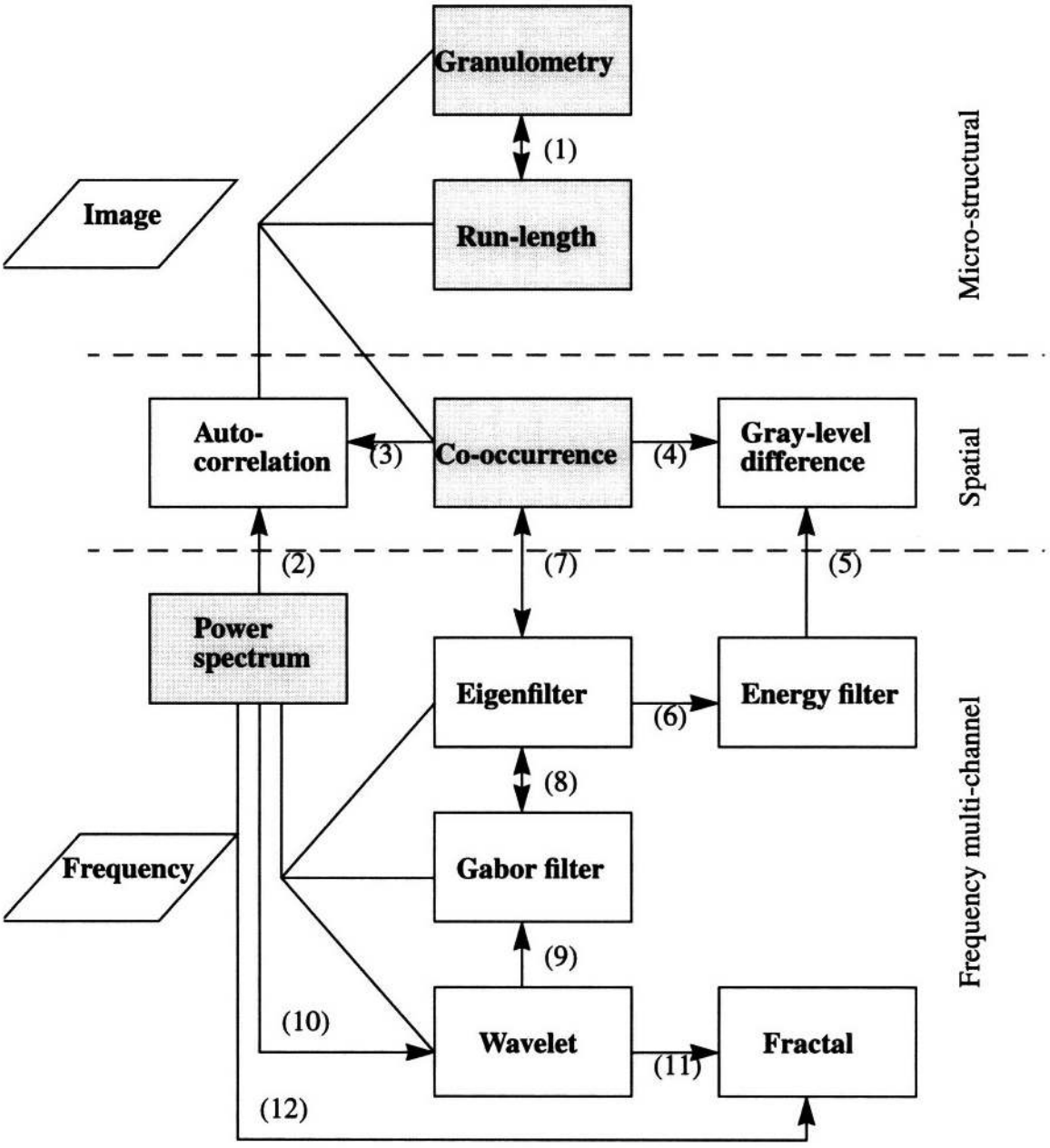


Figure 3.1: Interrelations among statistical texture analysis approaches.

where P and C are the power spectrum and autocorrelation function respectively. Since the power spectrum can also be computed directly from the original image using the Fast Fou-

rier Transform algorithm, as shown in Eqs. (2.2) and (2.3), the power spectrum method is in general preferred over the autocorrelation method.

(3). The autocorrelation function can also be computed from the co-occurrence matrix,

$$C(n_1, n_2) = \sum_{i=1}^{G} \sum_{j=1}^{G} M_{(n_1, n_2)}(i, j) \cdot i \cdot j, \tag{3.2}$$

where $M_{(n_1, n_2)}$ is the co-occurrence matrix for row shift n_1 and column shift n_2, as defined in Chapter 2. Equation (3.2) is in fact the definition of the autocorrelation function, with the co-occurrence matrix as the probability density function of gray-level variables i and j.

(4). As pointed out in Chapter 2, the gray level difference statistics can be computed from the co-occurrence matrix as well,

$$P_{(n_1, n_2)}(k) = \sum_{i=1}^{G} \sum_{\substack{j=1 \\ |i-j|=k}}^{G} M_{(n_1, n_2)}(i, j). \tag{3.3}$$

(5). The gray level difference features can also be seen as statistical features of images filtered by a set of edge-detection filters, as shown in Figure 3.2, for the four directional

$$\begin{bmatrix} 1 \\ -1 \end{bmatrix} \qquad \begin{bmatrix} 1 & -1 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \qquad \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

Figure 3.2: Gray level difference method filter masks.

nearest-neighbor gray level difference statistics. So the GLDM is also a multichannel processing method with filter banks similar to the energy filters in Figure 3.3 used by Laws [66],

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} -1 & 2 & -1 \\ -2 & 4 & -2 \\ -1 & 2 & -1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & -1 \\ 0 & 0 & 0 \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & -2 & 1 \\ 0 & 0 & 0 \\ -1 & 2 & -1 \end{bmatrix}$$

$$\begin{bmatrix} -1 & -2 & -1 \\ 2 & 4 & 2 \\ -1 & -2 & -1 \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & -1 \\ -2 & 0 & 2 \\ 1 & 0 & -1 \end{bmatrix} \quad \begin{bmatrix} 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \end{bmatrix}$$

Figure 3.3: Laws 3x3 energy filter masks.

(6). The design of Laws' texture energy filters is based on the observation that edge information is important for texture. Several directional edge filters and blob detectors are used to filter the image.

The eigenfilter method is a more formal way to design such directional filters. To compute the eigenfilter masks, I use the 256x256 texture image "pebble23" from Brodatz [9] shown in Figure 3.4. For each image pixel $f(n_1, n_2)$, we line scan the 3x3 neighborhood to form a 9-component feature vector $\boldsymbol{f}(n_1, n_2)$,

$$\begin{aligned} \boldsymbol{f}(n_1, n_2) = [&f(n_1 - 1, n_2 - 1), f(n_1 - 1, n_2), f(n_1 - 1, n_2 + 1), \\ &f(n_1, n_2 - 1), f(n_1, n_2), f(n_1, n_2 + 1), \\ &f(n_1 + 1, n_2 - 1), f(n_1 + 1, n_2), f(n_1 + 1, n_2 + 1)] \end{aligned} \tag{3.4}$$

Then the covariance matrix is estimated by,

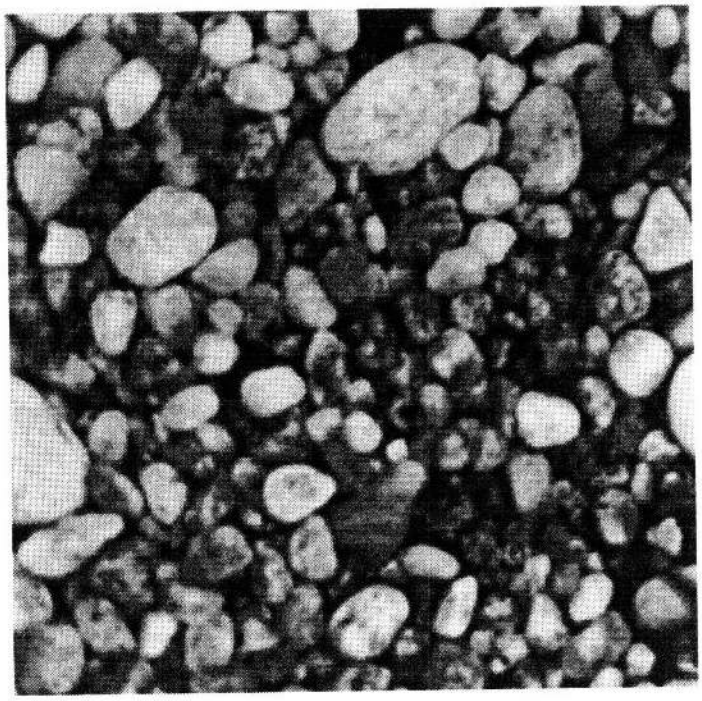


Figure 3.4: Brodatz texture "pebble23".

$$W = \frac{1}{N^2} \sum_{n_1 = 1}^{N} \sum_{n_2 = 1}^{N} (f(n_1, n_2) - \mu)(f(n_1, n_2) - \mu)^T, \tag{3.5}$$

where μ is the mean vector of $f(n_1, n_2)$. The nine eigenvectors computed from W form the 3x3 eigenfilter masks in Figure 3.5. Except for a scale factor, the eigenfilters are very similar to Laws' filters in Figure 3.3. The difference between the two approaches is that Laws uses a set of empirical filters based on experiments, but the eigenfilter approach lets the principle component analysis algorithm generate filter masks that reflect more closely the local texture structures [2].

(7). The eigenfilters are also closely related to the co-occurrence matrix [2]. For the 3x3 neighborhood vector in Eq. (3.4), the co-occurrence matrices are the second-order joint-probability functions of any two pixel elements in the vector. For a Gaussian process, such second order statistics are fully determined by the covariance matrix W in Eq. (3.5), which is equivalent to the diagonalized covariance matrix with the variances of the eigen-

$$\begin{bmatrix} 0.32 & 0.34 & 0.33 \\ 0.34 & 0.35 & 0.34 \\ 0.33 & 0.34 & 0.32 \end{bmatrix} \quad \begin{bmatrix} -0.48 & -0.11 & 0.29 \\ -0.42 & 0 & 0.42 \\ -0.29 & 0.11 & 0.48 \end{bmatrix} \quad \begin{bmatrix} -0.22 & 0.33 & -0.36 \\ -0.1 & 0.64 & -0.1 \\ -0.36 & 0.33 & -0.22 \end{bmatrix}$$

$$\begin{bmatrix} 0.28 & 0.42 & 0.48 \\ -0.1 & 0 & 0.1 \\ -0.48 & -0.42 & -0.28 \end{bmatrix} \quad \begin{bmatrix} 0.54 & -0.01 & -0.45 \\ -0.01 & -0.12 & -0.01 \\ -0.45 & -0.01 & 0.54 \end{bmatrix} \quad \begin{bmatrix} -0.43 & 0.46 & -0.09 \\ 0.3 & 0 & -0.3 \\ 0.09 & -0.47 & 0.43 \end{bmatrix}$$

$$\begin{bmatrix} -0.1 & -0.40 & -0.17 \\ 0.5 & 0.31 & 0.51 \\ -0.17 & -0.40 & -0.11 \end{bmatrix} \quad \begin{bmatrix} -0.09 & -0.31 & 0.42 \\ 0.47 & 0 & -0.47 \\ -0.42 & 0.31 & 0.09 \end{bmatrix} \quad \begin{bmatrix} 0.21 & -0.34 & 0.19 \\ -0.35 & 0.6 & -0.35 \\ 0.19 & -0.35 & 0.21 \end{bmatrix}$$

Figure 3.5: Eigenfilter masks computed from the texture image in Figure 3.4.

filtered images on the diagonal. Since it is impractical to compute all possible co-occurrence matrices, the eigenfilter variance features are a good simplification of co-occurrence features for near-Gaussian texture processes.

(8) & (9). Essentially, the texture energy method, the eigenfilter method, and the Gabor filter method are different filter design approaches, sharing the same basic idea of filtering the texture information into multiple channels. The only difference is between their filter design principles.

Although the Gabor filter can be tuned to various directions and frequencies, it suffers the same problem as the energy and eigenfilter methods: the output is overcomplete. Figure 3.6 compares these multichannel approaches with the wavelet and Fourier transform for one-dimensional signals. Figure 3.6 (a) shows the time domain representation. Figure 3.6 (b) shows the multichannel representation with the same number of signal elements as the original signal in every channel, which generates much redundant information. The wavelet representation in Figure 3.6 (c) is a compact multichannel representation with the same number of decomposition bases as the original signal. I choose the wavelet transform as a representative for the multichannel texture classification method to conduct further study in Chapter 4.

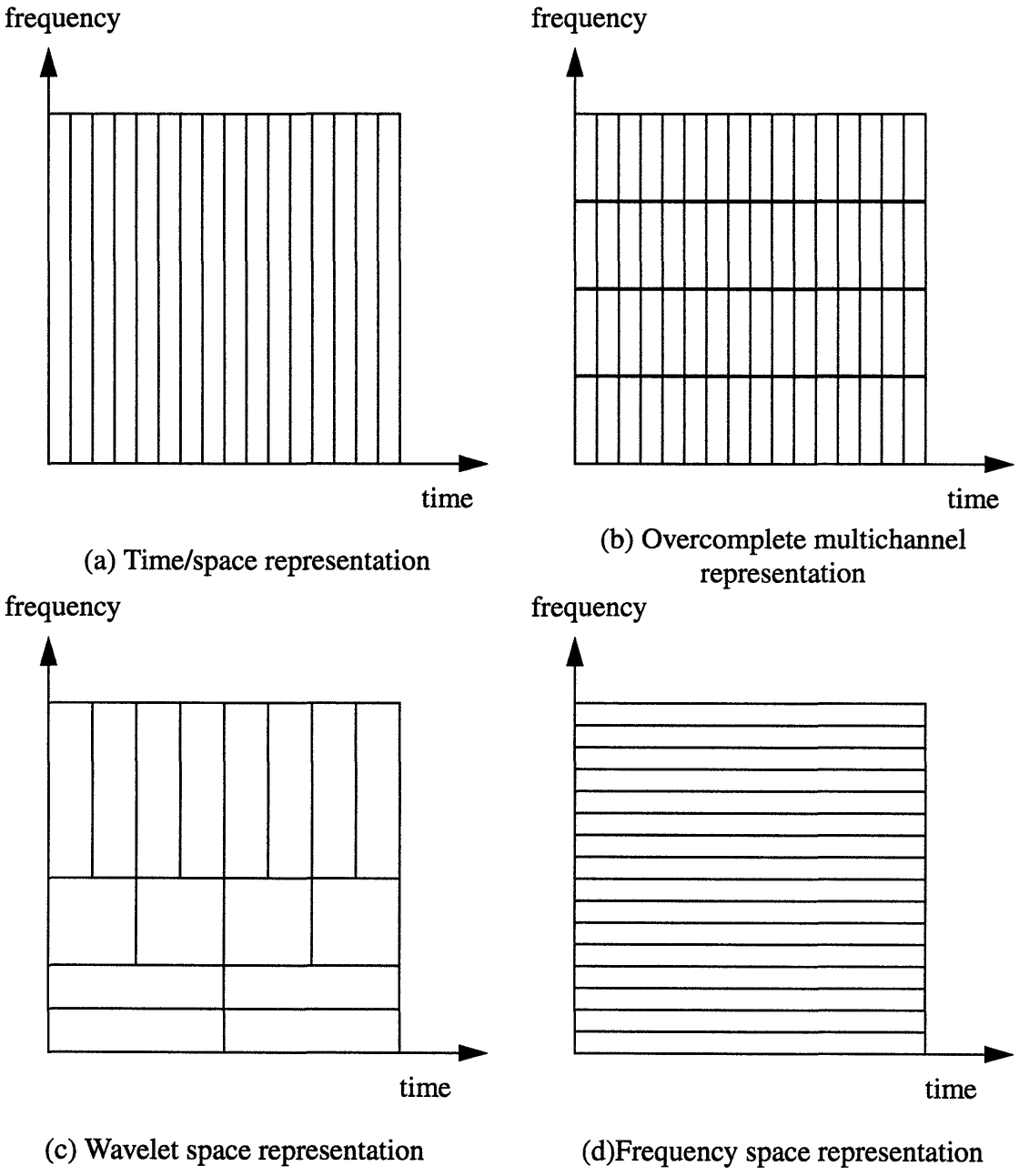


Figure 3.6: Illustration of time-space and multichannel frequency space representations of a one-dimensional signal.

(10). Figure 3.6 (d) illustrates that the Fourier transform can be treated as the highest level of a multichannel decomposition. Instead of painstakingly designing various filters, I take the maximum number of filter channels that can be obtained, then use the principle

component analysis technique to decide which channels to keep and how much of the energy in those channels to keep. The resultant coefficients, which represent the magnitude of each frequency channel's contribution, form a designed filter. In Chapter 4, I demonstrate the superiority of this novel approach.

(11) & (12). Fractals can be computed both from the wavelet transforms and the Fourier transforms. Mallat [71] proves that the ratio between wavelet detail signal energy is a constant related to the fractal dimension,

$$2^{2(3-D)} = \frac{\sigma_j^2}{\sigma_{j+1}^2}, \tag{3.6}$$

where σ_j^2 and σ_{j+1}^2 are the energy of the detail signal of the j and j+1 levels of wavelet decomposition, and D is the fractal dimension. As shown in Chapter 2, the fractal dimension can also be estimated from the power spectrum. Chaudhuri and Sarkar [14] computed fractal dimensions of several filtered images. Their method is equivalent to fitting the piece-wise slope of a power spectrum. So, the fractal features are simply features reflecting properties of the Fourier power spectrum.

Based on the above analysis of the interrelations of the 11 statistical texture analysis approaches, we can group them into three categories as shown by the dotted line in Figure 3.1: the spatial domain method, the micro-structural method, and the frequency multichannel method. The three methods capture texture information of increasing coherence length or correlation distance: the spatial domain features capture the interaction of the local neighborhood pixels; the frequency features capture the information resulting from averaging the whole image over all frequency channels. The micro-structural features fit nicely in the middle, capturing the medium-scale interactions of pixels.

The shaded methods in each of the three categories in the figure are the power spectrum method, the co-occurrence method, the run-length method, and the granulometry method. I conduct an in-depth study of these in the following four chapters, because they each offer a unique representation of textures that is not a subset of other methods.

It is generally agreed that texture classification methods are application dependent. One method may be the best for a particular set of textures but may fail on others. This sit-

uation is very similar to that described by an excellent analogy made by Meyer [77] in his discussion of optimal wavelet algorithms in signal processing:

> Each algorithm is presented in terms of a particular orthogonal basis. We can compare searching for the optimal algorithm to searching for the best point of view, or best perspective, to look at a statue in a museum. Each point of view reveals certain parts of the statue and obscures others. We change our point of view to find the best one by going around the statue. In effect, we make a rotation; we change the orthonormal basis of reference to find the optimal basis.

It is important to choose the best viewpoint for different applications. However, the many possible applications of texture analysis in medical imaging, remote sensing, and computer vision make it impossible to identify the best method for every application. What we can do is to select a library of unique approaches and try to make improvements to them. The previous analysis of the interrelations of the 11 methods can serve as a starting point for building such a library. I have just identified four viewing directions to assemble the picture of a texture "statue". In the following chapters, I describe further studies and improvements to the views.

In fact, sometimes it is difficult to determine the best point of view. Why not take pictures from all the good points of view and assemble them together to get the whole picture. In other words, since different transforms make explicit different information, we can combine them together to make explicit all the information. This, of course, will generate overlapping information. We can then employ a decorrelation algorithm to remove the overlapping information.

3.2 Feature selection

The representation matrix after the transformation is still too large for a classifier. Feature selection must be applied to the matrix to make it a more concise description. I now address this second information condensing process in the texture classification system, as depicted in Figure 1.1.

For a long time, researchers have been focusing on developing ad hoc features using special functions, mostly based on intuitive observation of the shape and statistics of the matrix, such as the major peak of the power spectrum matrix [67], entropy of the co-occurrence matrix [48], or long-run emphasis of run-length matrix [42]. There are several drawbacks to this approach. First, there is no theoretical proof that, given a certain number of features, maximum texture information can be extracted from the matrices. The number of possible features is unlimited. Second, many features are highly correlated with one another, thus containing redundant information. Some methods, such as the PSM and the GLRLM, have been shown to perform poorly on texture classification even after a large number of features have been extracted. Criticisms of these methods have been aimed at the transforms rather than the ad hoc way of feature computation.

In this thesis, I intend to give the PSM and the GLRLM another chance to prove their relevance to texture representation. I show that the poor performance is not a result of the transformation itself but of the way that features are extracted from the matrix. Instead of developing more special functions to compute features, I apply a principle component analysis, also called the Karhunen-Loeve Transform (KLT), directly on the transform matrix to extract dominant texture features. By doing so all information in the matrix is optimally preserved in a small number of features. I then use the Bhattacharyya distance measure to sort the KLT-extracted features according to their discriminatory power.

The difference between this new approach and the conventional feature extraction approach can be described in the statue-viewing terms: the traditional method selects a promising viewpoint, then starts to describe the statue in words; instead, we pull out a camera and take pictures of the statue.

Sometimes a picture may be too large for the KLT transform to compute. I develop a novel Multi-level Dominant Eigenvector Estimation (MDEE) method to compute the KLT transform in an efficient way.

3.2.1 Principle-component analysis

To compute the Karhunen-Loeve Transform, let x_i be a feature vector sample. We form an n by m matrix

$$A = \begin{bmatrix} x_1(1) & x_2(1) & \dots & x_m(1) \\ x_1(2) & x_2(2) & \dots & x_m(2) \\ \dots & \dots & \dots & \dots \\ x_1(n) & x_2(n) & \dots & x_m(n) \end{bmatrix}, \tag{3.7}$$

where n is the feature vector length and m is the number of training texture samples. The eigenvalues of the sample covariance matrix are computed in two ways, depending on the relative size of the feature vector and on the training sample number. If the feature vector length n is a small number, eigenvalues are computed by a standard procedure. The sample covariance matrix is estimated by

$$W = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu)(x_i - \mu)^T = \frac{1}{m}AA^T, \tag{3.8}$$

where μ is the mean vector. The eigenvalues and eigenvectors are computed directly from W.

3.2.2 Multi-level Dominant Eigenvector Estimation

However, n can be a very large number in many situations. For instance, for the feature vector formed by the four directional run-length matrices of a texture sample of size 32x32 with 32 gray levels, n can reach a maximum of 4096. This means the covariance matrix is of size 4096 by 4096. Direct computation of the eigenvalues and eigenvectors becomes impractical. Fortunately, if the sample image number m is much smaller than n, the rank of W will only be m-1. A more efficient way to compute the eigenvalues is the dominant eigenvalue estimation method [41]. Consider the eigenvectors e_i of A^TA/m, such that

$$\frac{1}{m}A^TAe_i = \lambda_i e_i. \tag{3.9}$$

By multiplying both sides by A, we have

$$\frac{1}{m}AA^T(Ae_i) = \lambda_i(Ae_i), \tag{3.10}$$

$$W(Ae_i) = \lambda_i(Ae_i). \tag{3.11}$$

This shows that Ae_i are the eigenvectors of the covariance matrix W. Therefore, we can compute the eigenvectors of a small m by m matrix A^TA/m then calculate the first m eigenvectors of W as Ae_i.

In the case where both m and n are large, we divide the training samples into $g = m/k$ groups of vectors,

$$A = \left[\underbrace{\begin{matrix} x_1(1) & \dots & x_k(1) \\ x_1(2) & \dots & x_k(2) \\ \dots & \dots & \dots \\ \dots & \dots & \dots \\ x_1(n) & \dots & x_k(n) \end{matrix}}_{A_1} \quad \underbrace{\begin{matrix} x_{k+1}(1) & \dots & x_{2k}(1) \\ x_{k+1}(2) & \dots & x_{2k}(2) \\ \dots & \dots & \dots \\ \dots & \dots & \dots \\ x_{k+1}(n) & \dots & x_{2k}(n) \end{matrix}}_{A_2} \quad \cdots \cdots \quad \underbrace{\begin{matrix} x_{(g-1)k+1}(1) & \dots & x_m(1) \\ x_{(g-1)k+1}(2) & \dots & x_m(2) \\ \dots & \dots & \dots \\ \dots & \dots & \dots \\ x_{(g-1)k+1}(n) & \dots & x_m(n) \end{matrix}}_{A_g}\right], \tag{3.12}$$

and apply the algorithm described above on each one of the g sample groups A_i. Then, the k dominant eigenvalues and eigenvectors are computed as the average of the computed g groups of eigenvalues and eigenvectors.

However, there are several practical implementation difficulties with this grouping approach. The number of samples in each group must be large enough and the samples must be uniformly selected from the whole data set to capture the dominant distribution directions of the original data set, so that the dominant eigenvectors in each group approximate the dominant eigenvectors of the whole data set. Furthermore finding the corresponding eigenvectors among all groups is a nontrivial process.

To avoid these problems, I developed a new Multi-level Dominant Eigenvector Estimation (MDEE) method. Instead of grouping column vectors as in Eq. (3.12), I group the

matrix in the row direction. By breaking the long feature vector into $g = n/k$ groups of small feature vectors of length k,

$$A = \begin{bmatrix} B_1\left\{\begin{bmatrix} x_1(1) & x_2(1) & \dots & \dots & x_m(1) \\ \dots & \dots & \dots & \dots & \dots \\ x_1(k) & x_2(k) & \dots & \dots & x_m(k) \end{bmatrix}\right\} \\ B_2\left\{\begin{bmatrix} x_1(k+1) & x_2(k+1) & \dots & \dots & x_m(k+1) \\ \dots & \dots & \dots & \dots & \dots \\ x_1(2k) & x_2(2k) & \dots & \dots & x_m(2k) \end{bmatrix}\right\} \\ \dots \quad \dots \quad \dots \quad \dots \quad \dots \\ \dots \quad \dots \quad \dots \quad \dots \quad \dots \\ B_g\left\{\begin{bmatrix} x_1((g-1)k+1) & x_2((g-1)k+1) & \dots & \dots & x_m((g-1)k+1) \\ \dots & \dots & \dots & \dots & \dots \\ x_1(n) & x_2(n) & \dots & \dots & x_m(n) \end{bmatrix}\right\} \end{bmatrix}, \quad (3.13)$$

we can perform the KLT on each of the g group short feature vector set B_i. Then a new feature vector is formed by the first few selected dominant eigenfeatures of each group. The final eigenvectors are computed by applying the KLT to this new feature vector. To prove that the eigenvalues computed by MDEE are a close approximation of the standard KLT, I study the two-group case here. The feature vector matrix and its covariance matrix are

$$A = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}, \quad (3.14)$$

$$W = AA^T = \begin{bmatrix} B_1B_1^T & B_1B_2^T \\ B_2B_1^T & B_2B_2^T \end{bmatrix} = \begin{bmatrix} W_1 & W_{12} \\ W_{21} & W_2 \end{bmatrix}. \quad (3.15)$$

The averaging coefficients are omitted in the equations for simplicity. Let the eigenvector matrices of the covariance matrices W_1 and W_2 be T_1 and T_2 respectively, then

$$T_1^T W_1 T_1 = \Lambda_1, \tag{3.16}$$

$$T_2^T W_2 T_2 = \Lambda_2. \tag{3.17}$$

where Λ_1 and Λ_2 are the diagonal eigenvalue matrices. The effective rotation matrix for the first-step group KLT is

$$T = \begin{bmatrix} T_1 & 0 \\ 0 & T_2 \end{bmatrix}. \tag{3.18}$$

T is also an orthogonal matrix, since

$$T^T T = \begin{bmatrix} T_1^T T_1 & 0 \\ 0 & T_2^T T_2 \end{bmatrix} = I \tag{3.19}$$

So, after the first-step group KLT, the covariance matrix of the rotated feature vector,

$$W_r = T^T W T = \begin{bmatrix} \Lambda_1 & T_1^T W_{12} T_2 \\ T_2^T W_{21} T_1 & \Lambda_2 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} \Lambda_{1b} & 0 \\ 0 & \Lambda_{1s} \end{bmatrix} \begin{bmatrix} C_{bb} & C_{bs} \\ C_{sb} & C_{ss} \end{bmatrix}^T \\ \begin{bmatrix} C_{bb} & C_{bs} \\ C_{sb} & C_{ss} \end{bmatrix} \begin{bmatrix} \Lambda_{2b} & 0 \\ 0 & \Lambda_{2s} \end{bmatrix} \end{bmatrix}, \tag{3.20}$$

is a similar matrix of the original feature vector covariance matrix *W*, because of the orthogonality of the rotation matrix *T*. Since similar matrices have the same eigenvalues, we can use the right most term of Eq. (3.20) to discuss the impact on *W* of keeping only the first few dominant eigenvalues in each group. In Eq. (3.20), Λ_{nb} and Λ_{ns} represent the larger dominant eigenvalue section and the smaller negligible eigenvalue section of the eigenvalue matrix Λ_n respectively, for $n = 1$ or 2. C_{xx}, where $x = b$ or s, represents the

cross-covariance matrix of the two groups of rotated features. By keeping only the dominant eigenvalues, the new feature vector covariance matrix becomes

$$W_d = \begin{bmatrix} \Lambda_{1b} & C_{bb}^T \\ C_{bb} & \Lambda_{2b} \end{bmatrix}. \tag{3.21}$$

The terms removed from W_r are Λ_{1s}, Λ_{2s}, C_{ss}, C_{bs} and C_{sb}. Since most energy is contained in the dominant eigenvalues, the loss of information due to Λ_{1s} and Λ_{2s} should be very small. The energy contained in the cross-covariance matrix of the two small energy feature vectors, C_{ss}, should therefore be even smaller.

We can also prove that C_{bs} and C_{sb} cannot be large either. If the two group features B_1 and B_2 are fairly uncorrelated with each other, then all the cross-covariance C_{xx} matrices in Eq. (3.20) will be very small. On the other hand, if the two group features are strongly correlated with each other, the dominant eigenfeatures of the two group will be very similar. Therefore the cross-covariance matrix C_{bs} of group-two large features with group-one small features will be similar to the cross-covariance matrix of the group-one large features with group-one small features, which is zero due to the decorrelation property of the KLT transform.

When the two group features B_1 and B_2 are partially correlated, the correlated part should be mostly signal, since noise parts of the variable B_1 and B_2 rarely correlate with each other. The basic property of the KLT is to preserve all signal energy in the first few large eigenvalues. Therefore, most signal energy in B_2, and especially most of the B_2 signal energy that is correlated with B_1, will be preserved in the large eigenvalue section of B_2 covariance matrix. The energy that is discarded in the small eigenvalue section of B_2 will contain little if any energy that is correlated with B_1. Therefore, C_{bs} and C_{sb} should be very small, and we will not lose much information by removing them from the covariance matrix W_r.

