

ROTATION AND GRAY-SCALE INVARIANT TEXTURE ANALYSIS



by

Abdul Jalil

**A dissertation submitted to M.A.J.U. in partial fulfillment of the
requirements for the degree of**

DOCTOR OF PHILOSOPHY

**Department of Electronic Engineering
Faculty of Engineering and Sciences**

**MOHAMMAD ALI JINNAH UNIVERSITY
Islamabad, Pakistan
2006**

Copyright © 2006 by Abdul Jalil.

All rights reserved. No part of the material protected by this copyright notice may be reproduced or utilized in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage and retrieval system, without the permission from the author.

**DEDICATED TO
HOLY PROPHET (P. B. U. H.)
THE GREATEST SOCIAL REFORMER**

&

MY WORTHY MOTHER, SPOUSE AND KIDS

(TEHNIA, RAFIA, MAARIJ, GHANIA AND AHMED)

Abstract

Texture analysis is an extremely active and useful area of research. In texture analysis the invariance to rotation, scale and translation are the most typical requirements. Moreover, gray-scale invariance is another important issue. It arises due to the reason that a texture may be subject to different levels of illumination. The purpose of this study is to investigate some inexpensive approaches that are rotation and gray scale invariant and to large extent translation invariant as well. There are three different types of approaches, which have been addressed in this dissertation.

In the first approach, we have done texture analysis using Radon Transform (RT) based Hidden Markov Model (HMM). We have introduced three different ways to extract feature vectors using RT. All three give rotation invariant features, while the last one gives rotation, as well as, gray scale invariant features. The textures in this case have been taken from Brodatz album. Due to the inherent property of the RT, we are able to capture the directional features of a certain texture having arbitrary orientation. This set of directional features is used for training of an HMM specifically for that particular texture. Once all the HMMs have been trained, the testing is carried out by using any one of these textures at random with arbitrary orientation.

The second approach is somewhat similar to the above one except that the modified or Differential Radon Transform (DRT) has been used instead of the ordinary RT. Hence, we are able to capture the features which are not only rotation but are also gray scale invariant. The reason for the later property is that, unlike the ordinary RT, the DRT is based on the differences between adjacent pixels instead of summing up the pixel values. These features have been used for training of HMMs, one for each texture, and finally testing is carried out. Similar experimentation has been done to extract features using both RT and DRT to give low pass and high pass features. The training and testing process using HMM has been done in a similar manner as above.

The third approach is quite different from the above two approaches. In this approach, some principal direction of a texture is defined. Once this direction is estimated, discrete wavelet transform is applied in that particular direction to extract features. These features are then used for classification by k-nearest neighbor classifier. There are two definitions of principal direction, which have been proposed in the dissertation. In case of

the first definition, Principal Component Analysis (PCA) has been used to estimate this principal direction. In the case of second definition, the direction has been found out by using DRT. This scheme is computationally lighter compared to the previous one. However, the third approach is limited to anisotropic textures only unlike the previous method

Considering the percentage of correct classification as figure of merit, we have carried out the performance evaluation of the above three approaches. The average result has been found to be 95% approximately and the best result has been close to 100%.

List of Publications and Submissions

1. **A. Jalil**, A. Manzar, A. Zahoor, I.M. Qureshi, “Rotation-Invariant Features for Texture Image Classification,” *IEEE International Conference on Engineering of Intelligent System*, pp.42-45, 22-23 April, 2006.
2. **A. Jalil**, I.M. Qureshi, T.A. Cheema and A. Naveed, “Feature Extraction of Hand Written Characters by using Non-linear and Unsupervised Neural Networks,” *Journal of Information Science and Engineering (JISE)*, vol. 21, no. 2, pp. 453-473, 2005.
3. N. Qaiser, M. Hussain, N. Qaiser, A. Hanif, S.M.J. Rizvi and **A. Jalil**, “Fusion of Optimized Moment Based and Gabor Texture Features for better Texture Classification,” *IEEE 8th International Multitopic Conference INMIC 2004*, Lahore, Pakistan, pp. 41-48, 2004.
4. **A. Jalil**, T.A. Cheema, I.M. Qureshi, “A New Rotation-Invariant Texture Analysis Technique using Radon Transform based Hidden Markov Models,” Submitted to IEICE.
5. **A. Jalil**, T.A. Cheema, A. Manzar, I.M. Qureshi, “Rotation and Gray-Scale-Invariant Texture Analysis,” Submitted to IEE Electronic letter
6. **A. Jalil**, T.A. Cheema, A. Manzar, I.M. Qureshi, “Rotation and Gray-Scale-Invariant Texture Analysis using Radon and Differential Radon Transforms based Hidden Markov Models,” *Pattern Analysis and Application* and conditionally accepted.
7. T.A. Cheema, I.M. Qureshi, **A. Jalil**, and A. Naveed, “Artificial Neural Networks for Blur Identification and Restoration of Nonlinearly Degraded Images,” *International Journal of Neural Systems*, vol. 11, no. 5, pp. 455–461, 2001.
8. T.A. Cheema, I.M. Qureshi, **A. Jalil**, A. Naveed, “Blurred Image Restoration of

- Nonlinearly Degraded Images using ANN and Nonlinear ARMA Model,” *Journal of Intelligent Systems*, vol. 11, no. 5, pp. 299–312, 2001.
9. T.A. Cheema, I.M. Qureshi, **A. Jalil**, and A. Naveed, “Blur and Image Restoration of Nonlinearly Degraded Images using Neural Networks based on modified Nonlinear ARMA Model,” Accepted in *Arabian Journal of Science and Engineering (AJSE)*.
 10. T.A. Cheema, I.M. Qureshi, **A. Jalil**, and A. Naveed, “Blur and Image Restoration of Nonlinearly Degraded Images using Neural Networks based on Modified ARMA Model,” *IEEE 8th International Multitopic Conference INMIC 2004*, Lahore, Pakistan, pp.102-107, 2004.
 11. A. Naveed, I. M. Qureshi, **A. Jalil** and T. A. Cheema, “Blind equalization and estimation of channel using artificial neural networks,” *IEEE 8th International Multitopic Conference INMIC 2004*, Lahore, Pakistan, pp.184-190, 2004.
 12. I.M. Qureshi, T.A. Cheema, A. Naveed and **A. Jalil**, “Genetic Algorithms Based Artificial Neural Networks for Blur Identification and Restoration of Degraded Images,” *Pakistan Journal of Information and Technology*, vol. 2, no. 1, pp. 21-24, 2003.
 13. **A. Jalil**, I.M. Qureshi, A. Naveed and T.A. Cheema, “Feature Extraction by using Non-linear and Unsupervised Neural Networks,” *Pakistan Journal of Information and Technology (PJIT)*, vol. 2, no.1, pp. 40-43, 2003.
 14. I.M. Qureshi, A. Naveed, T.A. Cheema and **A. Jalil**, “Artificial Neural Networks for Microstructure Analysis of Rolling Process,” *Pakistan Journal of Information and Technology (PJIT)*, vol. 2, no.1, pp. 65-68, 2003.
 15. I.M. Qureshi and **A. Jalil**, “Object Recognition Using ANN with Backpropagation Algorithm,” *Pakistan Journal of Applied Sciences*, 2(3): 281-287, March 2002.

Acknowledgement

*I stoop to praise and extend my gratification to **Lord of the Lords** Who bestowed upon me the wealth of health, cerebation and deliberation, sacrificing family, talented teachers, co-operative friends who enabled me to contribute to infinite realm of knowledge.*

*Peace and prayer for marvelous human and the torch bearer of wisdom **Muhammad (S.A.W)** Who's objective was enlightenment of the whole world.*

*I offer my special and sincere thanks to my supervisor **Dr. Ijaz Mansoor Qureshi** for his long illuminated guidance, illustrious advice, beneficial suggestions and condescending supervision throughout the research work. I staunchly believe that the completion of my dissertation sprouted up from his special personal interest, sparkling suggestions and gleaming criticism.*

*I express my thanks for scholastic guidance, selfless co-operation beaming advices and benevolent attitude to my co-supervisor **Dr. Tanveer Ahmad Cheema**. I found him eager to help me throughout my research. I am quite aware of this fact that no words can be strong enough to reflect my innermost thank for **Dr. Tanveer Ahmad Cheema**.*

*I am grateful to all my fellows and friends whose encouragement and inspiration propelled me in right direction. I offer my earnest thanks to **Dr. Aqdas Naveed Malik** for his help and guidance during course study which left indelible impacts on my mind and on my work. I am abundantly indebted to my friend **Mr. Anwar Manzar** for his precious support. My brother **Abdul Hamid (Deputy Chief Scientist)** rightly deserves my feelings of thankfulness that has been the beacon of moral support to me.*

I am wishful to appreciate my mother, brothers and sisters who remain desirous for my health and bright future.

***Mr. Ghias Malik** earned my gratefulness for his support in the typing of this manuscript.*

Lastly I thank my wife Qurra-tul-ain (Assistant Professor) who proved to be embodiment of endurance and sacrifice throughout untiring research years, who loves to see me flourishing, progressing and dominating.

(Abdul Jalil)

Table of Contents

Abstract	v
List of Publications and Submissions.....	vii
Acknowledgement.....	ix
List of Figures.....	xiv
List of Tables.....	xvi
Chapter 1	1
Introduction	1
1.1 Problem Statement.....	1
1.2 Contributions of the Dissertation	3
1.3 Organization of the Dissertation	5
Chapter 2	7
Texture Analysis and its Background	7
2.1 Introduction.....	7
2.2 Definition of Texture	7
2.3 Texture Analysis	9
2.4 Texture Feature extraction	10
Chapter 3	18
Computing Tools	18
3.1 Introduction.....	18
3.2 Principal Component Analysis	18
3.3 Radon Transform	20
3.4 Properties of Radon Transform.....	22
3.5 Wavelet Transforms.....	23
3.6 Classifiers	27
3.6.1 The Bayes Classifier	30
3.6.2 Neural Network Classifiers	33
3.6.3 Parametric Classifiers	36
3.6.4 Non-Parametric Classifier	38
3.6.5 Error Estimation.....	42
3.6.6 Dimensionality Reduction in Classifier	43
3.6.7 Choice of a Classifier.....	45
3.7 Hidden Markov Model (HMM).....	46
3.7.1 Assumptions in the Theory of HMMs.....	48
3.7.2 Three Basic Problems of HMMs.....	49

Chapter 452

Texture Analysis using Radon Transform and Hidden Markov Models52

4.1 Introduction..... 52

4.2 Background of Invariant Texture Analysis 52

4.3 Texture Analysis using RT for Case $s = 0$ 56

4.3.1 Feature Extraction using Radon Transform 57

4.3.2 Training using Hidden Markov Model..... 58

4.3.3 Testing and Classification 59

4.3.4 Simulation and Results..... 61

4.4 Texture Analysis using RT for Case $s \neq 0$ 64

4.4.1 Feature Extraction using RT 64

4.4.2 Training of HMM..... 65

4.4.3 Testing and Classification 66

4.4.4 Simulations and Results 66

4.5 Texture Analysis using Variance of RT for Case $s \neq 0$ 67

4.5.1 Feature Extraction using Variance of RT..... 67

4.5.2 Training of HMM..... 68

4.5.3 Testing and Classification 68

4.5.4 Simulations and Results 68

4.6 Summary..... 69

Chapter 571

Texture Analysis using Differential Radon Transform and Hidden Markov Models71

5.1 Introduction..... 71

5.2 Differential Radon Transform..... 72

5.3 Texture Analysis using Differential Radon Transforms and Hidden Markov Model 73

5.3.1 Feature Extraction using Differential Radon Transform..... 73

5.3.2 Training Hidden Markov Model using DRT Features 73

5.3.3 Testing and Classification 74

5.3.4 Simulation and Results..... 75

5.4 Texture analysis using Radon and Differential Radon Transform for the Case $s = 0$ 76

5.4.1 Feature Extraction using RT and DRT..... 76

5.4.2 Training using HMM 76

5.4.3 Testing and Classification 77

5.4.4 Simulations and Results 77

5.5 Texture analysis using Radon and Differential Radon Transform for the Case $s \neq 0$ 78

5.5.1 Feature extraction using RT and DRT 78

5.5.2 Training of HMM..... 79

5.5.3 Testing and Classification 79

5.5.4 Simulations and Results 79

5.6 Summary..... 80

Chapter 682

Texture Analysis using its Principal Direction and Discrete Wavelet Transform.....82

6.1 Introduction..... 82

6.2 Methods using Principal Direction 84

6.3 Proposed Method using DRT..... 87

6.4	Proposed Method using PCA.....	90
6.5	Selection of Dataset for Classification.....	94
6.6	Simulations and Results.....	97
6.7	Summary.....	101
Chapter 7	102
Conclusions	102
7.1	Summary of Results.....	102
7.2	Future Directions.....	104
References	106
Appendix	119

List of Figures

Fig. 1.1	Images form the Vis Tex-database: from top to bottom and left to right: Bark0,Bark4, Bark6, Bark8, Bark9, Brick1, Brick4, Brick5, Fabric0, Fabric4, Fabric7, Fabric9, Fabric11, Fabric13, Fabric16, Fabric17, Fabric18, Food0, Food2, Food5, Food8, Grass1, Sand0, Stone4, Tile1, Tile3, Tile7, Water6, Wood1, Wood2.....	2
Fig. 2.1	Components of Computer Vision System	9
Fig. 3.1	PCA Transformation (a) Eigenvector of an object (b) Object along eigen-axis	19
Fig. 3.2	Radon Transform	21
Fig. 3.3	Block diagram of separable wavelet filter bank in 2-D: (a) The analysis filter bank (b) The synthesis filter bank.....	25
Fig. 3.4	Two level decomposition of Brodatz texture D1 (a) Original image (b) Organization of the detail images within the wavelet transform (c) Decomposed image at level 2	26
Fig. 3.5	The Recognition Process	30
Fig. 3.6	Multilayer perceptron with an input layer (d neurons), a hidden layer (N_h neurons) and an output layer (c neurons).....	34
Fig. 4.1(a)	Anisotropic texture (D53), (b) Anisotropic texture (D53) rotated at 45° , (c) Fourier Transform of (a), (d) Fourier Transform of (b).....	53
Fig. 4.2(a)	Isotropic texture (D101), (b)) Isotropic texture (D101) rotated at 45° , (c) Fourier Transform of (a), (d) Fourier Transform of (b).....	54
Fig. 4.3	Radon Transform	56
Fig. 4.4(a)	Block diagram for the training phase and (b) Block diagram for the testing phase of the proposed model.....	60
Fig. 4.5	Pixel representation on lines	65
Fig. 6.1(a)	Anisotropic (directional) texture (D20) (b), variance of the projections along different orientations, (c) First order derivative of (b), (d) Second order derivative of (b).....	85
Fig. 6.2(a)	Anisotropic (directional) texture (D20) at rotation of 45° (b), variance of the projections along different orientations, (c) First order derivative of (b), (d) Second order derivative of (b).....	86
Fig. 6.3	Eight neighbors of a pixel.....	89

Fig. 6.4	Block Diagram for 4-level Wavelet Decomposition	90
Fig. 6.5	Block Diagram of Proposed Method	92
Fig. 6.6	Principal direction angle for Principal Eigenvector.....	92
Fig. 6.7	Original image of the texture, (b) Wavelet signature of (a), (c) Rotated image of (a) with Principal direction angle, (d) Wavelet signature of (c), (e) Rotation of (a) with angle 25, (f) Wavelet signature of (e), (g) Rotated image of (d) with Principal direction angle	93
Fig. 6.8	The CURET Dataset.....	95
Fig. 6.9	One hundred twelve (112) textures from Brodatz album (D01-D112). First row D01-D08, second row D09-D16, and so on, and 14 th row D105-D112.....	96
Fig. 6.10	Performance of four different methods in the presence of zero mean AWGN different signal to noise ratios	98

List of Tables

Table 4.1 PCC of 1200 test samples using proposed RT based method (case $s = 0$) for different number of feature vectors L and number of hidden states N	63
Table 4.2 PCC of 10 textures from Brodatz Album using method proposed by Chen and Kundu [53], for both Nonstationary and Stationary Transition HMM.	63
Table 4.3 PCC of 25 textures from Brodatz Album using the method proposed by Khouzani and Zadeh [60] for different k values of k -nn Classifier.....	63
Table 4.4 PCC of 1200 (60×20) test samples using proposed RT based method (case $s \neq 0$) for different number of feature vectors L and number of hidden states N	67
Table 4.5 PCC of 500 (25×20) test samples using proposed RT variance based method (case $s \neq 0$) for different number of feature vectors L and number of hidden states N	69
Table 4.6 Comparison of the best results of the proposed methods with some of the methods from the literature	69
Table 5.1 PCC of 1200 (60×20) test samples using proposed DRT based method (case $s=0$) for different number of feature vectors L and number of hidden states N	75
Table 5.2 PCC of 1200 (60×20) test samples using proposed combined RT and DRT based method (case $s = 0$) for different number of feature vectors L and number of hidden states N	78
Table 5.3 PCC of 1200 (60×20) test samples using proposed combined RT and DRT based method (case $s \neq 0$) for different number of feature vectors L and number of hidden states N	80
Table 6.1 Comparison of the three methods using 25 textures.....	99
Table 6.2 Comparison of the three methods using 112 textures.....	100

Chapter 1

Introduction

1.1 Problem Statement

Texture is an intuitive concept that describes properties like smoothness, coarseness and regularity of a region [1]. It is a source of information about the natural scene and adds richness to a design for designers. It is attractive for computer scientists and is an important component in image analysis for solving a wide range of applied recognition, segmentation and synthesis problems. It also provides a key for understanding the basic mechanisms that underlie human visual perception. Texture plays an important role in the composition of natural images. Most of the natural surfaces exhibit texture, therefore, a successful vision system must be able to deal with the texture world.

Texture analysis is an important and a useful area of study in machine vision, and its classification plays an important role in a variety of image processing applications such as robot vision, remote sensing, crop classification, content-based access to image databases, automatic tissue recognition in medical imaging, etc. Some examples of textures are shown in Fig. 1.1. These textures include recording of several kinds of wood, stone, soil, etc. If you are asked to label these pictures and for example, you are shown the picture of another part

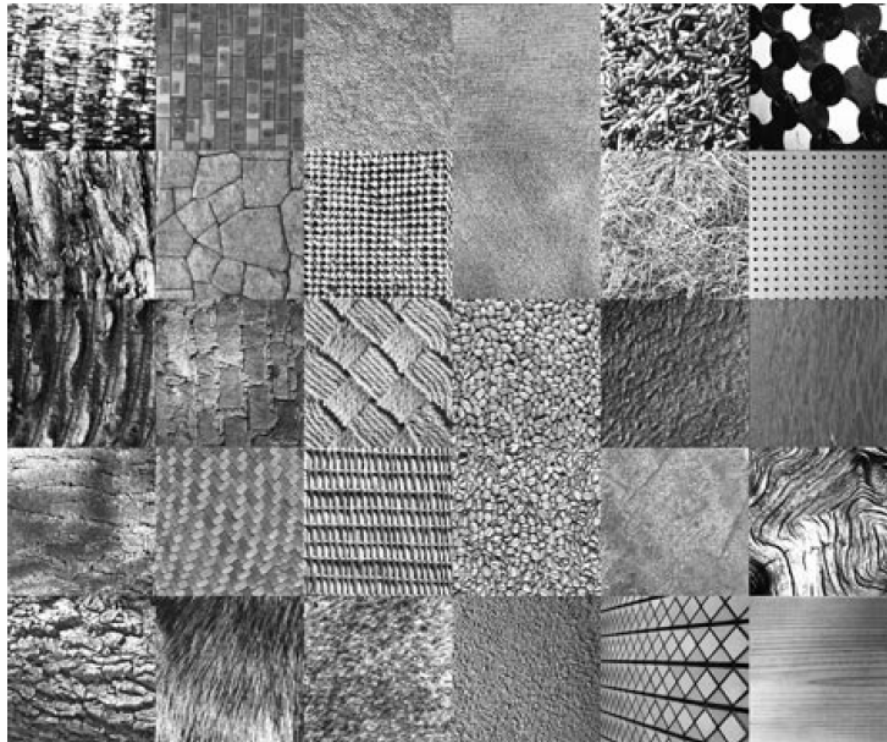


Fig. 1.1 Images from the Vis Tex-database: from top to bottom and left to right: Bark0, Bark4, Bark6, Bark8, Bark9, Brick1, Brick4, Brick5, Fabric0, Fabric4, Fabric7, Fabric9, Fabric11, Fabric13, Fabric16, Fabric17, Fabric18, Food0, Food2, Food5, Food8, Grass1, Sand0, Stone4, Tile1, Tile3, Tile7, Water6, Wood1, Wood2.

of the same tree as Bark0, you will be able to recognize it. If you make some errors initially, you may need some more training to become an expert in discriminating the different textures. However, if someone asks “how you recognized it?”, you will probably use arguments such as: “it looks more striped than the other”, and “this one is more blurry and rough” etc to describe the textures. This example demonstrates that there is no real definition for texture [2][3] and definitely not an obvious quantitative measure to characterize it. Characterizing a real-world view or an image into different texture classes is often a trivial task for the human visual system, but is one of the most challenging problems in the field of computer vision and image processing. Accurate results have been achieved traditionally through various schemes, but only working under certain assumptions or

limitations. With the increasing popularity of digital libraries, image and multimedia databases, the texture analysis has become a focus area of research since image regions can be described by their textural properties.

Texture analysis is a difficult problem due to diversity and complexity of natural textures. Over the years, it has received considerable attention in terms of both methodology and application. This has led to an ever-increasing interest in investigating the theoretical, as well as, the practical issues regarding texture feature extraction and classification. Translation, rotation, and scale invariant texture analysis methods have been areas of particular interest [4]. In these methods the rotation invariant texture analysis remains to be a challenge. A number of methods for rotation invariant texture classification have been proposed [5].

Apart from the requirements of rotation, scale and translation invariance for textures, the gray-scale invariance is another important issue. It arises due to the fact that a texture may be subject to different levels of illumination. The prime objective of this dissertation is to find out some new and novel techniques which shall give rotation, translation and gray-scale invariant texture analysis.