Now that we have seen that the covariance matrix W_d is a close approximation of W_r, and W_r is a similar matrix of W, we can say that the eigenvalues from W_d, i.e., by the

MDEE method, are indeed a close approximation of the eigenvalues computed from W, i.e., by the standard KLT method.

Significant reduction of computational time can be achieved by the MDEE over the standard KLT. Figure 3.7 demonstrates an example, where a feature vector of length n = 1000 is broken into 10 vector groups of length 100. With 10% of the eigenfeatures in each group saved for the second-level eigenvalue computation, the computational complexity for the MDEE is $11(n/10)^3$, nearly two orders of magnitude faster than the KLT. Furthermore, the algorithm offers an excellent opportunity for parallel computation. If all individual group KLTs are computed in parallel, a near three-order-of-magnitude speed increase can be achieved for the above example.

3.2.3 Statistical distance measure

It is well known that the KLT features are optimal for data representation. However, they are not necessarily the best for discrimination. To measure the class separability of each feature, some other criterion must be employed. I choose the Bhattacharyya distance measure in this work because it has a direct relation with the error bound of the Gaussian classifier and has a simple form for features with normal distributions. As indicated by Fukunaga [41], for a two-classes problem

$$\varepsilon_{(c_1, c_2)} \leq [P(c_1)P(c_2)]^{\frac{1}{2}} \exp[-\beta_{d(c_1, c_2)}], \tag{3.22}$$

where $P(c_i)$ is the prior probability of class c_i, ε is the probability of error for a Gaussian classifier and β_d is the Bhattacharyya distance. Because its inverse gives the upper bound on the probability of error, β_d can be an effective measure of class separability. For a normal distribution, β_d has the analytical form

$$\beta_{d(c_1, c_2)} = \frac{1}{8}(\mu_1 - \mu_2)^T \left(\frac{W_1 + W_2}{2}\right)^{-1} (\mu_1 - \mu_2) + \frac{1}{2} \ln \frac{\left|\frac{1}{2}(W_1 + W_2)\right|}{|W_1|^{1/2} |W_2|^{1/2}}. \tag{3.23}$$

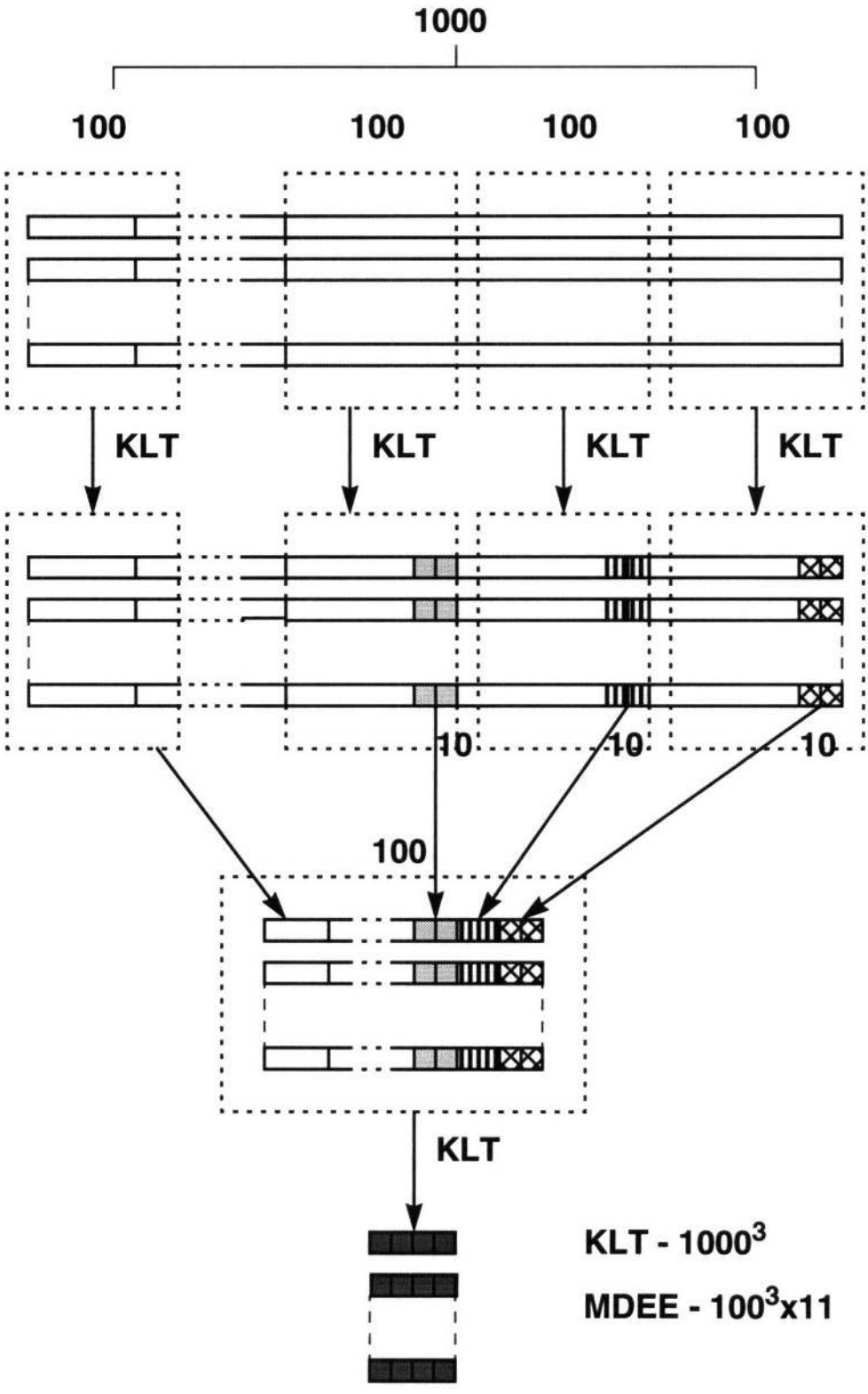


Figure 3.7: Multi-level Dominant Eigenvector Estimation (MDEE).

where μ_1, μ_2 and W_1, W_2 are the mean vectors and covariance matrices of the two class distributions. The many possible combinations of several features and the possibility of covariance matrix singularity make it impractical to compute the Bhattacharyya distance for several features at once. The one-at-a-time method is adopted instead. The formula is the same as Eq. (3.23), only with the covariance matrix W replaced by the variance and the mean vector μ replaced by the class mean. Figure 3.8 shows the situations in which a large β_d can be computed for two clusters. In (a), the two clusters have a large difference in mean values, resulting in a large value for the first term of Eq. (3.23). In (b), the two clusters differentiate in variances, so the large value of the second term of Eq. (3.23) still gives a large output of β_d. Figure 3.8 (c) illustrates the situation in which both means and variances are different for the two classes, and β_d is at its greatest. In all three cases, a Gaussian classifier is expected to give good performance.

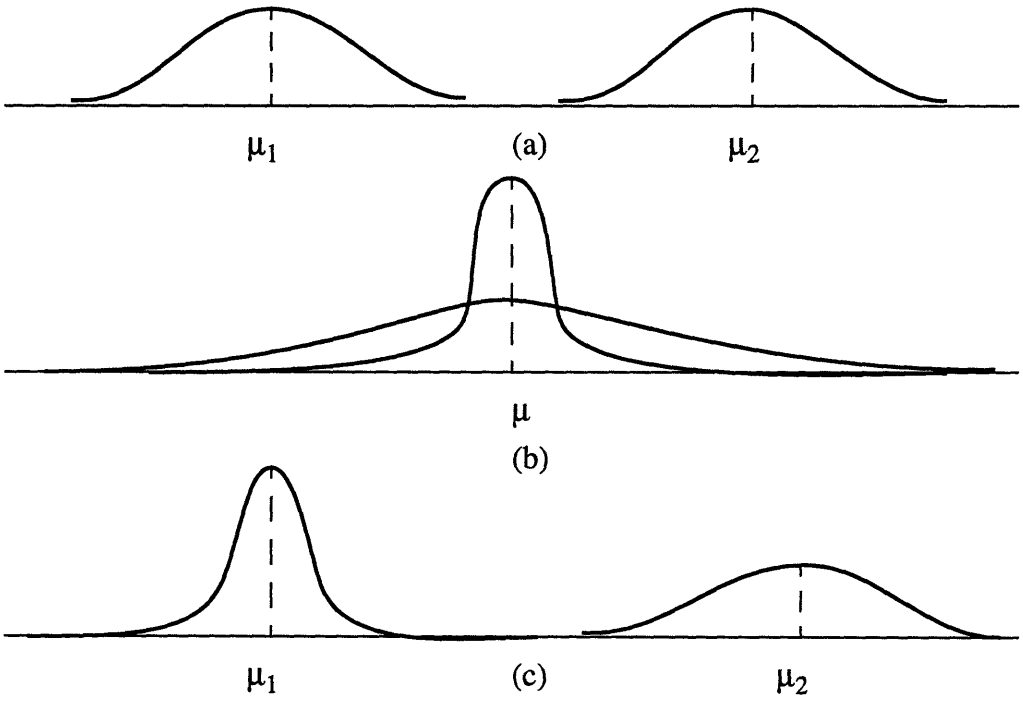


Figure 3.8: Illustration of the three situations when the Bhattacharyya distance measures between two class distributions are large: (a) large mean difference, (b) large variance difference, (c) large differences in both means and variances.

For multi-class problems, the overall probability of error can be bounded by [41]

$$\varepsilon \leq \sum_{i>j}^{M} \sum_{j=1}^{M} \varepsilon_{(c_i, c_j)}, \tag{3.24}$$

where ε and $\varepsilon_{(ci,cj)}$ ($i, j = 1,2,..., M$) are the probabilities of overall error and the pair-wise error between class i and j respectively. From Eqs. (3.22) and (3.24) we select features according to the minimum total upper error bound. Because the test data size is the same for all classes in our texture classification experiment, the prior probabilities $P(c_i)$ are equal for all classes. Thus, we select features with small values of

$$S_b = \sum_{i>j}^{M} \sum_{j=1}^{M} \exp[-\beta_{d(c_i, c_j)}]. \tag{3.25}$$

Throughout the experiments in the rest of the thesis, I select the first 30 features with larger eigenvalues, rank these KLT-decorrelated features by their S_b values, and use the first n features with the smallest S_b for classification. I run the feature length n from 1 to 30 to select the one that gives the best performance as the final feature vector length. This is apparently not an optimal searching approach, since a combination of the first n best individual features may not be the best length n feature vector. However, the experimental results suggest that it is a very close approximation. Since all features are first decorrelated by the KLT transform, as we increase the feature length each additional feature brings in new uncorrelated information and noise. When their S_b values increase to a certain point, the new features start to bring in more noise than information, suggesting that a suboptimal feature length is reached. The experiments show that most best feature lengths are between 10 and 20.

3.3 Statistical and neural network classifiers

3.3.1 Statistical classifier

Since our main focus in this thesis is the feature extraction algorithm instead of the classifier, I use a simple Gaussian classifier for most texture classification experiments.

I assume the feature vector x for each class c_i is a Gaussian distribution with mean m_i and covariance W_i. The distance measure is defined as [105]

$$D_i = (x-\mu_i)^T W_i^{-1}(x-\mu_i) + \ln|W_i| - 2\ln(P_i), \tag{3.26}$$

where P_i is the prior probability. The first term on the right of the equation is actually the Mahalanobis distance. The decision rule is

$$x \in c_L \quad \text{when } D_L = min\{D_i\}. \tag{3.27}$$

3.3.2 Neural network classifier

For the plankton classification application, I use the learning vector quantization classifier (LVQ) [63] [64] for feature classification because it makes weaker assumptions about the shapes of underlying feature vector distributions than traditional statistical classifiers. For this reason, the LVQ classifier can be more robust when distributions are generated by nonlinear processes and are strongly non-Gaussian, precisely the case for the plankton data.

A vector quantization process optimally allocates M codebook reference vectors, $\omega_i \in R^n$, to the space of n-dimensional feature vectors, $x \in R^n$, so the local point density of the ω_i can be used to approximate the probability density function $p(x)$ [63]. Consequently, the feature vector space is quantized into many subspaces around ω_i, the density of which is highest in those areas where feature vectors are more likely to appear and more coarse in those areas where feature vectors are scarce.

To use such a vector quantization process in a supervised pattern classification application, Kohonen [63] [64] developed the Learning Vector Quantization classifier. First, the codebook reference vectors ω_i are initialized by either M random feature vector samples or the mean value of the feature vectors. They are then assigned to a fixed number of known application-specific classes. The relative number of codebook vectors assigned to each class must comply with the a priori probabilities of the classes. A training algorithm is then used to optimize the codebook vectors. Let the input training vector x belong to class C_t, and its closest codebook vector ω_r labeled as class C_s. The codebook vector ω_i is updated by the learning rules [63],

$$\begin{aligned} \Delta\omega_r &= \alpha(x-\omega_r) && \text{if } C_s = C_t \\ \Delta\omega_r &= -\alpha(x-\omega_r) && \text{if } C_s \neq C_t \\ \Delta\omega_i &= 0 && \text{for } i \neq r \end{aligned} \quad , \tag{3.28}$$

where α is the learning rate. Only the closest of the vectors ω_i is updated, with the direction of the correction depending on the correctness of the classification. Effectively, these codebook vectors are pulled away from zones where misclassifications occur, i.e., away from the classification boundary region. After training, the nearest neighbor rule is used to classify the input test vector according to the class label of its nearest codebook vector.

A neural network architecture to implement the LVQ is shown in Figure 3.9. The network consists of two layers, a competitive layer and a linear output layer. The weights connecting all input neurons with the competitive layer neuron i form the codebook vector ω_i. The net input for each competitive layer neuron is the Euclidean distance between the input vector x and the weight vector ω_i. The output of each neuron is 0 except for the "winner" neuron, whose weight vector has the smallest distance to the input vector and whose output is 1.

The second layer transforms the competitive layer's neuron class into the final output class. As discussed above, the competitive layer neurons, i.e., the codebook vectors, are assigned to the output classes according to a priori probabilities of the classes. For the example shown in Figure 3.9, the first three neurons are assigned to Class 1, the next two to Class 2, and the final two to Class 3. Only the non-zero weight connections between the

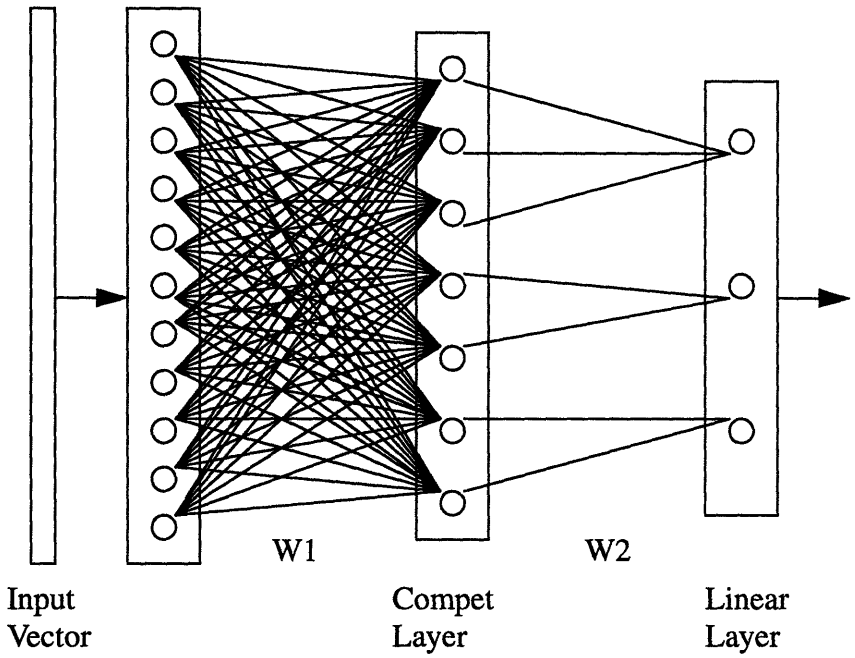


Figure 3.9: Learning vector quantization neural network.

competitive layer and the linear layer are shown in the figure. If any neuron in a particular class wins, the corresponding neuron class in the linear output layer will have an output of 1.

A drawback of the LVQ algorithm is the time-consuming training process. To improve it, I developed a statistical initial condition and a parallel training strategy. First, I initialize the neuron weight vectors in the competitive layer using the means and variances of the training vectors in each class. With the initial weight vectors in each class computed by adding random vectors of the class variance to the mean vector of the same class, the training process starts from a good statistical mapping position.

I make a second improvement by invoking a novel parallel training strategy. Traditionally, training samples are randomly selected to be fed into the network one at a time. Therefore, only one neuron is updated at each training epoch. The process is quite slow, especially when there are many neurons to update. That the order in which samples are presented to the network is not important suggests the idea of presenting several training samples in parallel. The only difference this may create is that there may be more than one

training sample very near the same neuron. In such a case I select only one of these few samples to update the neuron. This guarantees that parallel processing generates only one update operation on each winning neuron, with results similar to a serial training strategy. The parallel processing can be implemented by either hardware matrix operations or multiprocessor technology. The experimental results in Chapter 7 demonstrate the effectiveness of this method.

3.4 Summary

In this chapter, I offer new insights for each of the three components of a texture classification system: texture image transformation, feature extraction, and classification. Based on the extensive survey in Chapter 2, I conduct a unique theoretical investigation of texture analysis and study the interrelations among 11 types of texture analysis methods. A novel unification of the different methods defines a framework of transformation and representation in which three major classes of transform matrices capture texture information of increasing coherence length or correlation distance: the spatial domain method, the micro-structural method, and the frequency multichannel method.

A more concise vector representation of a selected transform matrix is then needed for input to a classifier. Unlike traditional methods, which use various special functions to describe the properties of each transform matrix, a new approach directly applies a principle component analysis technique to the transform matrix. The Karhunen-Loeve Transform extracts a vector of dominant features, optimally preserving texture information in the matrix. This approach is made possible by the introduction of a novel Multi-level Dominant Eigenvector Estimation algorithm, which reduces the computational complexity of the standard KLT by several orders of magnitude. The statistical Bhattacharyya distance measure is then used to rank dominant features according to their discrimination power.

In most classification experiments in the following chapters, a simple statistical Gaussian classifier is used. The plankton object recognition experiments use a Learning Vector Quantization (LVQ) neural-net classifier to achieve superior performance on the highly nonuniform plankton database. In this chapter, by introducing a new parallel LVQ learning scheme, the speed of network training is dramatically increased.

In the next three chapters, I apply the new feature extraction algorithms developed here on the three types of texture transform matrices—the frequency matrix, the run-length matrices, and the co-occurrence matrices—and improve the traditionally considered least efficient texture analysis methods, such as the power spectrum method and the run-length method, to be among the best.

Chapter 4

Frequency transform texture classification

In this chapter, I study the frequency transform texture classification method described in Chapter 3. I investigate and compare three methods here: the wavelet method, the traditional power spectrum method (PSM), and the new dominant spectrum method (DSM).

As mentioned earlier, recent advances in wavelet and wavelet packet theory [18] [24] [25] [71] [72] [77] [94] provide a promising multi-channel image processing technique. The texture research community is currently devoting considerable effort to wavelet applications in texture analysis and has achieved encouraging success [12] [13] [51] [65] [109]. However, most approaches are direct applications of existing wavelet processing algorithms, which are ideal for signal representation but not necessarily best for signal classification. To fully utilize the power of a wavelet packet transform, new techniques tailored to extracting features of greater discrimination ability must be developed. In this thesis, I use the feature selection algorithm described in Chapter 3 to combine and select frequency-channel features that give improved classification performance.

Since the Fourier transform can be considered as one of the highest possible level of multi-channel decompositions, it is reasonable to apply the same feature selection algorithms to the Fourier transform power spectrum. Just as the ideal tool for nonstationary signal analysis is a wavelet transform, the ideal tool for stationary signal analysis is a Fourier transform. Because texture signals are mostly stationary, I suspect that the Fourier transform power spectrum features may generate better results. I also compare the new Fourier features with the traditional power spectrum method and show that by using appropriate feature extraction algorithms, the discrimination power of the Fourier transform features can be significantly improved.

Section 4.1 of this chapter describes the texture feature extraction techniques, including the computation of wavelet features, the Fourier transform features, and the traditional PSM features. The texture classification experimental results are presented in Section 4.2. I summarize the conclusions in Section 4.3.

4.1 Texture feature extraction

4.1.1 Wavelet and wavelet packet transforms

For simplicity, a one-dimensional discrete signal $f(k)$ of length $n = 2^{n_0}$ is used for discussion in this section. The standard wavelet transform can be thought of as a smooth partition of the signal frequency axis. First, a lowpass filter $h(m)$ and a highpass filter $g(m)$, both of length M, are used to decompose the signal into two subbands, which are then downsampled by a factor of two. Let H and G be the convolution-downsampling operators defined as:

$$Hf(k) = \sum_{m=0}^{M-1} h(m) f(2k+m), \tag{4.1}$$

$$Gf(k) = \sum_{m=0}^{M-1} g(m) f(2k+m). \tag{4.2}$$

H and G are called perfect reconstruction quadrature mirror filters (QMFs) if they satisfy the following orthogonality conditions:

$$HG^* = GH^* = 0, \tag{4.3}$$

$$H^*H + G^*G = I, \tag{4.4}$$

where H^* and G^* are the adjoint (i.e., upsampling-anticonvolution) operators of H and G, respectively, and I is the identity operator.

This filtering and downsampling process is applied iteratively to the low-frequency subbands. At each level of the process, the high-frequency subband is preserved. When the process reaches the highest decomposition level, both the low- and high-frequency bands are kept. If the maximum processing level is L, the discrete wavelet coefficients of signal $f(k)$ are then $\{Gf, GHf, GH^2f, \ldots, GH^Lf, H^{L+1}f\}$ with the same length n as the original sig-

nal. Because of the orthogonality conditions of H and G, each level of transformation can be considered as a decomposition of the vector space into two mutually orthogonal subspaces. Let $V_{0,0}$ denote the original vector space R^n, and $V_{1,0}$ and $V_{1,1}$ be the mutually orthogonal subspaces generated by applying H and G to $V_{0,0}$. Then, the lth level of decomposition can be written as

$$V_{l,0} = V_{l+1,0} \oplus V_{l+1,1}, \tag{4.5}$$

for $l = 0, 1, ..., L$. Figure 4.1 shows such a decomposition process. Each subspace $V_{l,b}$ with $b = 0$ or 1 is spanned by $2^{n_0 - 1}$ wavelet basis vectors $\{\psi_{l,b,c}\}_{c=0}^{2^{n0-l}-1}$, which can be derived from H, G, and their adjoint operators. From the above iterative filtering operations, we can see that the wavelet transform partitions the frequency axis finely toward the lower frequency region. This is appropriate for a smooth signal containing primarily low frequency energy but not necessarily so for other more general types of signals, such as textures.

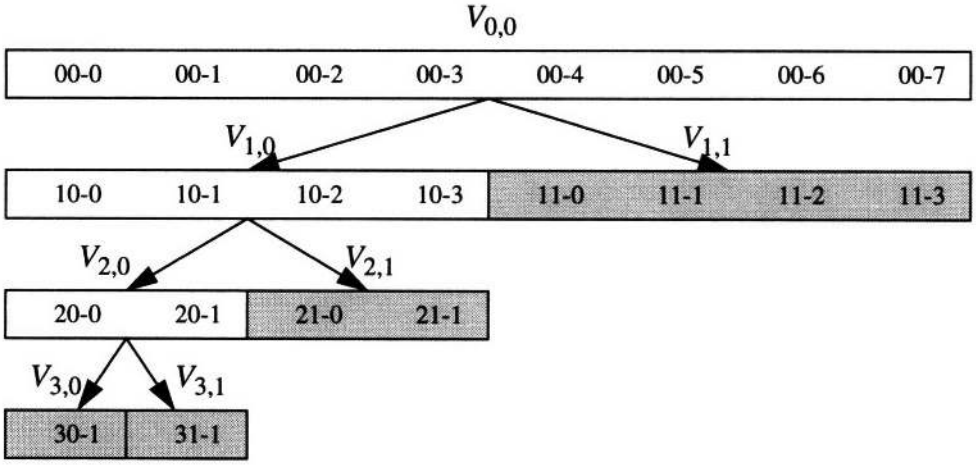


Figure 4.1: Standard wavelet transform binary tree.

A more generalized form of the standard wavelet transform is the wavelet packet transform, which decomposes both the high- and low-frequency bands at each iteration. Like the wavelet transform, two subbands, *Hf* and *Gf*, are generated at the first level of

decomposition. However, the second-level process generates four subbands, H^2f, GHf, HGf, and G^2f, instead of the two bands H^2f and GHf, as in the wavelet transform. If the process is repeated L times, Ln wavelet packet coefficients are obtained. In orthogonal subspace representation, the lth level of decomposition is

$$V_{l,b} = V_{l+1,2b} \oplus V_{l+1,2b+1}, \tag{4.6}$$

where $l = 0, 1, \ldots, L$ is the level index and $b = 0, \ldots, 2^l-1$ is the channel block index in each level. Figure 4.2 illustrates the wavelet packet decomposition of the original vector space $V_{0,0}$. Again, each subspace $V_{l,b}$ is spanned by 2^{n_0-l} basis vectors $\{W_{l,b,c}\}_{c=0}^{2^{n0-l}-1}$. For $b = 0$ and 1, W can be identified with ψ.

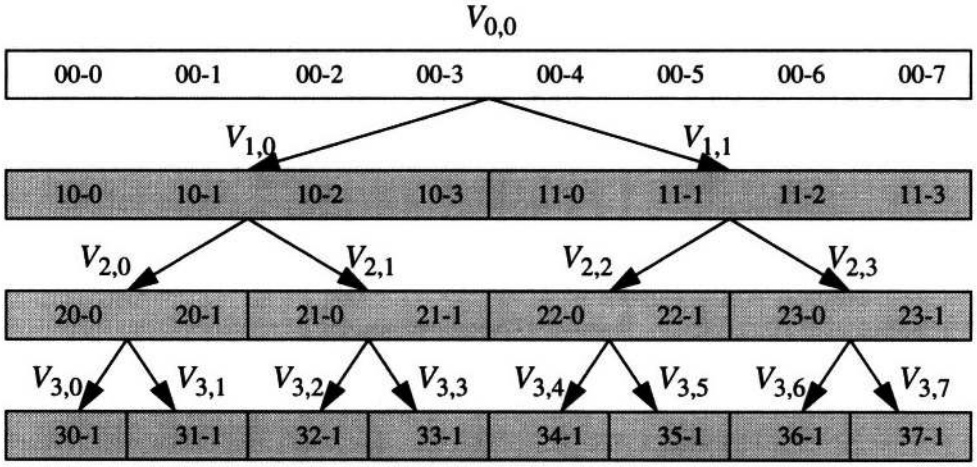


Figure 4.2: Wavelet packet transform binary tree.

For two-dimensional images, the wavelet or wavelet packet basis function can be expressed as the tensor product of two one-dimensional basis functions in the horizontal and vertical directions. The corresponding 2-D filters are thus:

$$h_{HH}(m,n) = h(m)h(n), \tag{4.7}$$

$$h_{HG}(m,n) = h(m)g(n), \tag{4.8}$$

$$h_{GH}(m,n) = g(m)h(n), \tag{4.9}$$

$$h_{GG}(m,n) = g(m)g(n). \tag{4.10}$$

In Figure 4.3, I show three sample textures and their wavelet packet coefficients for levels 1-4.

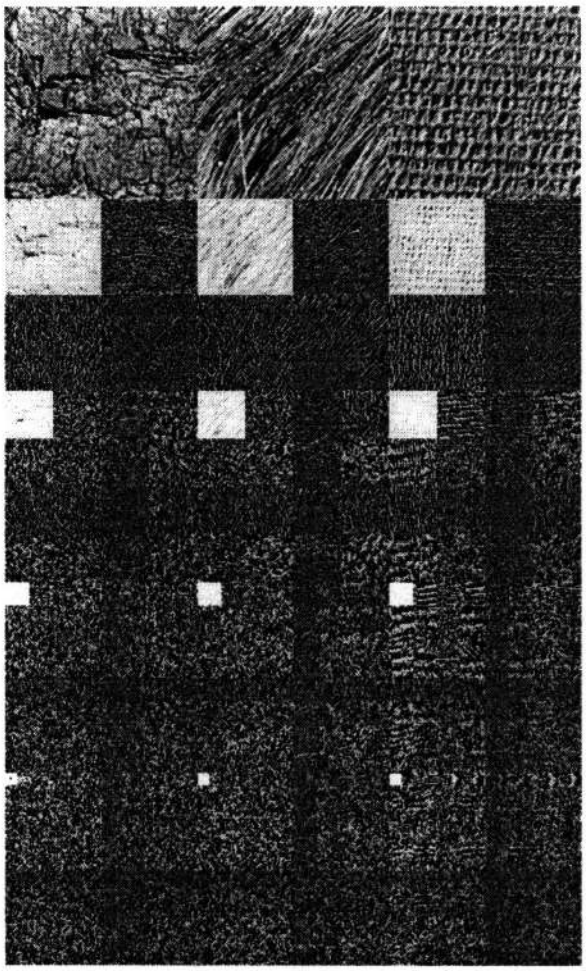


Figure 4.3: Three sample textures (row 1) and their wavelet packet coefficients at decomposition levels 1, 2, 3, and 4 (rows 2-5).

4.1.2 Wavelet texture feature extraction

After the wavelet packet coefficients are computed, I develop the algorithm by addressing the three main issues of multi-channel texture classification: feature extraction within each channel, channel selection, and feature combination among channels.

Since the wavelet coefficients are shift variant, they are not suitable for direct use as texture features, which must be shift-invariant. I choose to test the following shift invariant measures:

$$\mu = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N} x(i,j), \tag{4.11}$$

$$MNT_k = \frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N} (x(i,j)-\mu)^k, \tag{4.12}$$

$$ENT = -\sum_{i=1}^{M}\sum_{j=1}^{N} \frac{x(i,j)^2}{\|x\|_2^2}\log\left(\frac{x(i,j)^2}{\|x\|_2^2}\right), \tag{4.13}$$

where $x(i, j)$ denotes an element of the wavelet packet coefficient matrix x in each channel. To make the algorithm less vulnerable to the nonuniform illumination of images, the texture sample mean is removed before a feature vector is computed. Thus, the mean feature in Eq. (4.11) becomes zero. The four features I use in the experiment are: 1) variance feature VAR with $k = 2$ in Eq. (4.12), 2) the entropy feature ENT in Eq. (4.13), 3) the third moment MNT_3, and 4) the fourth moment MNT_4.

Because of the orthogonality condition of the wavelet transform, for the variance feature the following relation holds for any decomposition node and its four children nodes:

$$VAR_{l,b} = \frac{1}{4}\sum_{j=0}^{3} (VAR_{l+1,4b+j}). \tag{4.14}$$

The effect of this linear relationship on the classification accuracy of overcomplete wavelet packet features is seen in later experiments.

After the features are computed within each channel, the second issue is how to select good features among channels. One possible approach is to apply a statistical distance measure to each feature and to select those features with large distance measures. However, there are two drawbacks with this approach. The first is that neighborhood channel features tend to correlate with each other; thus, they contain similar information. If one has a large distance measure, the other will also, and both will be selected. Therefore, similar features will usually be selected. The second problem is that for some very small energy channels, a small amount of unexpected noise may cause the distance measure to be unrealistically large and the channel to be selected. To avoid these problems, I use the principal component analysis and statistical distance measures to combine the channel selection step and the channel combination step into one feature selection step.

The widely used Karhunen-Loeve transform is an appropriate feature reduction and selection procedure for the algorithm. The KLT decorrelates neighborhood channel features, and its energy packing property removes noisy channels and compacts useful information into a few dominant features. However, for a large feature vector, such as a vector comprising features of a higher level wavelet packet decomposition or a Fourier transform, the computation of the eigenvectors of the covariance matrix can be prohibitively expensive. I use the multilevel dominant eigenvector estimation method developed in Chapter 3 to overcome this problem then use the Bhattacharyya distance as a feature class separability measure to select the KLT decorrelated features.

In the following experiments, I test the algorithms on the following group of wavelet packet features:

1. Level 1: VAR, ENT, MNT_3,MNT_4, all,
2. Level 2: VAR, ENT, MNT_3, MNT_4, all,
3. Level 3: VAR, ENT, MNT_3, MNT_4, all,
4. Level 4: VAR, all,
5. Level 1&2: VAR, all,
6. Level 1&2&3: VAR, all,
7. Level 1&2&3&4: VAR,

8. Wavelet: VAR, all.

My goals are to test the discrimination power of each feature type in each individual level, the effects of overcomplete representation, and the classification power of the standard wavelet transform.

Another key experiment compares the texture classification performance of different wavelet filter design types and filter lengths. Since first being introduced, many types of wavelet filters have been proposed, but most previous texture analysis studies have only tested one particular type of wavelet filter design without any specific justification. In this Chapter, I demonstrate experimentally that the wavelet filter types and lengths do not, in fact, significantly affect texture classification performance.

4.1.3 Fourier transform features: Dominant spectrum method (DSM)

If we consider the Fourier transform as an extreme case of the wavelet packet transform, i.e., the highest possible level of wavelet packet decomposition, we can treat each Fourier frequency component as a frequency channel. Accordingly, we can treat the Fourier transform magnitude as an energy feature. Instead of painstakingly designing various multi-channel filters, we can take the maximum number of filter channels that can be obtained and then let the MDEE transform and the Bhattacharyya distance measure determine which channels to keep and how much of the energy in those channels to keep. The resulting coefficients, which represent the magnitude of each frequency channel's contribution, form a designed filter. I call this approach the dominant spectrum method (DSM). Since most texture images are generally stationary processes, which decompose canonically into a linear combination of sine and cosine waves in the same way that nonstationary signals decompose into linear combinations of wavelets, I expect DSM features to perform at least as well as the wavelet features. Only half the spectrum matrix is used because of the symmetric property of the Fourier transform of real functions.

The Fourier transform DSM features should not be confused with the traditional power spectrum method (PSM) described in Chapter 2. Early studies of PSM by Bajcsy [3] and Weszka et al. [120] concentrate on features computed by the summed spectral energy within circular or wedge-shaped frequency regions. These have been shown to perform

less well than most other texture analysis methods [19] [120]. Although Jernigan, et al. [55] [67] proposed to use entropy, peak, and shape measures to extract more texture features from the power spectrum, the performance improvement is limited, and the method is not widely accepted as an efficient texture analysis algorithm. Liu and Jernigan [67] gave an extensive summary of features that can be extracted by the PSM. In this chapter, I compare 20 features defined in [67] and listed in Appendix A.

Criticisms of the PSM have focused on use of the Fourier transform rather than on the way that texture features are computed from the power spectrum [120]. I believe that the fundamental problem with the PSM is that the feature extraction functions are, for the most part, ad hoc [102] [103], based on intuitive reasoning through human observation of the power spectrum shape. Instead of trying to develop more ad hoc features, I use the MDEE transform and Bhattacharyya distance as feature extraction algorithms to compute texture features from the Fourier transform magnitude matrix. All information in the spectrum is preserved and extracted in an optimal way. An experimental comparison of this approach with the PSM is presented in the following experiments.

Another problem addressed in this chapter is the so-called phase dilemma. In most previous research pertaining to PSM, researchers have tried to use the rich information contained in the Fourier transform phase [36] [67] [118]. This is mainly attributed to successes in the study of image reconstruction from partial Fourier transform information, where the Fourier phase has been shown to contain more information than the Fourier magnitude [21] [50] [80] [81]. So far, however, all results in texture analysis research reach the same conclusion that the textural content of the phase information is low.

In fact, the answer to this contradiction is imbedded in the basic property of the Fourier transform. The Fourier transform phase carries vital information representing the relative position of the harmonic basis functions essential for the image reconstruction. In most image reconstruction work, the images studied are all natural images with such large, smooth areas as sky or black background. The general shapes of their Fourier transform magnitudes are quite similar, with most energy concentrated in the low-frequency area. When only the phase information is used for reconstruction, usually an average of the Fourier magnitudes of many other irrelevant images is used as the initial magnitude. Thus, except for the small differences in the magnitude of some frequency components, most

overall structural information, i.e., the positions of all basis functions, are still there with small changes in magnitude. So the images can be mostly reconstructed with only gray scale changes in certain places.

For texture classification, the situation is completely different. Phase information is a special property of each individual texture sample, as important as that of the natural images used in reconstruction studies. However, since individual texture samples are delineated by an arbitrary grid definition within any class of image, statistically, the phase signals for all texture samples are essentially random regardless of texture classes and thus offer no discrimination power for texture classification. As discussed in Chapter 3, the phase signal is exactly the kind of noise signal we try to remove.

Although the absolute phase values are noise signals, the phase differences between different frequencies reflect the relative positions of the harmonic functions and thus may offer useful texture information. However, another important but overlooked property of the phase signal prevents this information from being extracted. No matter how small the energy of a frequency component, its phase value can be anything in a period of 2π. Even though this phase value may be controlled largely by random noise, it still has a value comparable with the phase value of the frequency component having the largest energy. This equal value property essentially renders the phase information useless for texture classification, because it makes the extraction of relative phase information impossible.

To confirm this analysis experimentally, I extract texture features by applying the KLT transform directly on the original texture images. This is equivalent to using both the Fourier magnitude and phase information while, at the same time, avoiding the problem of the equal phase value property.

4.2 Classification experiments

I use two data sets in the experiments. The first includes 16 types of natural optical images obtained from the MIT Media Lab Vistex texture database, shown in Figure 4.4. Table 4.1describes these 16 textures. The original 512x512 color images are converted to the same size gray scale images with 256 gray levels. Adaptive histogram equalization is applied so all images have the same flat histogram and are indistinguishable from each

other in terms of first order statistics. To test the sensitivity of the algorithms to noise, I add several levels of white noise to the data. By flattening the histogram and adding noise, I try to make the classification task more difficult so relative classification performance differences among different texture features become more apparent. To save computational time, the first round of extensive tests is conducted on the first eight image classes, with three levels of noise added to the images. The signal-to-noise ratios (SNRs) are 15, 10, and 1, respectively.

To further validate the hypothesis, I next select the most successful methods from the first comparison and test them on all 16 classes of Vistex images and on a second data set of noisy, real-world sidescan sonar images from a survey of an Arctic under-ice canopy [44]. The three classes of sidescan sonar texture images used are shown in Figure 4.5. These are first-year young ice, multiyear undeformed ice, and multiyear deformed ice. For all data sets, each image class is divided into 225 half-overlapping samples of dimension 64x64, of which 60 samples are used for training. Therefore, the total data sample number is 1800 for the first eight Vistex images, 3600 for all the 16 Vistex images and, 675 for the sidescan sonar data set, with 480, 960, and 180 samples for training, respectively.

4.2.1 Comparison of wavelet features with Fourier transform features

Table 4.2 shows the complete test results from the eight classes of Vistex images. It is difficult to draw conclusions directly from the large number of test results in the table, so I point out some apparent features and use figures to illustrate the other findings.

First, notice that for some feature groups the differences in classification accuracy between training and testing data are very large, more than 50% in some cases. In fact, except for the level-one features, which have only four channels, most other training data classifications achieve more than 95% accuracy, including the SNR-1 noisy data. This is not the case for the test data; since only the simple Gaussian classifier is used, we might expect these trends be even more apparent for a more sophisticated classifier, which can learn a more precise feature structure of the training data. The significance of this result is that it shows that the widely used leave-one-out testing scheme can be rather deceiving for testing new algorithms, since leaving out one sample does not affect much of the training

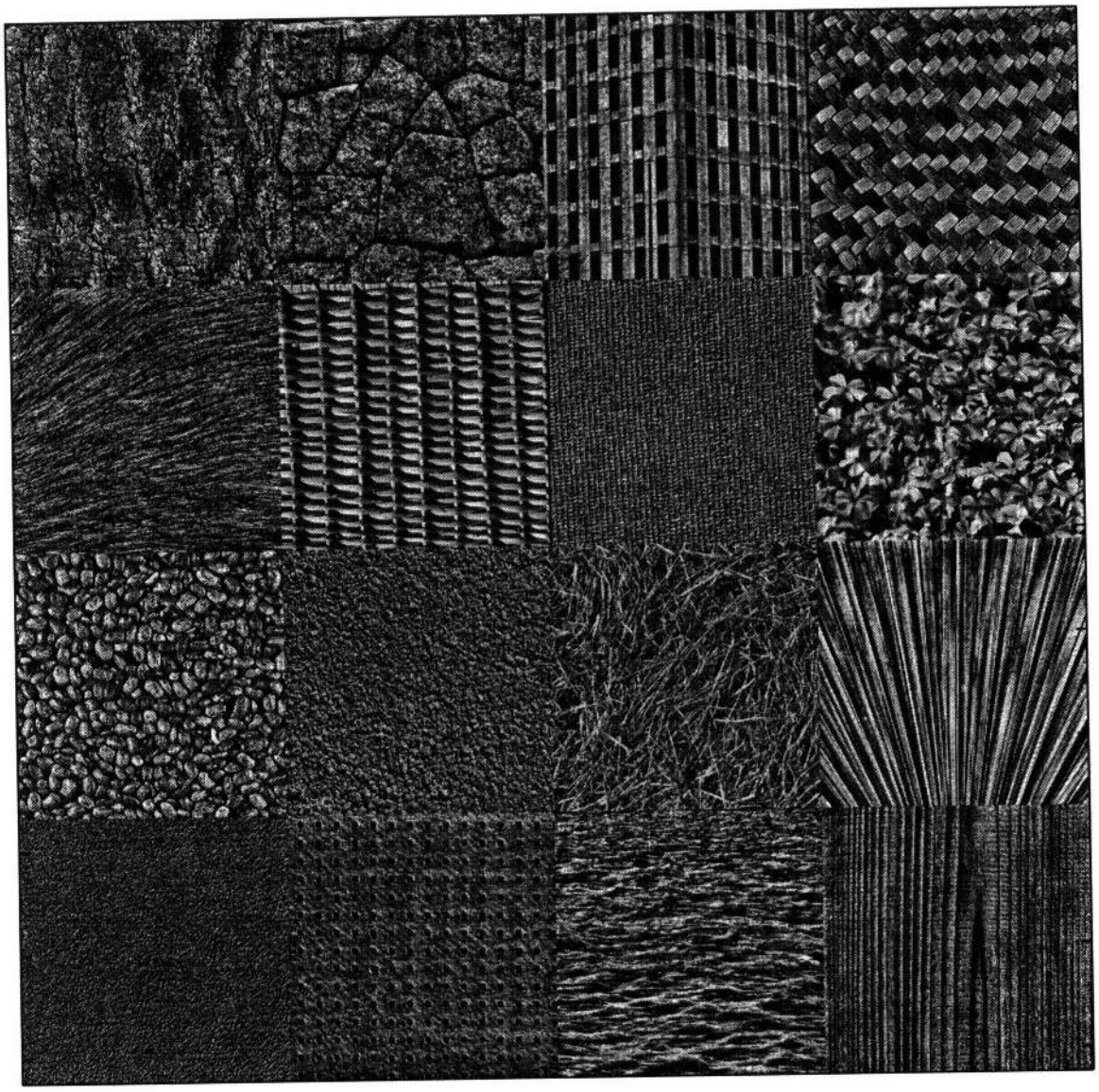


Figure 4.4: Sixteen Vistex textures. Descriptions are in Table 4.1. The eight images in the top two rows are used in the main experiment.

process. For the Gaussian classifier, the effect is minimal. This means that if the data set is too small, the results will not be conclusive.

Also note in the table that the number of features used to achieve best results for each group of features is mostly around 10, which is condensed from hundreds, sometimes thousands of original features. This demonstrates the effectiveness of the feature selection

Table 4.1: Vistex texture images description.

Image name	Contents	Lighting	Perspective
Bark.0008	tree bark	daylight direct right	frontal plane
Brick.0004	brick	daylight indirect right	frontal plane
Buildings.0009	building	daylight indirect	oblique
Fabric.0001	straw rattan	artificial incandescent	frontal plane
Fabric.0005	fur	artificial incandescent	frontal plane
Fabric.0013	wicker	daylight indirect	frontal plane
Fabric.0017.	carpet backing	daylight indirect	frontal plane
Flowers.0007	flowers	daylight direct	frontal plane
Food.0000	lima beans	artificial incandescent	frontal plane
Food.0005	coffee grounds	artificial strobe	frontal plane
Grass.0002	grass straw	daylight direct	frontal plane
Leaves.0002	plant leaf	daylight direct	frontal plane
Metal.0001	metal reflector sheet	artificial strobe	frontal plane
Tile.0007	ceiling tile	artificial strobe	frontal plane
Water.0006	water	daylight direct	oblique
Wood.0002	wood	daylight indirect	frontal plane

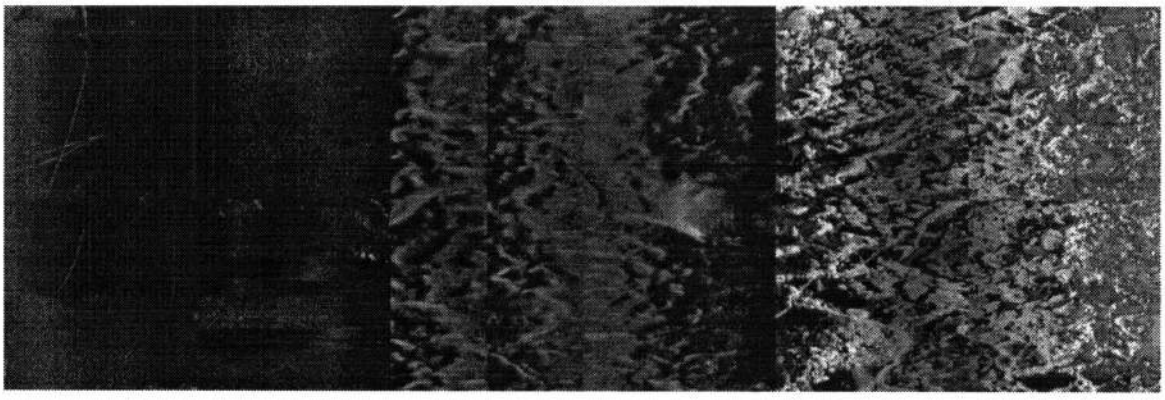


Figure 4.5: Sidescan sonar images of an Arctic under-ice canopy: (a) first-year ice, (b) multiyear undeformed ice, and (c) multiyear deformed ice.

Table 4.2: Complete test results on the eight Vistex texture images.

Data set			Original images				SNR 15 dB				SNR 5 dB				SNR 1 dB			
Feature group name			Training	Testing	All data	Feature num.	Training	Testing	All data	Feature num.	Training	Testing	All data	Feature num.	Training	Testing	All data	Feature num.
1	level 1	VAR	95.6	95.5	95.5	4	95.2	94.8	94.9	4	89.0	89.2	89.2	4	82.3	81.3	81.6	4
2		ENT	87.7	86.7	87.0	4	86.7	84.3	84.9	4	64.4	58.0	59.7	4	49.8	42.7	44.6	4
3		MNT_3	62.9	58.0	59.3	4	61.5	58.1	59.0	4	49.4	41.3	43.4	4	39.4	32.7	34.4	4
4		MNT_4	89.2	91.1	90.6	4	92.3	91.8	91.9	4	89.2	87.5	87.9	4	81.3	80.7	80.8	4
5		ALL	98.8	96.7	97.2	8	98.5	96.7	97.2	9	97.3	92.5	93.8	11	94.8	82.5	85.8	13
6	level 2	VAR	96.5	96.9	96.8	6	96.3	95.8	95.9	5	96.7	93.0	94.0	9	97.9	89.8	92.0	12
7		ENT	94.2	90.6	91.6	8	97.7	86.5	89.5	14	97.1	66.7	74.8	20	94.2	48.9	60.9	20
8		MNT_3	80.4	62.3	67.2	9	85.2	66.2	71.3	11	96.5	69.8	76.9	18	95.8	56.5	67.0	20
9		MNT_4	94.2	90.9	91.8	6	93.1	91.6	92.0	6	95.0	87.7	89.7	9	95.6	85.0	87.8	12
10		ALL	98.8	97.8	98.1	12	99.4	97.7	98.1	12	95.8	90.5	91.9	9	97.5	84.1	87.7	16
11	level 3	VAR	97.7	98.1	98.0	6	97.5	98.0	97.9	7	95.8	95.7	95.7	6	96.7	90.3	92.0	13
12		ENT	97.7	79.2	84.2	17	89.8	75.2	79.1	11	96.9	54.8	66.1	22	97.5	40.5	55.7	26
13		MNT_3	82.7	60.5	66.4	10	85.8	61.6	68.1	12	96.0	53.8	65.1	22	99.6	45.2	59.7	30
14		MNT_4	94.8	92.7	93.2	7	94.2	92.0	92.6	8	95.0	91.1	92.2	9	97.9	83.9	87.7	18
15		ALL	98.5	97.8	98.0	16	97.7	97.3	97.4	6	97.5	93.2	94.3	10	96.5	90.9	92.4	13
16	level 4	VAR	97.1	97.5	97.4	6	97.5	96.9	97.1	6	96.7	95.9	96.1	6	98.5	94.5	95.6	12
17		ALL	97.7	97.9	97.8	8	97.9	96.4	96.8	9	99.2	93.9	95.3	18	95.4	93.0	93.6	6
18	level 12	VAR	96.5	96.9	96.8	6	96.3	95.8	95.9	5	96.7	93.0	94.0	9	97.9	89.8	92.0	12
19		ALL	99.4	98.3	98.6	10	99.2	98.2	98.4	10	99.2	93.3	94.8	17	93.3	85.8	87.8	10
20	lev 123	VAR	98.5	98.6	98.6	7	98.3	98.6	98.6	7	96.9	95.3	95.7	9	97.9	90.5	92.4	13
21		ALL	99.8	98.6	98.9	20	99.2	98.2	98.4	11	98.8	94.5	95.7	13	97.7	91.7	93.3	14
22	lev1234var		99.0	97.2	97.7	14	98.8	96.7	97.3	12	97.9	96.0	96.5	12	98.1	93.7	94.9	11
23	wavelet	VAR	98.1	97.0	97.3	10	97.9	96.7	97.1	10	97.7	95.0	95.7	11	97.9	94.2	95.2	12
24		ALL	100	96.6	97.5	18	100.	96.4	97.4	18	97.9	93.8	94.9	14	98.1	90.2	92.3	19
25	FFT Mag. (DSM)		99.8	98.4	98.8	26	99.4	97.9	98.3	15	99.0	96.4	97.1	10	99.8	95.8	96.8	17

algorithms. A general trend is that noisier data tend to need more features to achieve best classification performance.

To help focus on the classification accuracy of the overall data set, Figure 4.6 shows a comparison of the four types of features and their combinations in the first three decomposition levels. The MNT_3 feature is the worst for all levels and for all data sets, and is apparently not a useful measure. Entropy also gives less satisfactory results than the variance feature, and the classification accuracy drops sharply for noisy data. This contradicts the conclusion given in [65], where entropy features perform about the same as variance measures. It is probably because they only experimented with original clean data and did not test with noise added to the data sets. The MNT_4 feature seems to give better results than the above two features but is still less successful than the variance feature. The performance differences between the MNT_4 and the variance are consistent over all data sets and all decomposition levels, because they are very closely correlated features.

The observation that variance features perform better than other features is consistent with Laws' [66] experiment with features extracted from empirical frequency channels. The remaining question is whether we need other measures to add new information. When a union operation is applied to the correct classification samples of the four types of features, the correct classification rate increases by about 5%, nearing 100% accuracy. This demonstrates that each feature has its own distinct classification power. By combining all features together, we get improved results for the lower decomposition level. Since the feature length is much smaller in these levels, an added dimension helps more than in the higher level decomposition case. This improvement is not as impressive as that for the union of results, because there is also more added noise, which may overwhelm the benefit of new information from added features.

Now consider in detail the variance measure results shown in Figure 4.7. For the individual levels of Figure 4.7 (a), the general trend is that the higher the decomposition level the better the result. This is predictable from Eq. (4.14), which shows that the lower level variance features are simply the average of their higher level children nodes. A KLT transform will do better than such a simple average operation in terms of extracting maximum information. To confirm this point, compare Figures 4.7 (a) and (b), which show that the following pairs of results are almost identical: level 1&2 vs. level 2, level 1&2&3 vs. level

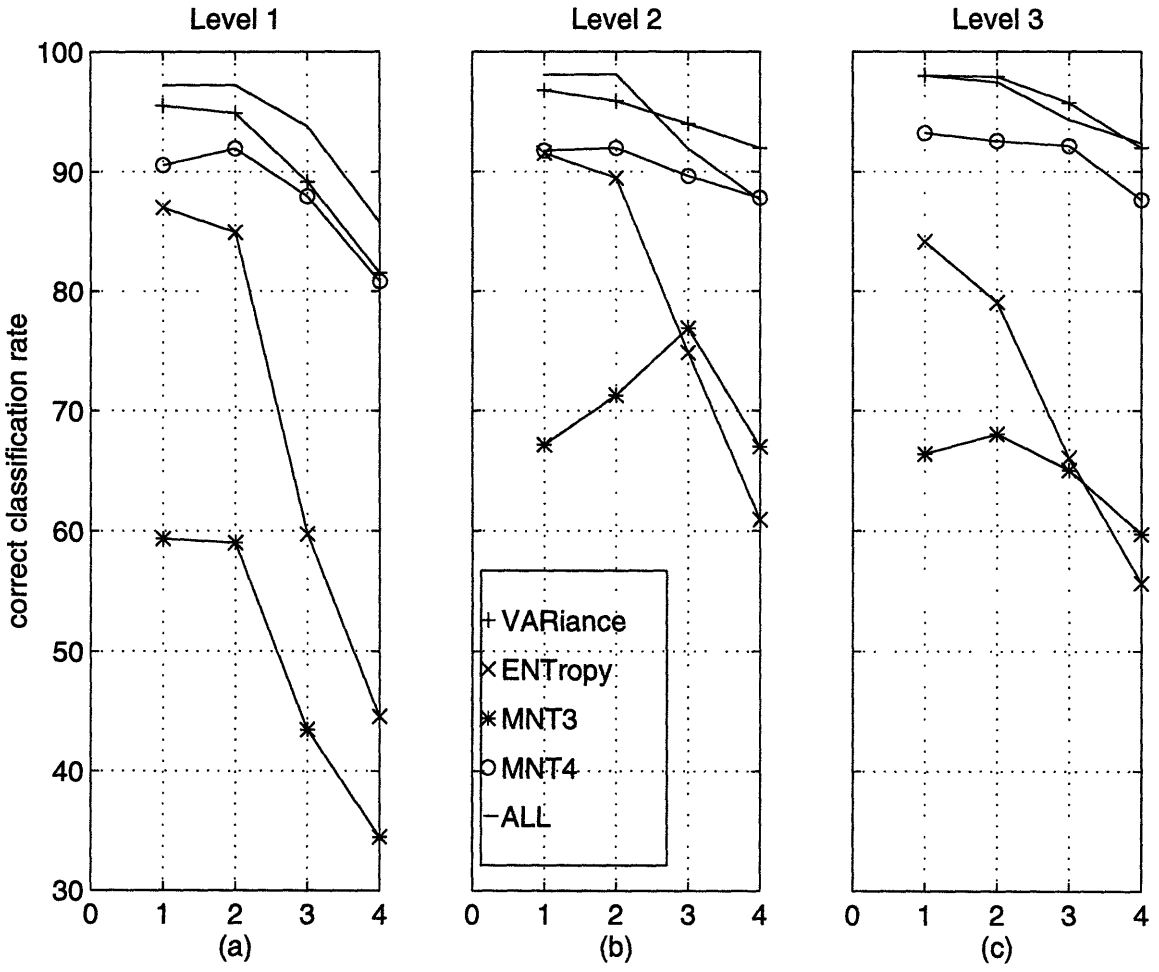


Figure 4.6: Comparison of the four types of features in the first three individual decomposition levels. The index of the horizontal axis represents signal-to-noise ratio (SNR) level: 1. original image, 2. SNR 15 dB, 3. SNR 5 dB, 4. SNR 1 dB.

3, level 1&2&3&4 vs. level 4. This means that lower level features are only a subset of higher level decomposition features. This is contrary to what Laine and Fan suggested in [65], that redundancy may provide additional discrimination power. The experiments show that better discrimination ability is not added by over completion. Instead, it is extracted by applying KLT to higher levels of finer channel decomposition, so the channel nodes are combined in an optimal way instead of by simple averaging.

Continuing this thread of analysis, we should expect that the Fourier transform provides even more information, because the Fourier transform is actually the extreme case of

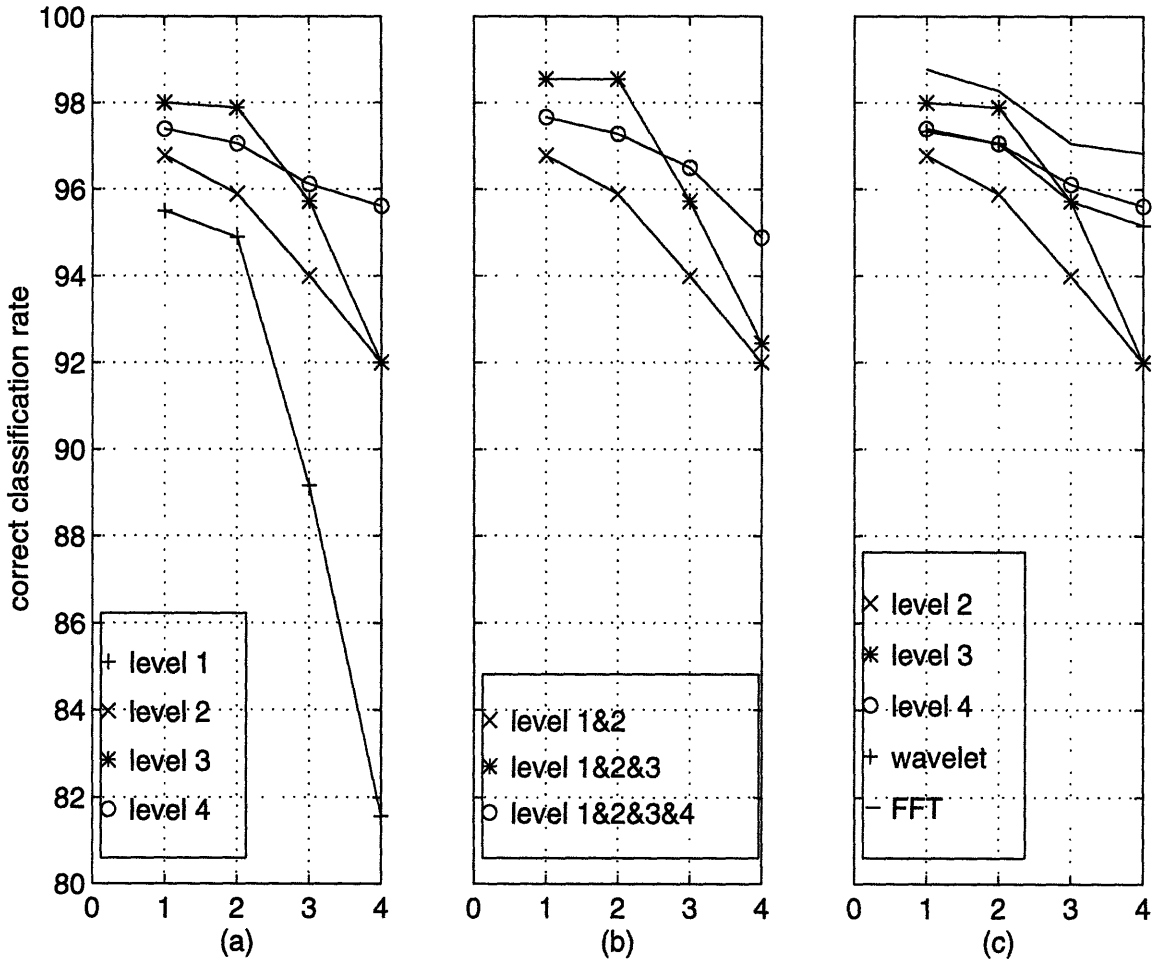


Figure 4.7: Comparison of variance features for individual decomposition levels, overcomplete levels, standard wavelet, and Fourier transform. The index of the horizontal axis represents the same SNR as in Fig. 7.

a wavelet packet transform, i.e., the wavelet packet transform at its highest possible level. Figure 4.7 (c) compares the performances of three levels of wavelet packet decomposition, the standard wavelet transform, and the Fourier transform. The Fourier transform indeed gives consistently better performance over all other feature groups on all levels of noisy data sets. This result should not be surprising, since the wavelet transform is optimal for nonstationary signal analysis, whereas the Fourier transform is optimal for stationary signal analysis. Most texture images are stationary periodic signals. Because of the symmetry property, only half the frequency matrix is used.