1.2 Contributions of the Dissertation

This dissertation contains three main contributions in the field of texture analysis, by which the rotation, gray-scale and to a large extent the translation invariant texture analysis has been investigated. The first contribution is the extraction of rotation invariant features using Radon Transform (RT) and training the Hidden Markov Model (HMM) on these features to use them for testing and classification. We have proposed three schemes to

extract the feature using RT. In the first scheme we suppress the offset of RT to zero. This implies the projection of image along lines passing through the centre at different angles. However, these features are only rotation invariant. In the second scheme, we do not suppress the offset and find out the RT of the image. The different samples of different projections are used as feature vectors, which are translation as well as rotation invariant. The third scheme uses the variances of the different samples of different projections of RT and formulates the feature vectors for training the HMM. These feature vectors are rotation, translation and gray-scale invariant. A separate HMM is trained for each texture using the extracted set of directional features. Once all the HMMs have been trained, the testing is carried out by choosing any one of these textures at random with arbitrary orientation.

The second major contribution is the rotation and gray-scale invariant texture analysis by using a newly proposed version of RT, which we named as “Differential Radon Transform” (DRT). The DRT gives absolute difference of the gray scale value of every two consecutive pixels along a line and then, all these differences are summed up. This process makes the features as rotation, as well as, gray-scale invariant. In the first case, pure DRT feature vectors are used to train the HMM for their arbitrary orientations. Testing and classification is then carried out by picking any one of these textures with arbitrary orientation. In the second case, we have used both RT, as well as, DRT features by keeping offsets equal to zero. This gave us somewhat low pass, as well as, high passes features. After training the HMM and carrying out the testing and classification, the results were even better than the previous case. In the third case, we keep the offset of RT and DRT as non-zero. A similar procedure was followed as in the above case. This gave us the best results among all the three cases.

The third contribution of this research work is the extraction of rotation and gray-scale invariant features by considering some definition of principal direction (PD) of the texture. Two schemes have been proposed on the basis of different definitions of PD. In the first scheme, a rotation invariant texture analysis technique has been developed by using the Principal Components Analysis (PCA) to find out the PD. In this scheme PD is defined as the direction of eigenvector belonging to the maximum eigenvalue. Once PD is estimated, the Discrete Wavelet Transform (DWT) is used to extract features in that principal direction, which are rotation invariant. k-nearest neighbor (k-nn) is used subsequently for the classification purpose. In the second scheme, the PD is defined as the direction along which there is maximum activity. This maximum activity can be found by using DRT. Then DWT is applied in the principal direction to extract the features which are rotation as well as gray-scale invariant. These features are also classified using the k-nn classifier. In third contribution the proposed schemes are suitable for anisotropic textures only.

The final contribution of the dissertation is towards the feature extraction of hand written characters and their classification through artificial neural networks. Since it is not in line with the rest of the dissertation, it has been given as appendix in the same form as it has been published in the *Journal of Information Science and Engineering*.

1.3 Organization of the Dissertation

Chapter 2 provides an overview of the techniques for texture analysis. It includes the definition and role of texture analysis in the machine vision. It also reviews the rotation, translation, scale and gray-scale invariant texture analysis methods.

Chapter 3 describes the mathematical tools used for texture analysis in this dissertation. These include the PCA, RT, DWT, HMM and some classifiers.

Chapter 4 is dedicated to rotation and gray-scale invariant texture analysis technique that uses RT to extract the features of different textures at different orientations which are used to train one dimensional HMM.

In Chapter 5, the concept and motivation of DRT has been given. Features are extracted using DRT to train the HMM. This technique provides the features of the textures which are rotation as well as gray-scale invariant. Testing is carried out for variety of textures at different orientations by using the trained HMMs. It has also been verified that combined behavior of RT and DRT features improves the results.

Chapter 6 deals with the rotation as well as gray-scale invariant texture analysis technique based on pre-defined PD. In the first scheme DRT has been used to find out PD. DWT is then used to extract the features. In the second scheme, definition of PD is different and it is found using PCA. Both the schemes are suitable to handle the anisotropic textures only.

Chapter 7 summarizes the work and gives suggestions for the future research.

Appendix presents the paper on feature extraction of handwritten characters and their training on Artificial Neural Networks (ANNs)

Chapter 2

Texture Analysis and its Background

2.1 Introduction

There are a number of things which can describe the texture. Examples are the physical roughness of a material, the spatial structure and self-similarity present in materials and the local spatial interactions to make up the whole. Texture is therefore, a very broad field and convincingly covers a wide range of concepts.

Texture analysis is an important issue with applications ranging from remote sensing and crop classification to object-based image coding and tissue recognition in medical images. The primary objective of different methods presented is the rotation and gray-scale invariant texture analysis. This chapter reviews the efforts relevant to this aspiration. Essential basics of image texture and surface texture are summarized and some important texture analysis techniques are appraised.

2.2 Definition of Texture

Despite the importance of texture analysis, there exists no precise definition of the texture. Its definition remains unclear in the literature [6][7][8]. The main reason is that natural textures often display different, yet contradicting properties, such as regularity versus randomness, uniformity versus distortion, which can hardly be described in a unified

manner. Texture is easily perceived by human and is believed to be a rich source of visual information about the nature and three-dimensional shapes of physical objects. Generally speaking, textures are complex visual patterns composed of entities, or sub-patterns that have characteristic such as brightness, color, slope, size, etc. Thus texture can be regarded as a similarity grouping in an image [9]. The local sub-pattern properties give rise to the perceived lightness, uniformity, density, roughness, regularity, linearity, frequency, phase, directionality, coarseness, randomness, fineness, smoothness, granulation, etc., of the texture as a whole [10]. However, different texture definitions have also been proposed by Coggins [11]. These different definitions usually lead to different computational approaches for texture analysis. Few of these are given below:

“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” [12].

“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” [13].

“An attribute representing the spatial arrangement of the grey levels of the pixels in a region” [14].

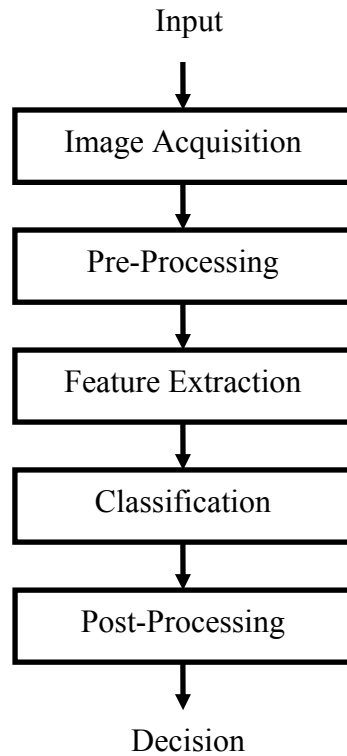


Fig. 2.1 Components of Computer Vision System

2.3 Texture Analysis

Major goals of texture research in computer vision are to understand, model and process texture. A typical computer vision system can be divided into the components which are given in the Fig. 2.1. Texture analysis might be applied to various stages of the process. At the preprocessing stage, images could be segmented into contiguous regions based on texture properties of each region. At the feature extraction and the classification stages, texture features could provide cues for classifying patterns or identifying objects.

As a fundamental basis for all other texture-related applications, texture analysis seeks to derive a general, efficient and compact quantitative description of the textures so

that various mathematical operations can be used to alter, compare and transform them. Most available texture analysis algorithms involve extracting the features and deriving an image coding scheme for representing the selected features. These algorithms may differ in the choice of features and the way of their representation. For example, a statistical approach describes a texture through image signal statistics which reflect the nondeterministic properties of spatial distribution of image signals. A spectral method extracts texture features from the spectral domain. A structural approach considers a texture as a hierarchy of spatial arrangements of well defined texture primitives. A probability model describes the underlying stochastic process that generates the textures. Following are the four major application domains relevant to the texture analysis [15]:

Feature extraction: To compute a characteristic of a digital image which may numerically describe its texture properties.

Texture classification: To determine to which of a finite number of physically defined classes (such as normal and abnormal tissue) a homogeneous texture region belongs.

Texture discrimination: To partition a textured image into regions, each region corresponding to a perceptually homogeneous texture (leads to image segmentation).

Shape from texture: To reconstruct 3D surface geometry from texture information.

2.4 Texture Feature extraction

Texture analysis has two major phases. The first phase is feature extraction in which the image information is reduced to a small set of descriptive features. The second phase deals with classification the features obtained from the texture. Various methods for texture feature extraction have been proposed during the last decades [16] [17] [18], but the texture analysis problem remains difficult and is still a subject of intensive research.

A wide variety of methods for describing the texture features have been proposed. Tuceryan and Jain [19] divided texture analysis methods into four major categories which are statistical, geometrical, model-based and signal processing based. The following discussion provides brief introduction to each of the four categories.

- ***Statistical methods***

Statistical methods analyze the spatial distribution of grey values, by computing local features at each point in the image, and deriving a set of statistics from the distributions of these features. With this method, the textures are described by statistical measures. Depending on the number of pixels defining the local feature, the statistical methods can be further classified into first-order (one pixel), second-order (two pixels) and higher-order (three or more pixels) statistics. The performance of these methods has been evaluated by Conner and Harlow [20].

- ***Geometrical methods***

The geometrical methods of texture are based on the view that textures are made up of primitives with geometrical properties. In these methods, it is common either to compute statistical features, or to identify the placement rules that describe the texture. In these methods the textures comprise of the primitives that appear in certain patterns with some placement rules [21]. In general, it is difficult to extract these elements from real textures. Structural methods may also be used for texture synthesis.

- ***Model-base methods***

Model-based texture methods try to capture the process that has generated the texture. In model-based features, some image model is assumed and its parameters are estimated for subimages. These model parameters are used as features. There are currently

three major model-based methods, which are, Markov Random Fields by Dubes and Jain [22], fractals by Pentland [23], and the multi-resolution autoregressive features introduced by Mao and Jain [24]. The detailed discussions of image models can be found in [25] by Kashyap, and [26] by Chellappa et al.

- ***Signal processing(Transform) methods***

Signal processing methods perform frequency analysis of the textures. It can be achieved by using spatial filters or through filtering in the frequency domain. Randen et al [27] presented a comparative study of filtering for texture classification. Some well known signal processing method are based on Law's Filter [28], Gabor filters [29][30], and pseudo-Wigner distribution [31].

A number of methods, mostly based on the above approaches, for describing texture features have been proposed. Each one has its own definition of the features which is used in the classification. Statistical approaches do not attempt to understand explicitly the hierarchical structure of the texture. Instead, they deal with the texture indirectly through the non-deterministic properties that govern the distributions and relationships between the grey levels of an image. Methods based on second-order statistics (i.e. statistics given by the pairs of pixels) have been shown to achieve higher discrimination rates as compared to the transform-based and structural methods [32]. Human texture discrimination in terms of texture statistical properties is investigated in [33]. Accordingly, the textures in grey-level images are discriminated spontaneously only if they differ in second order moments. Equal second order moments, but different third-order moments require deliberate cognitive effort. This may be an indication that for automatic processing, the statistics up to the second order may be the most important [34]. The most popular second-order statistical features for

texture analysis are derived from the so-called co-occurrence matrix [16]. These features have a potential for effective texture discrimination in biomedical-images [35][36]. The approach based on multidimensional co-occurrence matrices was developed to outperform wavelet packets (a transform-based technique) when applied to texture classification [37]. Also some works are based on the analysis of some of the second order statistical properties of the texture [38] such as the co-occurrence matrix [39].

Several stochastic models have also been proposed for texture modelling and classification. They include Gaussian Markov random fields models [40][41], Moving Average (MA), Autoregressive (AR), and Autoregressive Moving Average (ARMA) models [42][43]. A fractal model has been presented in [44][45], in which the statistical and harmonic features have been combined. Signal processing techniques are mainly based on texture filtering followed by energy evaluation. A review of major filtering approaches and a comparative study has been performed in [46]. Multi-channel texture analysis systems, which use a filter bank instead of a single filter, have been described in [47][48]. In particular, Gabor filtering has been extensively studied in [49][50].

A challenging problem in image classification is to extract rotation-invariant texture features. A Markov Random Field (MRF) [51] is a powerful tool to model the probability of spatial interactions in an image and has been extensively applied to extract texture features for image classification. As features based on MRF models are generally rotation-variant [52], hence application of MRF models for classifying rotated images is strictly limited. Cohen et al. [52] modelled the texture as Gaussian Markov Random Field (GMRF) and used the maximum likelihood techniques to estimate the rotation angle and scale

parameters. The problem with this method is that the likelihood function is highly nonlinear and local minima could also be there. Chen and Kundu [53] have used multi-channel sub-band decomposition along with an HMM to solve the problem. They have used a quadrature mirror filter for decomposition of the image into sub-bands, and then modelled the features of these sub-bands by an HMM. In that method, the textures with different orientations are assumed to be in the same class. Since textures with different orientations create different signal components for each subband, this increases the variations in the feature space. Hence, as the number of classes increases, the performance may deteriorate.

The GMRF model has been shown to be a powerful method of texture analysis and classification [41]. The GMRF parameters and noise source variance of a given model can be estimated for a texture using the least squares approach, which are often successfully employed as features for texture classification. However, the traditional GMRF models are not rotation invariant due to the structure of their neighbour sets.

Kashyap and Khotanzad [25] constructed an Isotropic Circular GMRF (ICGMRF) model to extract rotation-invariant features. The ICGMRF model is defined in a circular neighbourhood system. The values of the neighbours which are not located on the image grid are bilinearly interpolated. The values of all equiradius pixels are used to generate only one feature. The ICGMRF model, therefore, discards the directional information in the possibly anisotropic textures [52]. To capture directional information, the ICGMRF model was extended into a novel Anisotropic Circular GMRF (ACGMRF) model given in [54].

In the past decade, wavelet theory has been widely used for texture classification purposes [55][56][57][58][59][60]. In [61], the local spectral histograms, consisting of

marginal distributions of responses from a bank of filters, have been considered as the feature statistics. Although, many of the aforementioned methods have good classification performances, but they exhibit a high misclassification rate when the texture is rotated. Rotation invariant texture classification using wavelets are presented in [62] and [63]. Haley and Manjunath [63] employed a complete space-frequency model using Gabor wavelets to achieve a rotation-invariant texture classification. However, this method is also computationally complex. In [64], the authors present a rotation invariant model based texture classification method. In this work the texture is modelled as the output of a linear system driven by a binary image. The texture feature extraction is constituted of two steps. The first one consists of estimating the binary excitation, which is assumed to be the efficient presentation of the texture for classification purposes, by means of some blind deconvolution procedure. In the second step, a basis of moment invariants is employed to characterize the Autocorrelation Function (ACF) of the binary excitation. This technique is applied to 15 classes and Percentage of Correct Classification (PCC) is considered as the figure of merit which has been reported as 88.3%.

Wu and Wei [65] created one-dimensional (1-D) signals by sampling the images along a spiral path and used a quadrature mirror filter bank to decompose the signals into subbands and calculated several features for each subband. In this method, uniform fluctuations along the radial direction do not correspond to the uniform fluctuations in the 1-D signals which is due to the increasing radius of the spiral path. Therefore, the variational information in the radial direction is deteriorated. Do and Vetterli [66] used a steerable Wavelet-Domain HMM (WD-HMM) and a Maximum Likelihood (ML) estimator to find the model parameters. However, the rotation-invariant property of the estimated

model relies on the assumption that the ML solution of WD-HMM is unique and the training algorithm is able to find it. They examined the rotation invariance property of their method for 13 texture images from Brodatz album.

There are different techniques in literature to estimate the orientation of the image. These include the methods based on image gradients [67], angular distribution of signal power in the Fourier domain [68][69] and signal autocorrelation structure [67]. RT has been widely used in image analysis. Magli, et al. [70] used RT and 1-D continuous WT to detect linear patterns in the aerial images. Warrick and Delaney [71] used a localized RT with a wavelet filter to accentuate the linear and chirp-like features in synthetic aperture radar (SAR) images. Leavers [72] used RT to generate the taxonomy of shape for characterization of abrasive powder particles. Do and Vetterli [73] use ridgelet transform, which is a combination of finite RT and 1-D discrete wavelet transform, to approximate and denoise the images with straight edges. Ridgelet transform is also used to implement curvelet decomposition, which is used for image denoising [74].

Due to the inherent properties of RT, it is a useful tool to capture the directional information of the images. Khouzani and Zadeh [60] utilized RT to convert the rotation to translation and then apply a translation-invariant WT to the result to extract the texture features. Optimal numbers of projections for RT are proposed in this method. The extracted features generate an efficient orthogonal feature space. As a result of summing pixel values to generate projections in RT, this technique is robust to additive white noise. Although the polar concept has been previously used in the literature to achieve the rotation invariance, but the proposed method is different to the previous ones. Instead of using the polar

coordinate system, this method uses the projections of the image in different orientations. In each projection, the variations of the pixel intensities are preserved even if the pixels are far from the origin. Therefore, the method does not measure the intensity variations based on the location in the image. Furthermore, this method captures global information about the texture orientation, therefore, it preserves the directional information. This method is computationally complex. In the first step, the RT of the texture is calculated and then in the second step, a translation-invariant wavelet transform is used to calculate the frequency components and to extract the corresponding features.

Khouzani and Zadeh [75], also employed the RT to capture the directional information and to adjust the orientation of the texture for feature extraction. This is similar to the manual texture analysis in which, an unknown texture is rotated so that it could match with some known one. Analyzing the textures along their principal directions allows the creation of features with smaller intra-class variability, thus allowing higher separability. In this technique, the principal direction of the texture is estimated using RT and then the image is rotated to place the principal direction at zero degrees. DWT is then employed as a next step, to extract the features. The method for estimation of the principal direction is robust to additive white noise and illumination variations. Though the proposed method for principal direction estimation is well suited to most of the ordinary textures, however, complex textures may need complex techniques. For example, some textures may have straight lines along several directions. This may create an ambiguity for the direction estimation. In this situation, more complex techniques may be employed to estimate the principal direction.

Chapter 3

Computing Tools

3.1 Introduction

In this chapter we shall present some basic essentials of the computing tools used in this dissertation. These include Principal Component Analysis (PCA), Radon Transform (RT), Wavelet Transform (WT), Classifiers and Hidden Markov Model (HMM). In describing these essential tools, we have limited ourselves to the discussion relevant to this dissertation.

3.2 Principal Component Analysis

Principal component analysis (PCA) is a frequently used statistical technique, widely known as Karhunen-Loeve (KL) transformation, for optimal lossy compression of data under least square sense. It involves a mathematical procedure that transforms a larger number of correlated variables into a smaller number of uncorrelated variables called principal components. It is a way of identifying patterns in data, and expressing the data in such a way as to highlight their similarities and differences [76]. Since patterns in data can be difficult to find out especially in case of the data of high dimensions, where the luxury of graphical representation is unavailable, PCA is a powerful tool for analysis of such data.

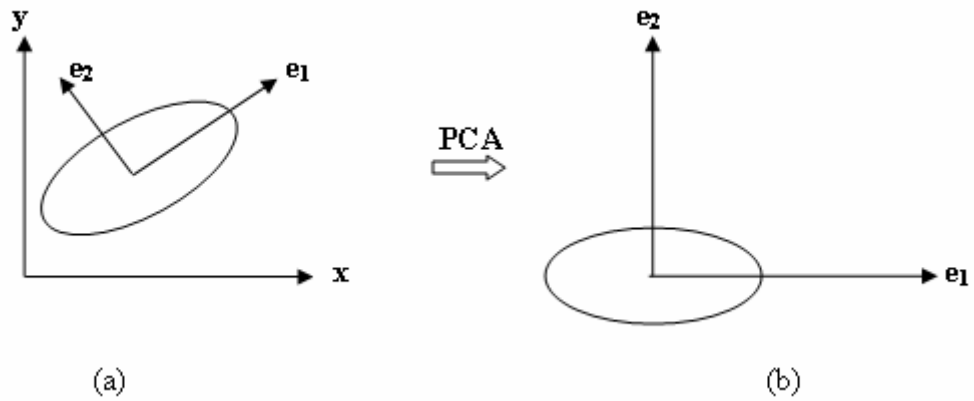


Fig. 3.1 PCA Transformation (a) Eigenvector of an object (b) Object along eigen-axis

PCA provides an orthogonal basis vector-space which is also known as eigen-space, for presentation of the original data. The eigen-space is a subspace of the image-space spanned by a set of eigenvectors of the covariance matrix of the trained images. The covariance matrix is constructed by performing PCA, which rotates the dataset to its primary axes that lie along the eigenvectors with the highest modes of variation as given in Fig. 3.1. Eigenvectors with the highest associated eigen-values represent the highest modes of variation in the dataset of images, whereas, the eigen-vectors with the lowest eigen-values represent the lowest modes of variation. In order to find out the principal components the covariance matrix, eigenvalues and eigenvectors are needed. Following is the mathematical explanation of PCA.

Consider a random vector population \mathbf{x} given as:

$$\mathbf{x} = (x_1, x_2, \dots, x_n). \quad (3.2.1)$$

The mean of this population will be

$$\mu_x = E\{\mathbf{x}\} \quad (3.2.2)$$

and the $n \times n$ covariance matrix will be

$$\mathbf{C} = E\{(\mathbf{x} - \mu_x)(\mathbf{x} - \mu_x)^T\} \quad (3.2.3)$$

With zero mean the covariance matrix becomes

$$\mathbf{C} = E\{(\mathbf{x})(\mathbf{x})^T\} \quad (3.2.4)$$

For an $n \times n$ matrix \mathbf{C} , there are n scalars, $\lambda_i, i = 1, 2, \dots, n$, such that

$$|\mathbf{C} - \lambda_i \mathbf{I}| = 0 \quad (3.2.5)$$

The λ_i are called eigen-values of the matrix. The set of n vectors \mathbf{e}_i such that

$$\mathbf{C}\mathbf{e}_i = \lambda_i \mathbf{e}_i \quad i = 1, 2, \dots, n \quad (3.2.6)$$

are called the eigenvectors of \mathbf{C} . They are $n \times 1$, and each corresponds to one of the eigenvalue. The data set which has the most significant amount of energy corresponds to the vector \mathbf{e}_i with maximum eigenvalue, λ_{\max} . The direction of this eigenvector is the principal direction of the image. We are using PCA method for this very purpose which is to find out the direction of the eigenvector corresponding to the largest eigenvalue.

3.3 Radon Transform

The Radon transform (RT) is a mathematical technique that maps events in an image or in a data set with different characteristics, e.g. curvatures or slopes, into different locations of a new space, called the Radon space. It has got popularity in seismic data processing, image processing, tomography, etc. The RT of a function $f(x, y)$ is defined as the integral along a straight line at a distance s from the origin and at angle of inclination θ from y-axis. Its mathematical expression can be written as

$$g(s, \theta) \triangleq \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) \delta(x \cos \theta + y \sin \theta - s) dx dy \quad (3.3.1)$$

where $-\infty \leq s < \infty$ and $0 \leq \theta < \pi$. The delta function defines integration only over this line. The new rotated coordinate system (s, u) is given as $s = x \cos \theta + y \sin \theta$ and $u = -x \sin \theta + y \cos \theta$. Similarly the inverse coordinates are given as $x = s \cos \theta - u \sin \theta$ and $y = s \sin \theta + u \cos \theta$. In terms of (s, u) , Eq. (3.3.1) can be written as

$$g(s, \theta) = \int_{-\infty}^{\infty} f(s \cos \theta - u \sin \theta, s \sin \theta + u \cos \theta) du \quad (3.3.2)$$

$g(s, \theta)$ is a summation of $f(x, y)$ along a line at an angle θ from y -axis and distance s from the origin as shown in Fig. 3.2. $f(x, y)$ is the given function and $R(s, \theta)$ indicates its RT. The RT maps the spatial domain (x, y) to the projection domain (s, θ) , in which each point corresponds to a straight line in the spatial domain.

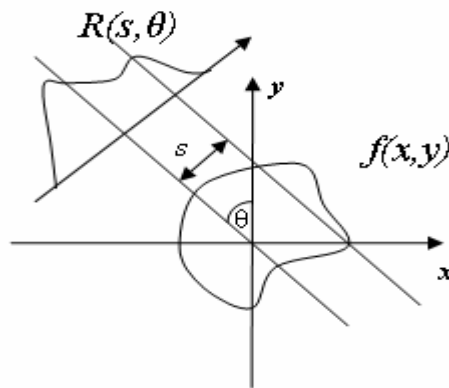


Fig. 3.2 Radon Transform

3.4 Properties of Radon Transform

The RT has several useful properties, which can be summarized as follows:

1. *Linearity*: If RT of $f_1(x, y)$ is given by $g_1(s, \theta)$ and RT of $f_2(x, y)$ is given by $g_2(s, \theta)$, then RT of linear combination $a_1 f_1(x, y) + a_2 f_2(x, y)$ will be given by $a_1 g_1(s, \theta) + a_2 g_2(s, \theta)$.

2. *Space limitedness*: If $f(x, y) = 0$, for $|x| \geq \frac{D}{2}$, $|y| \geq \frac{D}{2}$ then

$$g(s, \theta) = 0, \text{ for } |s| > \frac{D}{\sqrt{2}}$$

3. *Symmetry in (s, θ) domain*: RT of $f(x, y)$ i.e. $g(s, \theta)$ has the following property

$$g(s, \theta) = g(-s, \theta \pm \pi)$$

4. *Periodicity in (s, θ) domain*: RT of $f(x, y)$ i.e. $g(s, \theta)$ has the periodicity property given as

$$g(s, \theta) = g(s, \theta + 2k\pi)$$

where k is an integer.

5. *Shift*: If $f(x, y)$ goes through a translation i.e. $f(x - x_0, y - y_0)$, then its RT becomes $g(s - x_0 \cos \theta - y_0 \sin \theta, \theta)$.

6. *Rotation by θ_0* : If $g(s, \theta)$ is RT of $f_p(r, \phi) = f(x, y)$, then RT of $f_p(r, \phi + \theta_0)$ is $g(s, \theta + \theta_0)$.

7. *Scaling*: If $f(x, y)$ is scaled by a factor 'a', then RT of $f(ax, ay)$ is given as

$$\frac{1}{|a|} g(as, \theta), a \neq 0.$$

8. *Mass Conservation*: This is like Parsval's theorem i.e.

$$\int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x, y) dx dy = \int_{-\infty}^{+\infty} g(s, \theta) ds, \quad \forall \theta$$

9. *One-dimension Fourier Transform of RT*: One-dimensional Fourier transform of RT at constant θ is two-dimensional Fourier transform of $f(x, y)$ at constant θ . The one dimensional Fourier Transform of $g(r, \theta)$ is given as

$$\begin{aligned} G(\zeta, \theta) &\triangleq \int g(s, \theta) e^{-j2\pi\zeta s} ds \\ &= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(s \cos \theta - u \sin \theta, s \sin \theta + u \cos \theta) du e^{-j2\pi\zeta s} ds \end{aligned} \quad (3.3.3)$$

Perform the co-ordinate transformation from s, u to x, y the above equation becomes

$$duds = dxdy |J| \quad \text{where } J = 1 \quad (3.3.4)$$

$$\begin{aligned} G(\zeta, \theta) &\triangleq \int g(s, \theta) e^{-j2\pi\zeta s} ds \\ &= \int_{-\infty}^{+\infty} \int_{-\infty}^{+\infty} f(x, y) e^{-j2\pi\zeta(x \cos \theta + y \sin \theta)} dxdy \\ &= F(\zeta \cos \theta, \zeta \sin \theta) \end{aligned} \quad (3.3.5)$$

where $F(\zeta \cos \theta, \zeta \sin \theta)$ is the two dimensional Fourier Transform of $f(x, y)$.

3.5 Wavelet Transforms

A considerable interest has arisen in recent years regarding new transform techniques that specifically address the problems of image compression, edge and feature detection, and texture analysis. The techniques come under the headings of multi-resolution analysis, time frequency analysis, pyramid algorithms, and wavelet transforms [77]. Wavelets are well-suited for approximating data with sharp discontinuities. The Wavelet Transform uses multi-resolution technique by which different frequencies are analyzed with

different resolutions. Wavelet transform is an excellent scale analysis and interpreting tool [78]. It transforms image into multiresolution representation, which analyzes image variations at different scales and provides good energy compaction (high image information content) as well as adaptability to human visual system. Wavelet transform offers high temporal localization for high frequencies while offering good frequency resolution for low frequencies.

Wavelet coefficients of a signal $f(x)$ are the projection of the signal onto the multi-resolution subspaces

$$V_j = \overline{\text{span}\{\varphi_{j,k}(x), k \in Z\}} \quad (3.4.1)$$

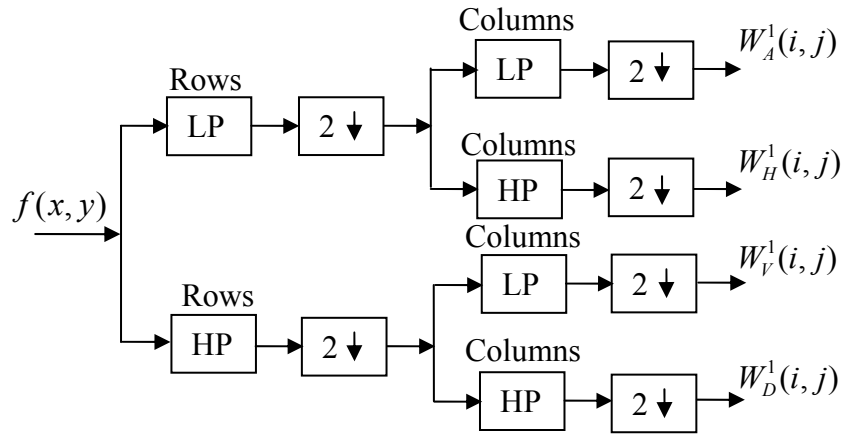
and

$$W_j = \overline{\text{span}\{\psi_{j,k}(x), k \in Z\}}, \quad j \in Z \quad (3.4.2)$$

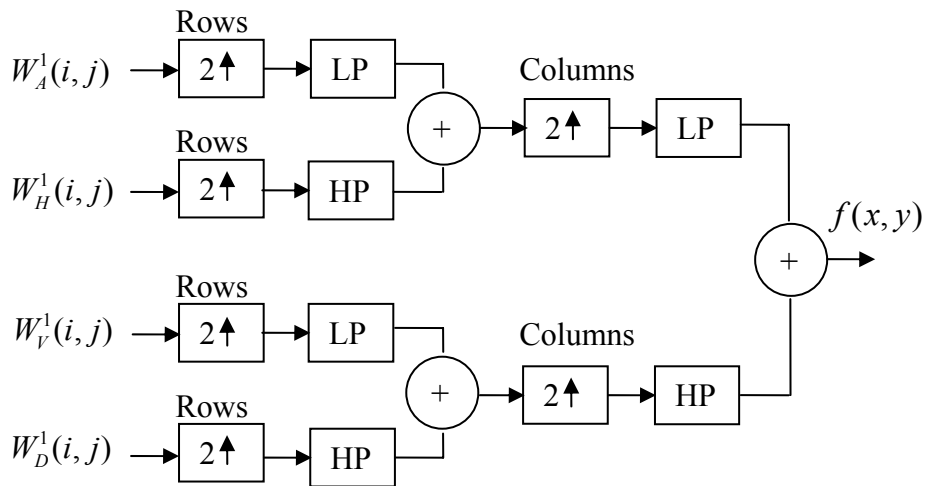
where the basis functions $\varphi_{j,k}(x)$ and $\psi_{j,k}(x)$ are constructed by dyadic dilations and translations of the mother scaling and mother wavelet functions $\varphi(x)$ and $\psi(x)$, respectively, given as

$$\left. \begin{aligned} \varphi_{j,k}(x) &= 2^{j/2} \varphi(2^j x - k) \\ \psi_{j,k}(x) &= 2^{j/2} \psi(2^j x - k) \end{aligned} \right\} \quad (3.4.3)$$

Similarly, the 2-D basis functions which are the products of scaling and wavelet functions are given as

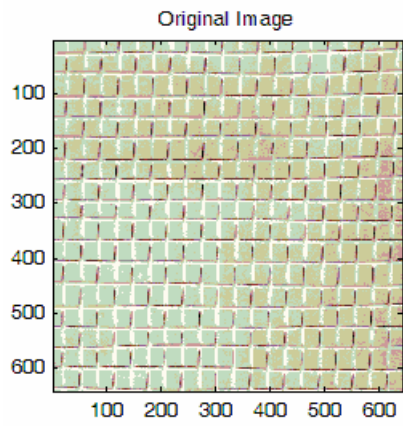


(a)

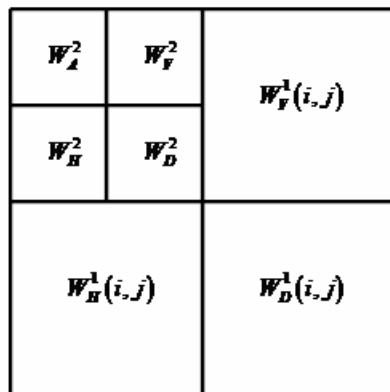


(b)

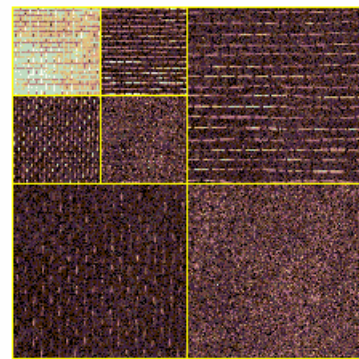
Fig. 3.3 Block diagram of separable wavelet filter bank in 2-D: (a) The analysis filter bank (b) The synthesis filter bank



(a)



(b)



Decomposition at level 2

(c)

Fig. 3.4 Two level decomposition of Brodatz texture D1 (a) Original image (b) Organization of the detail images within the wavelet transform (c) Decomposed image at level 2

$$\left. \begin{aligned} \varphi_{j,m,n}(x, y) &= \varphi_{j,m}(x)\varphi_{j,n}(y) \\ \psi_{j,m,n}^H(x, y) &= \psi_{j,m}(x)\varphi_{j,n}(y) \\ \psi_{j,m,n}^V(x, y) &= \varphi_{j,m}(x)\psi_{j,n}(y) \\ \psi_{j,m,n}^D(x, y) &= \psi_{j,m}(x)\psi_{j,n}(y) \end{aligned} \right\} \quad (3.4.4)$$

where ψ^H gives horizontal edges, ψ^V gives vertical edges and ψ^D gives variations along the diagonal. The wavelet decomposition of a 2-D signal can be achieved by applying the wavelet decomposition along the rows and columns of the image separately [79]. After decomposition, an original image has been divided into four subimages, where one is the low-resolution approximation sub-image and three difference sub-images. Three difference sub-images are categorized as horizontal detail image, vertical detail image and diagonal detail image. Fig. 3.3 (a) illustrates the block diagram of the image being decomposed into 4 wavelet subband images at the first level and Fig. 3.3 (b) describes the reconstruction of the image from the decomposed 4 wavelet subband images. Fig. 3.4 describes 2 level wavelet decomposition of Brodatz texture D1. In this example, the wavelet ‘db4’ is used for decomposition.

3.6 Classifiers

A work able definition of Pattern Recognition is given by Vapnik [80]:

A person (the instructor) observes the occurring situations and determines to which of the class, among c classes, each one of them belongs. It is required to construct a device (the classifier) which, after observing the instructors procedure, will carry out the classification approximately in the same manner as the instructor.

According to Jain et al. [81]:

Pattern recognition is a general term to describe a wide range of problems like recognition, description, classification, and grouping of patterns. These problems are important in a variety of engineering and scientific disciplines such as biology, psychology, medicine, marketing, artificial intelligence, computer vision and remote sensing.

The design of a pattern recognition system essentially involves the following four steps: data acquisition, preprocessing, feature extraction and finally the decision making. A well-defined and sufficiently constrained recognition problem will lead to a compact pattern representation and a simple decision making strategy. An important and desired attribute, of the most pattern recognition system, is learning from set of examples. The four best-known approaches for pattern recognition are: template matching, syntactic matching, statistical classification and neural networks. The simplest and earliest approach to pattern recognition is based on template matching. The pattern to be recognized is matched against the stored template taking into account the translation, rotation and scaling. The similarity measure, often a correlation, may be optimized based on the available training set. It is computationally demanding approach, but the availability of faster processors has now made this approach more feasible. In the syntactic approach [82][83] pattern is viewed as being composed of simple sub-patterns which are themselves built from yet simple sub-patterns. The elementary sub-patterns to be recognized are called primitives. The given complex pattern is represented in terms of the interrelationships between these primitives. This paradigm has been used in situations where the patterns have a definite structure which can be captured in terms of a set of rules, such as EKG waveforms, textured images, and

shape analysis of contours. In the statistical approach, the patterns are described as random variables, from which class densities can be inferred. Classification is thus based on the statistical modeling of data. Neural networks have the ability to learn complex non-linear input-output relationships. Sequential training procedures are being used in the networks to adapt them according to the data. This approach is strongly related to the statistical method, since they can be regarded as parametric models with their own learning schemes.

Following two could be the reasons of recognition of patterns. As a first reason, it is assumed that the input pattern is identified as a member of a predefined class and that use of this information is made during classifier design. This type is called *supervised classification*. In case of the reason the task is second task consists of *unsupervised classification* in which the pattern is assigned to a hitherto unknown class.

A schematic representation of a recognition system is shown in the Fig. 3.5. A preprocessing module can be applied to segment the pattern of interest from the background, remove noise, normalize the pattern, and any other operation which will contribute in defining a compact representation of the pattern. A recognition system can be operated in two modes. First one is training and classification. In the training mode the feature extraction module finds the appropriate features for representing the input patterns. Then, the classifier is trained to partition the feature space. A feedback path allows optimizing the pre-processing and feature extraction strategies. In the classification mode, the trained classifier assigns the input pattern to one of the pattern classes, based on the measured features. The dataset used during construction of the classifier, i.e. training set, is different from the one used for evaluation, i.e. the test set.

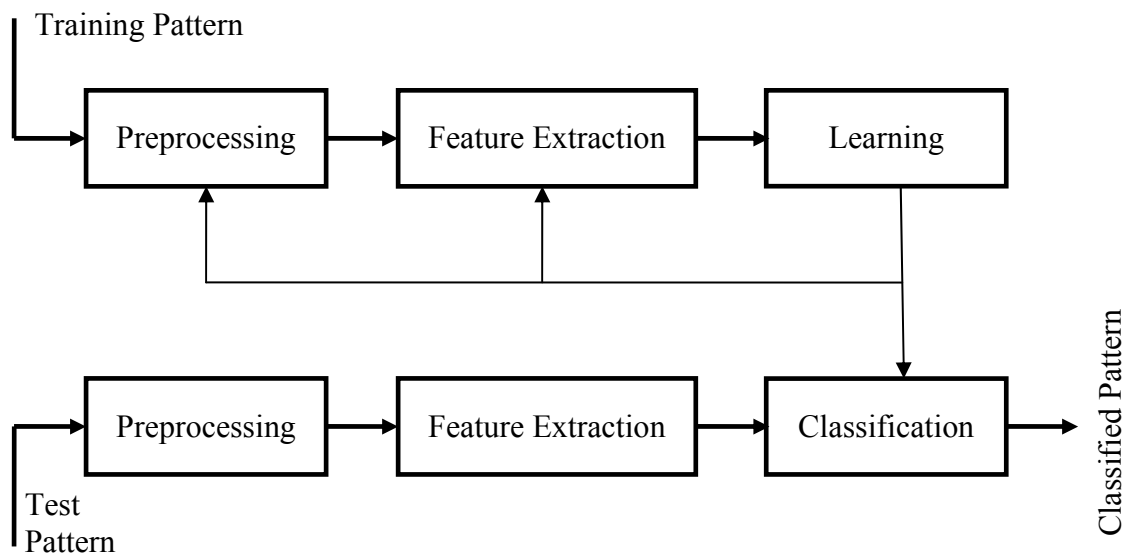


Fig. 3.5 The Recognition Process

It can be summarized that a classifier is a formula, algorithm or technique that outputs a class label for any feature vector, \mathbf{x} , applied to its input. Classifying patterns is then equivalent to assigning a class to a particular feature vector. Designing a classifier, means finding the decision rule i.e. the rule by which, given feature vector \mathbf{x} is assigns it to class m_i . It can be seen that the decision rule partitions the feature space into separate regions. If a feature vector is located inside a particular region, it will be assigned to the class associated with that region. Following are the different types of the classifier.

3.6.1 The Bayes Classifier

The fundamental idea used in statistical pattern recognition is Bayes decision theory [84]. The c classes, m_i ($i = 1, 2, \dots, c$), are treated as random entities that occur with *a priori*

probabilities $p(m_i)$, $i = 1, 2, \dots, c$. The *a posteriori probability* of being in class m_i for an observed feature vector \mathbf{x} is calculated using Bayes rule

$$p(m_i|\mathbf{x}) = \frac{p(m_i)p(\mathbf{x}|m_i)}{p(\mathbf{x})} \quad (3.5.1)$$

where $p(\mathbf{x}|m_i)$ is the class-conditional probability density function of \mathbf{x} , given the class m_i and $p(\mathbf{x})$ is the normalization term calculated by law of total probabilities

$$p(\mathbf{x}) = \sum_{i=1}^c p(m_i)p(\mathbf{x}|m_i) \quad (3.5.2)$$

so that the posteriori probabilities sum to 1

$$\sum_{i=1}^c p(m_i|\mathbf{x}) = 1 \quad (3.5.3)$$

$p(\mathbf{x})$ is the unconditional probability density governing the distribution of all \mathbf{x} irrespective of their class membership.

A number of well known decision rules, including the Bayes rule, the maximum likelihood rule and Neyman-Pearson rule are available to define the decision boundary. The Bayes rule is based on finding the class that gives the maximal *posterior probability* taking into account the loss. Define the loss, $l(m_i, m_j) = 1 - \delta_{ij}$, as the penalty of wrongly classifying a pattern in m_i when it should have been classified in class m_j [85]. The optimal Bayes decision rule for minimizing the risk can be stated as: the function $\hat{m}(\mathbf{x}) = m_i$, i.e. the function that assigns a class label to input feature vector for which the conditional risk

$$R_{m_i}(\mathbf{x}) = \sum_{j=1}^c l(m_i, m_j)p(m_j|\mathbf{x}) \quad (3.5.4)$$

is minimized. The loss function as defined above can be written as

$$l(m_i, m_j) = 1 - \delta_{ij} = \begin{cases} 1, & i \neq j \\ 0, & i = j \end{cases} \quad (3.5.5)$$

so that conditional risk becomes the conditional probability of misclassification. 0 loss is assigned to correct classification and 1 loss to any misclassification. For this choice of loss function, the Bayes decision rule also called the maximum posteriori (MAP) rule, which assign the input feature vector to class m_i if

$$p(m_i | \mathbf{x}) > p(m_j | \mathbf{x}), \quad \text{for all } j \neq i \quad (3.5.6)$$

Employing Bayes formula (3.5.1) gives the equivalent rule in terms of the class like-lihoods

$$\hat{m}(\mathbf{x}) = m_i \text{ if } p(\mathbf{x} | m_i) p(m_i) \geq p(\mathbf{x} | m_j) p(m_j), \quad \text{for all } j \quad (3.5.7)$$

since the estimation of the class a priori probabilities $p(m_i)$ is usually not a big problem.

The design of the Bayes classifier is based on the class conditional likelihoods given in eq. (3.5.7) or on the a posteriori probability densities as in eq. (3.5.6) which is optimal. Even when the Bayes classifier is optimal, still some misclassification errors can occur. This is due to the nature of the feature space in which the class likelihoods overlap so that vectors of class m_i may fall outside region R_i . The Bayes error ε defined as

$$\varepsilon = \sum \int \cdots \int_{\bigcup_{j \neq i} R_j} p(m_i) p(\mathbf{x} | m_i) d\mathbf{x} \quad (3.5.8)$$

is the expectation of finding feature vectors into wrong decision regions. Since the Bayes classifier is optimal, no other classifier exists which leads to a lower misclassification error. To reduce the misclassification error there is a need to construct other features so that less overlap exists between the class likelihoods.

Some practical limitations on constructing the Bayes classifier is that it requires perfect knowledge of the relevant probability densities which is not available in practice and must

be inferred from the available data. One approach is to assume a certain parametric form (like a Gaussian) for the class conditional likelihoods and estimate the parameters governing this distribution called parametric classifier. Another one is to estimate the class conditional likelihoods directly from the data in local neighborhood in feature space called non-parametric classifiers.

3.6.2 Neural Network Classifiers

The foundations of neural networks can be traced back to the single layer perceptron [86] which was capable of implementing linear decision boundaries in feature space. A variety of nets exist which are capable of modeling more complex decision boundaries. Some good textbooks covering the topic are [87] [88] [89] [90].