Next, notice in Figure 4.7 (c) that the Fourier transform and other higher levels of wavelet packet decompositions are very insensitive to noise. It is surprising to see that more than 95% accuracy is achieved at a SNR level of 1 dB, compared with the results in [12], where the tree-structured wavelet algorithm collapses to 70% accuracy at a 5-dB noise level. Noise insensitivity is a particular strength of subband image processing. Noise usually has a flat spectrum and, when divided into more subbands, the noise energy usually decreases. Yet, the energy of signals tends to concentrate in a few channels. Therefore, even when the total energy of the signal and noise are almost the same, as in the case of the testing data of SNR 1 dB, the signal-to-noise ratio is much higher in channels containing the most signal energy. The feature selection algorithms are designed in such a way that they pick up and condense the signal channels with high SNR into a compact representation of the data, with the incoherent noisy channels neglected.

4.2.2 Comparison of different wavelet filter types and lengths

The wavelet filter type used in the above experiments is the Daubechies minimum-support least asymmetric wavelet of filter length four [25]. In this section I compare this particular type and length of wavelet filter with others. The Daubechies minimum-support least asymmetric wavelet and the Coiflets [25] of various lengths are used for the comparison. Table 4.3 lists the classification results on the eight Vistex images with noise level of SNR 5. We see that almost all differences in classification accuracy among the different filter types and lengths are within 1%. No particular trend can be observed as the filter length increases, and the overall performance of the two types of filters are almost the same. These results indicate that the conclusions drawn in the previous section are valid for other general types of wavelets.

4.2.3 Experimental comparison of the KLT and MDEE

Except for some small feature vectors in the above experiments, I use the MDEE transform in place of the KLT. I did not list the specific grouping parameters in the table of results, because the numerical differences between the standard KLT and the MDEE transform are negligibly small. I show this with an experiment described in this section.

Table 4.3: Comparison of different wavelet filter types and lengths.

Data set name		Eight Vistex images with SNR5					
Wavelet type		Daubechies minimum-support least asymmetric			Coiflets		
Wavelet filter length		4	8	16	6	12	18
1	Level 2 VAR	94.0	94.7	95.4	95.7	94.7	95.8
2	Level 3 VAR	95.7	96.7	96.3	96.3	95.7	95.4
3	Level 4 VAR	96.1	95.4	95.3	94.9	94.7	95.9
4	All levels VAR	96.5	95.4	95.6	95.6	95.4	96.3

For this experiment, the data set comprises the original eight images from the previous experiment. The feature vector is formed by 420 frequency components in the center high-frequency region of the Fourier transform magnitude matrix. I use a smaller feature vector for this experiment so the brute-force KLT can be computed within a reasonable time with reasonable memory requirements. For the MDEE transform, the feature vector is broken into seven feature vectors of length 60 each. Then, the first $n = 20$, 10, and 5 dominant features in each group are selected to form the new feature vector of length $7 \cdot n$, from which the final dominant eigenvalues and eigenvectors are computed. Figure 4.8 (a) shows the results of the top 30 eigenvalues of the standard KLT and the MDEE transforms with the three values of n. We see that when 20 features are kept after the first-step eigenvalue computation, the final MDEE eigenvalues are almost the same as the standard KLT. When only 10 or 5 features are kept, the first 15 eigenvalues are still similar to KLT; the remaining eigenvalues start to lose a very small amount of energy. However, this does not affect the final classification results at all. Figure 4.8 (b) shows the percentage of correct classifications using the KLT and the MDEE plotted against the number of features used. All four groups of results overlap with each other almost completely, with the maximum classification accuracy difference being less than 0.5%.

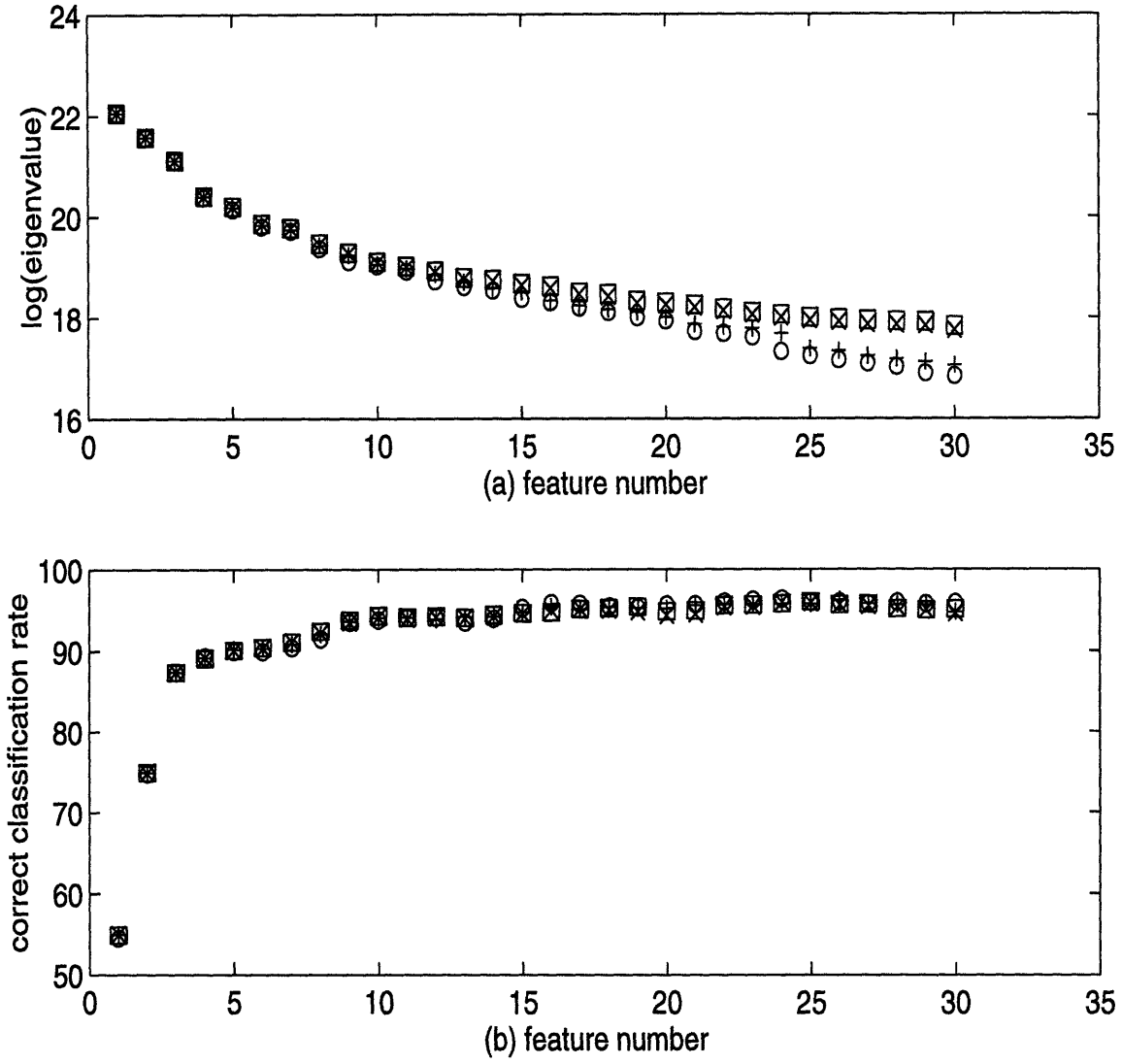


Figure 4.8: Comparison of the KLT and MDEE transforms: (a) plot of the top 30 largest eigenvalues; (b) plot of the correct classification rate against number of features used. In both plots, the square symbol is for the KLT and the other three are for MDEE: x for n=20; + for n=10; o for n=5.

4.2.4 Comparison of the DSM features with the PSM features

A comparison of the DSM features with the PSM features is conducted on the eight Vistex images with noise level of SNR 5. The 20 PSM features used are described in Appendix A. Results are given in Table 4.4, where the DSM features show nearly 10%

better performance than the PSM features. The performance discrepancy is further widened when tested on a larger data set, as described in the next section. This demonstrates that the optimal feature selection approach using MDEE and Bhattacharyya distance is better than an ad hoc feature extraction method. The first row in the table gives the results of using the original image directly as the feature vector. Its poor performance validates the conclusion that the phase information is only a noise signal for texture classification.

Table 4.4: Comparison of the DSM, PSM, and texture features extracted directly from spatial domain images.

Data set name		Eight Vistex images with SNR5			
Feature group name		Training data	Testing data	All data	Feature number
1	Original image	97.1	73.0	79.4	23
2	PSM	91.0	85.7	87.1	7
3	DSM	99.0	96.4	97.1	10

4.2.5 Further experiments with other data sets

Finally, I test the algorithms on the classification of a larger data set of 16 Vistex images and the noisy sidescan sonar images. Only the feature groups that performed best in the above experiment are used. Table 4.5 shows the classification results, which are consistent with the above results. An interesting observation is that the DSM method is very insensitive to noise. With SNR-5 noise added to the 16 images, the classification accuracy drops less than 1%. For the sidescan sonar images, although the image class number is smaller, each class of image is noisy and nonuniform. This added difficulty increases the classification accuracy difference between the wavelet packet and Fourier transform features. It again shows the superiority of the Fourier transform features over the more complex wavelet features.

Table 4.5: Classification results of the 16 Vistex images and sidescan sonar images.

Data set name		Original 16 Vistex images				16 Vistex images of SNR 5				Sidescan sonar images			
Feature group name		Training	Testing	All data	Feature num.	Training	Testing	All data	Feature num.	Training	Testing	All data	Feature num.
1	Level 3 VAR	97.6	97.3	97.4	8	97.2	94.0	94.8	11	93.3	83.2	85.9	12
2	Level 4 VAR	97.8	97.2	97.3	10	99.1	92.8	94.5	19	96.1	81.4	85.3	20
3	All levels VAR	98.3	97.3	97.6	10	98.9	93.1	94.6	19	98.3	85.3	88.7	23
4	PSM	83.5	81.4	81.9	5	72.9	71.6	71.9	4	93.9	89.1	90.4	8
5	DSM	99.1	96.8	97.4	13	99.2	95.8	96.7	16	98.9	92.5	94.2	9

4.3 Conclusions

Based on the above experiments, the following conclusions are drawn:

1). Variance (energy) measures are much better than entropy and higher order moments, but there may exist additional information in such features that is distinct from the energy information.

2). For variance features, overcomplete representation does not add more information than individual level features. Higher levels of decomposition perform better than lower levels. This leads to the conclusion that the Fourier transform magnitude features are better than the more complicated wavelet packet features.

3). Wavelet packet features are very insensitive to noise. Features from higher levels are less sensitive than the lower level features. Again, Fourier transform features are the best in terms of noise insensitivity.

4). The MDEE is an extremely close approximation of the KLT. MDEE plus Bhattacharyya distance measure are shown to be very effective in extracting texture features from both wavelet packet transforms and the Fourier transform.

5). Finally, the Fourier phase information is a noise signal for texture classification. However, the superior performance of DSM over the conventional PSM shows that the

Fourier transform magnitudes contain enough texture information for classification, if the right feature extraction algorithm is used.

Chapter 5

Run-length transform texture classification

Comparative studies suggest that run-length features are inefficient features for texture discrimination [19] [120]. Since first introduced by Galloway [42], their application has been very limited compared to other approaches.

In this chapter, I reinvestigate the run-length features from a new approach. By using the new multi-level dominant eigenvector estimation algorithm and the Bhattacharyya distance measure described in Chapter 3, I demonstrate that texture features extracted from the run-length matrix can give superb classification results. Experimental comparisons with the widely used co-occurrence features and the recently proposed wavelet features are also made.

This chapter is organized into three sections. Section 5.1 introduces the original definition of the run-length matrix and several of its variations, then reviews the traditional run-length features and describes the new run-length feature extraction algorithm. Section 5.2 presents the texture classification experimental results. The conclusions are summarized in Section 5.3.

5.1 Texture feature extraction

5.1.1 Definition of the run-length matrices

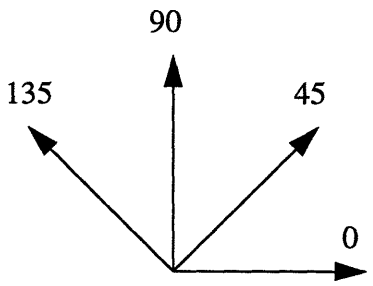
With the observation that, in a coarse texture, relatively long gray-level runs would occur more often and that a fine texture should contain primarily short runs, Galloway [42] proposes the use of a run-length matrix for texture features. For a given image, a run-length matrix $p(i, j)$ is defined as the number of runs with pixels of gray level i and run-length j. An example of the run-length matrices is shown in Figure 5.1, where four directional run-length matrices are computed from the original image. Various texture features

can be derived from these matrices. Section 5.1.2 lists some texture features developed in previous studies.

Original

1	1	1	2	2
3	4	2	2	3
4	4	4	4	4
5	5	3	3	3
1	1	3	4	5

Four directions



Run length

Gray level	0	1	2	3	4	5
	1	0	1	1	0	0
	2	0	2	0	0	0
	3	3	0	1	0	0
	4	2	0	0	0	1
	5	1	1	0	0	0

Run length

Gray level	45	1	2	3	4	5
	1	5	0	0	0	0
	2	0	2	0	0	0
	3	4	0	1	0	0
	4	5	1	0	0	0
	5	3	0	0	0	0

Gray level	90	1	2	3	4	5
	1	5	0	0	0	0
	2	2	1	0	0	0
	3	4	1	0	0	0
	4	5	1	0	0	0
	5	3	0	0	0	0

Gray level	135	1	2	3	4	5
	1	5	0	0	0	0
	2	4	0	0	0	0
	3	6	0	0	0	0
	4	5	1	0	0	0
	5	3	3	0	0	0

Figure 5.1: Four directional gray-level run-length matrices.

In this chapter, I design several new run-length matrices, which are slight but unique variations of the traditional run-length matrix. For a run-length matrix $p(i, j)$, let M be the

number of gray levels and N be the maximum run length. The four new matrices are defined as follows.

Gray Level Run Length Pixel Number Matrix - GLRLPNM:

$$p_p(i, j) = p(i, j) \cdot j. \tag{5.1}$$

Each element of the matrix represents the number of pixels of run-length j and gray-level i. Compared to the original matrix, the new matrix gives equal emphasis to all lengths of runs in an image.

Gray Level Run Number Vector - GLRNV:

$$p_g(i) = \sum_{j=1}^{N} p(i, j). \tag{5.2}$$

This vector represents the marginal distribution of the number of runs with gray-level i.

Run Length Run Number Vector - RLRNV:

$$p_r(j) = \sum_{i=1}^{M} p(i, j). \tag{5.3}$$

This vector represents the marginal distribution of the number of runs with run-length j.

Gray Level Run-Length-One Vector - GLRLOV:

$$p_o(i) = p(i, 1). \tag{5.4}$$

Figure 5.2 shows the four directional run-length matrices of several natural texture samples. Notice that the first column of each of the four directional run-length matrices is overwhelmingly larger than the other columns. This may mean that most texture information is contained in the run-length-one vector. The advantages of using this vector are that it offers significant feature length reduction and that a fast parallel run-length matrix computation can replace the conventional serial searching algorithm. For example, the positions of pixels with run length one in the horizontal direction can be found by a logical

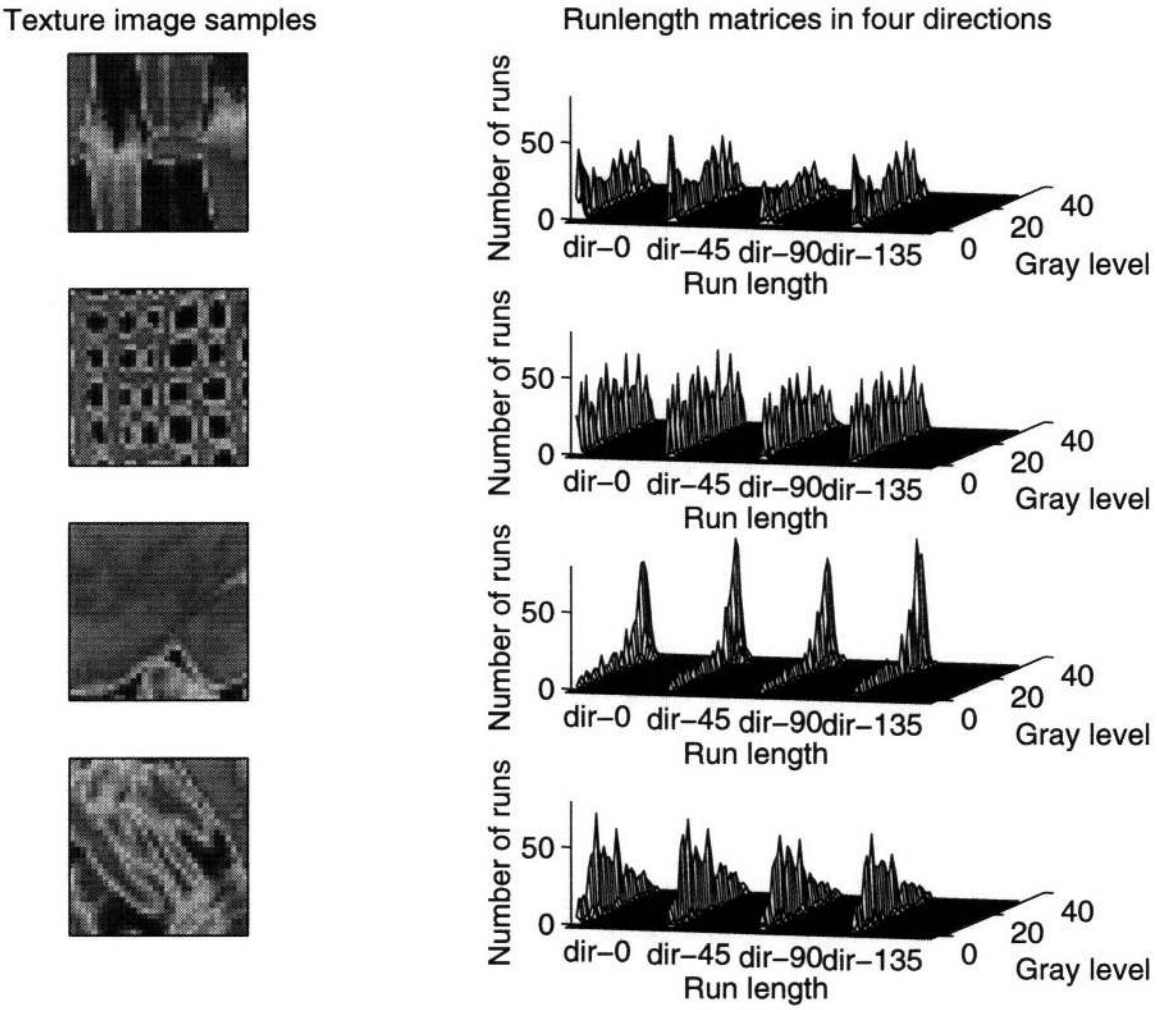


Figure 5.2: The four directional run-length matrices of several Brodatz texture samples. Each image sample is of size 32x32 with 32 gray levels. The four directional (0, 45, 90, and 135 degree directions) run-length matrices are combined into a single matrix. The leftmost column of each directional matrix is the run-length-one vector, which has much larger values than the other columns.

"and" operation on the outputs of the forward and backward derivative of the original image:

$$f(i, j) = x(i, j) - x(i, j-1), \tag{5.5}$$

$$b(i, j) = x(i, j-1) - x(i, j), \tag{5.6}$$

$$o(i, j) = f(i, j) \cap b(i, j), \tag{5.7}$$

where $x(i, j)$ is the texture image whose pixels outside the image boundary are set to zero, and $\cap$ represents the logical "and" operation. Then $p_o(i)$ can be obtained by computing the histogram of $x(i, j)_{o(i, j) = 1}$. To find the starting pixel position for runs with length two, a similar scheme can be employed,

$$f_2(i, j) = (f(i, j) \neq 0) - o(i, j), \tag{5.8}$$

$$b_2(i, j) = (b(i, j) \neq 0) - o(i, j), \tag{5.9}$$

$$o_2(i, j) = f_2(i, j) \cap b_2(i, j + 1). \tag{5.10}$$

In fact, the gray level run number vector $p_g(i)$ can also be obtained with the above approach by computing the histogram of $x(i, j)_{f(i, j) \neq 0}$.

The matrix and vectors defined above are not designed for the extraction of traditional features. Along with the original run-length matrix, they are used in the new feature extraction approach I developed in Chapter 3. The next section gives a review of the traditional feature extraction method.

5.1.2 Traditional run-length features

From the original run-length matrix $p(i, j)$, many numerical texture measures can be computed. The five original features of run-length statistics derived by Galloway [42] are:

Short Run Emphasis (SRE)

$$SRE = \frac{1}{n_r} \sum_{i = 1}^{M} \sum_{j = 1}^{N} p(i, j)/j^2 = \frac{1}{n_r} \sum_{j = 1}^{N} p_r(j)/j^2, \tag{5.11}$$

Long Run Emphasis (SRE)

$$LRE = \frac{1}{n_r}\sum_{i=1}^{M}\sum_{j=1}^{N} p(i,j)\cdot j^2 = \frac{1}{n_r}\sum_{j=1}^{N} p_r(j)\cdot j^2, \tag{5.12}$$

Gray Level Nonuniformity (GLN)

$$GLN = \frac{1}{n_r}\sum_{i=1}^{M}\left(\sum_{j=1}^{N} p(i,j)\right)^2 = \frac{1}{n_r}\sum_{i=1}^{M} p_g(i)^2, \tag{5.13}$$

Run Length Nonuniformity (RLN)

$$GLN = \frac{1}{n_r}\sum_{j=1}^{N}\left(\sum_{i=1}^{M} p(i,j)\right)^2 = \frac{1}{n_r}\sum_{j=1}^{N} p_r(j)^2, \tag{5.14}$$

Run Percentage

$$RP = \frac{n_r}{n_p}, \tag{5.15}$$

where n_r is the total number of runs and n_p is the number of pixels in the image. Based on the observation that most features are only functions of $p_r(j)$, without considering the gray level information contained in $p_g(i)$, Chu *et al.* [16] proposed two new texture features,

Low Gray-level Run Emphasis (LGRE)

$$LGRE = \frac{1}{n_r}\sum_{i=1}^{M}\sum_{j=1}^{N} p(i,j)/i^2 = \frac{1}{n_r}\sum_{i=1}^{M} p_g(i)/i^2, \tag{5.16}$$

High Gray-level Run Emphasis (HGRE)

$$HGRE = \frac{1}{n_r}\sum_{i=1}^{M}\sum_{j=1}^{N} p(i,j)\cdot i^2 = \frac{1}{n_r}\sum_{i=1}^{M} p_g(i)\cdot i^2, \tag{5.17}$$

to extract gray level information in the matrix. In a more recent study, Dasarathy and Holder [23] described another four feature-extraction functions following the idea of joint statistical measure of gray level and run length,

Short Run Low Gray-level Emphasis (SRLGE)

$$SRLGE = \frac{1}{n_r} \sum_{i=1}^{M} \sum_{j=1}^{N} p(i,j)/(i^2 \cdot j^2), \tag{5.18}$$

Short Run High Gray-level Emphasis (SRHGE)

$$SRHGE = \frac{1}{n_r} \sum_{i=1}^{M} \sum_{j=1}^{N} p(i,j) \cdot i^2/j^2, \tag{5.19}$$

Long Run Low Gray-level Emphasis (LRLGE)

$$LRLGE = \frac{1}{n_r} \sum_{i=1}^{M} \sum_{j=1}^{N} p(i,j) \cdot j^2/i^2, \tag{5.20}$$

Long Run High Gray-level Emphasis (LRHGE)

$$LRHGE = \frac{1}{n_r} \sum_{i=1}^{M} \sum_{j=1}^{N} p(i,j) \cdot i^2 \cdot j^2. \tag{5.21}$$

Dasarathy and Holder [23] tested all eleven features on the classification of a set of cell images and showed that the last four features gave much better performance. However, the data set they used was fairly small, with only 20 samples in each of the four image classes. In Section 5.2, I test these features on a much larger data set with 225 samples in each of eight image classes. An extensive comparison between these features and the features extracted by the new algorithm is also carried out.

These features are all based on intuitive reasoning, in an attempt to capture some apparent properties of the run-length distribution. For example, the eight features illus-

trated in Figure 5.3 are weighted-sum measures of the run-length concentration in the eight directions, i.e., the positive and negative 0-, 90-, 45-, and 135-degree directions. Similar to the way in which these texture features are derived, we could define more ad hoc features. The drawbacks of this approach would be the same as the traditional PSM discussed in Chapter 4: there is no theoretical proof that, given a certain number of features, maximum texture information can be extracted from the run-length matrix, and many of these features are highly correlated with each other. For example, for an image with high long-run emphasis, the short-run emphasis must be relatively small, so the long-run-emphasis features and the short-run-emphasis features essentially measure the same texture property. In the experiments of Section 5.2, I demonstrate that this overlapping information does more harm than good for classification.

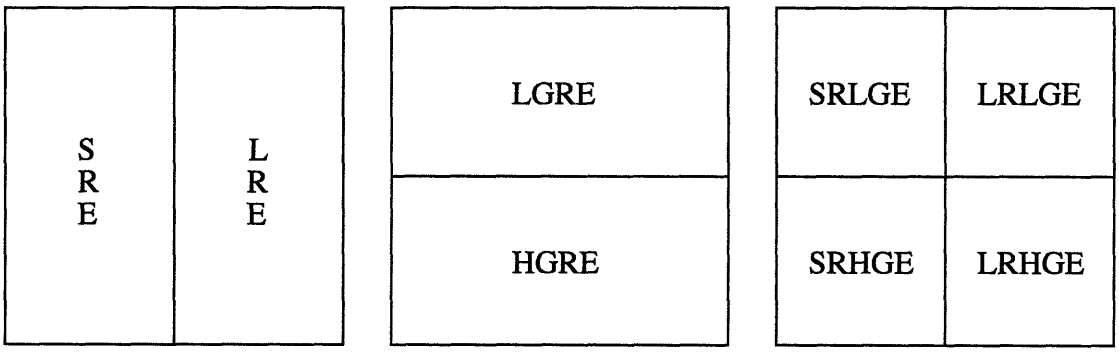


Figure 5.3: Run-emphasis regions of several traditional run-length texture features.

5.1.3 Dominant run-length features

Instead of developing new functions to extract texture information, I use the run-length matrix as the texture feature vector directly; this preserves all information in the matrix. However, this again introduces two problems. One is the high dimensionality of the feature vector. The other is the same as the traditional approach, i.e., that the neighborhood dimensions of the feature vector are highly correlated.

To alleviate the first problem, consider the run-length matrix more closely. Figure 5.2 shows that most nonzero values concentrate in the first few columns of the matrix. More-

over, the information in these first few columns, i.e., the short-run section, is correlated with that of the rest of the matrix, i.e., the long-run section, because for each row of the run-length matrix an image with a high long-run value has a smaller short-run value. By using only the first few columns as the feature vector, the information in the long-run section is not simply discarded but is mostly preserved in the feature vector. Another advantage of using only the first few columns is that the fast parallel run-length matrix computation algorithm described in section 5.1.1 can be employed. In the extreme case, only the first column of the matrix, the run-length-one vector, is used.

To further reduce the feature-vector dimension and to decorrelate neighboring element values in the matrices, I use the MDEE method and the Bhattacharyya distance measure described in Chapter 3. I call this new method the dominant run-length method (DRM).

5.2 Classification experiments

In this section, two separate data sets are used for the texture classification experiment. First, the traditional run-length features and the new run-length features are compared with each other, then with the co-occurrence features and the wavelet texture features on the classification of eight Brodatz images. Next, the best run-length features are tested on the Vistex data set used in Chapter 4.

The first data set comprises the eight Brodatz images [9], which are shown in Figure 5.4. Each image is of size 256x256 with 256 gray levels. The images are first quantized into 32 gray levels using equal-probability quantization. Each class is divided into 225 sample images of dimension 32x32 with 50% overlap. Sixty samples of each class are used as training data, so the training data size is 480 samples and the testing data size is 1320.

I did not use the Brodatz data set in the wavelet feature experiments in Chapter 4, because we only have the Brodatz images of size 256x256. They are relatively small if we need to cut a large number of image samples of size 64x64 for the 4-level wavelet packet decomposition. For other feature extraction methods, the sample image size can be 32x32. Given the great popularity of the Brodatz images in the texture research community, I

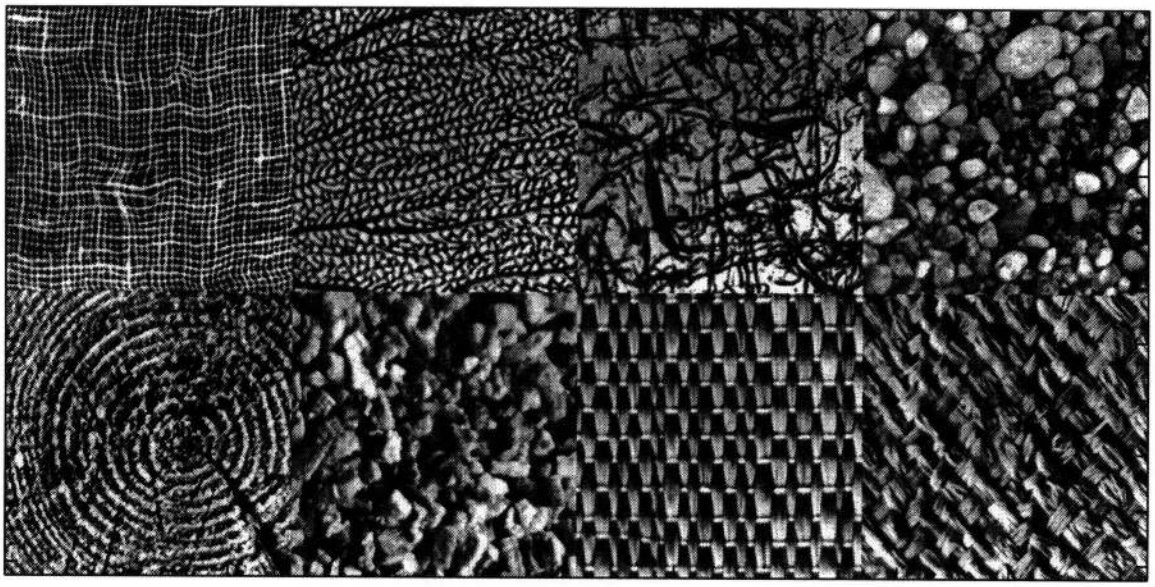


Figure 5.4: Eight Brodatz textures. Row 1: burlap, seafan, ricepaper, pebbles23; Row2: tree, mica, straw, raffia.

compare texture classification algorithms in the rest of the thesis on the Brodatz images, including the dominant power spectrum method proposed in Chapter 4.