A typical example of a neural network is the feedforward multilayer perceptron depicted in Fig. 3.6. It consists of several neurons grouped in several layers. Neurons between different layers are interconnected. In this feed-forward architecture the features are received in the input layer and propagated through the hidden layers to the output layer. Each neuron processes the signal it receives and sends the output to the following layer. The j th output of the neuron, u_j , of any layer is the weighted sum of all the inputs, i.e.

$$u_j = \sum_{i=0}^{n_s-1} w_{ij}^{[s]} x_i^{[s]} \quad (3.5.9)$$

where $w_{ij}^{[s]}$ represents the synaptic weights of the j^{th} neuron which are multiplied by the inputs $x_i^{[s]} = o_i^{[s-1]}$ to obtain outputs $o_i^{[s]}$, n_s is the number of neurons in the s^{th} layer where

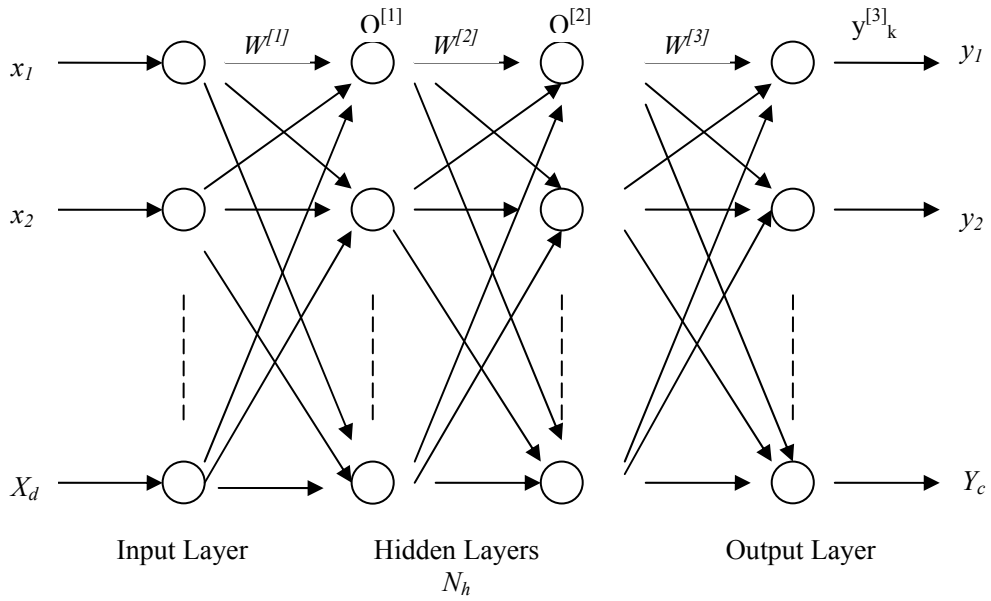


Fig. 3.6 Multilayer perceptron with an input layer (d neurons), a hidden layer (N_h neurons) and an output layer (c neurons)

$s = 1, 2, 3$. $\psi(u_j)$ is the activation function used for each unit. The output of the j^{th} unit in the s^{th} layer can be written as

$$y_j^{[s]} = \psi \left(\sum_{i=0}^{n_s-1} w_{ij}^{[s]} y_j^{s-1} \right), \quad (3.5.10)$$

where $y_j^{s-1} = o_i^{[s-1]}$.

The activation functions of the output layer are often the Heavyside step function ($\psi(x) = 1$ if $x > 0$ and 0 otherwise), so that the output of the neural network is a configuration of 0's and 1's. With each such configuration, a pattern class is associated. Designing the classifier, often called the training of the net, is done by assigning the weights an initial value, often random, and then by adjusting their values. The values are adjusted in

such a way that if a feature vector is fed at the input of the network, the output layer reflects the corresponding class. This is usually performed by iteratively feeding the samples of the design set at the input of the network and observing the output error which is used for adjusting the weights. The number of times a sample is drawn from the design set to update the weights is called the learning time and the rule to update them is called the learning rule. An important rule is the back-propagation rule [91] which made it possible to train multilayer perceptrons. A popular error function used to derive this rule is the sum of the squares of the difference of the output of the network and the desired output. This error function can be differentiated with respect to the network weights and these derivatives can then be used to find weight values which minimize the error. Placing this scheme into the same picture as before, it can be seen that neural networks also partition the feature space in several decision regions. The eq. (3.5.10) assigns a feature vector \mathbf{x} to a particular class via the output y_k . The possible forms that the decision boundaries can assume are specified by the architecture of the net. Architecture of the net means the number of nodes, their connectivity and activation functions. The exact form of the decision boundary is determined by the weights which are adapted during training.

The reason for calling this scheme a “neural network” is that it bears some resemblance with the structure of the brain. In a very simplified form the brain consists of neurons which are connected by dendrites through which signals (in the form of electrical and chemical impulses) propagate. Making this comparison could give neural networks more credit than they deserve: it makes them look like if they are easily capable of mimicking human intelligence. However, if we look at (3.5.10) it is obvious that they represent nothing more than a complex function depending on the weights and activation

functions, which will assign each point in feature space to a particular class, i.e. the network is a model for the decision boundary.

3.6.3 Parametric Classifiers

Parametric classifiers assume that the class conditional likelihoods have a known functional form which depends on a few parameters. The classifier is designed by estimating the distribution parameters from the design set. The pattern recognition problem can be seen as an optimization problem: given a features extraction scheme and a loss matrix, one desires to construct the classifier which minimizes the risk. This classifier is called the Bayes classifier. For the Bayes classifier, the decision rule is given by

$$\hat{m}(\mathbf{x}) = m_i \text{ if } R(m_i|\mathbf{x}) \leq R(m_j|\mathbf{x}) \quad (3.5.11)$$

The above decision rule is employed to classify the network. Alternatively, the decision boundary between class m_i and m_j contains all points which satisfy

$$d_i(\mathbf{x}) = d_j(\mathbf{x}) \quad (3.5.12)$$

where $d_i(\mathbf{x}) = p(\mathbf{x}|m_i)p(m_i)$. A vector is then classified according to the decision region to which it belongs. Many parametric classifiers can be designed using different forms or mixtures of parametric distributions. If we consider a Gaussian classifiers, which assumes that the class conditional probabilities are Gaussian, i.e.

$$p(\mathbf{x}|m_i) = \frac{1}{\sqrt{(2\pi)^N |\boldsymbol{\Sigma}_i|}} e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)} \quad (3.5.13)$$

N is dimensionality of the feature space and

$$\boldsymbol{\mu}_i = E[\mathbf{x}] \quad (3.5.14)$$

is the mean of the distribution and

$$\Sigma_i = E\left[(\mathbf{x} - \boldsymbol{\mu}_i)(\mathbf{x} - \boldsymbol{\mu}_i)^T\right] \quad (3.5.15)$$

is the covariance matrix and is symmetric by definition. Its diagonal element Σ_{jj} is the variance of the x_j of class m_i ; the off diagonal element Σ_{jk} is the covariance of x_j and x_k of class m_i . This distribution thus depends on $N + N(N-1)/2$ parameters. Estimators of these parameters are easily obtained by the maximum likelihood method (or by Bayesian inference theory) and can be found in any standard work on statistics or statistical pattern recognition.

To find the decision boundaries we evaluate the d_i using eq (3.5.12). Because of the exponential form it is convenient to take the logarithm of the d_i

$$\begin{aligned} \ln d_i(\mathbf{x}) &= \ln(p(\mathbf{x}|m_i)p(m_i)) \\ &= \ln p(m_i) - \frac{N}{2} \ln 2\pi - \frac{1}{2} \ln |\Sigma_i| - \frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_i)^T \Sigma_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i) \end{aligned} \quad (3.5.16)$$

Dropping the constant term the above express becomes

$$\ln d_i(\mathbf{x}) = \ln p(m_i) - \frac{1}{2} \ln |\Sigma_i| - \frac{1}{2} (\mathbf{x} - \boldsymbol{\mu}_i)^T \Sigma_i^{-1} (\mathbf{x} - \boldsymbol{\mu}_i) \quad (3.5.17)$$

The decision boundary for two class problem is given as

$$\ln d_i(\mathbf{x}) = \ln d_j(\mathbf{x}) \quad (3.5.18)$$

or

$$\begin{aligned} \ln \frac{p(m_1)}{p(m_2)} - \frac{1}{2} \ln \frac{|\Sigma_1|}{|\Sigma_2|} - \frac{1}{2} \left[(\mathbf{x} - \boldsymbol{\mu}_1)^T \Sigma_1^{-1} (\mathbf{x} - \boldsymbol{\mu}_1) \right] \\ + \frac{1}{2} \left[(\mathbf{x} - \boldsymbol{\mu}_2)^T \Sigma_2^{-1} (\mathbf{x} - \boldsymbol{\mu}_2) \right] = 0 \end{aligned} \quad (3.5.19)$$

Since this equation does not contain terms higher than the second order statistics in \mathbf{x} , hence this decision region is hyperquadric. A Gaussian classifier thus places a second

order decision surface between each pair of the pattern classes. If the class likelihoods are indeed Gaussian, no more complex boundaries are required to minimize the loss.

3.6.4 Non-Parametric Classifier

In contrast to the parametric classifier, this section deals with the classifiers which do not assume an *a priori* known parametric form for the class likelihoods $p(\mathbf{x}|m_i)$, but estimates them directly from the design samples. A parametric classifier is designed by estimating the required density parameters. After design, all relevant densities are known in the feature space and a sample is classified using the Bayes decision rule. All parametric classifiers operate somewhat similar manner. If a given sample is to be classified, only the densities around that sample are estimated from the design set. Thus non-parametric classifiers implement the decision rule locally and the likelihoods need to be estimated for each sample ordered to the classifier. Suppose L design samples are available and we need to estimate the probability density around \mathbf{x} in feature space. We could construct a region \mathfrak{R} around \mathbf{x} and then count the number of samples that belong to this region. Then an estimate of the density will be given as

$$p(\mathbf{x}) \approx \frac{k}{LV} \tag{3.5.20}$$

where k is the number of samples in the region \mathfrak{R} and V is the volume of the region \mathfrak{R} .

This estimate depends on L and the region \mathfrak{R} . Since the first is given and fixed, we can only control the latter. The larger the region \mathfrak{R} , the more the samples k it contains and the better will be the estimate. However, eq. (3.5.20) implicitly assumes that $p(\mathbf{x})$ remains constant over \mathfrak{R} and this assumption will become less valid for the large \mathfrak{R} . It follows that

the choice for \mathfrak{R} is not trivial and should depend on the given design set for optimum classifier design. Given the design set and a sample to be classified, there are two different approaches to evaluate (3.5.20):

1. *Parzen windows approach*: Fix the region \mathfrak{R} and count k .
2. *k-nn Classifier*: Fix count k and determine the volume V .

Both approaches are well documented in the literature. Since the latter has been frequently used in this dissertation, it will be explained in the following section.

3.6.4.1 k-nn Classifier

The k Nearest Neighbor (k-nn) classifier relies on eq (3.5.20) in which k is fixed. To estimate the density around \mathbf{x} , its k nearest neighbors from the design set are sought. By doing this one assumes that the feature space is a metric space, i.e. there exist some function $d(\mathbf{x}, \mathbf{y})$ which expresses the distance between two points \mathbf{x} and \mathbf{y} in feature space. A frequently used one is the Mahalanobis distance given as:

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^N (\mathbf{x} - \mathbf{y})^T \Sigma^{-1} (\mathbf{x} - \mathbf{y}) \quad (3.5.21)$$

where Σ is the covariance matrix estimated from the design set. If Σ is the identity, we obtain the Euclidian distance

$$d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^N (\mathbf{x} - \mathbf{y})^2 \quad (3.5.22)$$

If we use the Mahalanobis distance, the region \mathfrak{R} is a hyperellipsoid in feature space, whereas, in the Euclidean case it is a hypersphere. When the Euclidean distance measure is used, some features (the ones with the largest variance across the design set)

tend to dominate this measure. It is therefore useful to normalize the features. If we denote the i^{th} feature from the j^{th} pattern by x_i^j then normalization is performed by

$$x_i'^j = \frac{x_i^j - \bar{x}_i}{\sigma_i} \quad (3.5.23)$$

where \bar{x}_i is the mean and σ_i^2 is the variance of the i^{th} feature defined as

$$\bar{x}_i = \frac{1}{L} \sum_{j=1}^L x_i^j, \quad \sigma_i^2 = \frac{1}{L} \sum_{j=1}^L (x_i^j - \bar{x}_i)^2 \quad (3.5.24)$$

This normalization is unaffected by outliers. These are points in feature space which, due to uncontrollable causes, deviate from the underlying probability density and deteriorate the estimates of \bar{x}_i and σ_i^2 . A solution is to replace \bar{x}_i by the median $med(x_i)$ of x_i^j and σ_i^2 by

$$\sigma_i^2 = \frac{1}{L} \sum_{j=1}^L (x_i^j - med(x_i))^2 \quad (3.5.25)$$

This normalization will be less effected by outliers.

It is now clear that how the k-nn approach can be employed to estimate probability densities. Let us do so for a classification problem again having L design samples in c classes and with L_i samples in class m_i . To classify \mathbf{x} , we find the k samples closest to it and determine the volume V in which these samples reside. Suppose that there are k_i samples of class m_i among that k nearest neighbors. The class likelihoods, the unconditional density and the class priors can be estimated as follows:

$$p(\mathbf{x}|m_i) \approx \frac{k_i}{L_i V}, \quad p(\mathbf{x}) \approx \frac{k}{L V}, \quad p(m_i) \approx \frac{L_i}{L} \quad (3.5.26)$$

Using the Bayes rule as define in eq. (3.5.7) becomes

$$\begin{aligned} \hat{m}(\mathbf{x}) &= m_i \text{ if } p(\mathbf{x}|m_i)p(m_i) \geq p(\mathbf{x}|m_j)p(m_j), \quad \forall j \\ &\Rightarrow k_i > k_j \quad \forall j \end{aligned} \tag{3.5.27}$$

This leads to a very simple classification procedure, i.e. the sample \mathbf{x} should be assigned to that class to which most of its k nearest neighbors belong. Since this rule is derived from the Bayes rule, it is optimal in the sense that it minimizes the conditional risk. However, this last statement has to be relaxed. The loss is minimized according to the quality of the estimates of the class likelihoods and priors and we may expect a slightly higher error rate for the k -nn classifier due to inaccuracies of the estimates in eq (3.5.26). In fact, an asymptotic analysis shows that the k -nn classifier converges to the Bayes classifier, i.e. the one designed with the true densities and thus having minimal error, as $L \rightarrow \infty$ while k/L remains finite.

A question is the choice of k for a given design set. Fukunaga [92] studied this problem and found that the optimal k depends heavily on the form of the class likelihoods. Trial and error is often the only way to find out a suitable k for a given problem. The number of the nearest neighbors i.e. k should be odd in order to avoid ties, and it should be kept small, since its large value tends to create misclassifications unless the individual classes are well-separated. The performance of a k -nn classifier is always at least half of the best possible classifier for a given problem. One of the major drawbacks of k -nn classifiers is that it needs all available data. This may lead to considerable overhead, if the training data set is large.

3.6.5 Error Estimation

The design of a classifier is based on the performance of the classifier. A good classifier should accurately classify all the possible patterns for which it is designed. To evaluate the performance the sampled used should be different to the used for design purpose. This is the way of training the generalization ability of the classifier. It seems appropriate to quantify classifier performance by the Bayes error rule given in eq.(3.5.8) , which expresses the expected value of misclassification. However, the evaluation of the integrals in eq. (3.5.8) is mostly difficult if not impossible to perform. Furthermore, using this quantity assumes that the Bayes decision rule has been implemented based on the accurate probability densities. Direct computation of eq. (3.5.8) as a measure for classifier performance thus doesn't seem to be feasible in practice. In analogy to the Bayes error, let us define the true error rate E_C as the probability of misclassification associated with the classifier C. Estimation of this quantity is usually performed by error counting. In this case, a collection of data samples not used for training (the test set) is presented to the classifier and the percentage of samples classified falsely is determined. Suppose that there are S_t test samples and \hat{h}_i be a variable which takes the value 0 if the i^{th} test sample is classified correctly and 1 otherwise. The classification performance of the classifier is then estimated by the (observed) error rate:

$$\hat{E}_C = \frac{1}{S_t} \sum_{i=1}^{S_t} \hat{h}_i \quad (3.5.28)$$

E_C depends on the test set and is therefore itself a random variable. Therefore this estimate should be accompanied by a confidence interval which reflects its accuracy. For a large enough test set ($S_t > 30$) a 95% confidence interval is given by [93]:

$$\widehat{E}_c - \widehat{f}(\widehat{E}_c) \leq E_c \leq \widehat{E}_c + \widehat{f}(\widehat{E}_c) \quad (3.5.29)$$

with

$$\widehat{f}(\widehat{E}_c) = 1.96 \sqrt{\frac{\widehat{E}_c(1-\widehat{E}_c)}{S_t}} \quad (3.5.30)$$

In practice, only a finite number L of data samples is available. Using all samples for designing and afterwards for testing the classifier (the re-substitution method) gives a negative bias to the error rate since test and design set are not independent. Subdividing the available data into a design set (containing L_d samples) and a test set (L_t samples, $L_d + L_t = L$) and estimating the error rate as above is called the holdout method.

The holdout method is not very economical since part of the data is not used for classifier design. This problem is solved by the leave-k-out method. In this case the k samples are used as a test set, while the remaining $(L - K)$ samples are used for design. Next, the K test samples are added to the design set from which K other samples are now used for testing. This procedure is repeated until each available sample is exactly used once for testing. It is clear that in this manner a large design set and at the same time an independent test set is available. The drawback is that the classifier needs to be designed several times ($L=K$ times).

3.6.6 Dimensionality Reduction in Classifier

It may seem logical that using more features extracted from the patterns always leads to a better characterization and thus a better classifier with a lower error rate. However, in practice, the contrary is observed. For a given problem the error rate initially drops with increasing number of features, but at certain point the error rate saturates or even

risks with the use of additional features. This phenomenon is called the *curse of dimensionality*. The origin of this phenomenon is the fact that classifier design relies on the inference of statistical properties.

Reducing the dimensionality to enhance the performance of the underlying technique is a popular solution to the curse of dimensionality [94][95][96]. There are two main reasons to keep the dimensionality of the pattern representation to as small as possible: measurement cost and classification accuracy. A limited yet salient feature set simplifies both the pattern representation and the classifiers that are built on the selected representation. Consequently, the resulting classifier will be faster and will use less memory. Moreover, a small number of features can alleviate the curse of dimensionality when the number of training samples is limited. On the other hand, a reduction in the number of features may lead to a loss in the discrimination power and thereby lower the accuracy of the resulting recognition system. It is important to make a distinction between feature selection and feature extraction. The term feature selection refers to algorithms that select the best subset of the input feature set. Methods that create new features based on transformation or combination of the original feature set are called feature extraction algorithms.

The main issue in dimensionality reduction is the choice of a criterion function. A commonly used criterion is the classification error of a feature subset. One has to be careful, since the classification error itself cannot be reliably estimated when the ratio of sample size to the number of features is small. In addition to the choice of a criterion function, we also need to determine the appropriate dimensionality of the reduced feature space.

3.6.7 Choice of a Classifier

Once a feature selection or classification procedure is decided properly, the choice of a classifier is a difficult problem to tackle with, for the recognition of a particular pattern. This is by no means a trivial problem and cannot be answered straightforward. An interesting study on this subject is described in [97], which contains the results of an experiment in which several classifiers are used on different types of problems (images, financial and biological data, etc.). The main conclusion is that

- 1) Performances of different classifiers on same data set may differ significantly.
- 2) If one classifier performs well on one data set, it can perform rather badly on another.

The choice of a classifier is thus data-dependent and there is no such thing as “the best classifier”.

In [97] the comparison of the classifiers on the image data set is given. A striking conclusion made by the researchers is that “k-nn is best for images”. This statement is supported by the fact that (on average) the k-nn classifier yields lowest error rates. Furthermore, the k-nn classifier requires no training. However, it requires lot of memory since all design patterns must be stored which can also slow down the classification. The LVQ network gave comparable results to the k-nn classifier (somewhat higher error rates) but requires less memory. On the average over all data sets, it was observed that neural nets were slightly better. However, the marginal improvement in results they provide over other classifiers goes at the expense of a much more laborious designing stage. The number of layers, neurons with their activation functions and connectivity must be specified. A

learning rule and a training time must also be specified. Because of these arguments, we have found that the k-nn classifier was a convenient classifier which will be used several times in this thesis for the comparison of feature sets.

However, all considerations made in this chapter departed from an already constructed set of features extracted by some means from the available patterns. Given this feature set, the lowest error one can achieve is the Bayes error given in eq (3.5.8), which depends on the class likelihoods. To improve the classification (i.e. to lower the error), it is necessary to construct other features so that the likelihoods overlap less and consequently lead to a lower Bayes error down. This leads to the following conjecture: If a good feature set is available, then any classifier will do the job, otherwise, even the most sophisticated classifiers will fail to solve a pattern recognition problem.

3.7 Hidden Markov Model (HMM)

An HMM is a doubly stochastic process, In this case the with an underlying stochastic process is not observable, but can only be observed through another set of stochastic processes that produce the sequence of observed symbols. HMMs are a set of statistical models used to characterize the statistical properties of a signal. It is a statistical method that uses probability measures to model sequential data represented by sequence of observation vectors.

In other words, HMM is a finite set of states, each of which is being associated with its respective probability distribution. Transitions among the states are governed by a set of probabilities called transition probabilities. In a particular state an observation can be generated with the probability given by the associated probability distribution.

An HMM is characterized by the following parameters.

1. The number of observation symbols in the alphabet is finite for the discrete alphabet size and is denoted by O . The observation symbols correspond to the physical output of the system being modeled. If the observations are continuous then O is infinite.
2. The set of state transition probabilities, $A = \{a_{ij}\}$ where

$$a_{ij} = p\{q_{t+1} = j | q_t = i\}, \quad 1 \leq i, j \leq N \quad (3.6.1)$$

where q_t denotes the state at time t and N is the total number of available states of the model.