As these results show, most of the new algorithms give perfect classification. To further compare the performance of these new algorithms and their consistency when applied to a larger natural image set, I conducted a second experiment on the 16 Vistex texture images used in Chapter 4. Unlike Brodatz images, which were mostly obtained in well controlled studio conditions, the Vistex images were taken under natural lighting conditions. They pose a more realistic challenge for texture classification algorithms. Each original color image is converted to a 256x256 gray-scale image with 256 gray levels. The same 32-gray-level quantization is applied to each image. This quantization gives all image classes the same flat histogram, indistinguishable by mean and variance features. However, no adaptive histogram equalization is applied to the images to compensate the nonuniform lighting. This makes the classification more difficult. Each class is again divided into 225 samples of dimension 32x32 with 50% overlap. Sixty samples of each

class are used as training data. So the training data has 960 samples and the testing data 2640 samples.

5.2.1 Classification using the traditional run-length features

Table 5.1 shows the classification results using the traditional run-length features directly on the Brodatz images. Similar to [23], the feature groups tested are the original five features of Galloway [42], the two features of Chu *et al.* [16], and the four new features of Dasarathy and Holder [22]. All four-direction features are used. Contrary to the good classification results on only four classes of 80 samples in [22], all groups of features performed poorly here. With only 35% classification accuracy, the result of using all three feature groups together is much worse than any single feature group. However, by applying the feature selection algorithms, i.e., KLT plus Bhattacharyya distance measure, to the feature vector before classification, improved results are shown in Table 5.2. In this case, the feature vector containing all three group features achieves 88% accuracy, far better than any single group features. This is mainly because of the close correlation of the three groups of features.

Table 5.1: Brodatz texture classification results using the traditional run-length features.

Feature name	Original feature length	Number of selected features	Correct classification rate		
			Training data	Testing data	All data
G5	20	20	64.6	60.7	61.7
C2	8	8	61.2	41.8	47.0
D4	16	16	84.4	59.1	65.8
ALL	44	44	35.6	35.4	35.4

To see the degree of correlation, I compute the auto-correlation coefficient matrix of the complete run-length feature vector shown in Figure 5.5. Many coefficient values in the matrix are close to one and the high correlations can also be seen in the scatter plots of several strongly correlated features, as illustrated in Figure 5.6. The poor classification

Table 5.2: Brodatz texture classification results using the new feature selection method on the traditional run-length features.

Feature name	Original feature length	Number of selected features	Correct classification rate		
			Training data	Testing data	All data
G5	20	12	88.5	74.9	78.6
C2	8	8	61.2	41.8	47.0
D4	16	16	84.4	59.1	65.8
ALL	44	24	99.4	83.7	87.9

performance of correlated features indicates that additional features bring in a great deal of noise, which overwhelms any marginal benefit of mostly redundant information contained in the added features. This shows the importance of using the KLT transform to extract decorrelated information.

5.2.2 Classification using the dominant run-length features.

Figures 5.7 and 5.8 show the scatter plots of the top eight features obtained by applying the MDEE transform on the original run-length matrix and on the run-length-one vector, respectively. Almost perfectly separable clustering can be seen for most of the eight image classes in both cases, in sharp contrast to the overlapping clusters in Figure 5.6 using the traditional feature vector.

The classification results with the DRM described in Section 5.1.3 are summarized in Table 5.3. Notice the dramatic reduction of feature length from several hundred to around ten, comparable with the traditional feature vector length. The results indicate that a compact, optimal run-length feature vector can be extracted by the MDEE method, without resort to ad hoc functions.

With only such a small number of features, perfect classification is achieved with the original matrix and with most of the new matrices and vectors. The only exceptions are the RLRN vector and the long-run region of the run-length matrix. The poor performance of the long-run region matrix and the good performance of the short-run region matrix indicate that most texture information is indeed concentrated in the short-run region. This also

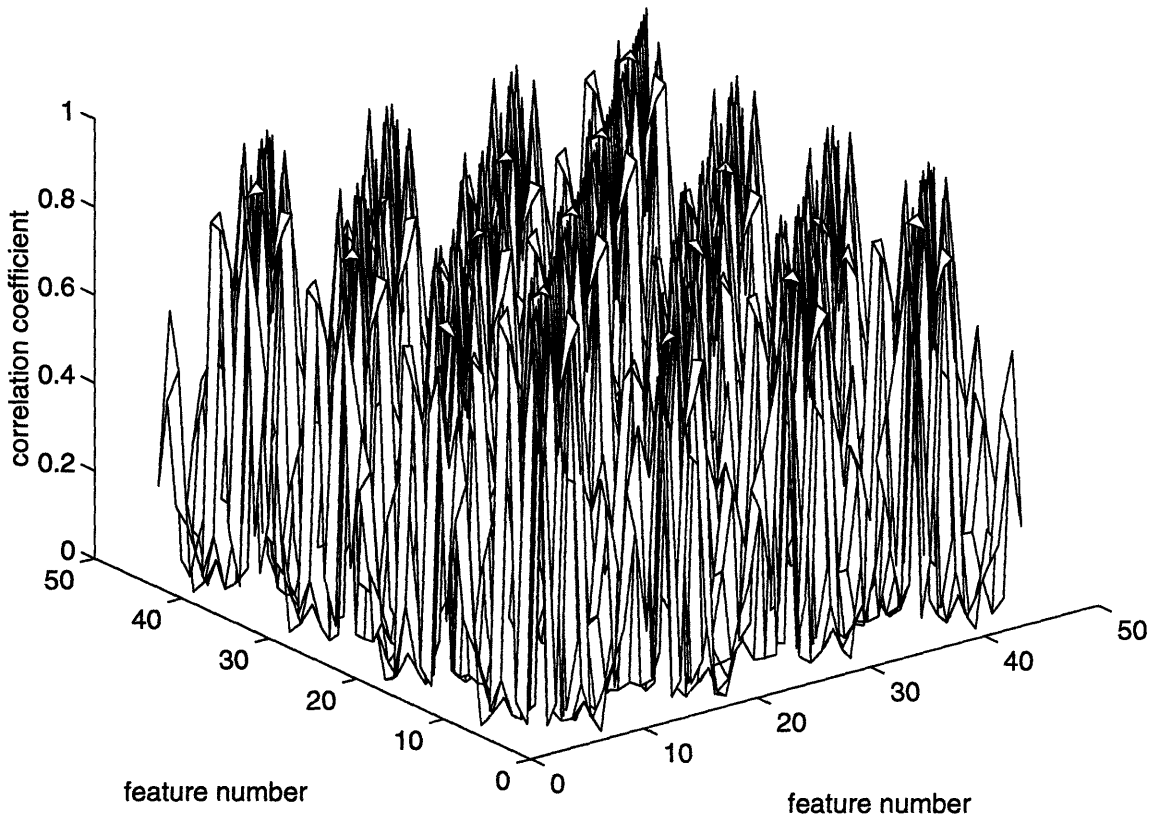


Figure 5.5: Auto-correlation coefficient matrix of the traditional run-length features.

helps to explain the poor performance of the RLRN vector. Since most information is stored in the first few columns of the run-length matrix, the only important features in RLRN are the first few features, which are the summation of the first few columns. The gray-level information is totally lost.

5.2.3 Comparison with co-occurrence method

Since the co-occurrence features are one of the most popular and powerful sources of features for texture characterization and have been ranked among the best by several comparative studies [19] [120], I compare these with the new run-length features. (Further studies of the co-occurrence features using the new feature extraction approach are described in Chapter 6.)

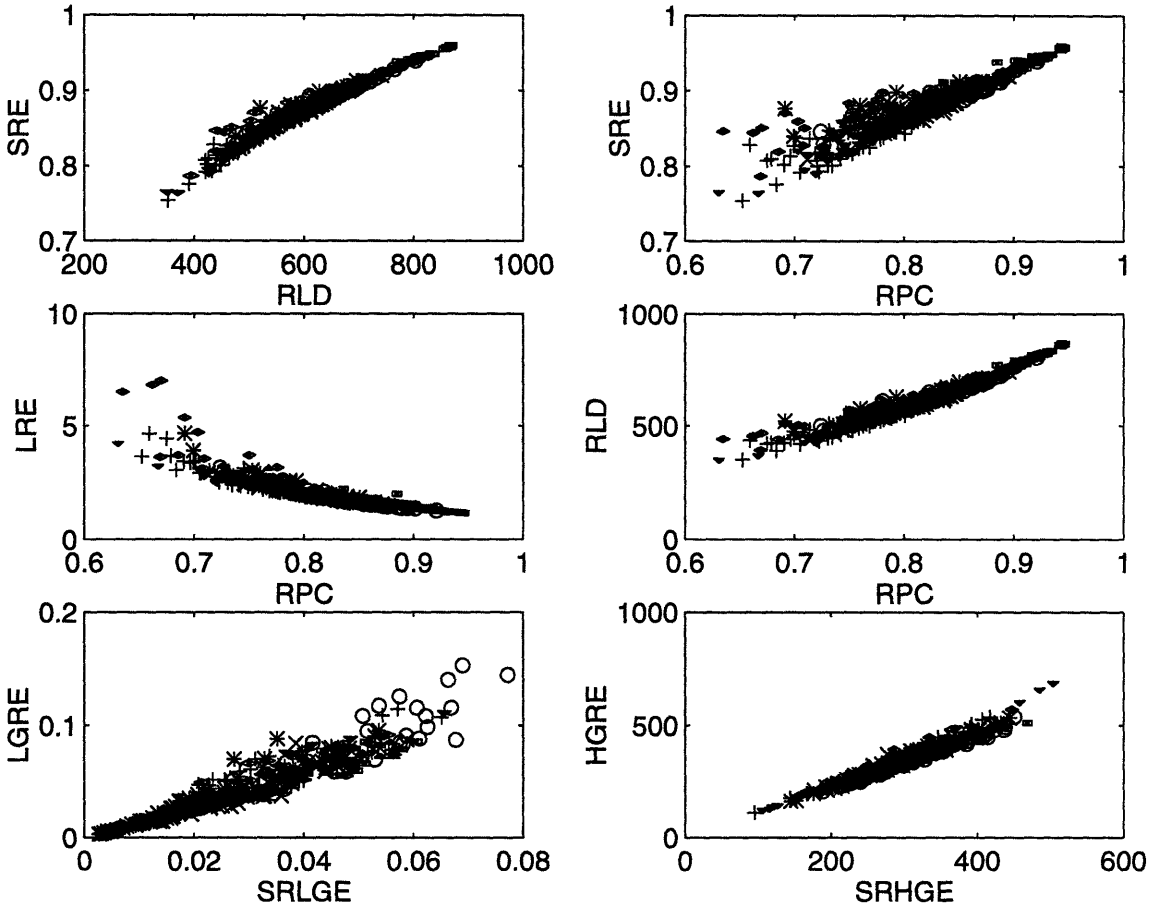


Figure 5.6: Scatter plots of several highly correlated traditional run-length texture features of the eight Brodatz textures. Due to overlap, not all eight class symbols can be discerned.

In this next experiment, I compare the performance of the traditional co-occurrence features with the run-length features using the Brodatz data set. Thirteen co-occurrence features are computed for each of the four directions—Contrast, Correlation, Entropy, Variance, etc.—as described in [48]. The KLT and Bhattacharyya distance feature selection method is applied to the feature vector. The classification results are shown in the first row of Table 5.4. Comparing this with the results in Table 5.2, we see that the co-occurrence features are better than the traditional run-length features. This is consistent with the conclusions in [19] and [120]. However, by comparing the result to Table 5.3, I draw a completely different conclusion: the new run-length matrix approaches give much better

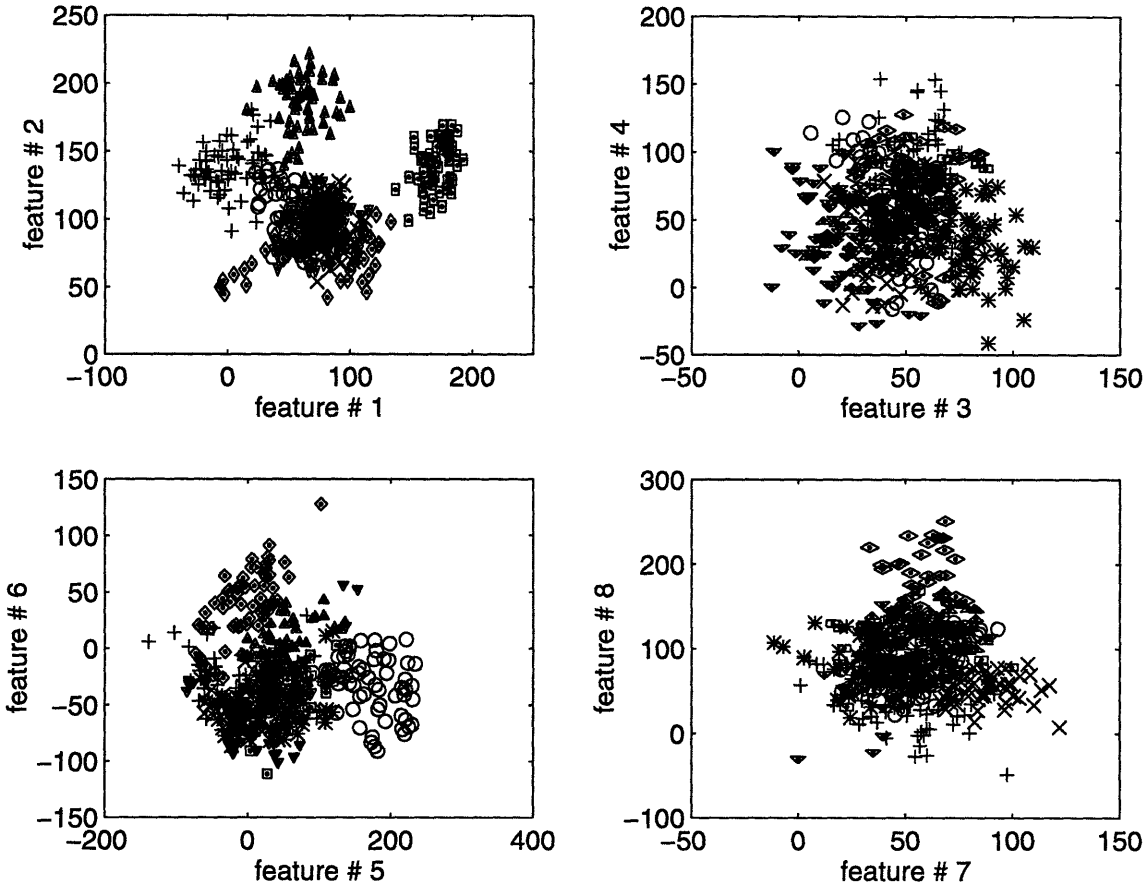


Figure 5.7: Scatter plots of the top eight features extracted by applying a MDEE transform on the original run-length matrices of the Brodatz textures. Linearly separable clustering is observed for most of the eight texture classes.

classification performance than the co-occurrence method. This demonstrates that there is rich texture information contained in the run-length matrices and that a method of extracting such information is of paramount importance to successful classification.

5.2.4 Comparison with wavelet method

I now compare the wavelet features described in Chapter 4 with the run-length features on the same eight classes of Brodatz textures. The texture feature used for each wavelet decomposition channel is the energy feature.

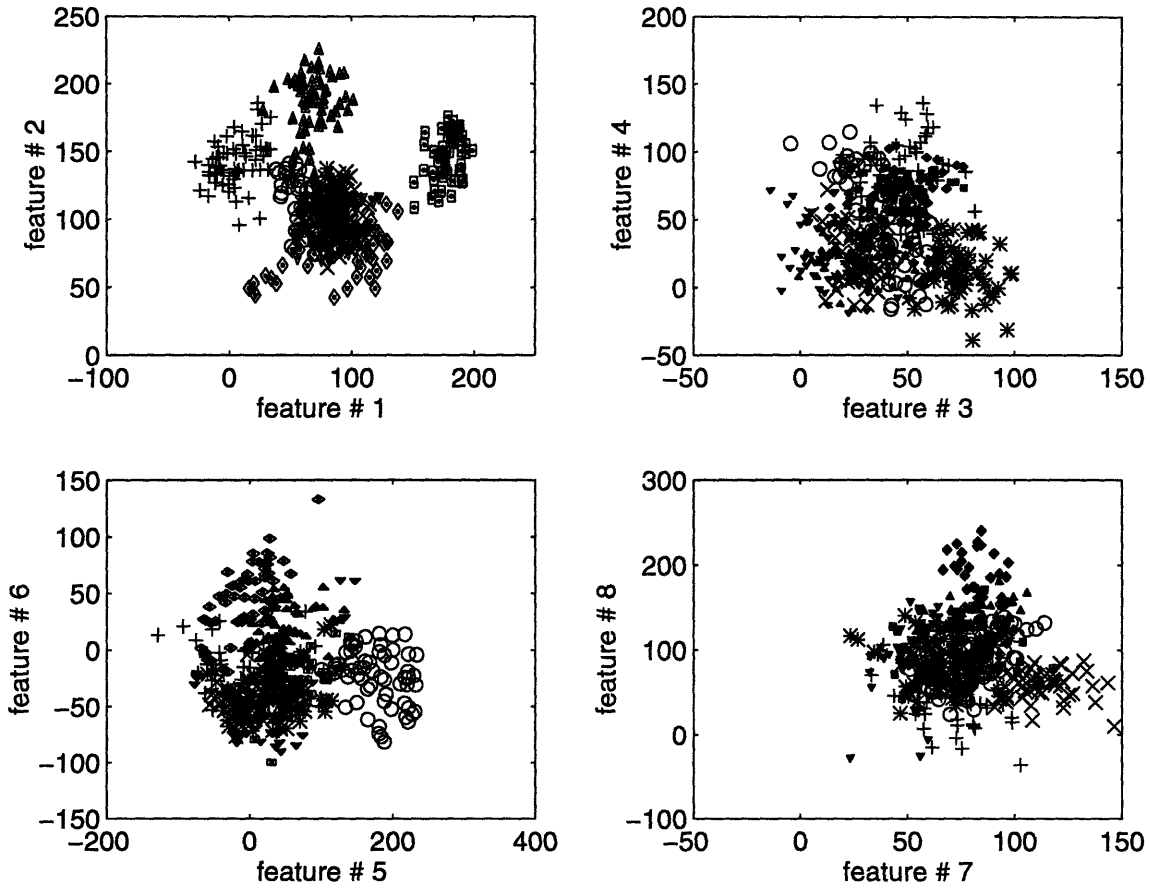


Figure 5.8: Scatter plots of the top eight features extracted by applying a MDEE transform on the run-length-one vector of the Brodatz textures. Linearly separable clustering is observed for most of the eight texture classes.

I apply the feature selection algorithm on both the standard wavelet features and the three levels of wavelet packet decomposition features. Classification results are shown in Table 5.4. The best classification rate is only 83% for the second-level wavelet packet features, which is less than that for the co-occurrence and run-length features. This is largely due to the small texture sample size, 32x32, which is not large enough to estimate a stable energy/frequency distribution. Still, it is important for any texture classification algorithm to give good performance on small images, so that they can be used for the more difficult image-segmentation applications.

Table 5.3: Brodatz texture classification results using the new dominant run-length matrix features.

Feature name	Original feature length	Number of selected features	Correct classification rate		
			Training data	Testing data	All data
p: columns 1:4	512	11	100.0	100.0	100.0
p: columns 5:32	3584	8	53.3	41.3	44.5
p: whole matrix	4096	11	100.0	100.0	100.0
p_p: columns 1:4	512	7	100.0	100.0	100.0
p_p: columns 5:32	3584	17	69.6	41.4	48.9
p_p: whole matrix	4096	10	100.0	100.0	100.0
p_g: GLRNV	128	8	100.0	100.0	100.0
p_r: RLRNV	128	20	95.2	63.9	72.3
p_o: GLRLOV	128	11	100.0	100.0	100.0

Table 5.4: Brodatz Texture classification results using co-occurrence and wavelet features.

Feature name		Original feature length	Number of selected features	Correct classification rate		
				Training data	Testing data	All data
co-occurrence		52	25	100.0	95.8	96.9
Wavelet packet	ALL	84	28	98.3	70.4	77.8
	LE1	4	4	67.5	65.3	65.9
	LE2	16	14	95.4	78.6	83.1
	LE3	64	27	98.8	69.8	77.6
	Standard	10	10	87.3	78.4	80.8

To summarize classification results discussed so far, the new DRM features significantly improve classification performance over the traditional run-length features. The new run-length features perform even better than the traditionally superior co-occurrence

features and the recently proposed wavelet features. Several of the proposed run-length matrices and vectors achieve perfect classification for the eight texture classes. To further compare these matrices and to test the hypothesis that only the first column of the run-length matrix is an effective feature vector, I apply the algorithms to the larger and more difficult Vistex data set.

5.2.5 Results on a larger data set

Sixteen images from the Vistex texture image database are used in this section. As described earlier, all image classes have the same flat histogram and no adaptive histogram equalization is applied to the images to compensate the nonuniform lighting. This makes the classification more difficult and the result a closer reflection of real-world applications.

Even for such a difficult data set, about 97% classification accuracy is achieved with the run-length matrices, as shown in Table 5.5. An especially interesting result is that the run-length-one vector gives excellent performance, similar to that of the original full matrix. This confirms that the proposed fast parallel processing algorithm can be used to extract useful run-length texture features. The overall good performance of all the run-length matrices indicates that the new feature extraction and selection scheme is successful.

Table 5.5: Vistex texture classification results using the new dominant run-length matrix features.

Feature name	Original feature length	Number of selected features	Correct classification rate		
			Training data	Testing data	All data
p: columns 1:4	512	17	99.9	96.8	97.6
p: whole matrix	4096	18	99.9	98.0	98.5
p_p: columns 1:4	512	19	100.0	96.8	97.6
p_p: whole matrix	4096	24	100.0	97.5	98.1
p_g: GLRNV	128	23	100.0	93.9	95.6
p_o: GLRLOV	128	18	99.8	97.0	97.8

5.3 Conclusion

In this chapter, I extract a new set of run-length texture features that significantly improve image classification accuracy over traditional run-length features. By directly using part or all of the run-length matrix as a feature vector, much of the texture information is preserved. This approach is made possible by the new multi-level dominant eigenvector estimation method introduced in Chapter 3. Combined with the Bhattacharyya distance measure, they form an efficient feature selection algorithm.

The advantage of this approach is demonstrated experimentally by the classification of two independent texture data sets. Perfect classification is achieved on the eight Brodatz images. The 97% classification accuracy on the 16 Vistex images further confirms the effectiveness of the algorithm. Experimentally, I observe that most texture information is stored in the first few columns of the run-length matrix, especially in the first column. This observation justifies development of a new fast, parallel run-length matrix computational algorithm.

Comparisons of this new approach with the co-occurrence and wavelet methods demonstrate that texture information contained in the run-length matrices possesses greater discriminatory power than conventional texture features. I hope this work will renew interest in run-length texture features and promote the success of more applications.

Chapter 6

Multi-view texture classification

In this chapter, I extend the multi-level dominant eigenfeature analysis one level further to a multi-view texture analysis approach. Using the same MDEE algorithm, I combine the feature vectors extracted from several different transform matrices—including the Fourier spectrum matrix, the run-length matrices, and the co-occurrence matrices—to form a more complete description of texture images. Although similar to many previous studies using combined texture feature vectors, this approach is a logical extension of the MDEE algorithm, which serves to combine new information and remove overlapping information from different texture matrices.

Before studying the multi-view analysis, I first conduct an experiment on the co-occurrence matrices using the MDEE and Bhattacharyya distance to extract texture features directly from the transform matrices. As discussed in Chapter 3, co-occurrence features offer a unique view of texture images that cannot be replaced by other methods, so I incorporate them into the multi-view feature vectors.

6.1 Texture feature extraction

6.1.1 Co-occurrence features

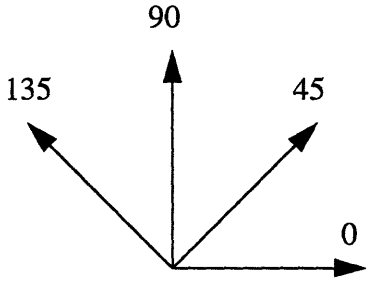
To restate the definition of a co-occurrence matrix, it is an estimate of the joint probability density function of texture image pixels separated by a particular row and column shift. Figure 6.1 shows an example of the four directional nearest neighbor co-occurrence matrices of an image sample.

The number of co-occurrence matrices that can be computed from an image is very large, because the row and column shift can vary from one to almost the window size. However, as the shift value increases, the information content of the co-occurrence matrix decreases, because the correlation between pixels separated by a large distance drops sig-

Original

1	1	1	2	2
3	4	2	2	3
4	4	4	4	4
5	5	3	3	3
1	1	3	4	5

Four directions



Gray level

Gray level	0	1	2	3	4	5
	1	6	1	1	0	0
	2	1	4	1	1	0
	3	1	1	4	2	1
	4	0	1	2	8	1
	5	0	0	1	1	2

Gray level

Gray level	45	1	2	3	4	5
	1	0	0	2	1	1
	2	0	4	0	2	0
	3	2	0	2	4	0
	4	1	2	4	2	2
	5	1	0	0	2	0

Gray level

Gray level	90	1	2	3	4	5
	1	0	1	1	1	2
	2	1	2	1	2	0
	3	1	1	2	6	1
	4	1	2	6	2	2
	5	2	0	1	2	0

Gray level

Gray level	135	1	2	3	4	5
	1	0	2	0	1	1
	2	2	0	1	2	0
	3	0	1	0	5	2
	4	1	2	5	2	1
	5	1	0	2	1	0

Figure 6.1: Four directional gray-level co-occurrence matrices.

nificantly in natural imagery. Weszka et al. [120] reported that small row- and column-shift values yield the best results in practice, so I test only the four directional nearest neighbor co-occurrence matrices.

The size of each co-occurrence matrix is also a problem, since it is the same as the image gray level number. An image quantization is necessary before the matrix computa-

tion. In the experiments, I use the same 32-level quantization as in the run-length experiments of Chapter 5.

From the co-occurrence matrices many features can be computed. Since Haralick et al. [48] developed the first 14 texture functions, there have been many studies of co-occurrence features. Most concentrate on analyzing the relative discriminatory power of individual features and how each individual feature performs for various applications. It is even suggested that the drawback to co-occurrence features is the huge number of potential features and the lack of any compelling underlying theory that would guide studies of any particular set of features [79].

Similar to the previous two chapters, I apply the new feature extraction method developed in Chapter 3 directly on the co-occurrence matrices to extract maximum texture information in an optimal way. In the experimental section, I compare the new features with the traditional features proposed by Haralick [46] on the Brodatz textures used in Chapter 5.

6.1.2 Combining feature vectors: multi-view classification

In the studies of extracting maximum texture information directly from transform texture matrices I have applied the MDEE method on the frequency, run-length, and co-occurrence matrices. By breaking each matrix into pieces, extracting useful information, then reassembling the information into a final representation, we have extracted very concise and highly informative texture features. To carry this approach one step further, I not only use the MDEE to assemble texture information from sections of a single texture matrix but also to combine information from different texture matrices.

To see the advantage of this approach, consider Meyer's analogy, as discussed in Section 3.1: choosing the right algorithm is like choosing the best view point of a statue in a museum; the orthogonal base rotation is like walking around a statue to look at its different sides.

However, sometimes we walk around the statue and find many good points of view. Each point of view reflects different characteristics of the art work, making it very hard to choose the best point of view. So we take several pictures from all the good viewing direc-

tions and put them in an album together to get the whole picture. For the texture classification study, the situation is similar: since different transforms make explicit different information, we can mosaic them together to make explicit all the texture information. The MDEE method is an excellent tool to conduct such a mosaicking task.

Figure 6.2 illustrates the structure of the multi-view processing scheme. Using MDEE we first extract texture information from each individual transform matrix then combine them into a more complete description of a texture image. The three transform texture matrices each offer a unique view of the texture image: the co-occurrence features capture the interaction of the local neighborhood pixels; the frequency features yield global information from averaging the whole image in the frequency channel; while the run-length features fit in the middle, capturing the medium-scale pixel interactions. These three features complement each other; together they expose texture information at all scales. Of course, they also offer overlapping and spurious information. The MDEE method serves to decorrelate this information and to compact it into a concise, complete description of texture.

There are apparent drawbacks of the multi-view analysis, most notably, the increased computational complexity. In previous chapters, to partially compensate for the speed loss, I use such faster methods as the FFT and parallel run-length algorithms. These transforms also can be carried out in parallel. Depending on different applications, we have to balance the trade-off between performance and computational complexity.

6.2 Classification experiments

The data sets used in this section are the same as those used for run-length features in Chapter 5. First, the co-occurrence features are compared on the eight Brodatz images, then all the new features extracted directly from the spectrum matrix, the run-length matrices, and the co-occurrence matrices, are compared on the 16 Vistex images. The sample image size and quantization levels are all the same as in Chapter 5. The classifier is also the same Gaussian classifier described in Section 3.3.1.

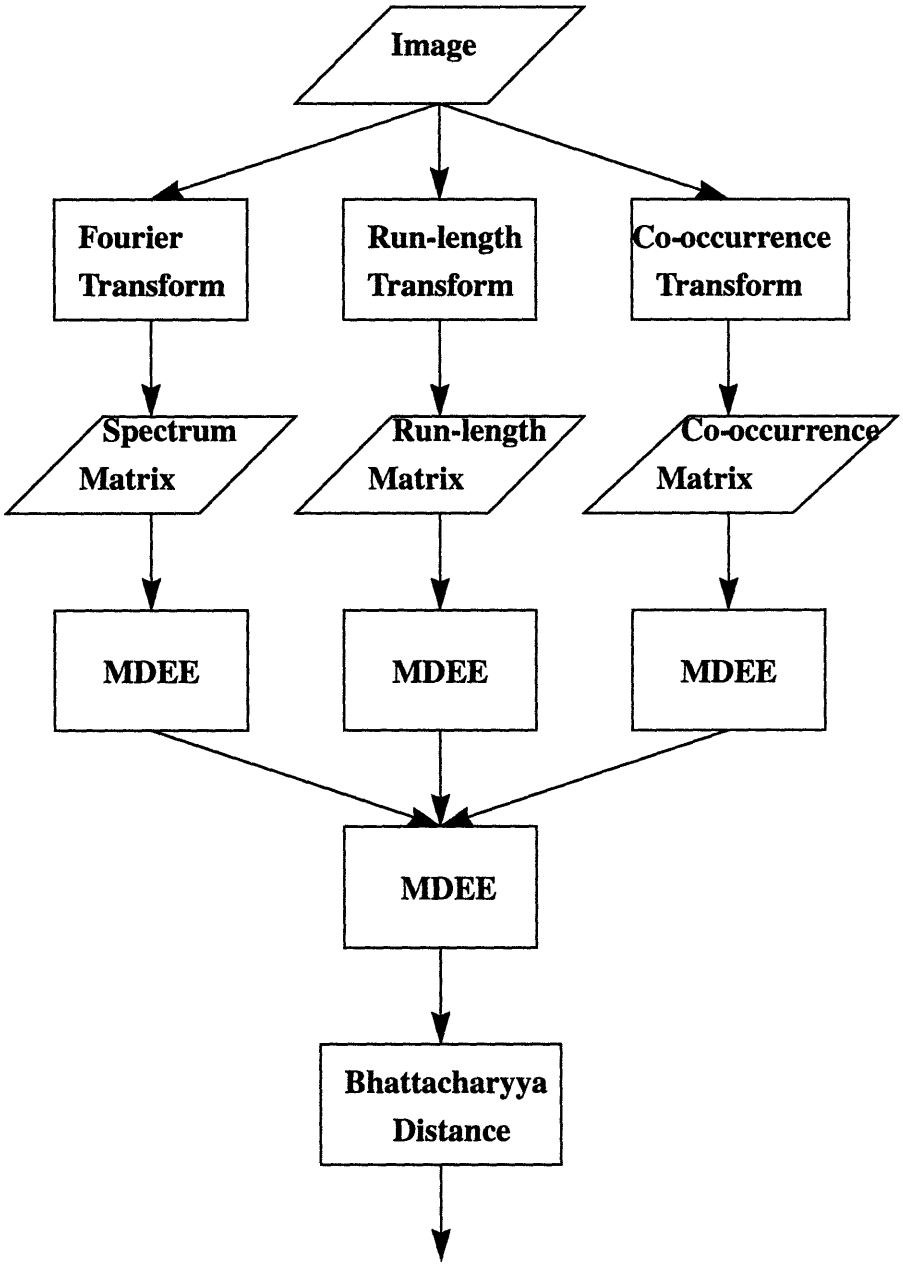


Figure 6.2: Combining three transform matrices of texture images.

6.2.1 Comparison of the traditional and the new co-occurrence features

In this experiment, I compare the performance of the traditional co-occurrence features with the new features on the eight Brodatz images shown in Figure 5.4. For the tradi-

tional approach, the first 13 co-occurrence features are computed for each of the four directions, namely, Contrast, Correlation, Entropy, Variance, etc., as described in [48]. The KLT and the Bhattacharyya distance feature selection methods are applied to the feature vector. The classification results are shown in Table 6.1. The combination of the four-direction features gives the best result, which is much better than the traditional run-length features in Table 5.2. This is consistent with previous comparative studies [19] [120].

Table 6.1: Brodatz texture classification results using the traditional co-occurrence features.

Feature name	Original feature length	Number of selected features	Correct classification rate		
			Training data	Testing data	All data
Dir-0	13	13	94.2	87.0	88.9
Dir-45	13	13	90.4	83.6	85.4
Dir-90	13	13	92.1	83.0	85.4
Dir-135	13	13	92.9	83.5	86.0
All	52	25	100.0	95.8	96.9

Table 6.2 shows the Brodatz texture classification results using the texture features extracted directly from the co-occurrence matrices by the MDEE and Bhattacharyya distance feature selection algorithm. Since the co-occurrence matrix is symmetric, only the upper triangle of the matrix including the diagonal is used. Comparing this to Table 6.1, the near-perfect classification again shows the success of this new approach. The new features are then applied to the more difficult Vistex data set. Good results are observed in Table 6.3.

6.2.2 Comparison of individual features with combined features

I now summarize the performances of all three new transform texture features I have studied so far, namely, the DSM features, the DRM features, and the new co-occurrence

Table 6.2: Brodatz texture classification results using the new co-occurrence matrix features.

Feature name	Original feature length	Number of selected features	Correct classification rate		
			Training data	Testing data	All data
Dir-0	528	27	100.0	99.1	99.3
Dir-45	528	26	100.0	99.0	99.3
Dir-90	528	28	100.0	99.3	99.5
Dir-135	528	29	100.0	99.0	99.3
All	2112	26	100.0	99.6	99.7

Table 6.3: Vistex texture classification results using the new co-occurrence matrix features.

Feature name	Original feature length	Number of selected features	Correct classification rate		
			Training data	Testing data	All data
Dir-0	528	25	98.4	78.7	84.0
Dir-45	528	28	98.8	70.7	78.2
Dir-90	528	28	99.5	80.8	85.8
Dir-135	528	29	99.1	75.8	82.0
All	2112	30	100.0	90.4	93.0

features. Table 6.4 shows the classification accuracies of the three transform matrix features on the 16 Vistex textures. All three methods give very good results. Notice that the DSM features give good results even on the 32x32 texture sample size, much better than the 83% classification accuracy of the wavelet features in Table 5.4.

To further improve the classification accuracy, I combine the three feature vectors using the MDEE algorithm. When combining the feature vectors, a normalization of the feature vectors is applied first so that all three feature vectors have similar absolute magni-

Table 6.4: Vistex texture classification results using the individual transform matrix features.

Feature name	Original feature length	Number of selected features	Correct classification rate		
			Training data	Testing data	All data
DSM	528	17	99.2	95.2	96.2
DRM	512	17	99.9	96.8	97.6
CO-OC	2112	30	100.0	90.4	93.0

tudes. Table 6.5 shows the improved results. The combination of any two feature vectors gives an impressive 99% classification accuracy. All three vectors together give near-perfect classification. Notice also that the selected feature vector length is similar to the selected feature vector length of each individual method in Table 6.4. This indicates that the overlapping information is mostly removed by the MDEE, demonstrating the advantage of the multi-view classification strategy.

Table 6.5: Vistex texture classification results using the combined transform matrix features.

Feature name	Original feature length	Number of selected features	Correct classification rate		
			Training data	Testing data	All data
CO-OC & DSM	2640	20	100.0	98.6	99.0
CO-OC & DRM	2624	22	100.0	98.9	99.2
DSM & DRM	1040	17	99.9	98.8	99.1
ALL THREE	3152	22	100.0	99.5	99.6

6.3 Conclusions

Through experiments on the co-occurrence matrix, I have again demonstrated the effectiveness of the feature extraction algorithm developed in Chapter 3. I carry the multi-level dominant feature extraction approach one step further to a multi-view analysis

approach. The near-perfect classification rate shows that by combining features from different viewpoints, we can extract more information with greater discriminating power.

Chapter 7

Multi-view analysis for textured object recognition

An ultimate goal of image analysis by computer—including texture analysis, image segmentation, or shape analysis—is to automatically recognize objects in the image. In this chapter, I apply the multi-view texture analysis approach to multi-view object recognition. For object recognition, the viewing points are not required to come from various texture transformations but may include any descriptions of the object. We can then use the multi-view analysis method to combine the description features to form a more complete representation of the object. Again, I use the name "multi-view" not to present a significantly new algorithm; rather, it is another logical extension of the MDEE analysis approach. It is a convenient word to express an object recognition approach analogous with "viewing a statue in a museum," as introduced in Chapter 3.

In the following experiments, I show a successful application of this approach to underwater plankton image recognition. I integrate such popular shape descriptors as moment invariants and Fourier boundary descriptors with the granulometric texture features to compose a very effective feature vector for the plankton images. Then, using the improved Learning Vector Quantization (LVQ) neural network classifier developed in Chapter 3, I classify the plankton images into several taxonomic categories.

This chapter is organized as follows. An introduction to plankton image recognition is given in Section 7.1. The three feature extraction methods—moment invariants, Fourier boundary descriptors, and granulometric features—are described in Section 7.2. Section 7.3 describes real-time data acquisition and preprocessing and reports experimental results from the classification of the six plankton taxa. Section 7.4 summarizes the conclusions.

7.1 Introduction to plankton recognition

At the base of the food chain in the ocean, plankton have a large impact on marine ecosystem dynamics, and it is important to study how changing climate and human activities impact plankton populations. Such studies require large-scale mapping at high spatial and temporal resolutions of plankton distribution and abundance together with taxonomic and size composition. The surveys also should be repeated at regular intervals so seasonal growth, population cycles, and productivity of the surveyed areas can be studied on a quantitative basis. Until recently, however, it has been very difficult or impossible to conduct such extensive experiments, because most plankton move continuously in three dimensions and are affected by varying small- and large-scale physical phenomena [83].

Traditionally, plankton surveys are conducted with such equipment as towed nets, pumps, and Niskin bottles. Because of the laborious deployment process and limited sample storage space on ship, the spatial sampling rate is extremely low. The painstaking and error-prone post-processing (manual counting of samples through a microscope and data entry) may take months or years, which effectively prohibits large-scale, high-resolution, three-dimensional surveys over periods of time. Some previous models of production and growth that are based on feeding or environmental conditions may be flawed if the interactions of organisms with one another and with the local environment are estimated from samples drawn at inappropriate intervals of space or time [82].

To overcome the limitations of traditional plankton sampling instruments, a new Video Plankton Recorder (VPR) was developed [26]. As the VPR is towed through the water, it continuously captures magnified plankton images, providing a spatial resolution of plankton distributions that cannot be obtained with other equipment. For large-area survey, the amount of image data can be overwhelming, necessitating an automated approach to plankton recognition if all data are to be processed. This would not only save a great deal of man power but also make the real-time sorting of plankton possible. Real-time abundance and distribution data on zooplankton and accompanying environmental variables are needed to guide researchers during field studies on population and community processes, just as physical oceanographers have for decades used real-time measurements of

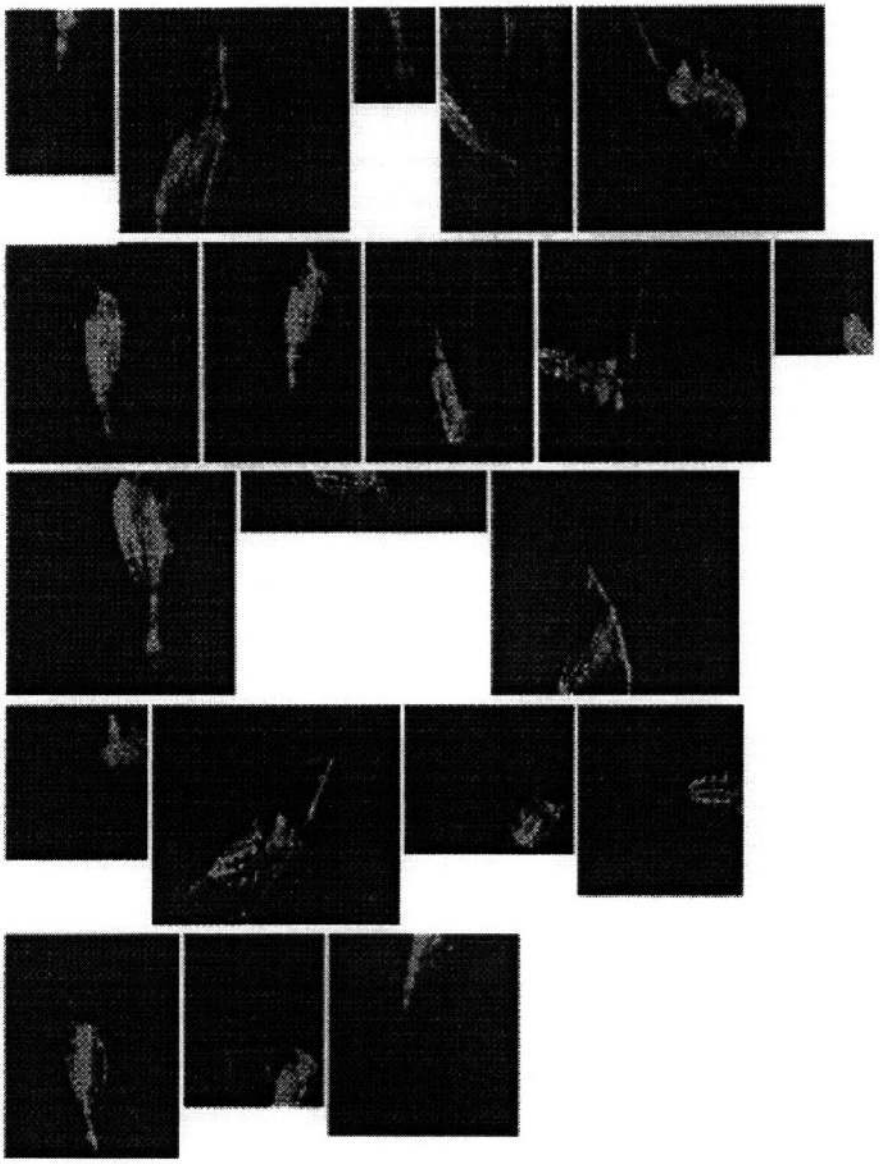
temperature and conductivity to adjust their survey strategy according to observed phenomena [83].

Now that high-quality images of individual plankton can be obtained with the VPR, our approach to the full automation of at-sea analysis of plankton size and taxonomic composition focuses on the development of an image analysis and pattern recognition system for real-time processing of the large volume of image data being acquired. The development approach includes three parts: 1) a hardware/software system for preprocessing of the images (including real-time image capture, object detection, and in-focus analysis) and digital storage of detected object images; 2) pattern recognition algorithms for automated identification and classification of planktonic taxa; 3) incorporation of the pattern recognition algorithms into a high-performance image analysis system to achieve a real-time processing capability. Development of a preprocessing and acquisition system as described in Step 1 has been completed and used to detect and save subimages of planktonic taxa in real-time while at sea [27].

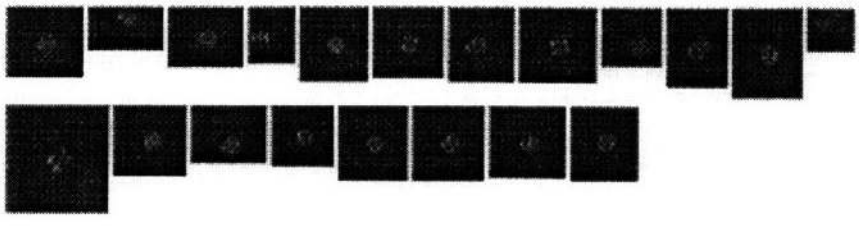
In this chapter, I mainly address Step 2 and demonstrate an automated approach to plankton image classification. The experimental data sets differ from those used for most previous object recognition research in four aspects: 1) the underwater images are much noisier, 2) many objects are in partial occlusion, 3) the objects are deformable, and 4) images are projection variant, i.e., the images are video records of three-dimensional objects in arbitrary positions and orientations. Figure 7.1 shows example subimages extracted from the larger video fields, illustrating the diversity of images within individual taxa.

7.2 Feature Extraction

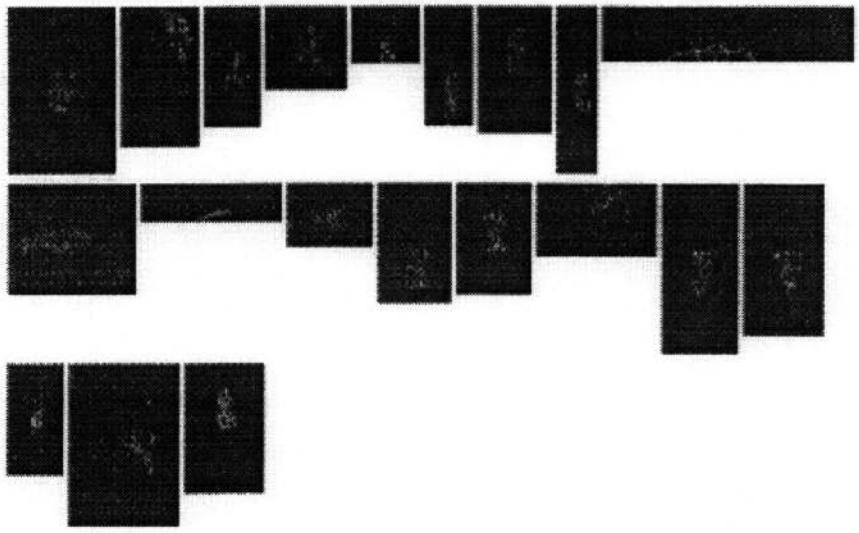
Classification of two-dimensional shapes regardless of position, size, and orientation is an important problem in pattern recognition. Applications range from industrial inspection and scene analysis to optical character recognition. The two most widely used features are moment invariants and Fourier boundary descriptors. Classification of three-dimensional projection-variant objects is even more difficult.



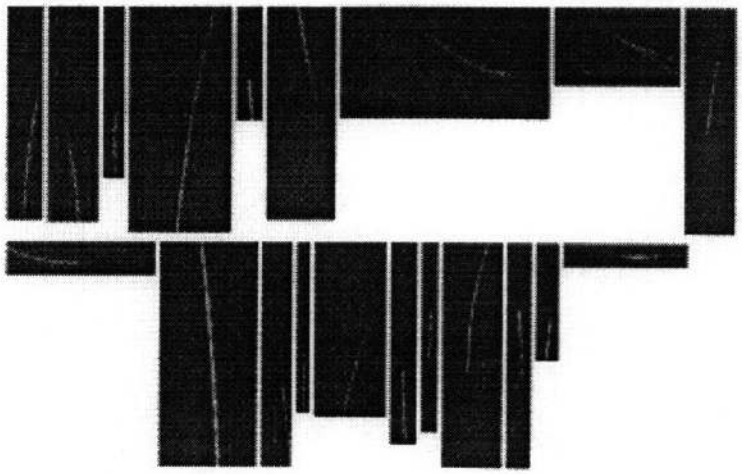
(a). CALANUS



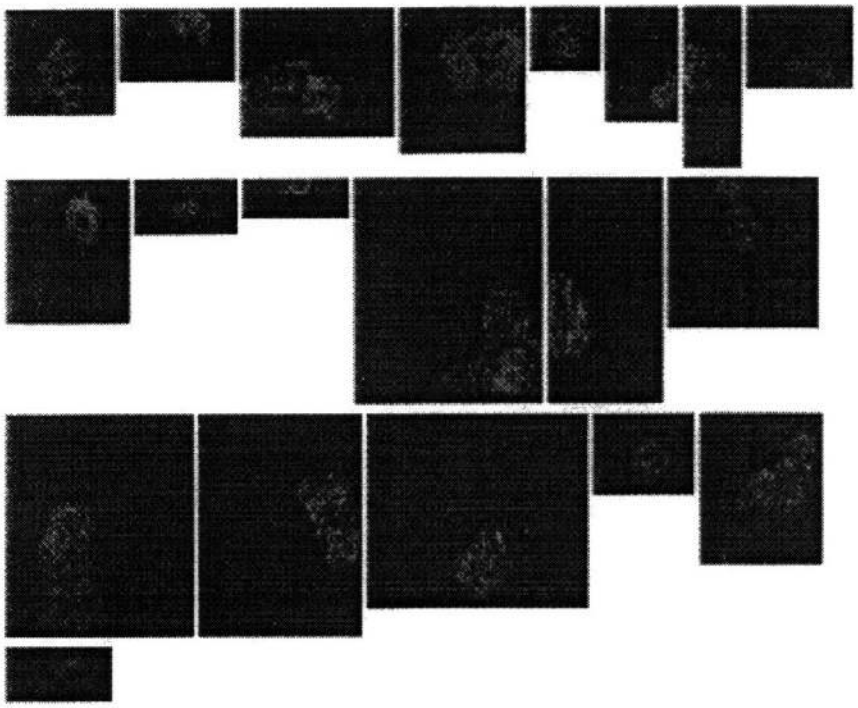
(b). DIAT-CENTR



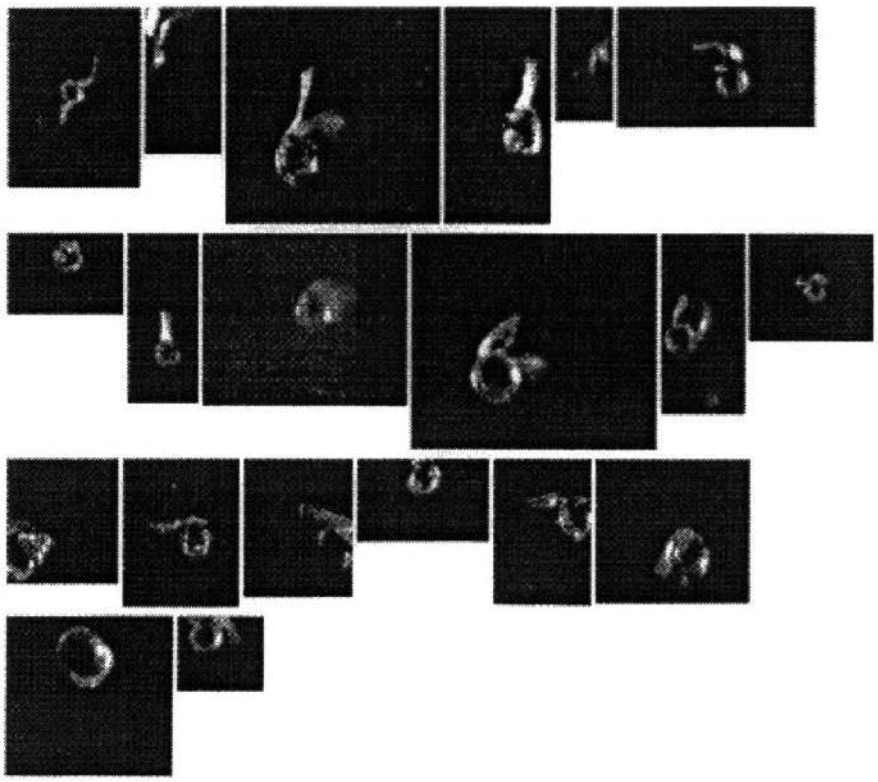
(c). DIAT-CHAET



(d). DIATOM



(e). DIATOMCOLO



(f). PTEROPOD

Figure 7.1: Twenty sample images for each of the six types of plankton.

In this section, I use gray-scale granulometric features as a powerful pattern descriptor, which captures both shape and texture signatures, as a step toward addressing the three-dimensional problem. I also use the multi-view analysis technique to combine the three types of feature vectors to form a more complete description of the plankton patterns. I briefly review the three feature types here.

7.2.1 Moment Invariants

The concept of moments as invariant image features was first introduced by Hu [53] and later revised by Reiss [91]. Moments and functions of moments have been used as pattern features in many applications. Some examples and comparisons of different features are found in [45] [90] [104]. In the experiments, I use the seven invariant moments described by Hu [53] and Gonzalez [45].

A $(p+q)$th order moment of a continuous image function $f(x, y)$ is defined as

$$m_{pq} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} x^p y^q f(x, y)dxdy \qquad p, q = 0, 1, 2, \ldots . \tag{7.1}$$

For digital images the integrals are replaced by summations and m_{pq} becomes

$$m_{pq} = \sum_x \sum_y x^p y^q f(x, y). \tag{7.2}$$

The central moments of a digital image can be expressed as

$$\mu_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q f(x, y), \tag{7.3}$$

where

$$\bar{x} = \frac{m_{10}}{m_{00}} \qquad \bar{y} = \frac{m_{01}}{m_{00}}.$$

The normalized central moments are derived from these central moments as

$$\eta_{pq} = \frac{\mu_{pq}}{\mu^{\gamma}_{00}}, \qquad \gamma = \frac{p+q}{2} + 1, \qquad p+q = 2, 3, \ldots. \tag{7.4}$$

Based on methods of algebraic invariants, Hu [53] derived seven invariant moments, ϕ_i, i = 1... 7, using nonlinear combinations of the second and third normalized central moments. These invariant moments possess the desirable properties of being translation, rotation, and scale invariant. For the analytical definition of the seven invariant moments, refer to [53] or [45].

7.2.2 Fourier Descriptor

The first in depth study of the Fourier descriptor was given by Zahn and Roskies [121] and later refined by Persoon and Fu [87]. Recently, more research effort has been devoted to shape classification based on Fourier descriptors [90] [60] [92]. The most common boundary models include the curvature function, centroidal radius, and complex contour coordinates. Kauppinen et al. [60] give a detailed experimental comparison of different models. I use the radius Fourier Descriptor (FD) and complex contour Fourier descriptor, which were shown to be the best among FDs tested in [60].

Consider a closed boundary defined by a closed sequence of successive boundary-pixel coordinates (x_i, y_i). The centroidal radius function expresses the distance of boundary points from the centroid (x_c, y_c) of the object,

$$r_i = \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2}. \tag{7.5}$$

A complex contour coordinate function is simply the coordinates of the boundary pixels in an object-centered coordinate system represented as complex numbers,

$$z_i = (x_i - x_c) + j(y_i - y_c). \tag{7.6}$$

Since both functions are computed around the center of the object, they are automatically translation invariant. To achieve rotation and scale invariance, a Fourier transformation of the boundary signature is generally used. For digital images I use the discrete

Fourier transform (DFT). The shift-invariant DFT magnitude gives a rotation-invariant feature vector. Scale invariance is achieved by normalizing all DFT magnitudes by a non-zero DFT magnitude value, because of the linear property of the DFT,

$$F(nf(x)) = nF(f(x)). \tag{7.7}$$

Usually, the first non-zero frequency component is used.

The feature vector for a radius Fourier descriptor is

$$FD_r = \left[\frac{|F_1|}{|F_0|} \cdots \frac{|F_{N/2}|}{|F_0|}\right], \tag{7.8}$$

where N is the boundary-function length and F_i denotes the ith component of the Fourier spectrum. Notice that only half the spectrum need be used because of the symmetric property of the Fourier transform of real functions. This Fourier descriptor is very similar to the dominant spectrum method I developed for texture classification. In fact, we can think of the object boundary as a one-dimensional texture and use the DSM to extract texture features.

The contour Fourier method transforms the complex coordinate function in Eq. (7.6) directly. The feature vector is

$$FD_c = \left[\frac{|F_{-(N/2-1)}|}{|F_1|} \cdots \frac{|F_{-1}|}{|F_1|}\frac{|F_2|}{|F_1|} \cdots \frac{|F_{N/2}|}{|F_1|}\right]. \tag{7.9}$$

In this case, both positive and negative halves of the complex spectrum are retained. Because the complex function is centered around the coordinate origin, F_0 has zero value and F_1 is used for the normalization.

7.2.3 Definition of granulometric features

The above traditional features are mostly developed in a well controlled pattern environment. Test images are usually shifted, scaled, and rotated versions of a very small set of perfect images of such simple objects as hammers, scissors, airplane silhouettes, and let-

ters. However, most boundary features do not perform well when a small amount of noise is added to the original images [60]. As mentioned earlier, the experimental data sets are not only much noisier but the plankton are non-rigid, projection-variant, and often in partial occlusion.

To overcome or, at least, partially alleviate these difficulties, I turn to features based on mathematical morphology [76] [98] known as granulometries. Granulometries are introduced in the sixties by Matheron as tools to extract size distributions from binary images [76]. The approach is to perform a series of morphological openings of increasing size and to map each opening size to the number of image pixels whose value goes from 1 to 0 at this size. The resulting curve, often called the pattern spectrum [74], maps each size to a measure of the image parts with this size. Typically, the peak of this curve provides the dominant object size in the image. For examples of the application of granulometries, refer to [111] [112].

Beyond pure size information, granulometries actually provide a "pattern signature" of the image to which they are applied and can be used successfully as elements of a feature vector for shape classification problems [97]. Furthermore, the concept of granulometries can easily be extended to gray-scale images. In this context, granulometric curves capture information on object texture as well as shape. Additionally, various types of gray-scale granulometric curves can be computed, depending on the underlying family of openings or closings used. For example, curves based on openings with line segments capture information on bright, linear image features, whereas curves based on closings with disk-shaped elements capture information on dark, "blobby" image parts (see [112] for other applications of gray-scale granulometries).

To capture information both on linear, elongated and "blobby" image parts, whether dark or bright, I use four types of gray-scale granulometries. The curves are normalized for invariance to average background gray level. For example, Figure 7.2 shows an image of copepod oithona together with the corresponding pattern spectrum characterizing bright, blobby image parts. The peak of this curve characterizes the size and contrast of the body of the organism. Figure 7.3 shows a similar curve for an organism known as a pteropod. The corresponding granulometric curve is distinctly different from the previous one, reflecting the more complex body configuration of the pteropod.

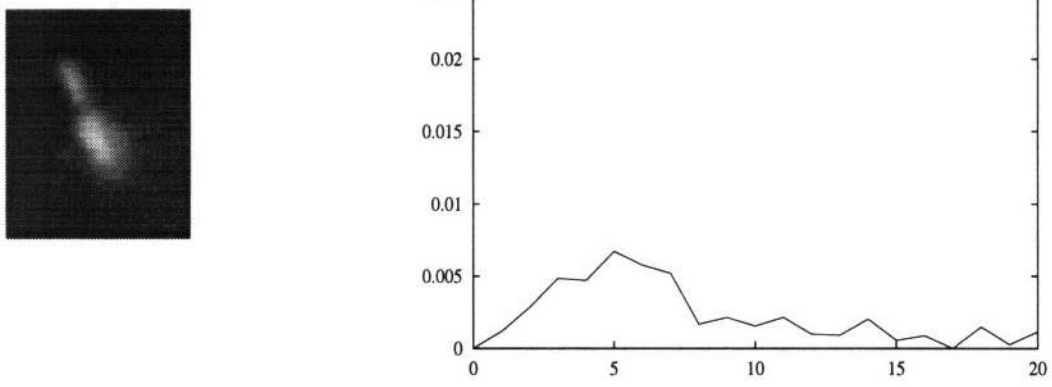


Figure 7.2: Copepod oithona image and its corresponding granulometric curve.