Transition probabilities follow the general stochastic constraints given as

$$a_{ij} \geq 0, \quad 1 \leq i, j \leq N \quad (3.6.2)$$

and

$$\sum_{j=1}^N a_{ij} = 1, \quad 1 \leq i \leq N \quad (3.6.3)$$

3. The probability of observing some specific symbol from a specific state,

$B = \{b_j(k)\}$ such that

$$b_j(k) = p\{\mathbf{o}_t = v_k | q_t = j\}, \quad 1 \leq j \leq N, 1 \leq k \leq O \quad (3.6.4)$$

where v_k denotes the k^{th} observation symbol in the alphabet and \mathbf{o}_t is the vector of outputs up to time t .

Following are the stochastic constraints which must be satisfied

$$b_j(k) \geq 0, \quad 1 \leq j \leq N, 1 \leq k \leq O \quad (3.6.5)$$

and

$$\sum_{j=1}^N b_j(k) = 1, \quad 1 \leq j \leq N \quad (3.6.6)$$

4. The initial state distribution is given by the vector, $\boldsymbol{\pi} = \{\pi_i\}$ where

$$\pi_i = p\{q_0 = i\}, \quad 1 \leq i \leq N \quad (3.6.7)$$

which is the probability of starting the process from a specific state.

A complete specification of an HMM requires specification of the three probability measures \mathbf{A} , \mathbf{B} and $\boldsymbol{\pi}$. Therefore, the compact notation $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ is used to denote the complete parameter set of HMM with discrete probability distributions.

3.7.1 Assumptions in the Theory of HMMs

Following are the assumptions made in the theory of HMMs for the sake of mathematical convenience and to avoid the computational complexity:

Markov assumption: As given by the definition of HMMs transition probabilities are defined as

$$a_{ij} = p\{q_{t+1} = j | q_t = i\} \quad (3.6.8)$$

In other word it is assumed that the next state is dependent only upon the current state. This is called the Markov assumption and due to this one step memory the resulting model becomes a first order HMM.

Stationarity assumption: It is assumed that state transition probabilities are independent of the time i.e. transition probabilities do not change with time. It can be expressed as

$$p\{q_{t_1+1} = j | q_{t_1} = i\} = p\{q_{t_2+1} = j | q_{t_2} = i\} \quad (3.6.9)$$

for any t_1 and t_2 .

Independence outputs assumption: Under this assumption, the current observed output o_t is statistically independent of the previously observed, o_{t-1} . We can formulate this assumption mathematically, considering a sequence of observations, $\mathbf{O} = \mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_T$.

Then according to the assumption for an HMM λ ,

$$p\{\mathbf{O}|q_1, q_2, \dots, q_T\} = \prod_{t=1}^T p(\mathbf{o}_t|q_t, \lambda) \quad (3.6.10)$$

3.7.2 Three Basic Problems of HMMs

Once we have generated an HMM, there are three basic problems of interest, evaluation problem, learning problem and decoding problem. Since only the first two problems are being exploited in this dissertation therefore, their statements along with their solutions are being mentioned below.

Evaluation Problem: Given an HMM λ and a sequence of observations $\mathbf{O} = o_1, o_2, \dots, o_T$, what is the probability that the observations are generated by the model, i.e. what is $p\{\mathbf{O}|\lambda\}$?

Some details about the solution of this problem are given as follow. First of the all, we define a forward variable, $\alpha_t(i)$ which is the probability of the partial observation sequence o_1, o_2, \dots, o_t when the process terminates at the state i , i.e.

$$\alpha_t(i) = p(o_1, o_2, \dots, o_t, q_t = i|\lambda) \quad (3.6.11)$$

The following recursion relationship holds for this forward variable

$$\alpha_{t+1}(j) = b_j(o_{t+1}) \sum_{i=1}^N \alpha_t(i) a_{ij} \quad (3.6.12)$$

where $1 \leq j \leq N$ and $1 \leq t \leq T-1$. Using the above relationship we can find out $\alpha_T(i), 1 \leq i \leq N$. Then the required probability $p(O|\lambda)$ is given as:

$$p(\mathbf{O}|\lambda) = \sum_{i=1}^N p\{\mathbf{O}, q_T = i|\lambda\} = \sum_{i=1}^N \alpha_T(i) \quad (3.6.13)$$

The above probability can also be solved by defining backward variable $\beta_t(i)$

which is the probability of the partial observation sequence $o_{t+1}, o_{t+2}, \dots, o_T$, given that the current state is i . It is written mathematically as follow

$$\beta_t(i) = p(o_{t+1}, o_{t+2}, \dots, o_T | q_t = i, \lambda) \quad (3.6.14)$$

Then $p(O|\lambda)$ is calculated in terms of forward and backward variables as follow

$$p(O|\lambda) = \sum_{i=1}^N p\{O, q_T = i|\lambda\} = \sum_{i=1}^N \alpha_T(i) \beta_T(i) \quad (3.6.15)$$

The learning Problem: Given an initial model λ and a set of observation sequences $\mathbf{O} = \mathbf{o}_1, \mathbf{o}_2, \dots, \mathbf{o}_L$, how can we re-estimate the model parameters $\{\mathbf{A}, \mathbf{B}, \boldsymbol{\pi}\}$ so as to increase the likelihood of generating this set of sequences, where L is the total number of observation sequences.

There are typically two methods to solve the learning problem. One is the Baum Welch algorithm and the other is the gradient based algorithm. Since Baum Welch algorithm has been used in this dissertation, hence, some of its details are being presented.

First of all two variables are defined. The first variable is $\xi_t(i, j)$ which is the probability of being in state i at time $= t$ and in state j at time $= t+1$

$$\begin{aligned}
\xi_t(i, j) &= p(q_t = i, q_{t+1} = j | O, \lambda) \\
&= \frac{p(q_t = i, q_{t+1} = j, O | \lambda)}{p(O | \lambda)} \\
&= \frac{p(q_t = i, q_{t+1} = j, O | \lambda)}{\sum_{i=1}^N \sum_{j=1}^N p(q_t = i, q_{t+1} = j, O | \lambda)}
\end{aligned} \tag{3.6.16}$$

The second variable is $\gamma_t(i)$ which is the *posterior probability* is given as

$$\begin{aligned}
\gamma_t(i) &= p(q_t = i | O, \lambda) = \frac{p(q_t = i, O | \lambda)}{p(O | \lambda)} \\
&= \frac{p(q_t = i, O | \lambda)}{\sum_{i=1}^N p(q_t = i, O | \lambda)} = \frac{\alpha_t(i) \beta_t(i)}{\sum_{i=1}^N \alpha_t(i) \beta_t(i)}
\end{aligned} \tag{3.6.17}$$

According to the Baum Welch algorithm given an initial model $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$. Then forward and back variables α_s and β_s are calculated. Then calculate ξ_s and γ_s using (3.6.16) and (3.6.17). Finally the HMM parameters are updated according to the following equations

$$\bar{\pi} = \gamma_1(i), \quad 1 \leq i \leq N \tag{3.6.18}$$

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i, j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \quad 1 \leq i \leq N, \quad 1 \leq j \leq N \tag{3.6.19}$$

$$\bar{b}_j(k) = \frac{\sum_{t=1}^T \gamma_t(j)}{\sum_{t=1}^T \gamma_t(j)}, \quad 1 \leq i \leq N, \quad 1 \leq j \leq O \tag{3.6.20}$$

Chapter 4

Texture Analysis using Radon Transform and Hidden Markov Models

4.1 Introduction

This chapter covers three proposed schemes to extract features of the textures using RT. All these schemes carried out training of HMM, one for each texture. Then, finally extensive testing is carried out using the evaluation problem of HMM. Simulations and results are given with each scheme.

4.2 Background of Invariant Texture Analysis

One usually looks for texture analysis methods that are translation, rotation and scale invariant [4]. Wavelet transform (WT) and wavelet packets have played an important role in texture description and analysis [98]. Wavelets provide the appealing property of representing texture compactly both, in frequency and time domain. However, ordinary WT, which had been used for texture analysis [55][78], are not rotation-invariant [55][99][78][100]. Some attempts have been made towards rotation invariant texture analysis using WT [101][102][63]. The reason is that, textures have different frequency

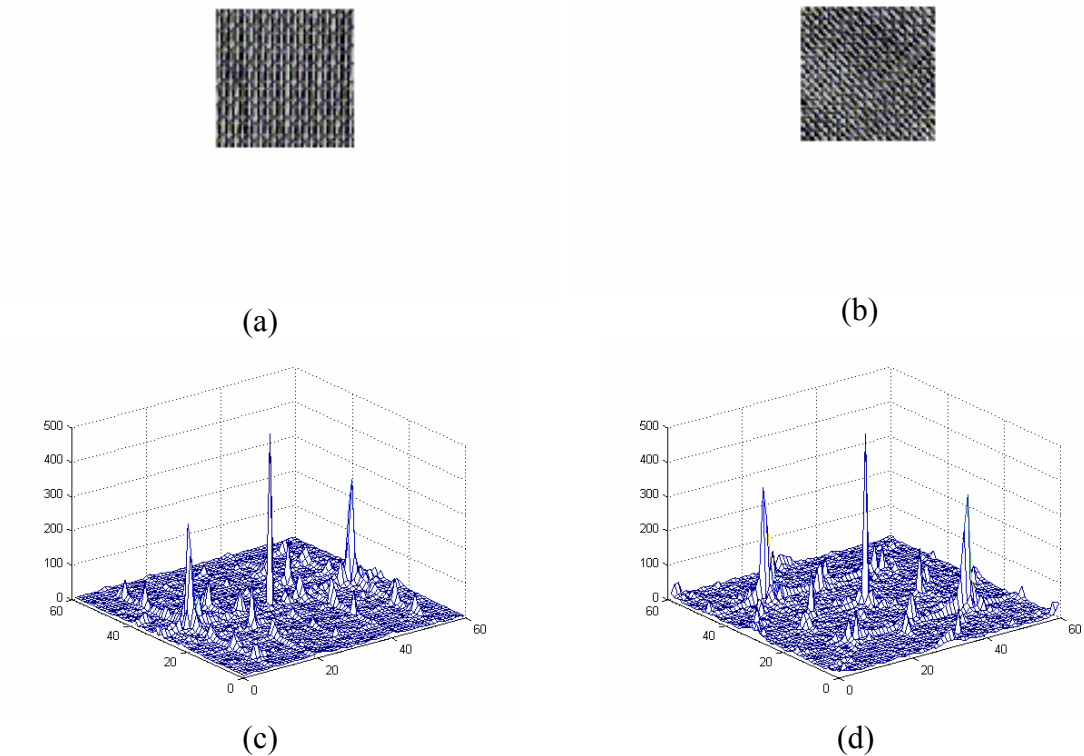


Fig. 4.1 (a) Anisotropic texture (D53), (b) Anisotropic texture (D53) rotated at 45° , (c) Fourier Transform of (a), (d) Fourier Transform of (b)

components along different directions. Ordinary WT captures variation only along vertical, horizontal, and diagonal directions. This happens because of the separability of the basis functions. Fig. 4.1 (a) and (b) show a directional (anisotropic) texture sample from Brodatz album (D53) in two different orientations, and their Fourier transforms (FT) in Fig. 4.1 (c) and (d), respectively. The FT rotates as the image is rotated, as shown in the Fig. 4.1 (c) and (d) it changes significantly when the image is rotated. However, some textures have no specific direction, which are called isotropic textures. This means that the frequency components of the texture do not change significantly at different orientations. In other words, its Fourier transform is almost circularly symmetric. Therefore, the wavelet features

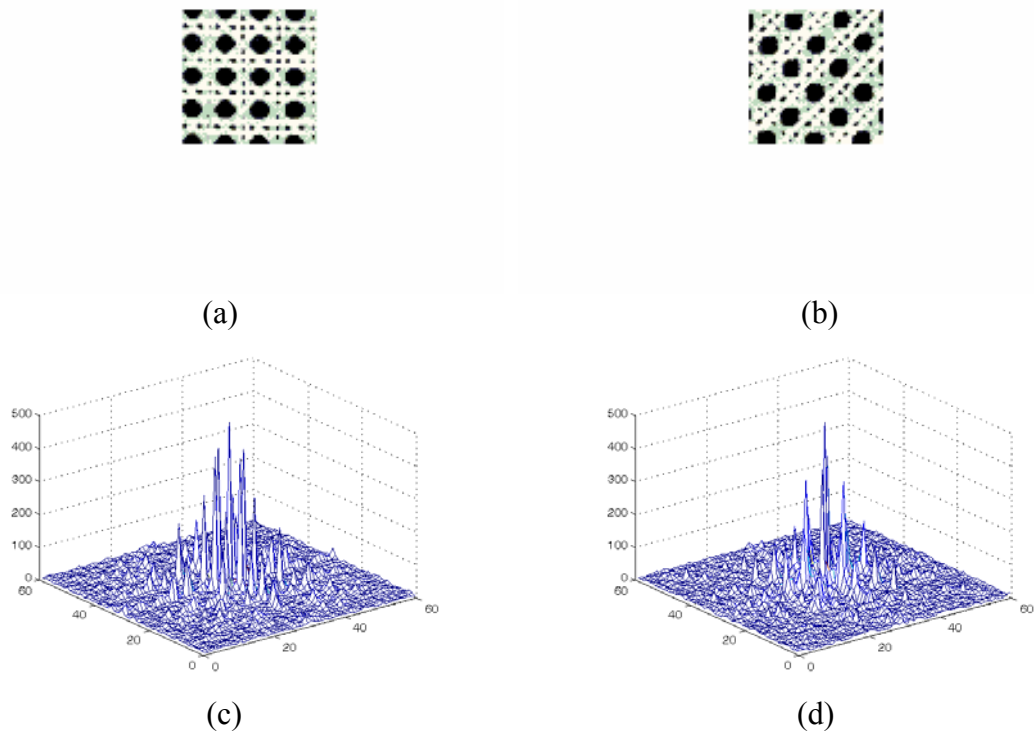


Fig. 4.2 (a) Isotropic texture (D101), (b) Isotropic texture (D101) rotated at 45o, (c) Fourier Transform of (a), (d) Fourier Transform of (b)

are approximately invariant to rotation. Fig. 4.2 (a) and (b) are two different orientations of the same texture (D101) and (c) and (d) are their FTs, respectively. It means that the FT does not change significantly when the image is rotated.

Mao and Jain [103] have used rotation-invariant symmetric autoregressive random field model in which neighborhood points of a pixel are defined on several circles around it. This approach, however, overlooks the global information of the texture. Some approaches have used HMM [104][53][66]. Chen and Kundu [53] decomposed the image into subbands using quadrature mirror filter, and then modeled these subbands by an HMM. Unfortunately, as the number of classes (textures) increases, the performance deteriorates. Do and Vitterli [66] have used a steerable wavelet domain HMM along with a maximum

likelihood solution for model parameters, but they have experimented on a limited scale and used only thirteen images from Brodatz album. Khouzani and Zadeh [60] have also proposed preprocessing step to make the analysis invariant to rotation. They utilize the RT to convert the rotation into translation and then apply translation invariant WT to create rotation invariant features.

In this chapter, we have presented a new technique which uses RT for feature extraction and one-dimensional (1-D) HMM, which is trained on these features and hence used for classification. RT is suitable for extracting the rotation invariant features for ordinary texture as well as for complex texture. Due to the directional properties of RT, we have proposed to use it to capture the directional information of each texture with different orientations. The RT has two components. First component is the angle of line with y-axis and it is denoted as θ . The second component is the perpendicular offset, s . Here, we will discuss two cases for feature extraction of texture images. In the first case ($s = 0$), the former component of the RT is considered to be varying, while the later component is suppressed. We shall call this RT as modified RT (MRT). In the second case ($s \neq 0$), both components, s and θ , are taken as varying. For second case there are again two subcases. In the first one we take the values of RT as features of the texture, while in the second subcase we take the variances of the different projections of RT and consider them as feature vectors. These feature vectors, attained by any scheme, are considered as observation vectors in order to train 1-D HMM to give us representation of this texture. Compared to [104][53][66], we have used only 1-D HMM instead of 2-D HMM. For testing purpose, we picked up any one of these textures with an arbitrary orientation. We then found out its observation vector using RT and calculated the best match between this observation vector

and HMM. We have given comparison of the percentage of the correct classification (PCC) with some other schemes in the literature.

4.3 Texture Analysis using RT for Case $s = 0$

An important area of research within pattern recognition is to explore the techniques which are employed to extract rotation-invariant features of the texture images. The application of RT has potential to use it for this purpose. The RT of 2-D function $f(x, y)$ is defined as

$$g(s, \theta) \triangleq R[f(x, y)] = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} f(x, y) \delta(x \cos \theta + y \sin \theta - s) dx dy \quad (4.3.1)$$

where θ is the angle formed by the line along which the integral is calculated, and s is the perpendicular distance (offset) of this line from the origin as shown in Fig. 4.3. Thus $g(s, \theta)$ is simply 1-D projection of $f(x, y)$ at an angle θ with distance s from the origin. RT is always applied on a disk shape area selected at the centre of the image.

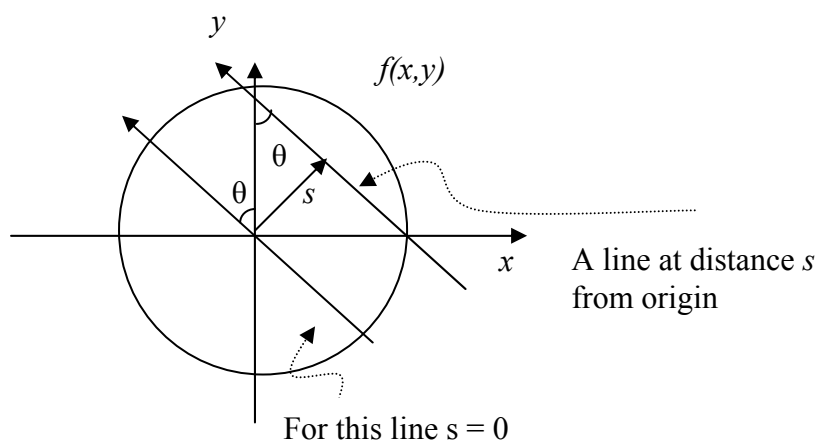


Fig. 4.3 Radon Transform

4.3.1 Feature Extraction using Radon Transform

When, we take $s = 0$, it means suppressing the offset. Thus, the RT of the texture, for any orientation, is projection of lines at different angles passing through the centre of the disk shape texture. It means that only θ varies from 0° to 180° , with discrete steps of $\Delta\theta$. Almost each line contains the same number of pixels. Hence, there is no need to average out each projection. This gives the feature vector of length L depending upon the discrete step $\Delta\theta$. Each arbitrary orientation of a texture gives a corresponding vector. Obviously if we take L arbitrary orientations we will get L vectors given as

$$\mathbf{O} = [\mathbf{o}_1 \ \mathbf{o}_2 \ \cdots \ \mathbf{o}_L]^T \quad (4.3.2)$$

This formulates an observation sequence for one HMM. This way of extracting features of the texture image has the following disadvantages:

- There is no possibility of inverting the transformation to obtain something approximating the original texture.
- The suppression of s is potentially serious for more generalized classification. This is due to the choice of texture centre, which have a great effect on the features extracted. Thus this problem loses the translational invariance.

The selection of $\Delta\theta$ may affect the dimensions of the feature vector but it is not an obstacle to classification. On the basis of compromising to fix the choice of texture centre and avoiding the inverse transformation, the extracted features of the texture have following advantages:

- Robust from classification point of view, if the translational invariance is not an issue.
- Process is computationally less demanding.

4.3.2 Training using Hidden Markov Model

One HMM can be trained on feature vectors obtained from RT. We have an observation sequence of L vectors given as $\mathbf{O} = [\mathbf{o}_1 \mathbf{o}_2 \cdots \mathbf{o}_L]^T$, where is the L number of orientations of a texture. Each \mathbf{o}_i is a column vector representing feature vector of i^{th} orientation. T stands for the transpose of a matrix. This sequence of observations is modeled by a unique HMM whose salient features [105] are discussed in section 3.7.

Each texture with different orientations is represented by a separate model which is usually denoted by $\lambda_i = (\mathbf{A}_i, \mathbf{B}_i, \boldsymbol{\pi}_i)$ $i = 1, 2, \dots, M$ where M is the number of Brodatz textures.

This model is trained for the separate texture independently. Therefore, we have M HMM models for M number of textures. We choose $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ such that $p(\mathbf{O}|\lambda)$ is locally maximized using an iterative procedure such as Baum–Welch algorithm. For this purpose re-estimation of HMM parameters is required. Two variables $\xi_t(i, j)$ and $\gamma_t(i)$, defined in section 3.7, are used for the re-estimation of HMM parameters. A set of reasonable re-estimation formulas for $\boldsymbol{\pi}, \mathbf{A}$ and \mathbf{B} is given by eqs. (3.6.18), (3.6.19) and (3.6.20). If we define the current model as $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ then the re-estimation model will be defined as $\bar{\lambda} = (\bar{\mathbf{A}}, \bar{\mathbf{B}}, \bar{\boldsymbol{\pi}})$, which is determined by the above re-estimation formulas. It was proved in the Baum-Welch algorithm that either the initial model λ defines a critical point of the likelihood function, in which case $\bar{\lambda} = \lambda$, or model $\bar{\lambda}$ is more likely than model λ in the sense that $p(\mathbf{O}|\bar{\lambda}) > p(\mathbf{O}|\lambda)$. In other words, we have found a new model $\bar{\lambda}$ from which the observation sequence is more likely to have been produced. Based on this procedure, if we iteratively use $\bar{\lambda}$ instead of λ and repeat the re-estimation calculation, we can improve

the probability of observation sequence, \mathbf{O} , being observed from the model until some limiting point is reached.

The training is performed off line and only once. For each new texture, a new class is considered, and a new model is trained. Fig. 4.4 (a) is a schematic diagram giving all the steps for training phase of each texture with different orientations. There is no need to retrain the others. A particular tolerance factor is given to ensure the proper training and convergence of the HMM.