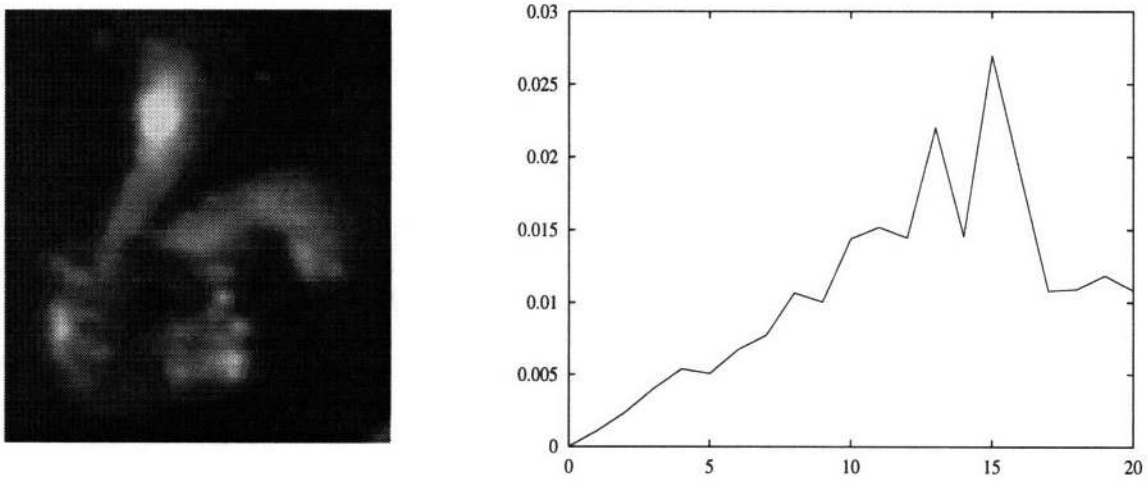


Figure 7.3: Pteropod image and its corresponding granulometric curve.

However, traditional granulometry algorithms involve sequential opening and closing operations with structuring elements of increasing size and are therefore extremely computationally intensive. This has limited the application of granulometry in image analysis. It also poses a problem for the experiments in this thesis, since eventually we would like to incorporate the granulometry algorithm into a real-time processing system. To overcome this problem, I use several novel gray-scale granulometry algorithms developed by Vincent [112] to compute the granulometric curves, which are several orders of magnitude faster than traditional techniques. This will allow these algorithms to be incorporated in a complete system implementation for real-time plankton classification.

Similar to the way I handle the DSM and the DRM, I use the gray-scale granulometric curves directly as feature vectors, instead of the moments of the curves as used by previous researchers [31] [32].

7.2.4 View combination

I now apply the multi-view texture analysis approach to multi-view object recognition. The "statue in the museum" described in Chapter 3 now becomes the plankton image. For object recognition, we don't have to limit our viewpoint only to texture transformations. To describe the plankton images, we can choose the three points of view described above: body moments, object boundary, and object texture and shape. We stand a much better chance of recognizing the object if we collect all types of information available, combine them, and make selections. Exactly the same multi-view analysis algorithms developed in Chapter 6 can be used here. By replacing the three texture transformations in Figure 6.2 with the three feature vectors described above, we have the processing structure for this object recognition work. In the following section, I demonstrate the advantage of this approach experimentally.

7.3 Classification experiments

7.3.1 Data acquisition and preprocessing

Data acquisition and processing for the current implementation are carried out in two phases. First, a real-time hardware/software system detects in-focus objects, defines a sub-image around each detected object, then saves the much lower bandwidth data stream to disk. Digital storage requirements are reduced by more than two orders of magnitude and the time required to manually identify training images is accelerated by a similar factor, when compared with manually jogging a videotape editing machine to search for organisms. In a second phase, more computationally expensive algorithms are applied to the condensed data set for segmentation, feature extraction, and classification.

Real-time data acquisition and focused-object detection

The VPR uses a video camera with telephoto lens and a red strobe to obtain magnified images of plankton. The strobe is synchronized with the camera at 60 fields per second. Together, the camera's high resolution (570x485 pixels) and the strobe's short pulse duration allow detailed imaging of the plankton (10-μm resolution for the 0.5-cm field of view) [26]. The goal of the VPR's real-time video processing system is to archive to tape digital images of all sufficiently large, bright, and in-focus objects as they appear in the video stream.

Live or recorded video and time-code data are sent to the video processing system, which currently consists of a Sun SPARCstation 20/72 connected to an Imaging Technologies 151 pipelined image processor and a Horita time-code reader. The image processor can perform real-time (60 field per second) digitization, 3x3 convolutions, rank value filtering, and frame buffer data exchanges with the host workstation. A multi-threaded algorithm on the host is used to supervise the image processor, collect time-code data, compute edge strength, and transfer in-focus subimages to disk.

A simple but effective algorithm has been devised to detect and record objects in real-time. First, bright large blobs are located in a median-filtered image by finding the connected components from run-length lists computed in hardware. For each large blob, the

first derivative of the Sobel edge intensity (basically a second derivative of intensity) along the blob's perimeter is used to reject objects that are out of focus. A gray-scale subimage surrounding any in-focus targets is immediately passed to the workstation for archival and taxonomic classification.

The three main parameters of these algorithms are image intensity threshold, edge strength threshold, and minimum object area in pixels. These are adjusted empirically before data collection based on various factors including lens magnification, transmissivity of the water, and lighting characteristics. The objective is to achieve a very high probability of detection with a reasonable probability of false alarm, since more intensive processing can be applied to the much smaller data set that results.

On average, about 3 out of 60 video fields contain an in-focus object, and only a subimage surrounding each object is saved to disk as an individual file. These object files are time-stamped using the corresponding video time code for precise correlation with ancillary hydrographic and position data. This culling process typically reduces the amount of image data to be stored and classified by a factor of 100 or more, thus making the remaining processing computationally feasible.

Data description and preliminary processing

In this experiment, six classes obtained from nearly 2,000 plankton subimages captured by the VPR are used to test the pattern classification algorithms. They include 133 Calanus, 269 Diat-centr, 658 Diat-chaet, 126 Diatom, 641 Diatomcolo, and 42 Pteropod images. Figure 7.1 shows 20 sample images for each of the six plankton taxa. Half of the images are used as training data and half for testing.

Each gray-scale image is first segmented into a binary image using a simple mean-shift method. I use the mean value of the image to threshold the image, then the mean values of the object and the background are computed. The average of the two mean values is used as a new threshold to segment the image again. The process iterates until a stable threshold is reached. Since the images are mostly bimodal, only two iterations give a very good segmentation result, as shown in Figure 7.4. We are currently developing a more robust connectivity-based thresholding technique and will compare the two methods in future work.

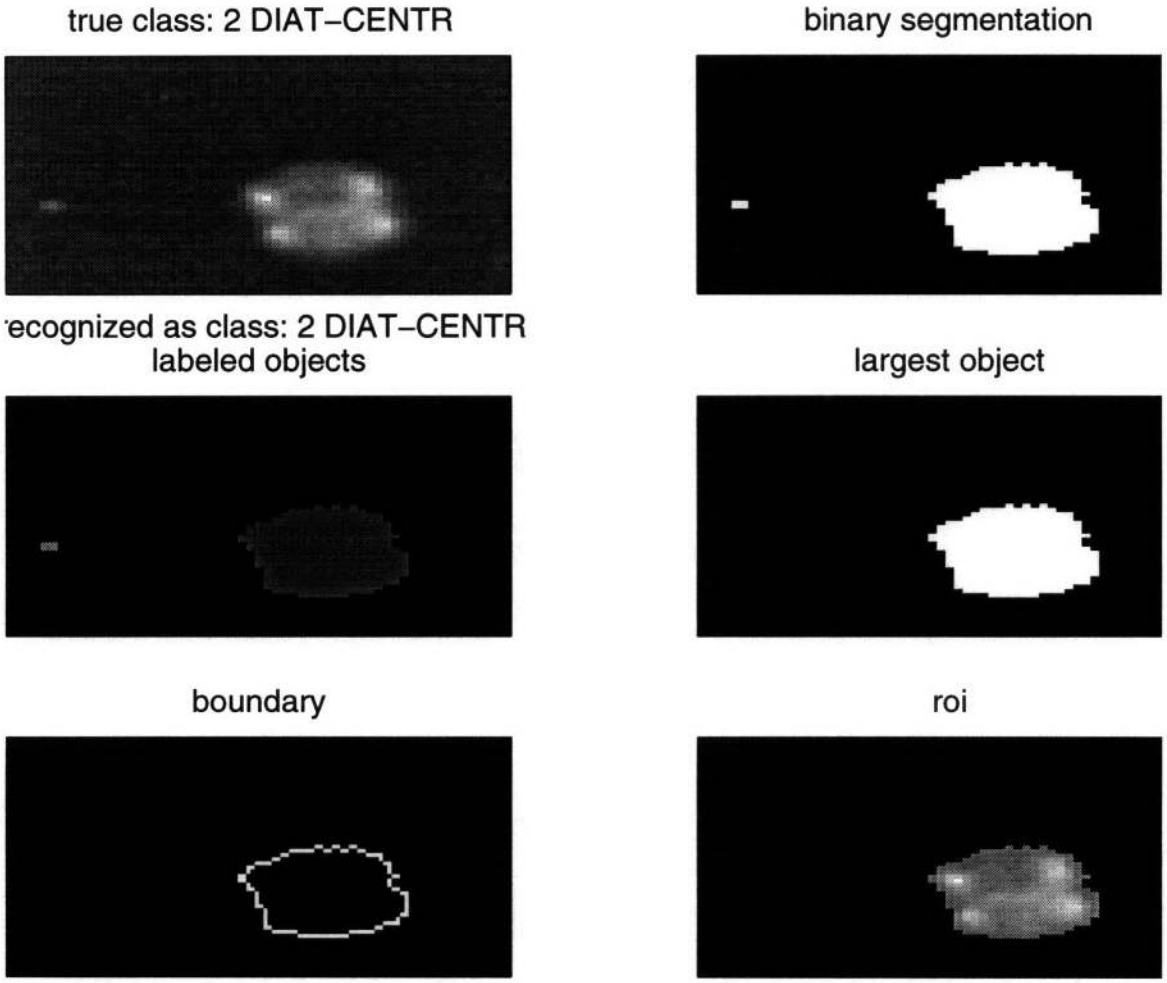


Figure 7.4: Illustration of the intermediate results of the image processing steps.

Next, the largest binary object is used to compute the boundary descriptors. This binary image is also used to mask the original grey-scale image to obtain the region of interest (ROI) image, from which moment features are computed. Figure 7.4 illustrates the results for each of these processing steps. Granulometric features are computed directly from the gray-scale images.

A suite of experiments was conducted to study the performance of the three types of feature vectors, their combinations, and the improved LVQ classifier described in Chapter 3. All classification results shown in the following tables are obtained from the data set described above. Since there seems to be no simple solution for determining the best net-

work configuration, no exhaustive search was conducted to determine the best network parameters. However, I investigated several network configurations and selected the one that appears to be most appropriate for the application. Throughout the experiment, I use the following parameters: 200 competitive layer neurons, learning rate 0.1, parallel training sample number 120 per epoch, i.e., 120 training samples are feed into the classifier at each training epoch.

7.3.2 Classification results using individual feature vectors

Table 7.1 summarizes classification results using the three individual feature vectors. Only 65% classification accuracy is achieved using moment features. This may be attributed in part to the relatively short feature vector (seven moment invariants), but the many images of the same class are also very different in shape because of variations in projection direction, organism body motion, and image occlusions. This shape inhomogeneity also affects the performance of the FD features. For the testing data, only 69% classification accuracy is achieved by the contour FDs. The radius FD features outperform the contour FD features by 10%. This contrasts with [60], where contour FDs give better results than the radius FDs. Such a discrepancy in results may be caused by the differences in the data sets. The sampling number I use for the boundary function is 360 points, much higher than for many previous studies, because plankton have noisier, more irregular boundaries requiring a higher sampling rate to capture high-frequency information. The granulometry features give the best performance, better than 90% accuracy. This demonstrates the features' insensitivity to occlusion, image projection direction, and body motion because of the rich three-dimensional texture and shape information captured.

7.3.3 Classification results using combined feature vectors

The confusion matrices of the classification results using the three individual feature types are given in Table 7.2, 7.3, and 7.4. Notice that the shapes of these matrices are quite different. The moments are good at distinguishing Diat-centr and Diatom; the radius FD generates good results on Diat-centr and Diat-chaet; the granulometry features perform well on all classes except the Diat-centr. All these suggest that the discrimination abilities

Table 7.1: Classification results of the six classes of plankton images using individual feature vectors.

Feature types	Training	Testing	All data	Feature length
Moment invariants	67.74	63.13	65.44	7
Contour FD	83.7	69.2	76.5	28
Radius FD	94.6	78.0	86.3	21
Granulometry	97.8	86.4	92.1	29

of the three feature types are distinct. They make explicit different information regarding the plankton images. To form a more complete description of the plankton patterns, I use the multi-view analysis technique to combine all three feature types into a single feature vector. The combined vector yields 95% classification accuracy as shown in Table 7.5, which is comparable to what a human observer can achieve by rapid visual inspection. Notice also that the feature length is condensed to around 20 from more than 300, demonstrating the efficiency of the feature selection approach.

Table 7.2: Confusion matrix of the moment invariant features

Names	Calanus	Diat-centr	Diat-chaet	Diatom	Diatomcolo	Pteropod
Calanus	41	0	71	0	20	4
Diat-centr	3	241	3	0	65	8
Diat-chaet	67	3	446	8	180	11
Diatom	0	0	10	116	0	0
Diatomcolo	21	23	127	2	373	13
Pteropod	1	2	1	0	3	6

Not all combinations show an improvement in results, for example, the combined contour and radius FDs. Adding moment features to the granulometry feature vector only improves results slightly. The short moment feature vector seems to contain only a subset of the information contained in the granulometry features.

Table 7.3: Confusion matrix of the radius Fourier descriptors

Names	Calanus	Diat-centr	Diat-chaet	Diatom	Diatomcolo	Pteropod
Calanus	71	0	9	3	26	4
Diat-centr	0	262	2	0	4	1
Diat-chaet	22	6	602	5	66	3
Diatom	2	0	3	117	1	3
Diatomcolo	34	1	42	1	539	9
Pteropod	4	0	0	0	5	22

Table 7.4: Confusion matrix of the granulometry features

Names	Calanus	Diat-centr	Diat-chaet	Diatom	Diatomcolo	Pteropod
Calanus	117	2	15	2	7	0
Diat-centr	0	239	7	0	0	0
Diat-chaet	15	27	600	5	25	0
Diatom	1	0	16	116	2	0
Diatomcolo	0	1	20	3	607	0
Pteropod	0	0	0	0	0	42

Table 7.5: Classification results of the six classes of plankton images using combined feature vectors

Feature types	Training	Testing	All data	Feature length
Contour FD & Radius FD	90.4	76.8	83.6	29
Moments & Granulometry	97.5	87.5	92.5	29
Granulometry & Radius FD	98.7	91.5	95.1	24
Moments & Granulometry & Radius FD	98.0	92.2	95.1	19

From the combined feature vector confusion matrix in Table 7.6, we see that the Diat-chaet images are misclassified more than any other type, probably because the images have structures similar to those of other image classes. For example, the second Diat-chaet

image in the second row of Figure 7.1 (c) is quite similar to DIATOM, and some small samples of Diat-chaet may be confused with DIAT-CENTR. All images are hand picked by a highly trained biologist and sometimes only very subtle characteristics are used to judge the occluded plankton images. Given the data quality, the overall classification rate is very encouraging.

Table 7.6: Confusion matrix of the combined moments, radius FDs, and granulometries features

Names	Calanus	Diat-centr	Diat-chaet	Diatom	Diatomcolo	Pteropod
Calanus	120	0	9	1	8	1
Diat-centr	0	262	7	0	1	2
Diat-chaet	13	6	627	2	23	0
Diatom	0	0	5	120	0	0
Diatomcolo	0	1	10	3	609	0
Pteropod	0	0	0	0	0	39

7.3.4 Comparison of the LVQ training methods

Figure 7.5 compares a traditional serial training LVQ method to the new parallel algorithm described in Section 3.3.2. The three lines show the progression of classification accuracy with increasing numbers of training epochs using three training methods. The dashed line represents the traditional training method with all neurons initialized with the mean value of the training samples; the dotted line shows the traditional training method with the statistical initial condition; the solid line gives the result of the new parallel training method. The new method reaches 98% training accuracy within 100 epochs, while the traditional methods achieve 95% accuracy using nearly 10,000 epochs.

Figure 7.6 compares the traditional method with the new parallel training method using scatter plots of the first two dimensions of the feature vectors and their corresponding neuron weight vectors at several training stages. The "+" markers represent the six classes of training samples and the 'o' markers represent the competitive layer neurons. As the training progresses, the neuron weight vectors gradually start to match the topology of

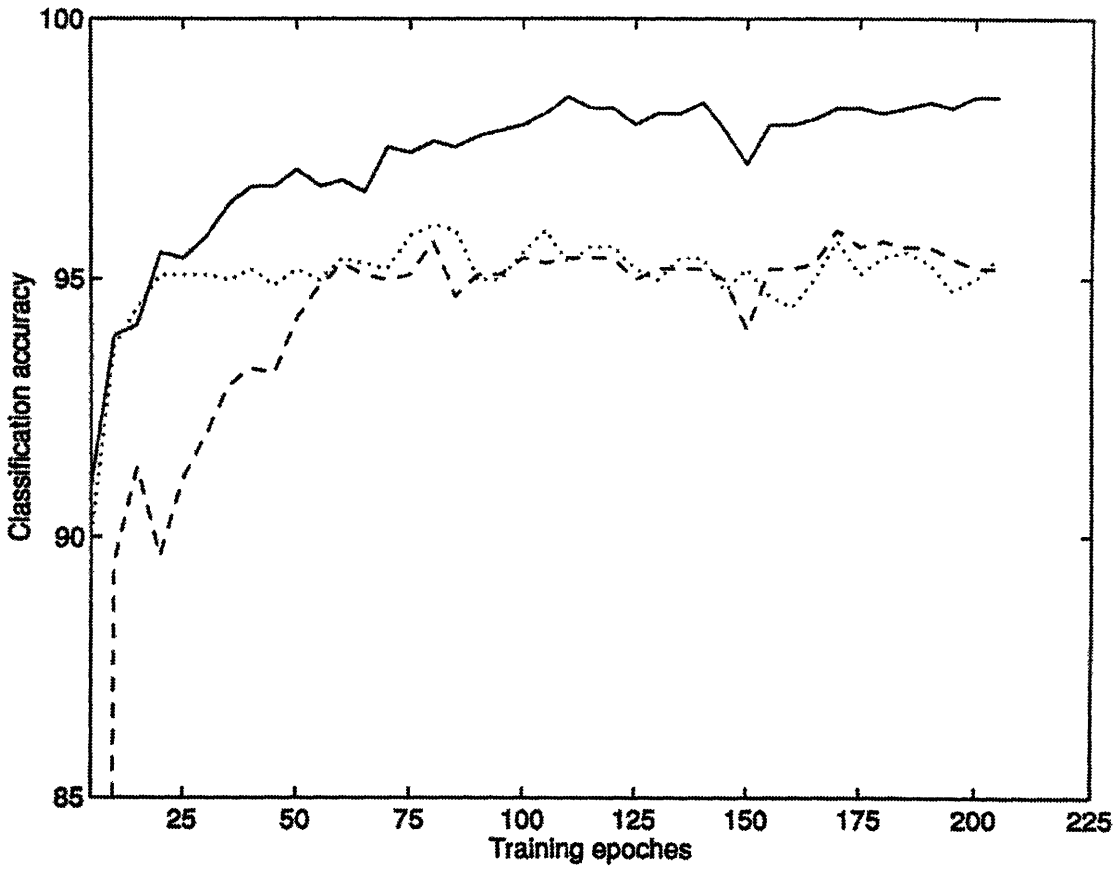
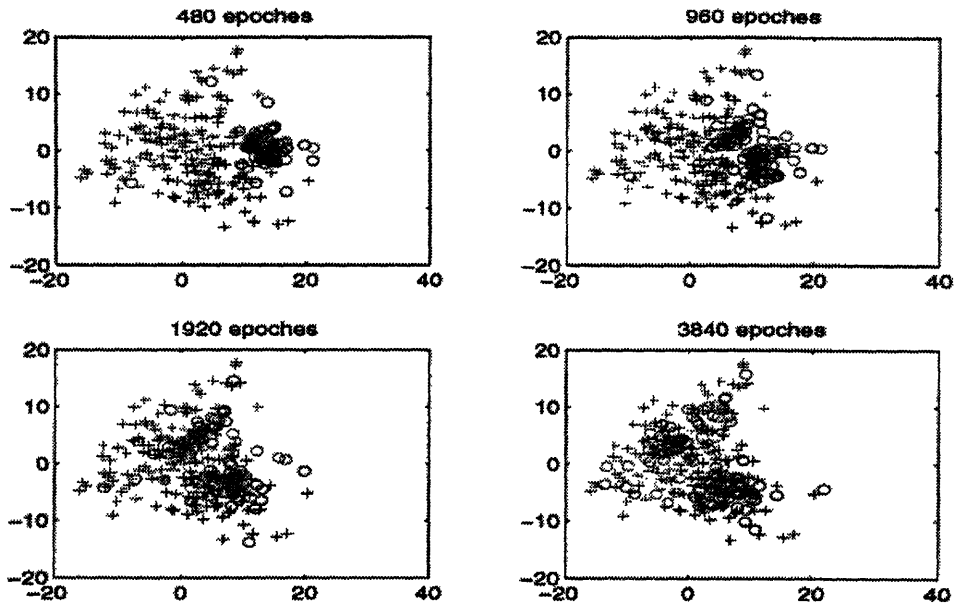


Figure 7.5: Comparison of the three training methods. The dashed line represents the classification accuracy progression with training epochs using the traditional training method and the mean value initial condition. The dotted line is for the traditional training method with the statistical initial condition. The solid line is the result of the new parallel training method. The epoch numbers on the horizontal axis for the two traditional training methods should be multiplied by 120.

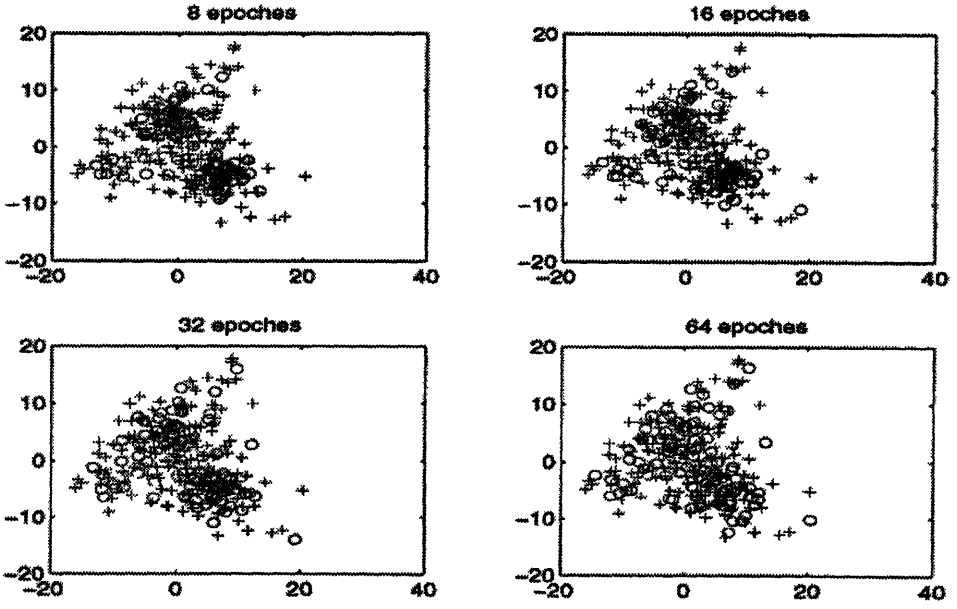
the feature vector space. Comparing Figure 7.6 (a) and (b), we see that the new method apparently maps the density of the feature vector space much better and more quickly.

7.4 Conclusions

Among the three individual feature types, the granulometry vector contains more information than conventional shape descriptors and is much more insensitive to occlu-



(a)Training using traditional method.



(b)Parallel training with statistical initial condition.

Figure 7.6: Comparison of the traditional and the new parallel training methods using scatter plots of the first two feature dimensions and their corresponding neuron weights vector at several training stages. The "+" markers represent the six classes of training samples and the 'o' markers represent the hidden layer neurons.

sion, body motion, and projection direction variation. By directly using the granulometry feature curves, we preserve all the textural and shape information in the vector.

The experimental results demonstrate that the combined feature vector is better than any single feature type, which again validates the multi-view analysis approach. Together with the improved LVQ classifier, 95% classification accuracy is achieved.

Chapter 8

Conclusions and future work

The goal of this thesis is to construct an efficient texture classification system by addressing the three main system components: texture transformation, feature extraction and selection, and classifier design. My main focus has been on developing algorithms to extract the maximum texture information available. In Section 8.1, I summarize what has been discovered and established.

A successful research project will not only conclude many leads, but also open many new doors for future work. In the second section of this chapter, I point out some things I consider worthy of further exploration.

8.1 Summary of major contributions

I have developed new algorithms for each of the three major components of the texture classification system depicted in Figure 1.1.

Texture transformation

I first give an extensive review of existing texture analysis methods. Based on my understanding of these methods, I conduct a unique theoretical study of the interrelations of 11 types of statistical texture analysis approaches and unify them into three major categories: the spatial domain method, the micro-structural method, and the frequency multichannel method. Within each category, I identify a method that extracts texture information nearly a superset of other methods in that category. These are the co-occurrence method, the run-length method, and the Fourier spectrum method.

Feature extraction and selection

After choosing a transform for texture representation or description, the next step is to make it a more concise description that the classifier can use directly. I have developed a novel approach using the principle component analysis technique directly on the transform matrix to extract dominant texture features. This guarantees that all texture information in the matrix is optimally preserved in a small number of features. The approach is made possible by the introduction of a novel multi-level dominant eigenvector estimation algorithm, which reduces the computational complexity of the standard KLT by several orders of magnitude. The Bhattacharyya distance measure is then used to rank the KLT extracted features according to their discriminatory power.

To fully utilize the power of a wavelet packet transform, I use the feature selection algorithm described above to combine and select frequency-channel features that give improved classification performance. The Fourier transform is then considered as one of the highest level of multi-channel decomposition, and the MDEE algorithm is used to extract texture features directly from the Fourier spectrum matrix. For the first time, I demonstrate that the Fourier transform spectrum features generate better results than the more complicated wavelet method.

I then compare the new dominant spectrum features (DSM) with the traditional power spectrum method, and show that using appropriate feature extraction algorithms the discrimination power of the Fourier transform features can be significantly improved.

The novel dominant run-length method (DRM) also demonstrates that, by using the new MDEE algorithm and the Bhattacharyya distance measure for texture feature extraction from the run-length matrices directly, much of the run-length texture information is preserved. Perfect classification is achieved on the eight Brodatz textures using DRM, compared to the upper 80% accuracy of the conventional run-length features traditionally considered least efficient. Based on the observation that most texture information is contained in the first few columns of the run-length matrix, especially the first column of the run-length matrix, I develop a new, fast, parallel run-length matrix computation scheme.

Next, I extend the multi-level dominant eigenfeature analysis approach one level further to a multi-view texture analysis approach. Using the same MDEE algorithm, I com-

bine the feature vectors extracted from several different transform matrices—including the Fourier transform matrix, the run-length matrices, and the co-occurrence matrices—to form a more complete description of texture images. Although similar to many previous studies using combined texture feature vectors, this approach is a logical extension of the MDEE algorithm. A 99.6% classification accuracy on the 16 Vistex images is achieved with the combined feature vector.

Finally, the multi-view texture analysis approach is applied to an object recognition application. For object recognition, the viewing points do not have to be from various texture transformations, they can be any descriptions of the object. The multi-view analysis method can then be used to combine the description features to form a more complete representation of the object. Again, I use the name "multi-view" not to present a significantly new algorithm but to demonstrate another logical extension of the MDEE analysis strategy.

A successful application of this approach to underwater plankton image recognition is then presented. I integrate such popular shape descriptors as moment invariants and Fourier boundary descriptors with the granulometric texture features to compose a very effective feature vector for the plankton images.

Classifier design.

In most texture classification experiments, a simple statistical Gaussian classifier is used. The plankton object recognition experiments use a Learning Vector Quantization (LVQ) neural-net classifier to achieve superior performance on the highly non-uniform plankton database. By introducing a new parallel LVQ learning scheme and a statistical initial condition, the speed of network training is dramatically increased. Experiments show a 95% classification accuracy on six plankton taxa taken from nearly 2,000 images. This result is comparable with what a trained biologist can accomplish by traditional manual techniques, making possible for the first time a fully automated, at-sea approach to real-time mapping of plankton populations.

8.2 Future research directions

Refine MDEE

In my thesis experiments I use a simple uniform grouping of the large feature vector and select the same number of large eigenvalue features in each group. In fact, there are several ways to improve the grouping process.

We can perform non-uniform grouping to keep high-SNR vector sections in one larger sub-vector, while breaking the low-SNR vector section into smaller pieces. In most situations the low-SNR section of the texture matrix is much longer in length, for example, the high-frequency section of the spectrum matrix.

When deciding how many dominant eigenvalues in each group to keep for the next level computation we can preset certain energy thresholds—for example, keeping 95% of the energy in important groups and 60% in noisier groups—then use those energy thresholds to decide the number of eigenvalues to keep in each group. We can also compute the energy in the cross-covariance matrices (discarded in the first step computation). If it is above a certain level, we can reset the number of discarded eigenvalues to a smaller number.

I did not implement any of the above techniques because the current algorithm has been working very well for our applications so far, but these modifications may benefit other applications. The MDEE can certainly be applied to several other pattern recognition research areas, such as face recognition, optical character recognition, and medical image analysis.

Transform texture segmentation

Although I showed that the Fourier transform features give better texture classification performance than the wavelet features, wavelets may be more appropriate for texture image segmentation. Texture segmentation is usually much more involved than classification, since boundaries between two classes of textures are difficult to characterize. A major concern for segmentation is the size of the spatial window over which the texture features are extracted. For classification, the choice is mainly a trade-off between compu-

tation and performance. In the context of segmentation there is also the constraint of boundary localization, since the ambiguity of the boundary location increases with window size.

We may also use different window sizes for different frequency channels. Figure 8.1 shows the wavelet space-frequency tiling for texture segmentation. For low frequency channels, a large window size is needed to capture the distinct frequency signature of difference texture classes; for high frequency channels, a small neighborhood window is sufficient. In comparison, Figure 8.2 shows the space-frequency tiling of a regular block processing method using the Fourier transform. It is similar to the texture classification problem, where each sample image size is fixed. The uniform window size for all frequency channels introduces more segmentation boundary ambiguities.