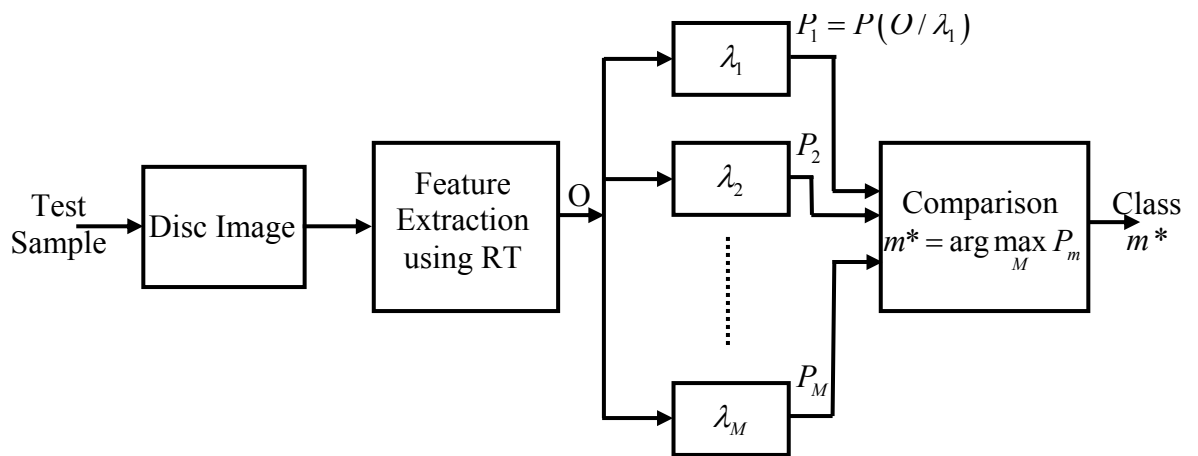
4.3.3 Testing and Classification

Once all the HMM's have been trained, the classification stage is straightforward. The preprocessing and feature extraction are repeated in the testing and classification phase as shown in the Fig. 4.4 (b). In the testing, we do not need to find out the complete observation matrix $\mathbf{O} = [\mathbf{o}_1 \mathbf{o}_2 \cdots \mathbf{o}_L]^T$. Any arbitrary orientation of any texture can be taken and its relevant feature vector, \mathbf{o}_i , can be found using RT. We have used the evaluation problem of HMM for testing and classification purpose. The evaluation problem can be enunciated as follows: *“given a model and a feature vector, how do we compute the probability that the feature vector was produced by the model?”*[105]

We can also view the problem as one of scoring how well a given model matches a given observation vector. The latter viewpoint is extremely useful. If we consider the case in which we are trying to choose among several competing models, the solution to evaluation problem allows us to choose the model which best matches with the observations.



(a)



(b)

Fig. 4.4 (a) Block diagram for the training phase and (b) Block diagram for the testing phase of the proposed model

We have developed one HMM model for each type of texture. Therefore, we will have M kinds of models denoted by $\lambda_1, \lambda_2, \dots, \lambda_M$. First we find out the feature vector \mathbf{o}_i for an unknown texture and then carry out the evaluation problem in HMM i.e. $P(\mathbf{o}_i|\lambda_m)$ for $m = 1, 2, \dots, M$. The class of the texture can be found from $P(\mathbf{o}_i|\lambda_m)$, as given below

$$m^* = \arg\{\max_m P(\mathbf{o}_i|\lambda_m)\} \quad (4.3.3)$$

This unknown texture belongs to class m^* .

4.3.4 Simulation and Results

We have taken first 60 textures from Brodatz album (D01-D112), as shown in Fig. 6.9, for the evaluation of the proposed algorithm. Each texture is treated as a class. The performance of algorithms have been evaluated using Percentage of Correct Classification (PCC) as a figure of merit and is defined as

$$PCC = \left(\frac{T_s - M_s}{T_s} \right) \times 100 \quad (4.3.4)$$

where T_s is total number of samples and M_s is number of misclassified samples.

After selection of any texture, its RT is taken at constant discrete steps of 2^0 from 0^0 to 180^0 . Length of each feature vector is thus 90 and is kept same throughout the simulations. Number of orientations for each texture is taken as L , which is also the number of feature vectors to train an HMM model. These different orientations are taken at random angles. In simulations $L = 20, 40, 60, 80$ and 100 have been tried. The number of states has

been given as N . In simulations the number of hidden states has been taken as $N = 3, 4, 6, 8$ and 10 has been tried to see their effect on PCC.

We have used all 60 textures at 20 arbitrary orientations which make up test set as 60×20 (1200 in number) for testing purpose. We observe that as the number of feature vectors L increases, on which the HMM has been trained, the model is more robust and gives better value for PCC. The states in HMM do not have any explicit physical meaning. One cannot say that increase or decrease of states would result in a better model in terms of PCC. However, the Table 4.1 shows that $N = 5$ seems to be the best choice in terms of PCC for larger values of L (60, 80, and 100). Although, one may increase the number of training feature vectors in order to get a better HMM for any texture but it becomes computationally cumbersome. The best value of PCC achieved is 98.25% using the proposed algorithm as shown in Table 4.1. The results proposed by Chen and Kundu [53] are also shown in Table 4.2. The results of Khouzani and Zadeh [60] are given in Table 4.3. The comparison of the results shows that the best result of Chen and Kundu [53] is 93.33%, Ojala et. al. [106] is 95.8% and Khouzani et. al. [60] is 97.9%. Therefore, the results of proposed method, i.e. 98.25%, show better performance of the method in terms of PCC. This technique, for $s = 0$, is computationally light especially while extracting the features of a texture. Moreover, it is has been tested on 60 textures while the schemes compared with have tested on 25 textures. However, the proposed scheme is only rotation invariant, but not translation or gray scale invariant.

Table 4.1 PCC of 1200 test samples using proposed RT based method (case $s = 0$) for different number of feature vectors L and number of hidden states N

$L \backslash N$	3	4	5	6	8	10
20	90.50%	87.50%	84.00%	85.33%	85.33%	85.08%
40	95.75%	95.50%	95.00%	94.08%	93.41%	94.58%
60	96.91%	96.41%	96.83%	95.83%	95.50%	95.41%
80	97.91%	97.33%	97.66%	97.00%	96.83%	96.50%
100	98.16%	97.83%	98.25%	97.83%	97.66%	97.08%

Table 4.2 PCC of 10 textures from Brodatz Album using method proposed by Chen and Kundu [53], for both Nonstationary and Stationary Transition HMM.

State no.	Nonstationary Transition HMM			Stationary Transition HMM		
	$\alpha=1.0, \beta=0$	$\alpha=0.5, \beta=20$	$\alpha=0.5, \beta=70$	$\alpha=1.0, \beta=0$	$\alpha=0.5, \beta=20$	$\alpha=0.5, \beta=70$
6	91.67%	91.67%	91.67%	83.33%	83.33%	83.33%
8	93.33%	91.67%	91.67%	86.67%	86.67%	86.67%
10	90.00%	90.00%	90.00%	85.00%	86.67%	86.67%
12	91.67%	91.67%	91.67%	86.67%	86.67%	86.67%

Table 4.3 PCC of 25 textures from Brodatz Album using the method proposed by Khouzani and Zadeh [60] for different k values of k -nn Classifier.

Wavelet Bases	k			
	1	3	5	7
db ₂	96.70%	97.60%	97.90%	97.40%
db ₄	97.40%	97.70%	97.90%	97.90%
db ₆	96.40%	97.70%	97.60%	97.80%
db ₁₂	96.30%	96.60%	96.90%	96.90%

4.4 Texture Analysis using RT for Case $s \neq 0$

Two dimensional RT given in eq (4.3.1) is again utilized for texture analysis for the case $s \neq 0$. In this case both the components of RT are being considered as varying. Thus the texture analysis carried out is rotation and translation invariant.

4.4.1 Feature Extraction using RT

The RT of any texture is taken along lines at different orientations θ and different offsets from the origin, s . The angle θ is varying from 0° to 180° in discrete steps of $\Delta\theta$. In this case there will be different number of pixels for different values of s for a particular angle θ . For small value of s the line is close to centre and so there will be a large number of pixels on that line. For large value of s , line is far from the centre and so has less number of pixels on it as shown in Fig. 4.5. The total sum of grayscale values of pixels on each line is divided by the total number of pixels on that line. This gives us an average of gray scale values on a particular line with parameters (s, θ) . Only one orientation of a texture gives us the observation sequence

$$\mathbf{O} = [\mathbf{o}_1 \ \mathbf{o}_2 \ \cdots \ \mathbf{o}_L]^T \quad (4.4.1)$$

There is one observation vector for each θ . L is the number of vectors, not the number of orientations as in case of $s = 0$. In fact L will be equal to the total number of discrete values of θ between 0° and 180° . If we take $\Delta\theta = 2$, then L , the number of sequence vectors, will be 90. The length of each vector is equal to the number of discrete steps taken for offset Δs . Now these features are translation invariant which was not true in the previous case. However, these features have disadvantage of being computationally heavier. The distance s changes in discrete steps of $\Delta s = 10^\circ$, which is in fact 10 pixels. For

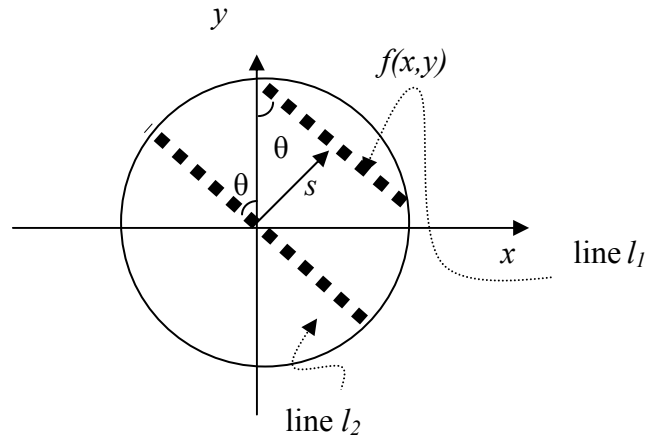


Fig. 4.5 Pixel representation on lines

any fixed value of θ , there will be projections of the image along different lines at different s from the origin. This will formulate one feature vector of the texture. Number of extracted feature vectors for a particular texture is denoted by L . For every texture we have L number of feature vectors or observation sequence denoted as $\mathbf{O} = [\mathbf{o}_1 \mathbf{o}_2 \cdots \mathbf{o}_L]^T$.

4.4.2 Training of HMM

The observation sequence formulated in the above section is used to train an HMM model with the help of Baum-Welch algorithm [105]. The model parameters, $\lambda_i = (\mathbf{A}_i, \mathbf{B}_i, \boldsymbol{\pi}_i)$, are adjusted in order to maximize the probability of the given sequence of observation sequence, $p(\mathbf{O}|\lambda_i)$. Here i denotes the number of the texture for which HMM is being trained. At the end, the number of HMMs is equal to the number of textures, M .

4.4.3 Testing and Classification

We use evaluation problem [105] of HMM for testing and classification purpose. In testing, we have found out the observation sequence $\mathbf{O} = [\mathbf{o}_1 \mathbf{o}_2 \cdots \mathbf{o}_L]^T$ by using RT on the texture being tested for classification. Then calculated $p(\mathbf{o}_j | \lambda_m)$ for all $j = 1, 2, \dots, L$ and $m = 1, 2, \dots, M$. Then the class is found by using

$$m^* = \arg \left\{ \max_m \sum_{j=1}^L P(\mathbf{o}_j | \lambda_m) \right\} \quad (4.4.2)$$

This unknown texture is either texture m^* of the Brodatz album, or it is closest to the m^* texture.

4.4.4 Simulations and Results

The proposed method was implemented on sixty textures from Brodatz album (D1-D60). The performance criterion is PCC which is defined in eq (4.3.4). The algorithm has been tested by training HMM for different number of feature vectors ($L = 36, 45, 60, 90$) which are achieved by using discrete steps of as ($5^\circ, 4^\circ, 3^\circ, 2^\circ$), respectively. Different number of states ($N = 3, 4, 5, 6, 8, 10$) have been used. The main thrust was to find out the optimal values of L and N which give the best PCC. For testing we have used 60 textures at 20 arbitrary orientations to make the test set of 60×20 (1200). Table 4.4 shows that as the number of feature vectors, L , increases, the PCC also increases for any specific hidden state. The best results of 98.80% have been achieved for $L = 90$ and $N = 6$. Results may be further improved but it needs heavy computation to train the HMMs for higher values of L at different states N . These results are better than that of Chen and Kundu [53] (for 10 textures), Ojala et. al. [106] (16 textures), and Khouzani and Zadeh [60] (25 selected

Table 4.4 PCC of 1200 (60×20) test samples using proposed RT based method (case $s \neq 0$) for different number of feature vectors L and number of hidden states N

L \ N	3	4	5	6	8	10
36	86.40%	85.20%	84.90%	83.20%	81.62%	81.45%
45	90.12%	93.45%	95.60%	93.70%	92.45%	90.16%
60	96.92%	97.56%	98.20%	97.60%	93.41%	92.50%
90	97.65%	98.30%	98.50%	98.80%	97.90%	97.56%

textures). These algorithms produce PCC of 93.33%, 95.8% and 97.9%, respectively, as shown in Table 4.1, Table 4.2 and Table 4.3.

4.5 Texture Analysis using Variance of RT for Case $s \neq 0$

In this case the RT of a texture is carried out for various θ and s , but we have not used the values of RT as feature vectors. Instead we have used variance of RT values. The texture analysis shall be rotation, translation and gray scale invariant.

4.5.1 Feature Extraction using Variance of RT.

As in the previous case we carry out RT of a texture for θ varying from 0° to 180° in discrete step of $\Delta\theta$. For any particular value of θ , the projection of RT gives different values for different values of s . We find out the variance $\sigma^2(\theta)$ of these values given by

$$\sigma^2(\theta) = \sum_s (g(s, \theta) - \bar{g}(\theta))^2 / N \quad (4.4.3)$$

where $\bar{g}(\theta) = \sigma^2(\theta) = \sum_s g(s, \theta) / N$ and N is the number of samples in each projection.

We find variance of the samples of these projections for all discrete values of θ to formulate a vector given by

$$\mathbf{o}_i = (\sigma^2(0), \sigma^2(\Delta\theta), \sigma^2(2\Delta\theta), \dots)^T \quad (4.4.4)$$

This will be considered as one feature vector of that texture. Now we shall take L different arbitrary orientations of the texture and find out similar feature vectors which we can denote by matrix \mathbf{O}

$$\mathbf{O} = [\mathbf{o}_1 \ \mathbf{o}_2 \ \dots \ \mathbf{o}_L] \quad (4.4.5)$$

This now becomes the observation sequence on which our HMM shall be trained. In a similar manner we formulate observation sequences, one for each texture.

4.5.2 Training of HMM

Baum-Welch algorithm is used for training purpose. Again we shall train an HMM for each texture and model parameters $\lambda_i = (\mathbf{A}_i, \mathbf{B}_i, \boldsymbol{\pi}_i)$ will be adjusted to maximize the probability $p(\mathbf{O}|\lambda_i)$.

4.5.3 Testing and Classification

In this case we take the RT of the test sample which is a texture at any arbitrary orientation. Formulate the variance vector \mathbf{o}_t . Using evaluation problem [105] of HMM we calculate $p(\mathbf{o}_t|\lambda_m)$ for all $m = 1, 2, \dots, M$. The decision has been taken as

$$m^* = \arg \left\{ \max_{1 \leq m \leq M} p(\mathbf{o}_t|\lambda_m) \right\} \quad (4.4.6)$$

The test texture is either texture m^* of the Brodatz album or closest to the m^* texture.

4.5.4 Simulations and Results

In this case we have used the first twenty five textures of the Brodatz album. The hidden number of states for the HMM has been taken as $N = 3, 4, 5, 6, 8$ and 10 . Since each

orientation of a texture gives us only one feature vector, hence we have used L orientations of each texture to give us L number of feature vectors in order to train an HMM for each texture. We have taken $L = 10, 15, 20, 25$. We can see the trend from the Table 4.5 that as L increases for a fix number of N , the PCC gets better. For $L = 25$ and $N = 5$ we get PCC equal to 100%. This scheme seems to be the best among all the three schemes mentioned in the chapter.

Table 4.5 PCC of 500 (25×20) test samples using proposed RT variance based method (case $s \neq 0$) for different number of feature vectors L and number of hidden states N

L \ N	3	4	5	6	8	10
10	94.3%	94.0%	94.2%	93.8%	92.6%	92.5%
15	96.2%	96.3%	96.0%	95.8%	95.0%	95.2%
20	96.2%	99.0%	98.8%	98.8%	98.4%	98.6%
25	99.8%	99.8%	100%	99.8%	99.8%	99.5%

Table 4.6 Comparison of the best results of the proposed methods with some of the methods from the literature

Methods from the literature			Proposed methods		
Chen and Kundu (10 textures)	Khouzani and Zadeh (25 textures)	Ojala et. al. (16 textures)	Case ($s = 0$) RT based (60 textures)	Case ($s \neq 0$) RT based (60 textures)	Case ($s \neq 0$) RT variance (25 textures)
93.33%	97.90%	95.80%	98.25%	98.80%	100%

4.6 Summary

We have presented three schemes for extracting the features. After feature extraction, the remaining procedure is common for all the schemes. The first scheme (case $s = 0$) is computationally lighter than the other two. Its results are comparable or better than

the other techniques given in [53], [60] and [106]. However, this scheme is rotation invariant.

The second scheme (case $s \neq 0$) uses RT for feature extraction. It is computationally comparable to the first one. This scheme also gives us better results than the above scheme. Moreover, it is translation invariant as well unlike the above scheme.

The third scheme (case $s \neq 0$) uses variance of RT for feature extraction. It is computationally heavier compared to the previous two schemes. Its best result is 100% for $L = 25$ and $N = 5$. Moreover, the texture analysis is rotation, translation and gray scale invariant. It is quite clear from Table 4.6 that all the three proposed schemes have given better results than the other schemes from the literature, although our first two schemes have been tested over 60 textures.

In all these techniques we have not tested their robustness against zero mean additive white Gaussian noise (AWGN). The reason being that since the RT of the discrete image is simply addition of the pixel values, the RT of the noise will turn out to be the mean of the noise which is zero. Thus zero mean AWGN has no effect on the RT of the image. Hence these techniques are robust against this kind noise.

Chapter 5

Texture Analysis using Differential Radon Transform and Hidden Markov Models

5.1 Introduction

This chapter presents yet another approach to provide rotation, as well as, gray scale invariant texture analysis. For this purpose, we have proposed modified form of Radan transform. Ordinary RT is simply the projection of the image along the line (s, θ) , which ends up in adding up the gray scale values of the pixels lying on that line. In the proposed modified RT, the absolute differencing between the adjacent pixels is carried out. Thus we may call it as “Differential Radon Transform” (DRT). By virtue of its definition, the features extracted by using DRT shall be gray scale invariant apart from being rotation invariant.

We proposed three schemes in this approach to extract feature vectors. In the first scheme we use only DRT to extract features. In the second one we used RT as well as DRT with $s = 0$ to give us the features which looked more like low pass, as well as, high pass features of the texture. In the last scheme, we used RT as well DRT but this time keeping the offset $s \neq 0$.

5.2 Differential Radon Transform

The Radon transform (RT) is simply a projection of function $f(x, y)$ on the (s, θ) line as we have discussed in section 4.3.1. If we consider $f(x, y)$ as a discrete image, then the RT is a sum of the gray scale pixels falling on this line inclined at an angle θ with y -axis. This implies that features extracted by using RT are like low pass features of the image which involves simply summing or averaging. These features are definitely not gray-scale-invariant. Thus we propose a modified version of RT. The modified RT of function $f(x, y)$, denoted as $g_D(s, \theta)$, can be defined as

$$g_D(s, \theta) \triangleq \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} |f(x, y) - f(x + \Delta x, y + \Delta y)| \delta(x \cos \theta + y \sin \theta - s) dx dy \quad (5.2.1)$$

where $f(x, y)$ and $f(x + \Delta x, y + \Delta y)$ are the gray scale values of the function at (x, y) and $(x + \Delta x, y + \Delta y)$ respectively, which are constrained to be on the same line inclined at an angle θ with y -axis and at a distance s from the origin. We have taken the absolute differences and integrate them along (s, θ) line. In case of discrete images, the integral changes into summation. We take absolute difference of the gray scale value of every two consecutive pixels along the same line and sum up these differences. Then we may average it over the number of pixels on the (s, θ) line, according to the situation and the type of the problem. The way we take the difference between any two consecutive pixels rather than the sum, the name “differential Radon transform” is fairly suggestive. This DRT thus provides us gray scale invariance, because it only depends upon the difference between the gray scale values and not on the absolute values. We may consider the DRT features more like high pass features of the image.

5.3 Texture Analysis using Differential Radon Transforms and Hidden Markov Model

The features can be extracted using DRT given in eq (5.2.1). The features thus extracted will be called as DRT features. These features were then used to train HMM models. Now, we will discuss the procedure to formulate observation sequence for HMM based on DRT.

5.3.1 Feature Extraction using Differential Radon Transform

The observation sequence is collection of DRT feature vectors of the texture at different orientations. The main difference to calculate DRT features from RT features is that instead of summing up the pixel values along any line with some orientation, we sum up the absolute difference of the consecutive pixel values along that line. Now, we define a filter $H = [-1, 1]$. If we convolve H with the line, l , we get the difference of the consecutive pixel values along that line. Summing up the absolute of these values we get one value for this particular line. The orientation of these lines changes with θ from 0° to 178° with discrete steps of $\Delta\theta = 2^\circ$. For $\Delta\theta = 2^\circ$, we get ninety values for one arbitrary orientation of texture. This gives one feature vector for a particular orientation of a texture. It is obvious that L orientations of a texture will give us L feature vectors, which formulate observation sequence for training of HMM. The features extracted through DRT will ultimately give us gray-scale invariant features as discussed in section 5.2. We have kept $s = 0$ in this case.

5.3.2 Training Hidden Markov Model using DRT Features

We have used the compact notion $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ to represent the HMM. The use of DRT gives us one feature vector for each orientation of a texture. We have an observation

sequence of L vectors given as $\mathbf{O} = [\mathbf{o}_1 \mathbf{o}_2 \cdots \mathbf{o}_L]^T$ for L number of random orientations of a texture. Each \mathbf{o}_i is a column vector representing feature vector of i^{th} orientation. This sequence of observation vectors is used to train an HMM with the help of Baum-Welch algorithm [105]. It is an iterative algorithm that uses the forward and backward probabilities to solve the problem of training by parameters estimation. The model parameters $\lambda_m = (\mathbf{A}_m, \mathbf{B}_m, \boldsymbol{\pi})$ are to be adjusted in order to maximize the probability of that sequence of observation vectors, $p(\mathbf{O}|\lambda_m)$. This algorithm, which is the implementation of the expectation maximization (EM) algorithm [107] in the HMM case, guarantees the model to converge to a local maximum of the probability of observation of training set according to the maximum likelihood estimation (MLE) criterion. This maximum depends strongly on the initial HMM parameters. One model is trained for each texture independently. For M textures we have to build M HMM models. If a new class is added, we need to train for that class only. The training is done off line and once only. The training procedure is explained in Fig. 4.4 (a). A particular tolerance factor is given in order to ensure proper training and convergence of the HMM.

5.3.3 Testing and Classification

The same procedure will be carried out for testing and classification as discussed in section 4.3.3. The difference is based on the features. Here, we are using DRT features instead of RT features. Assumed there are M kinds of models denoted by $\lambda_1, \lambda_2, \cdots, \lambda_M$. The trained model of each texture is available. First, we find the DRT feature vector \mathbf{o}_i for any

unknown texture. Then, we carry out the simple evaluation problem in HMM, i.e., calculate $p(\mathbf{o}|\lambda_m)$ for $m = 1, 2, \dots, M$. At last we will find

$$m^* = \arg \max_m P(\mathbf{o}|\lambda_m) \quad (5.3.1)$$

This unknown texture belongs to class m^* .

5.3.4 Simulation and Results

The proposed algorithms have been implemented on sixty textures from Brodatz album (D01-D112) included isotropic and anisotropic textures as given in Fig. 6.9. The performance of the algorithm has been evaluated by using PCC as figure of merit as defined in eq (4.3.4).

We have obtained DRT of any observation using eq (5.2.1) at constant discrete steps of 2° from 0° to 178° . The feature vectors thus obtained are used to train HMM for $L = 20, 40, 60, 80, 100$ and $N = 3, 4, 6, 8, 10$. We have used all 60 textures at 20 arbitrary orientations to make the test set of 60×20 (1200) as used in RT base technique. Table 5.1 shows that as the number of features (L), on which the HMM has been trained, increases the

Table 5.1 PCC of 1200 (60×20) test samples using proposed DRT based method (case $s = 0$) for different number of feature vectors L and number of hidden states N

L \ N	3	4	5	6	8	10
20	79.17%	75.67%	71.42%	69.75%	65.83%	60.33%
40	90.00%	87.92%	87.17%	85.50%	82.92%	82.92%
60	91.17%	91.00%	91.00%	90.67%	90.58%	88.25%
80	91.08%	91.33%	91.50%	89.92%	90.67%	89.00%
100	93.50%	93.50%	93.67%	94.50%	95.92%	96.33%

PCC also increase for a given number of states. On the other hand, the model produces similar behavior for different number of hidden states, i.e. $N = 3, 4, 6, 8, 10$. The best results are obtained at $L=100$ and $N=10$. Although, one may increase the number of training feature vectors to get a better HMM for any texture, but it becomes computationally cumbersome. It is observed that the RT based algorithm has performed better as compared to DRT based algorithm in terms of PCC.

5.4 Texture analysis using Radon and Differential Radon Transform for the Case $s = 0$

In this section, we have tried to classify the texture on bases of considering the combined behavior of RT and DRT features. In other words, the low pass features of the image and high pass features of the image are being combined to analyze the texture. In this section the case $s = 0$ is considered, which has already been elaborated in section 4.3.

5.4.1 Feature Extraction using RT and DRT

RT and DRT features are being extracted by implementing eq (3.3.1) and eq (5.2.1), respectively. The procedures for extracting both type of features have already been discussed in sections 4.3.1 and 5.3.1. In both cases $s = 0$ and a constant discrete step of $\Delta\theta = 2^\circ$ from 0° to 180° has been taken during the process of feature extraction. Separate observation sequences have been formulated for each texture for training the HMM.

5.4.2 Training using HMM

Once the observation sequence is formulated for each texture on the basis of RT and DRT features, HMMs are trained for each texture, for these observation sequences.

Training procedure for RT based HMM and for DRT based HMM is the same as given in section 4.3.2. Thus for each texture, there will be two HMM models, one trained on RT and the other on DRT features.

5.4.3 Testing and Classification

An arbitrary orientation of any texture can be taken and its relevant RT feature vector, \mathbf{o}_{RT} , and DRT feature vector, \mathbf{o}_{DRT} , can be found. We use evaluation problem of HMM for testing and classification purpose as discussed in section 4.3.3 and section 5.3.3 for RT and DRT feature vectors, respectively.

For \mathbf{o}_{RT} , we calculate $P_{RT}(\mathbf{o}_{RT}|\lambda_m)$ and for \mathbf{o}_{DRT} $P_{DRT}(\mathbf{o}_{DRT}|\lambda_m)$ for $m = 1, 2, \dots, M$. Then we find

$$m^* = \arg \left\{ \max_{1 \leq m \leq M} [P_{RT}(\mathbf{o}_{RT}|\lambda_m) + P_{DRT}(\mathbf{o}_{DRT}|\lambda_m)] \right\} \quad (5.3.2)$$

The tested texture belongs to class m^* .

5.4.4 Simulations and Results

For simulation we have taken the first 60 textures of Brodatz album (D1-D60) and carried out simulations for varying number of feature vectors ($L = 20, 40, 60, 80, 100, 120$) and different number of states ($N = 3, 4, 5, 6, 8, 10$). We have used all the 60 textures at 20 arbitrary orientations to make the test set of 60×20 (1200) as used in the previous sections.

Table 5.2 shows that as the number of features, L , on which the HMM has been trained, increases, the PCC also increases for any specific hidden state. The best results of

Table 5.2 PCC of 1200 (60×20) test samples using proposed combined RT and DRT based method (case $s = 0$) for different number of feature vectors L and number of hidden states N

L \ N	3	4	5	6	8	10
20	91.50%	90.17%	86.00%	86.92%	85.17%	81.42%
40	95.83%	95.67%	94.17%	95.58%	94.67%	94.75%
60	95.75%	95.50%	95.50%	95.33%	95.08%	94.75%
80	95.92%	96.08%	95.58%	95.00%	95.83%	95.25%
100	96.67%	97.00%	96.17%	96.50%	97.17%	98.08%
120	97.08%	97.92%	98.00%	97.50%	98.50%	98.75%

98.75% have been achieved for $L=120$ and $N=10$. These results are much better than that of Chen and Kundu [53], Ojala et. al. [106] and Khouzani and Zadeh [60] although they have performed simulation on 25 textures and less and that too have been selected in some cases, see Table 4.6.

5.5 Texture analysis using Radon and Differential Radon Transform for the Case $s \neq 0$

In this section, we have also tried to classify the texture on bases of considering the combined behavior of RT features and DRT features for the case $s \neq 0$. As in the previous section we shall train two HMMs for every texture.

5.5.1 Feature extraction using RT and DRT

We have discussed the procedure to find the RT features for the case $s \neq 0$ in section 4.4.1. The DRT features for the case $s \neq 0$ have been calculated in the similar manner keeping the discrete steps Δs and $\Delta \theta$ same. If there are a total number of L discrete steps of θ , then the observation sequence has L feature vectors. These feature vectors extracted

by RT and DRT, of a texture, formulate the observation sequence for the training of two HMMs.

5.5.2 Training of HMM

Once the observation sequence is formulated for each texture on the basis of RT features and DRT features for the case $s \neq 0$, two HMMs are trained for each texture, one on RT features (low pass features) and the other on DRT features (high pass features). Training is carried out by using Baum-Welch algorithm as given in section 4.3.2.

5.5.3 Testing and Classification

In case of testing, we have to find out observation sequence $\mathbf{O}_{RT} = [\mathbf{o}_1 \mathbf{o}_2 \cdots \mathbf{o}_L]^T$ by using RT on the texture being tested for classification. Similarly find the observation sequence $\mathbf{O}_{DRT} = [\mathbf{o}_1 \mathbf{o}_2 \cdots \mathbf{o}_L]^T$ using DRT. Then calculate $P(\mathbf{o}_j^{RT} | \lambda_m)$ for all $j = 1, 2, \dots, L$ and $m = 1, 2, \dots, M$. Similarly, calculate $P(\mathbf{o}_j^{DRT} | \lambda_m)$ for all $j = 1, 2, \dots, L$ and $m = 1, 2, \dots, M$. Then the class is found out by using

$$m^* = \arg \left\{ \max_m \left[\sum_{j=1}^L P(\mathbf{o}_j^{RT} | \lambda_m) + \sum_{j=1}^L P(\mathbf{o}_j^{DRT} | \lambda_m) \right] \right\} \quad (5.3.3)$$

This unknown texture is either texture m^* of the Brodatz album, or it is closest to the m^* texture.

5.5.4 Simulations and Results

The feature vectors thus obtained are used to train HMM for $L = 20, 40, 60, 80, 100, 120$ and $N = 3, 4, 6, 8, 10$. We have used all 60 textures at 20 arbitrary orientations to make the

test set of 60×20 (1200) as used in the previous sections. Table 5.3 shows that as the number of features (L), on which the HMM has been trained, increases the PCC also increase for any specific hidden state. The best results of 99.10% have been found for $L=120$ and $N=8$. These results, although performed on 60 textures, are more accurate than that of Chen and Kundu [53], Ojala et. al. [106] and Khouzani and Zadeh [60], which have been performed on 25 textures or even less.

Table 5.3 PCC of 1200 (60×20) test samples using proposed combined RT and DRT based method (case $s \neq 0$) for different number of feature vectors L and number of hidden states N

L \ N	3	4	5	6	8	10
20	90.50%	90.75%	89.00%	88.92%	85.20%	81.40%
40	93.60%	95.30%	95.10%	95.00%	94.67%	94.30%
60	94.60%	96.50%	97.50%	96.33%	95.98%	95.75%
80	96.92%	97.08%	98.20%	97.90%	97.53%	96.25%
100	98.20%	98.32%	98.40%	98.50%	98.10%	97.60%
120	98.38%	98.62%	98.70%	98.88%	99.10%	98.25%

5.6 Summary

In this chapter, we have introduced the concept of differential Radon transform which instead of summing up carries out the absolute differencing between adjacent pixels. This results in giving us the high pass features of the texture. We have suggested three schemes and tried them on first sixty textures of Brodatz album. In the first scheme we have used only DRT for feature extraction by suppressing the offset ($s = 0$). The results are not very promising in this case. In the second scheme we have used RT as well as DRT for feature extraction by keeping the offset $s = 0$. Two HMMs are trained for each texture, one on RT and the other on DRT features and the result does show an improvement. In the third scheme, the offset $s \neq 0$ and RT and DRT both, are used for feature extraction. Same

procedure as in the second scheme has been repeated. Results have improved but at the cost of computation, because $s \neq 0$. The best result with PCC 99.10 % have been found for $L = 120$ and $N = 8$.

Chapter 6

Texture Analysis using its Principal Direction and Discrete Wavelet Transform

6.1 Introduction

This chapter discusses the issue of feature extraction and classification using a different approach. The approach is based on the definition of principal direction (PD), finding the orientation of this PD and applying discrete wavelet transform (DWT) along this direction to find out the features of the texture. These features are finally used to classify the textures with the help of k-nn classifier.

Applying the wavelet transform on digital images requires a discretization of the transform parameters from the Continuous Wavelet Transform (CWT) and leads to DWT. Since eighties the DWT has been thoroughly studied [108] in the one-dimensional space. A fast and powerful scheme for implementing the 1D-DWT using a filter bank has been designed by Mallat in [109]. The two-dimensional extension is obtained by applying this filterbank along the rows and columns of an image. Due to the separability of the filters, the two-dimensional separable discrete wavelet transform (S-DWT) is strongly oriented in the

horizontal and vertical directions, which makes it impractical to extract rotation-invariant features from it.

Greenspan et al. [110] tried to solve this problem by employing 4 angular filters and interpolated their responses to obtain the rotation-invariant features. Other researchers have tried to incorporate the rotation-invariance in the classification strategy, by e.g. including rotated examples in the learning data [53]. Yet another approach consists of a spiral re-sampling of the data, to obtain a 1-dimensional signal, where rotation-invariance is reflected as translation-invariance [65]. Another solution is to implement non-separable filter banks using non-separable sub-sampling lattices in the decomposition scheme like the quincunx lattice [111][112]. However, a sufficient angular localization is still difficult to obtain in this manner. Despite all these efforts, extracting the features from the original data and incorporating rotation-invariance clearly will improve the results.

Among all texture feature extraction methods such as co-occurrence matrix [16], Markov random fields (MRF) [113], Gibbs random model, Gabor filters and wavelet transform [30][114], the wavelet transform and wavelets packets [98] attracts more and more attention due to its powerful ability in texture representation. The problem which remains there is that of translation, rotation, and scale invariant texture analysis.

In this chapter, two new rotation invariant and gray scale invariant texture analyses have been proposed. The first technique uses DRT and ordinary DWT. The DRT is first used to calculate the angle of the PD of the texture. Then, the texture is rotated in the opposite direction by the same angle as detected by DRT. Finally, DWT is applied to the preprocessed texture to extract features which are rotation invariant. The second technique

is identical to the first one except that instead of DRT, PCA is used to find the orientation of the PD.

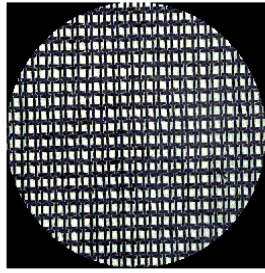
6.2 Methods using Principal Direction

Methods have been proposed in the literature to find orientation of predefined PD of the textures. Once the orientation of the PD is found, wavelet transform is applied in this direction. One such method which is of interest and relevance to us has been proposed in Khouzani et al [75]. Its details are presented below. The RT is used to detect linear trends in texture. Khouzani et. al. defined texture PD as the direction along which there are more number of straight lines. The RT along this direction usually has larger variations. Therefore, the variance of the projection in this direction is locally maximum. A disk shape area from the middle of the image has been selected before calculating the RT. Fig. 6.1 (a) shows an anisotropic (directional) texture (D20) and Fig. 6.1 (b) shows the variance of the projections along different orientations. As shown, the variance of the projections has two local maxima at 2° and 92° . The local maxima at 2° is narrower compared with the local maximum at 92° , because there are more straight lines along 2° . Thus, the derivative of the variance changes more rapidly at 2° as shown in Fig. 6.1 (c). To distinguish between these two local maxima, they calculated the second derivative of the variance as follows

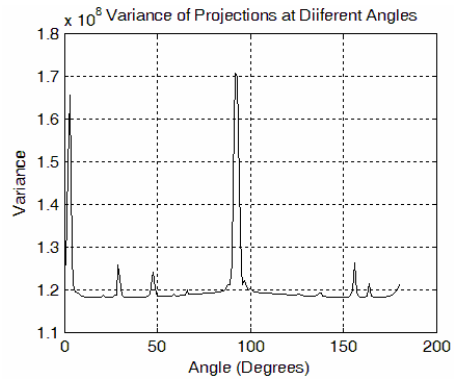
$$\alpha = \arg \left(\min_{\theta} \frac{d^2 \sigma_{\theta}^2}{d\theta^2} \right) \quad (6.2.1)$$

where $\sigma_{\theta}^2 = 1/N \sum_s (g(s, \theta) - \mu_{\theta})^2$ is the variance of the projection at θ ,

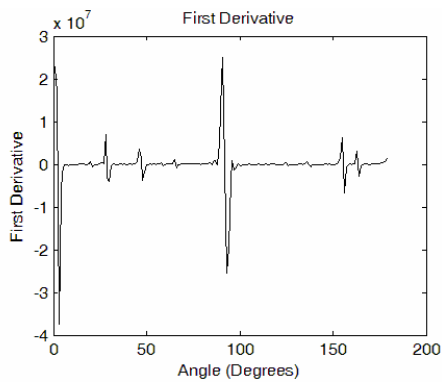
$\mu_{\theta} = 1/N \sum_s g(s, \theta)$, $g(s, \theta)$ is the RT of the image at orientation θ and N is the number



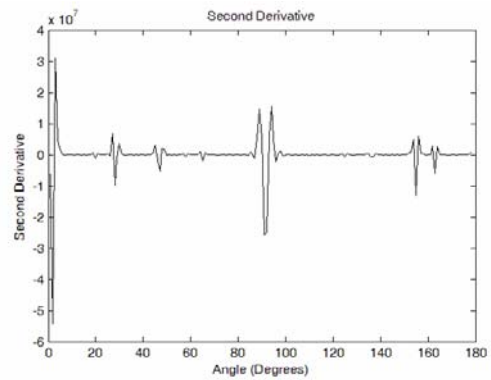
(a)



(b)



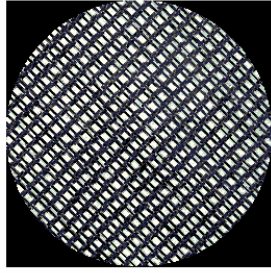
(c)



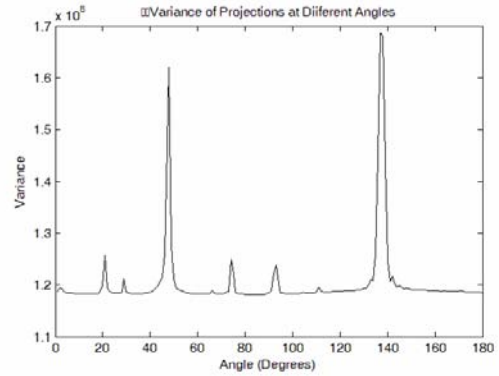
(d)

Fig. 6.1 (a) Anisotropic (directional) texture (D20) (b), variance of the projections along different orientations, (c) First order derivative of (b), (d) Second order derivative of (b)

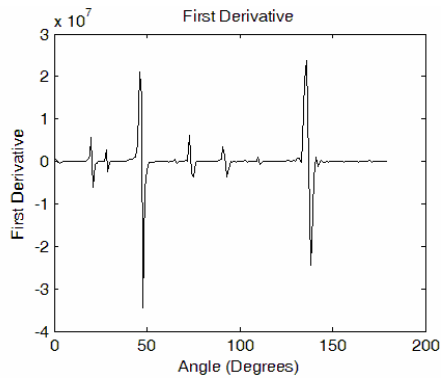
of samples in each projection. The image is then rotated by the angle $-\alpha$ to adjust the orientation of the texture. In particular example α is calculated as 2° . This is demonstrated in Fig. 6.1 (d). Now rotate the image at -2° and calculate the wavelet feature for this texture as proposed in chapter 3. In the Fig. 6.2 (a) the texture is rotated with an angle 45° . The variance of its projections along different orientations, first and second derivatives of the



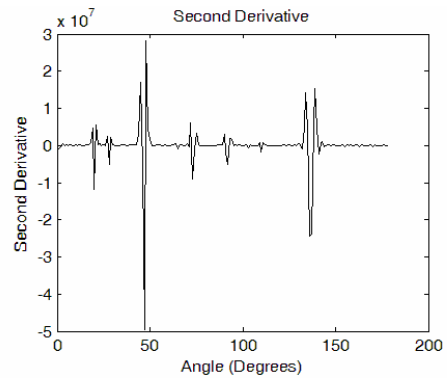
(a)



(b)



(c)



(d)

Fig. 6.2 (a) Anisotropic (directional) texture (D20) at rotation of 45° (b), variance of the projections along different orientations, (c) First order derivative of (b), (d) Second order derivative of (b)

projections are shown in the Fig. 6.2 (b), (c) and (d), respectively. The minimum for this case is at 47° calculated by eq. (6.2.1). The PD is 47° . After going through rotation of -47° , this rotated texture will be at zero orientation. Now we get wavelet features of the texture.

6.3 Proposed Method using DRT

The proposed method also defines the PD of a texture but it uses DRT to find the orientation of this PD. Before giving our definition for PD of a texture, we give our definition of activity. Normally activity around a point or pixel at (i, j) is defined as

$$A(i, j) = \sum_{s=-1}^{+1} \sum_{r=-1}^{+1} (f(i, j) - f(i+r, j+s))^2 \quad (6.3.1)$$

where $f(i, j)$ is the gray scale value of pixel at (i, j) . Thus activity at (i, j) involves eight neighbors around it as shown in the Fig. 6.3. Using the above definition people have defined the activity of the block by adding up the activity around every pixel in the block. Similarly, one can talk of activity on a line by adding the activities of all the pixels lying on that line. We modify the definition of activity on a line as follows. Activity on a line is the absolute difference between the gray scale values of the consecutive pixels on that line. This is exactly the same as DRT. Thus we propose the following definition for PD of a texture.

“The principal direction of a texture is the direction along which there is maximum activity”

Like Khouzani’s definition of PD, this definition will work more efficiently for directional (anisotropic) textures and may fail completely for perfectly isotropic textures. To find out the PD with the proposed definition, we use DRT to find projections at different angles, θ . The DRT as defined by (5.2.1) is applied to a texture for a particular value of θ and all the samples of the projections in this direction are summed up and averaged over the number of samples. This is denoted by $G_D(\theta)$ and defined as

$$G_D(\theta) \triangleq \sum_s g_D(s, \theta) / N \quad (6.3.2)$$

where N is the number of samples in a projection. We take projection along angle from 0° to 180° in discrete steps of $\Delta\theta$. We try to find more values of projection near any expected peak (i.e. decrease value of $\Delta\theta$) in order to get a sharper maximum.

Moreover, there may be more than one maxima competing with each other, we will use the second derivative of the function in order to find a maximum which is relatively more localized. Thus, in this case

$$\theta_{PD} = \arg \left\{ \min_{\theta} \left(\frac{d^2 G_D(\theta)}{d\theta^2} \right) \right\} \quad (6.3.3)$$

After finding this orientation of the PD, the image is rotated by $-\theta_{PD}$ before applying DWT in order to extract the features. These features will give same classification effect for the same texture at any orientation.

Let us make a comparison of this definition with the one given by Khouzani and Zadeh [75]. Firstly, if there is a texture with straight lines, then the PD as defined above will be the direction orthogonal to the direction along which there is maximum number of straight lines i.e. it will be orthogonal to the PD defined by Khouzani and Zadeh. Secondly, this definition is more generalized than that of Khouzani and applicable to all those textures which have no straight lines at all. Thirdly, since DRT finds the absolute difference between gray scale values of consecutive pixels, this method will also be gray scale invariant. Hence, the method is immune to illumination changes across the image. If we further use second derivative of DRT, we shall be totally getting rid of low frequency components, thus making the method even more robust to illumination changes.