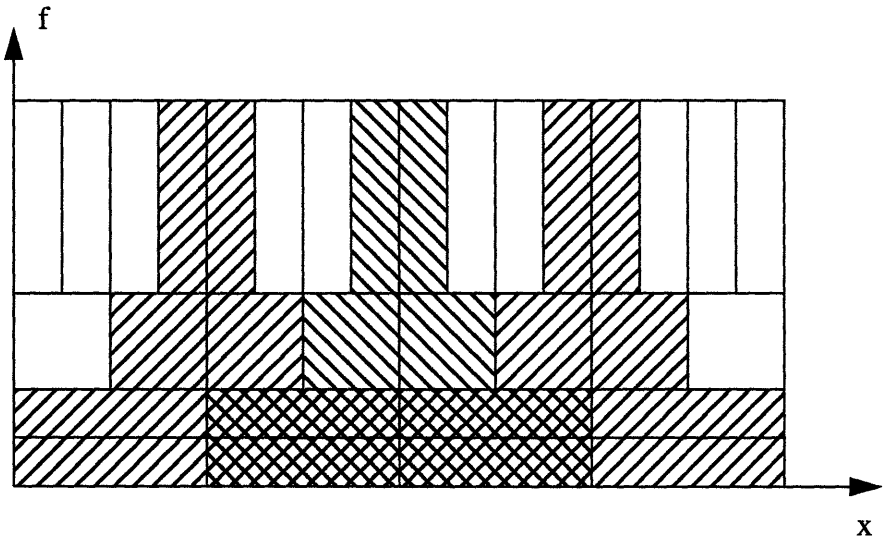


Figure 8.1: Wavelet transform segmentation.

A drawback for wavelet space-frequency tiling is its emphasis on lower frequency channels, even though the most significant information in many textured images often appears in the middle-frequency channels. A more efficient technique, better suited to textures with different energy distributions among frequency channels, is the wavelet packet transform. As demonstrated in Figure 8.3, the channel selection and window size selection are considered simultaneously.

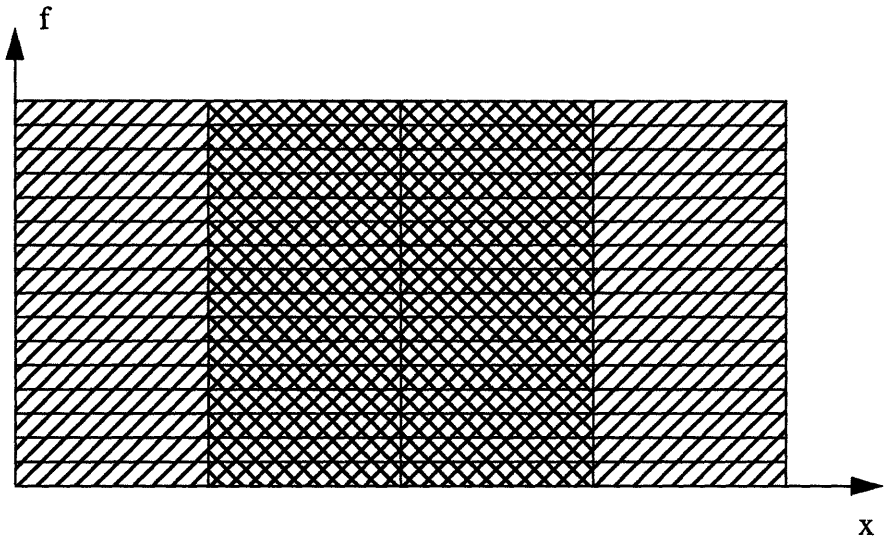


Figure 8.2: Fourier transform segmentation.

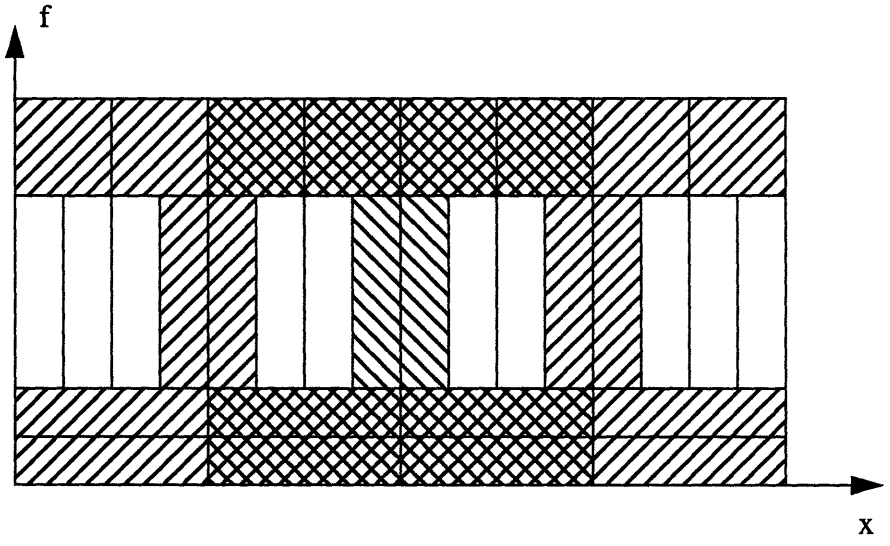


Figure 8.3: Wavelet packet transform segmentation.

More features and better classifiers for plankton research

With the VPR and the pattern classification system developed in this thesis, the real-time automatic sorting of plankton into taxonomic categories becomes possible for the first time. This will allow rapid acquisition of size-dependent taxonomic data over a range

of scales in a dynamic oceanographic environment and provide new insights into the biological and physical processes controlling plankton populations [28].

To achieve this goal, though, much work remains. First, biologists are interested in classifying many more plankton species than are tested in this thesis, many of which are quite similar. To maintain a high degree of classification accuracy, we must develop more distinct pattern features. A hierarchical classification system may also be necessary to classify major species and sub-species in several steps. We also intend to integrate the classification algorithms with the current VPR processing system to carry out real-time taxonomic identification at sea.

Appendix A

Power spectrum features

Let $P(u,v)$ be the power spectrum and let

$$p(u,v) = \frac{P(u,v)}{\sum_{u,v \neq 0} P(u,v)},$$

be the normalized power spectrum. The 20 frequency domain features are defined as

1) Energy in the major peak:

$$f_1 = p(u_1, v_1) \times 100,$$

where (u_1, v_1) is the position of the maximum peak of $P(u,v)$ in the frequency domain.

2) Laplacian of the major peak:

$$f_2 = \nabla^2 P(u_1, v_1) = P(u_1 + 1, v_1) + P(u_1 - 1, v_1) + P(u_1, v_1 + 1) -$$

$$P(u_1, v_1 - 1) - 4P(u_1, v_1),$$

3) Spread of the major peak.

f_3 = number of adjacent neighbors of (u_1,v_1)

$$\text{with} \quad P(u,v) \geq \frac{P(u_1, v_1)}{2},$$

4) Squared major-peak frequency:

$$f_4 = u_1^2 + v_1^2,$$

5) Isotropy of the power spectrum:

$$f_5 = \frac{|\sigma_u - \sigma_v|}{[(\sigma_u + \sigma_v)^2 - 4\sigma_{uv}^2]^{1/2}},$$

where

$$\sigma_u = \sum_u \sum_v u^2 p(u, v),$$

$$\sigma_v = \sum_u \sum_v v^2 p(u, v),$$

$$\sigma_{uv} = \sum_u \sum_v uv p(u, v).$$

6) Major peak horizontal frequency:

$$f_6 = u_1,$$

7) Major peak vertical frequency:

$$f_7 = v_1,$$

8) Secondary peak horizontal frequency:

$$f_8 = u_2,$$

9) Secondary peak vertical frequency:

$$f_9 = v_2,$$

10) Squared distance between major and secondary peak:

$$f_{20} = (u_1 - u_2)^2 + (v_1 - v_2)^2 .$$

where (u_2,v_2) is the position of the secondary maximum peak of $P(u,v)$ in the frequency domain.

11) Moment of inertia in quadrant I:

$$f_{11} = \sum_{u>0} \sum_{v>0} (u^2 + v^2)^{1/2} p(u, v),$$

12) Moment of inertia in quadrant II:

$$f_{12} = \sum_{u<0} \sum_{v>0} (u^2 + v^2)^{1/2} p(u, v),$$

13) Moment ratio:

$$f_{13} = \frac{f_{12}}{f_{11}},$$

14) Percentage of energy in quadrant I:

$$f_{14} = \sum_{u>0} \sum_{v>0} p(u, v),$$

15) Percentage of energy in quadrant II:

$$f_{15} = \sum_{u<0} \sum_{v>0} p(u, v),$$

16) Ratio of nonzero components in quadrant I and II:

$$f_{16} = \frac{n_1}{n_2},$$

where n_i is the number of nonzero frequency components in quadrant i.

17) Relative entropy, *R1*:

$$f_{17} = \left[- \sum_{u, v \in R_1} p_1(u, v) \log p_1(u, v) \right] / \log K_1,$$

where

$$p_i = P(u, v) / \sum_{u, v \in R_i} P(u, v), i = 1, 2, 3, 4,$$

K_i is the number of distinct frequencies in R_i,

$$R_i = \left\{ u, v \left| \left[\frac{(i-1)}{4} u_m < |u| < \frac{i}{4} u_m \right] \right. \right.$$

$$\left. \left. and \left[\frac{(i-1)}{4} v_m < |v| < \frac{i}{4} v_m \right] \right\} \right.,$$

and u_m, v_m are maximum frequency components for the local spectrum.

18) Relative entropy, *R2*:

$$f_{18} = \left[- \sum_{u, v \in R_2} p_2(u, v) \log p_2(u, v) \right] / \log K_2,$$

19) Relative entropy, *R3*:

$$f_{19} = \left[- \sum_{u, v \in R_3} p_3(u, v) \log p_3(u, v) \right] / \log K_3 ,$$

20) Relative entropy, *R4*:

$$f_{20} = \left[- \sum_{u, v \in R_4} p_4(u, v) \log p_4(u, v) \right] / \log K_4 .$$

Bibliography

[1] K. Abend, T. J. Harley, and L. N. Kanal, "Classification of binary patterns," *IEEE Trans. on Information Theory*, vol. 11, pp. 538-544, 1965.

[2] F. Ade, "Characterization of textures by `eigenfilters'," *Signal Processing*, vol. 5, pp. 451-457, 1983.

[3] R. Bajcsy, "Computer description of textured surfaces," in *Proc. 3rd Int. Joint Conf. Artificial Intelligence*, pp. 572-579, Aug. 1973.

[4] M. Barnsley, *Fractals Everywhere*, New York, Academic Press, 1988.

[5] J. R. Bergen and E. H. Adelson, "Early vision and texture perception," *Nature*, vol. 333, pp. 363-364. 1988.

[6] J. Besag, "Spatial interaction and the statistical analysis of lattice systems," *Journal of the Royal Statistical Society*, Series B, 6, pp. 192-236, 1974.

[7] J. Besag, "On the statistical analysis of dirty pictures," *Journal of the Royal Statistical Society*, Series B, 48, pp. 259-302, 1986.

[8] C. Bouman and B. Liu, "Multiple resolution segmentation of textured images," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 13, pp. 99-113, Feb. 1991.

[9] P. Brodatz, *Textures: A Photographic Album for Artists and Designers*, Dover, New York, 1966.

[10] B. A. Burns, "SAR image statistics related to atmospheric drag over sea ice," *IEEE Trans. Geosci. Remote Sensing*, vol. GE-28, no. 2, pp. 158-165, March 1990.

[11] L. Carlucci, "A formal system for texture languages," *Pattern Recognition*, vol. 4, pp. 53-72, 1972.

[12] T. Chang and C.-C. J. Kuo, "Texture analysis and classification with tree-structured wavelet transform," *IEEE Trans. Image Processing*, vol. 2, pp 429-441, Oct. 1993.

[13] T. Chang and C.-C. J. Kuo, "Tree-structured wavelet transform for textured image segmentation," *Advanced Signal Processing Algorithms, Architectures, and Implementations II*, pp. 394-405, San Diego, CA. July 1992.

[14] B. B. Chaudhuri and N. Sarkar, "Texture segmentation using fractal dimension," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 17, no. 1, pp. 72-77, 1995.

[15] P. C. Chen and T. Pavlidis, "Segmentation by texture using correlation," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 5, pp. 64-69, Jan. 1983.

[16] A. Chu, C. M. Sehgal and J. F. Greenleaf, "Use of gray value distribution of run lengths for texture analysis," *Pattern Recognition Letters*, vol. 11, pp 415-420. June 1990.

[17] J. M. Coggins and A. K. Jain, "A spatial filtering approach to texture analysis," *Pattern Recognition letters*, vol. 3, pp. 195-203, 1985.

[18] R. R. Coifman and M. V. Wickerhauser, "Entropy-based algorithms for best basis selection," *IEEE Trans. Inform. Theory*, vol. 38, pp. 713-719, Mar. 1992.

[19] R. W. Conners and C. A. Harlow, "A Theoretical Comparison of Texture Algorithms," *IEEE Tran. on Pattern Analysis and Machine Intelligence*, vol. PAMI-2, No. 3, pp. 204-222, May 1980.

[20] G. R. Cross and A. K. Jain, "Markov random field texture models," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 5, pp. 25-39, Jan. 1983.

[21] S. R. Curtis, A. V. Oppenheim and J. S. Lim, "Signal reconstruction from Fourier transform sign information," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 33, pp. 643-657, June 1985.

[22] E. M. Darling and R. D. Joseph, "Pattern recognition from satellite altitudes," *IEEE Trans. Syst., Man, Cybern.*, vol. 4, pp. 38-47, 1968.

[23] B. V. Dasarathy and E. B. Holder, "Image characterizations based on joint gray-level run-length distributions," *Pattern Recognition Letters*, vol. 12, pp. 497-502, 1991.

[24] I. Daubechies, "Orthonormal bases of compactly supported wavelets," *Comm. Pure Appl. Math.*, vol 41, pp. 909-996, 1988.

[25] I. Daubechies, *Ten Lectures On Wavelets*, SIAM 1992.

[26] C. S. Davis, S. M. Gallager, N. S. Berman, L. R. Haury and J. R. Strickler, "The Video Plankton Recorder (VPR): design and initial results," *Arch. Hydrobiol. Beith.*, vol. 36, pp 67-81, 1992.

[27] C. S. Davis, S. M. Gallager, M. Marra, and W. K. Stewart, "Rapid Visualization of Plankton Abundance and Taxonomic Composition Using the Video Plankton Recorder," *Deep-Sea Research* (in press).

[28] C.S. Davis, S. M. Gallager and A. R. Solow, "Microaggregations of oceanic plankton observed by towed video microscopy," *Science*, May 1992.

[29] L. S. Davis, S. A. Johns and J. K. Aggarwal, "Texture analysis using generalized co-occurrence matrices," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 1, pp. 251-259, July 1979.

[30] H. Derin and H. Elliott, "Modeling and segmentation of noisy and textured images using Gibbs random fields," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 9, pp. 39-55, 1987.

[31] E. R. Dougherty, E. J. Kraus, and J. B. Pelz, "Image segmentation by local morphological granulometries," Proc. of IGARSS'89, vol. 3, pp. 1120-1223, Vancouver, BC, July 1989.

[32] E. R. Dougherty, J. B. Pelz, F. Sand, and A. lent, "Morphological image segmentation by local granulometric size distributions," *Journal of Electronic Imaging*, vol. 1(1), Jan. 1992.

[33] R. C. Dubes and A. K. Jain, "Random field models in image analysis," *J. of App. Statistics*, Vol. 16, No. 2, pp 131-164, 1989.

[34] D. Dubuc, S. W. Zucker, C. Tricott, J. F. Quiniou and D. Wehbi, "Evaluating the fractal dimension of surfaces," *Proc. R. Soc. London A*, vol. 425, pp. 113-127, 1989.

[35] D. Dunn, W. E. Higgins and J. Wakeley, "Texture segmentation using 2-D Gabor elementary functions," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 16, pp. 130-149, Feb. 1994.

[36] J. O. Eklundh, "On the use of Fourier phase features for texture discrimination," *Comput. Graphics and Image Processing*, vol. 9, pp. 199-201, 1979.

[37] D. T. Eppler and L. D. Farmer, "Texture analysis of radiometric signatures of new sea ice forming in Arctic leads," *IEEE Trans. Geosci. Remote Sensing*, vol. GE-29, no. 2, pp. 233-241, March 1991.

[38] J. Feder, *Fractals*, New York, Plenum Press, 1988.

[39] I. Fogel and D. Sagi, "Gabor filters as texture discriminator," *Biol. Cybern.*, vol. 61, pp. 103-113, 1989.

[40] K. S. Fu, *Syntactic Pattern Recognition and Applications*, Prentice-Hall, New Jersey, 1982.

[41] K. Fukunaga, *Introduction to statistical pattern recognition.* New York: Academic Press, 1972.

[42] M. M. Galloway, "Texture analysis using gray level run lengths," *Comput. Graphics Image Processing*, vol. 4, pp. 172-179, June 1975.

[43] S. German and D. German, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images, *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 6, pp. 721-741, 1984.

[44] J. A. Goff, W. K. Stewart, H. Singh, and X. Tang, "Quantitative analysis of ice draft II: application of stochastic modeling to intersecting topographic profiles," *J. Geophys. Res.*, vol. 100, No. C4, pp. 7005-7017, April, 1995.

[45] R.C. Gonzalez and P. Wintz, *Digital Image Processing*. Addison-Wesley, 1987.

[46] L. V. Gool, P. Dewaele and A. Oosterlinck, "Texture analysis anno 1983," *Computer Vision, Graphics, and Image Processing*, vol. 29, pp. 336-357, 1985.

[47] R. M. Haralick, "Statistical and structural approaches to texture," *Proceedings of the IEEE*, vol. 67, no. 5, pp 786-804, May 1979.

[48] R. M. Haralick, K. S. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-3, no. 6, pp. 610-621, 1973.

[49] J. K. Hawkins, "Textural Properties for Pattern Recognition," In *Picture Processing and Psychopictorics*, B. Lipkin and A. Rosenfeld (editors), Academic Press, New York, 1969.

[50] M. H. Hayes, J. S. Lim and A. V. Oppenheim, "Signal reconstruction from phase or magnitude," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 28, pp. 672-680, Dec. 1980.

[51] M. B. Henke-Reed and S. N. C. Cheng, "Cloth texture classification using the wavelet transform," *Journal of Imaging Science and Technology*, vol. 37, pp. 610, Dec. 1993.

[52] Q. A. Holmes, D. R. Nuesch and R. A. Shuchman, "Texture analysis and real-time classification of sea-ice types using digital SAR data," *IEEE Trans. Geosci. Remote Sensing*, vol. GE-22, no. 2, pp. 113-120, March 1984.

[53] M.K. Hu, "Visual pattern recognition by moment invariants," *IRE Trans. Information Theory*, vol. 1T-8, pp. 179-187, Feb. 1962.

[54] A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using gabor filters," in *Proc. of the IEEE Inter. Conf. on Sys. Man and Cyber.*, pp. 14-19, Los Angeles, CA, 1990.

[55] M. E. Jernigan and F. D'Astous, "Entropy-based texture analysis in the spatial frequency domain," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 6 pp. 237-243, March 1984.

[56] B. Julesz, "Visual pattern discrimination," *IRE Trans. Information Theory*, vol. 8, no. 2, pp. 84-92, 1962.

[57] B. Julesz and J. R. Bergen, "Textons, the fundamental elements in preattentive vision and perception of textures," *Bell Syst. Tech. J.*, pp. 1619-1645, 1981.

[58] H. Kaizer, "A quantification of textures on aerial photographs," Tech. Note 121, AD 69484, Boston Univ. Res. Labs, 1955.

[59] R. L. Kashyap, R. Chellappa and A. Khotanzad, "Texture classification using features derived from random field models," *Patt. Recogn. Lett.*, vol. 1, pp. 43-50, 1982.

[60] H. Kauppinen, T. Seppanen and M. Pietikainen, "An experimental comparison of autoregressive and Fourier-Based descriptors in 2D shape classification," *IEEE Trans. on Pattern Analysis and Machine Intelligence,* vol. 17, no. 2, pp. 201-207, Feb. 1995.

[61] J. Keller, R. Crownover and S. Chen, "Texture description and segmentation through fractal geometry," *Computer Vision, Graphics, and Image Processing*, vol. 45, pp. 150-160, 1989.

[62] J. Key and A. S. McLaren, "Fractal nature of the sea ice draft profile," *Geophys. Res. Letters*, vol. 18, no. 8, pp. 1437-1440, Aug. 1991.

[63] T. Kohonen, *Self-Organization and Associative Memory*, 2nd Edition, Berlin: Springer- Verlag, 1987.

[64] T. Kohonen, "The self-organizing map," *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464-1480, Sept. 1990.

[65] A. Laine and J. Fan, "Texture classification by wavelet packet signatures," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15 pp. 1186-1191, Nov. 1993.

[66] K. I. Laws, *Textured Image Segmentation*, Ph.D. dissertation, Image Processing Inst., Univ. of Southern California, 1980.

[67] S. Liu and M. E. Jernigan, "Texture analysis and discrimination in additive noise," *Computer Vision, Graphics, and Image Processing*, vol. 49, pp. 52-67, 1990.

[68] S. Y. Lu and K. S. Fu, "A syntactic approach to texture analysis," *Comput. Graph. Image Processing*, vol. 7, no. 3, pp. 303-330, 1978.

[69] J. D. Lyden, B. A. Burns and A. L. Maffett, "Characterization of sea ice types using synthetic aperture radar," *IEEE Trans. Geosci. Remote Sensing*, vol. GE-22, no. 5, pp. 431-439, Sept. 1984.

[70] J. Malik and P. Perona, "Preattentive texture discrimination with early vision mechanisms," *J. Opt. Soc. Am.* A, Vol. 7, No. 5, pp 923-932, 1990.

[71] S.G. Mallat, "A theory for multiresolution signal decomposition: the wavelet representation," *IEEE Trans. Pattern Analysis Machine Intelligence*, vol. 11, pp. 674-693, July 1989.

[72] S.G. Mallat, "Multifrequency channel decompositions of images and wavelet models," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pp. 2091-2110, Dec. 1989.

[73] B. B. Mandelbrot, *The Fractal Geometry of Nature*, New York, W. H. Freeman, 1983.

[74] P. Maragos, "Pattern Spectrum and Multiscale Shape Representation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 11, no. 7, pp. 701-716, July 1989.

[75] D. Marr, *Vision*, San Francisco, CA: Freeman, 1982.

[76] G. Matheron, *Random Sets and Integral Geometry*, John Wiley and Sons, New York, 1975.

[77] Y. Meyer, *Wavelets: Algorithms and Applications*, SIAM, Philadelphia, PA, 1993, Translated and revised by R. D. Ryan.

[78] J. A. Nystuen and F. W. Garcia, Jr., "Sea ice classification using SAR backscatter statistics," *IEEE Trans. Geosci. Remote Sensing*, vol. GE-30, no. 3, pp. 502-509, May 1992.

[79] P. P. Ohanian and R. C. Dubes, "Performance evaluation for four classes of textural features," *Pattern Recognition*, vol. 25, no. 8, pp. 819-833, 1992.

[80] A. V. Oppenheim, M. H. Hayes and J. S. Lim, "Iterative procedures for signal reconstruction from Fourier transform phase," *Optical Engineering*, Vol. 21, no. 1, pp. 122-127. Jan. 1982.

[81] A. V. Oppenheim, J. S. Lim and S. R. Curtis, "Signal synthesis and reconstruction from partial Fourier-domain information," *J. Opt. Soc. Am.*, vol. 73, no. 11, pp. 1413-1420, 1983.

[82] R. W. Owen, "Microscale and finescale variations of small plankton in coastal and pelagic environments," *Journal of Marine Research*, vol. 47, pp 197-240, 1989.

[83] G. A. Paffenhofer, T. B. Stewart, M. J. Youngbluth and T. G. Bailey, "High-resolution vertical profiles of pelagic tunicates," *Journal of Plankton Research*, vol. 13, no. 5, pp 971-981, 1991.

[84] N. G. Pace and C. M. Dyer, "Machine classification of sedimentary sea bottoms," *IEEE Trans. Geosci. Remote Sensing*, vol. GE-17, pp. 52-56, 1979.

[85] N. G. Pace and H. Gao, "Swathe seabed classification," *IEEE J. Oceanic Eng.*, vol. 13, no. 2, pp. 83-90, 1988.

[86] A. P. Pentland, "Fractal-based description of natural scenes," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 6, pp. 661-674, Nov. 1984.

[87] E. Persoon and K. S. Fu, "Shape discrimination using Fourier descriptors," *IEEE Trans. on Sys., Man, and Cyber.*, vol. 7, no. 3, pp. 170-179, Mar. 1977.

[88] R. K. Raney, "Probing ice sheets with imaging radar," *Science*, vol. 262, no. 3, pp 1521-1522, Dec. 1993.

[89] T. B. Reed, IV and D. M. Hussong, "Digital image processing techniques for enhancement and classification of SeaMARC II sidescan sonar imagery," *J. Geophys. Res.*, vol. 94, pp. 7469-7490, 1989.

[90] A.P. Reeves, R. J. Prokop, S. E. Andrews, and F. P. Kuhl, "Three-dimensional shape analysis using moments and Fourier descriptors," *IEEE Trans. on Pattern Analysis and Mach. Intell.*, vol. 10, no. 6, pp. 937-943, Nov. 1988.

[91] T.H. Reiss, "The revised fundamental theorem of moment invariants," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 13, no. 8, pp. 830-834, Aug. 1991.

[92] T. Reti and I. Czinege, "Shape characterization of particles via generalized Fourier analysis," *Journal of Microscopy*, Vol. 156, Pt 1, pp. 15-32, Oct. 1989.

[93] Z. Reut, N. G. Pace and M. J. P. Heaton, "Computer classification of sea beds by sonar," *Nature*, vol. 314, pp. 426-428, 1985.

[94] O. Rioul and M. Vetterli, "Wavelets and signal processing," *IEEE Signal Processing Mag.*, vol. 8, pp. 14-37, Oct. 1991.

[95] A. Rosenfeld and E. Troy, "Visual texture analysis," Tech. Rep. 70-116, University of Maryland, College Park, MD, June 1970.

[96] D. A. Rothrock and A. S. Thorndike, "Geometric properties of the underside of sea ice," *J. Geophys. Res.*, vol. 85, no. c7, pp. 3955-3963, July 1980.

[97] M. Schmitt and J. Mattioli, "Shape Recognition Combining Mathematical Morphology and Neural Networks," *SPIE: Application of Artificial Neural Network*, Orlando, Florida, April 1991.

[98] J. Serra, *Image Analysis and Mathematical Morphology*, Academic Press, London, 1982.

[99] J. Sklansky, "Image Segmentation and Feature Extraction," *IEEE Transactions on Systems, Man and Cybernetics*, SMC-8, pp. 237-247, 1978.

[100] W. K. Stewart, M. Jiang, M. Marra, "A neural network approach to classification of sidescan sonar imagery from a midocean ridge area," IEEE J. Oceanic Eng., Vol. 19, no. 2, pp. 214-224, 1994.

[101] H. Tamura, S. Mori, and Y. Yamawaki, "Textural Features Corresponding to Visual Perception," *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-8, pp. 460-473, 1978.

[102] X. Tang and W. K. Stewart, "Texture classification using principle-component analysis techniques," *Proc. Image and Signal Processing for Remote Sensing*, SPIE vol. 2315, pp. 22-35, Rome, Italy, 1994.

[103] X. Tang and W. K. Stewart, "Texture classification using wavelet packet and Fourier transforms," *Proc. MTS-IEEE Oceans'95*, vol. 1, pp. 387-396, San Diago, CA, Oct. 1995.

[104] C.H. Teh and R. T. Chin, "On image analysis by the methods of moments," *IEEE Trans. on Pattern Analysis and Machine Intelligence* vol. 10, no. 4, pp. 496-513, July. 1991.

[105] C. W. Therrien, *Decision Estimation and Classification, An Introduction to Pattern Recognition and Related Topics*, Wiley, New York, 1989.

[106] F. Tomita and S. Tsuji, *Computer Analysis of Visual Textures*, Kluwer Academic Publishers, 1990.

[107] M. Tuceryan and A. K. Jain, "Texture analysis," *The handbook of Pattern Recognition and Computer Vision*, by C. H. Chen, L. F. Pau, and P. S. P. Wang (eds), World Scientific Publishing Co., 1992.

[108] F. T. Ulaby, F. Kouyate, B. Brisco and T. H. L. Williams, "Textural information in SAR images," *IEEE Trans. Geosci. Remote Sensing*, vol. GE-24, no. 2, pp. 235-245, March 1986.

[109] M. Unser, "Texture discrimination using wavelets," in P*roc. of the 1993 IEEE Computer Soc. Conf. on Comput. Vision and Patt. Recognition*, New York, NY, 1993.

[110] M. Unser, "Local linear transforms for texture measurements," *Signal Processing*, Vol. 11, No. 1, pp. 61-79, July 1986.

[111] L. Vincent, "Fast opening functions and morphological granulometries," *Proc. Image Algebra and Morphological Image Processing*, SPIE 2300, pp. 253-267, San Diego, July 1994.

[112] L. Vincent, "Fast Grayscale Granulometry Algorithms," *EURASIP Workshop ISMM'94, Mathematical Morphology and its Applications to Image Processing*, pp. 265-272, France, Sept. 1994.

[113] H. Voorhees and T. Poggio, "Computing texture boundaries from images," *Nature*, vol. 333, pp. 364-367. 1988.

[114] P. Wadhams, "Sea ice thickness distribution in Fram Strait," *Nature*, vol. 305, pp. 108-111, Sept. 1983.

[115] P. Wadhams, "The underside of Arctic sea ice imaged by sidescan sonar," Nature, vol. 333, pp. 161-164, Sept. 1983.

[116] P. Wadhams and T. Davy, "On the spacing and draft distributions for pressure ridge keels," J. Geophys. Res., vol. 91, no. C9, pp. 10697-10708, Sept. 1986.

[117] P. Wadhams, A. S. McLaren and R. Weintraub, "Ice thickness distribution in Davis strait in February from submarine sonar profiles," J. Geophys. Res., vol. 90, no. C1, pp. 1069-1077, Jan. 1985.

[118] A. E. J. Walford and M. E. Jernigan, "The phase spectrum and its contribution to human texture perception," *IEEE 1985 Proc. of the Inter. Conf. on Sys. Man and Cybernetics*, pp. 701-704, 1985.

[119] H. Wechsler, "Texture analysis - a survey," *Signal Processing*, vol. 2, pp. 271-282, 1980.

[120] J. S. Weszka, C. R. Dyer, and A. Rosenfeld, "A Comparative Study of Texture Measures for Terrain Classification," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-6, No. 4, pp. 269-285, 1976.

[121] C. Zahn and R. Z. Roskies, "Fourier descriptors for plane closed curves," *IEEE Trans. on Computers*, vol. 21, no. 3, pp. 269-281, Mar. 1972.

[122] S. Zucker, "Toward a model of texture," *Comput. Graph. Image Processing*, vol. 5, no. 2, pp. 190-202, 1976.