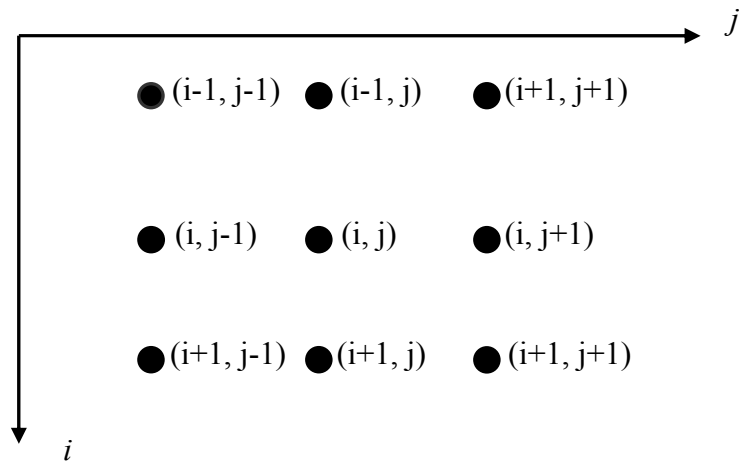


Fig. 6.3 Eight neighbors of a pixel

Lastly there is another basic difference between the proposed method and Khauzani,s method. In that, one has to calculate the variance of the different projections of RT, while we simply take the average of the sum of DRT projections in different orientations. This makes our method computationally lighter.

By applying the DWT to this preprocessed image for different levels a number of sub-bands are produced. We have chosen four levels. At each level a texture image is decomposed into four subbands, (A,H,V,D) , where A represents the approximation image and H, V, D represent the horizontal, vertical and diagonal details of the image, respectively. $W_s^l(i, j)$ represents the wavelet coefficients at the l^{th} level for subbands where s can be A,H,V or D . The wavelet decompositions at each stage is done for approximation subband, A , of the previous image. Fig. 6.4 shows the four level wavelet decomposition of the image $f(x,y)$.

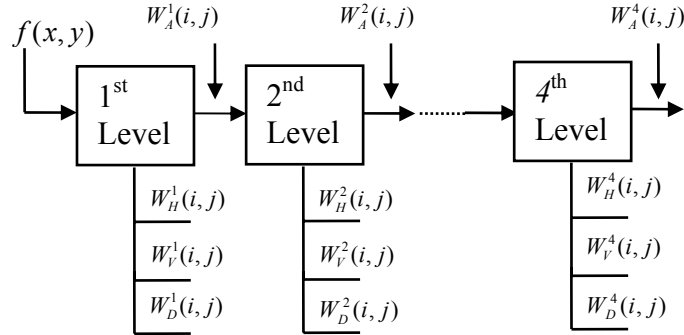


Fig. 6.4 Block Diagram for 4-level Wavelet Decomposition

The energy features of thirteen subbands are calculated from the wavelets coefficients produced by four levels of ordinary wavelet decomposition using Daubechies wavelet of length 4 (db_4). For each subband, we calculate the statistical feature (standard deviation of the subband) in the following way

$$e^{(l,s)} = \sqrt{\sum_{i,j} (W_s^l(i, j) - \bar{W}_s^l)^2} \quad (6.3.4)$$

where $\bar{W}_s^l = \frac{1}{MN'} \sum_i \sum_j \bar{W}_s^l(i, j)$, M' and N' are the dimensions of each subband,

where $M' = M/2^l$ and $N' = N/2^l$. For classification, we use the k-nearest neighbors (k-nn) classifier.

6.4 Proposed Method using PCA

In order to develop the rotation invariant texture features, we have proposed another method using PCA to find out PD for each texture and then find out wavelet features along that direction. The block diagram of the proposed method is shown in the Fig. 6.5. We denote the texture image by $f(x,y)$, where f is the gray level value of the pixel with spatial

coordinates (x, y) . These three elements x , y , and f can be expressed in the form of 3-D column vector, \mathbf{x} , where

$$\mathbf{x} = \begin{bmatrix} x \\ y \\ f \end{bmatrix} \quad (6.4.1)$$

This one vector represents one pixel in the texture. For the texture of size $M \times N$, there will be a total of MN three dimensional vectors from which we formulate the covariance matrix of size 3×3 according to eq.(3.2.3). We find its eigenvalue and corresponding eigenvectors. The eigenvector corresponding to the highest eigenvalue is called the principal eigenvector and its direction is the PD of the texture with certain estimated angle θ_e as shown in the Fig. 6.6. Now we rotate the image by $-\theta_e$ to adjust the orientation of the texture. For different orientation of the same texture, the PD will have different values of θ_e , but once rotated by $-\theta_e$, all the orientations of the same image will almost be similar. For example, take D97 from Brodatz album. Fig. 6.7 (a) and (b) are the original image of the texture and its wavelet signature, respectively. The procedure to find the wavelet signature will be discussed later. Fig. 6.7 (c) and (d) are the rotated texture of image (a) at principal angle found through PCA and its wavelet signature.

The original image rotated with angle 25 as shown in Fig. 6.7 (e) and its wavelet signature is shown in Fig. 6.7 (f), (g) and (h) are rotated image of texture at principal angle which has been found out through PCA and its wavelet signature, respectively. The wavelet signature of (d) and (h) are same while the signature of the (a) and (f) are different from each other and also different from (d) and (h).

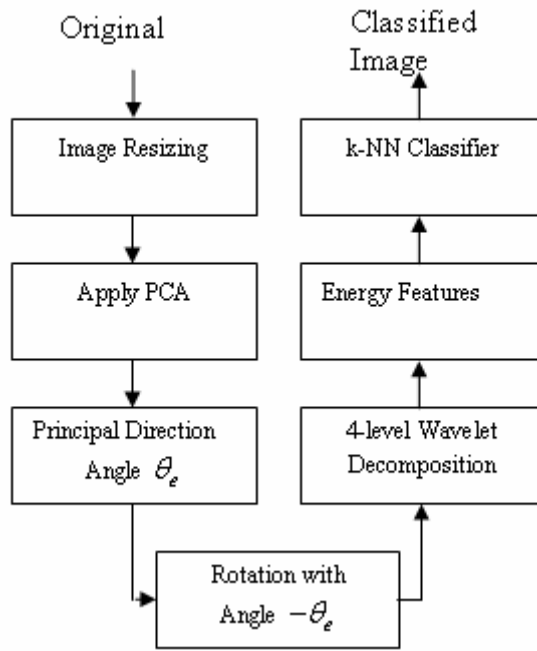


Fig. 6.5 Block Diagram of Proposed Method

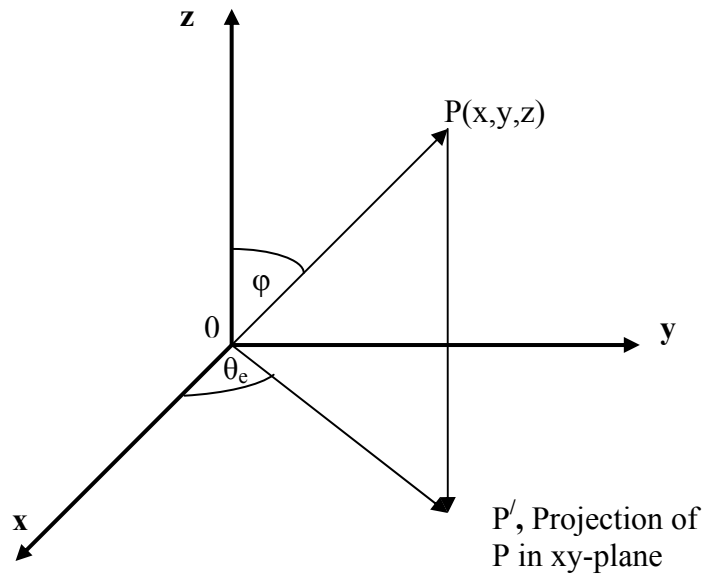
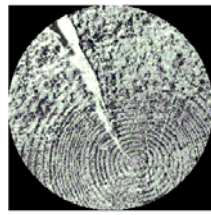
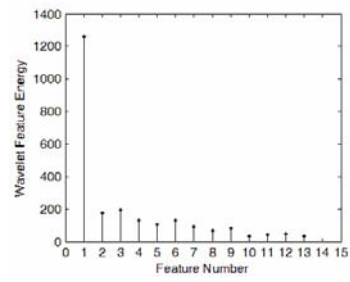


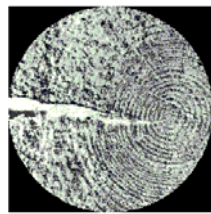
Fig. 6.6 Principal direction angle for Principal Eigenvector



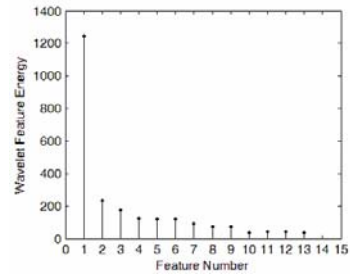
(a)



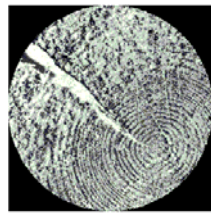
(b)



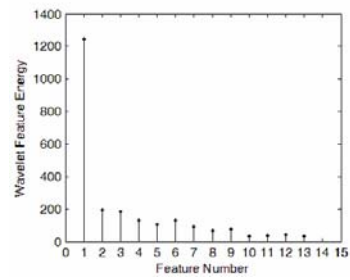
(c)



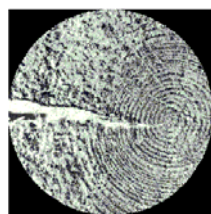
(d)



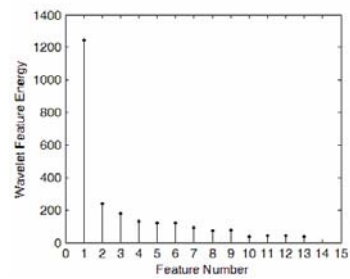
(e)



(f)



(g)



(h)

Fig. 6.7 Original image of the texture, (b) Wavelet signature of (a), (c) Rotated image of (a) with Principal direction angle, (d) Wavelet signature of (c), (e) Rotation of (a) with angle 25, (f) Wavelet signature of (e), (g) Rotated image of (d) with Principal direction angle

After finding this angle, the image is rotated by $-\theta_e$ to adjust the orientation of the texture. DWT is applied to the rotated image to extract the features. The procedure to calculate energy features through DWT is discussed in section 6.3. These features will give same classification effect for the same texture at any orientation.

6.5 Selection of Dataset for Classification

The famous datasets are CURET and Brodatz. The CURET dataset [115] provides a challenging dataset to classify as shown in the Fig. 6.8. The database contains images of 61 texture classes. For each texture class there are 205 images of each texture under different illuminations and directional conditions. The CURET database contains images of a number of natural textures from a wide variety of angles and illuminations. However, many of these textures do not exhibit patterns that exist in several real world textures and very few are synthetic in nature. These features are also important for texture classification and fit in with our definition of texture. In order to try and test the classifiers ability to handle both synthetic and patterned textures, the Brodatz database [116] considered for experiment is shown in Fig. 6.9. This database includes patterned and synthetic textures both at small and large scales. It contains 112 different texture classes. Unlike the CURET dataset this dataset consists of a single high resolution image per texture, subsequently there are no variations in illumination or rotation within each texture class. This database is a good test for texture classification as it includes regularly patterned, highly irregular and many natural textures. This dataset, although smaller, does contain significantly more texture classes than in the CURET database (112 compared to 61).



Fig. 6.8 The CURET Dataset

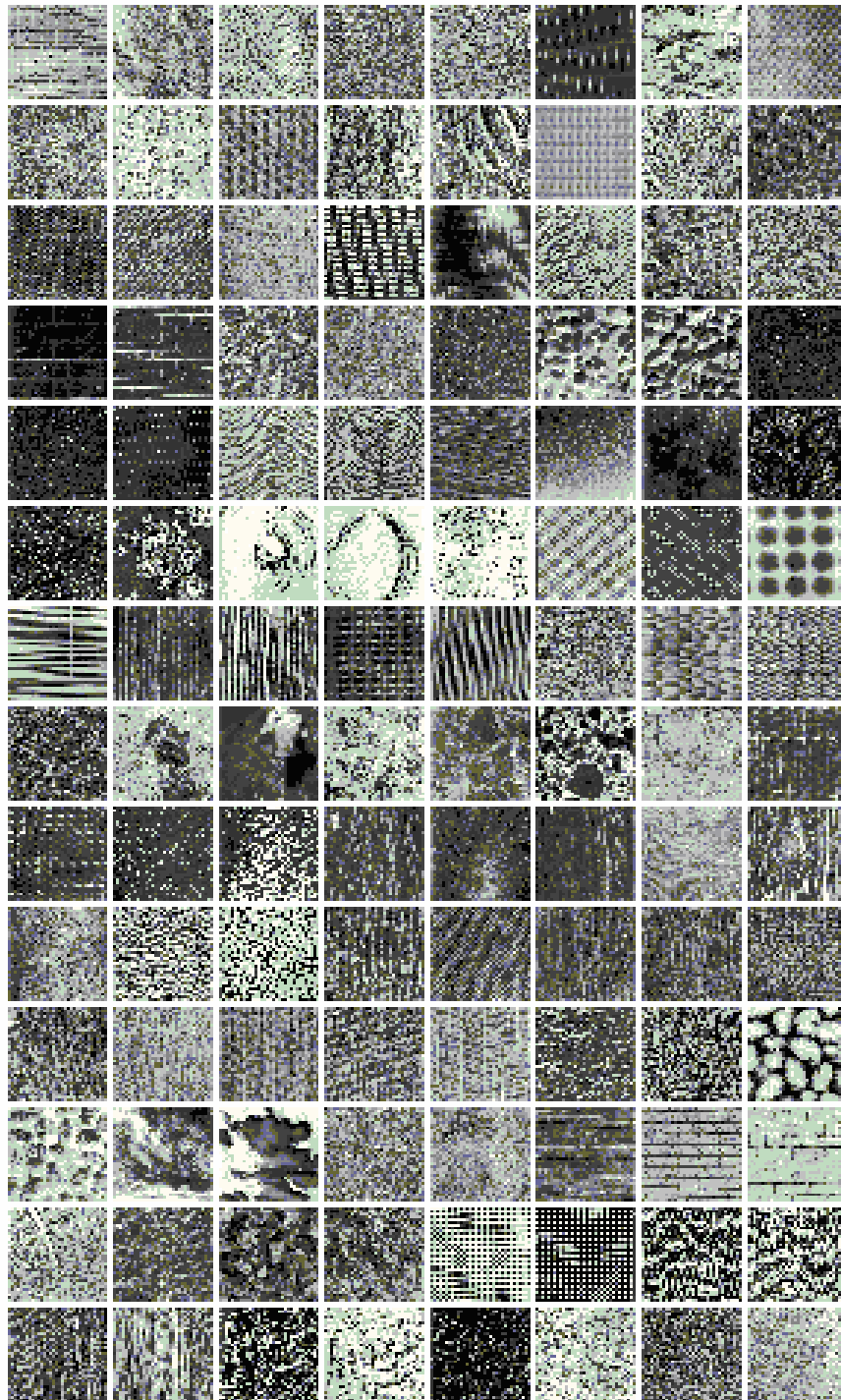


Fig. 6.9 One hundred twelve (112) textures from Brodatz album (D01-D112). First row D01-D08, second row D09-D16, and so on, and 14th row D105-D112

6.6 Simulations and Results

Brodatz textures [116] have been used for simulation of these two proposed schemes. Original size of the textures is 640×640 . Four cases were taken for each Brodatz texture, resized as 256×256 , 128×128 , 64×64 , and 32×32 pixels. In the first case we took only 25 textures and compared the results of two proposed schemes with that of Khouzani. In the second case we took 112 textures and compared the results of these three schemes. The texture was rotated from 0 degrees to 180 degrees with a discrete step of 5 degrees thus providing 37 different orientations for each texture. Therefore, in case of 112 textures we created a total number of 4144 (112×37) samples for each case. To estimate the PD, we have calculated the PCA for each rotation. Seven samples of each texture were taken as training set. The total number of samples for training was 784 (112×7). Thirty samples of each texture were taken as testing set. The total number of samples for testing was 3360 (112×30). The Table 6.1 compares these two schemes with that of Khouzani and Zadeh [75] for 25 textures and Table 6.2 for 112 textures. As we observe the results grow better as the pixel size of the texture reduces smaller and smaller. 32×32 has the worst PCC while 256×256 has the best PCC. Moreover, we also observe that in general $k = 1$ classifier gives the best results. Comparing these three techniques, PCA gives the best results in general but the scheme is computationally the heaviest. DRT is computationally the lightest and gives results slightly inferior to that of PCA. Moreover, 32×32 image results by Khouzani's method are no match to the results given by DRT or PCA method.

The noise robustness against zero mean additive white Gaussian noise (AWGN) has been checked for the proposed methods. In Fig. 6.10, proposed PCA-DWT and proposed

DRT-DWT methods have been compared in presence of zero mean AWGN with log polar-method and multichannel Gabor filtering as reported in [59]. The method using DRT-DWT is less robust to zero mean AWGN as compared to PCA-DWT. The performance of both the proposed methods is better than that of log polar-method and multichannel Gabor filtering. Unlike RT, DRT is carrying out absolute differencing between adjacent pixels. Hence DRT of this noise is not zero. Due to this property it is fairly less immune to the noise as compared to the techniques using RT.

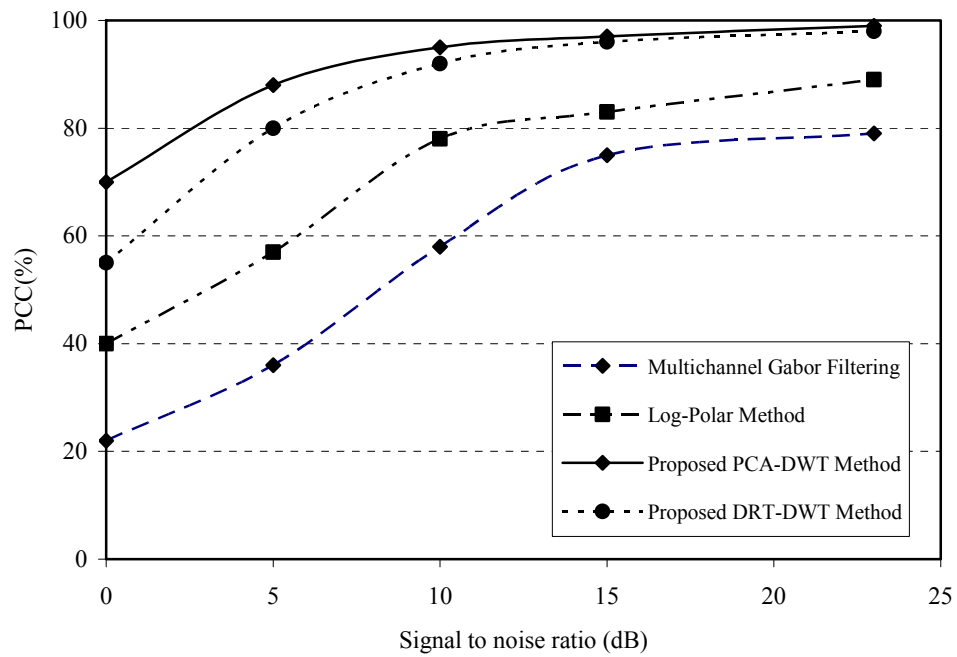


Fig. 6.10 Performance of four different methods in the presence of zero mean AWGN different signal to noise ratios

Table 6.1 Comparison of the three methods using 25 textures

Wavelet Bases	Texture Size	Khouzami Method (%)					Proposed PCA based Method (%)			Proposed DRT based Method (%)			
		k=1	k=3	k=5	k=7	k=1	k=3	k=5	k=7	k=1	k=3	k=5	k=7
db ₂	32×32	65.8	65.16	62.1	60.1	92	88.8	87.8	86.6	89.0	87.3	86.5	86.0
	64×64	95.8	93.6	92.0	89.3	98.6	99.1	98.5	93.8	97.8	98.8	98.1	96.1
	128×128	97.9	98	91.2	89.5	100	100	100	99.3	98.5	98.8	98.1	96.1
db ₄	256×256	100	100	100	98.3	100	100	100	99.8	100	100	99.8	98.5
	32×32	69.0	68.3	67.5	61.5	92.3	91	91.1	90.7	86.5	86.5	86.2	85.6
	64×64	91.3	90	88.5	85.9	98.6	98.5	97.1	96.5	92.5	92.5	91.6	91.0
db ₆	128×128	97.8	98.2	92.4	94.5	100	100	100	98.8	98.8	98.9	95.3	95.1
	256×256	99.6	99.6	99.6	96.5	100	100	100	99.0	100	99.7	98.9	98.0
	32×32	67.8	67.0	65.5	61.9	88.3	88.3	87.8	85.6	81.0	82.1	80.0	78.5
db ₈	64×64	94.0	91.8	90.6	89.8	100	100	100	94.6	88.0	85.1	84.8	82.3
	128×128	98.8	98.1	92.1	90.1	100	100	100	96.8	95.0	96.3	94.5	94.0
	256×256	99.8	99.8	99.8	97.7	100	100	100	97.8	100	100	99.7	98.6
db ₁₂	32×32	67.5	6	60.5	55.5	96.0	96.0	95.5	85.5	78.6	79.5	78.3	77.5
	64×64	93.1	92.3	91.0	85.8	99.6	99.3	98.6	93.9	89.7	89.6	88.7	87.6
	128×128	98.3	98.3	97.1	93.6	100	100	100	95.9	98.7	98.8	98.5	98.3
db ₁₂	256×256	99.5	99.6	97.3	94.4	100	100	100	97.1	99.8	99.8	98.9	98.3
	32×32	75.6	75.5	73.8	66.3	95.0	97.1	95.8	82.0	79.5	79.6	78.5	78.7
	64×64	87.3	8	79.1	78.	98.3	98.0	95.8	89.3	88.6	89.5	88.5	88.0
db ₁₂	128×128	98.2	98.3	90.8	90.3	100	100	100	95.5	98.6	98.7	97.0	96.5
	256×256	99.5	99.5	96.6	93.1	100	100	100	96.9	99.7	99.9	98.9	97.8

Table 6.2 Comparison of the three methods using 112 textures

Wavelet Bases	Texture Size	Khouzani Method (%)					Proposed PCA based Method (%)			Proposed DRT based Method(%)			
		k=1	k=3	k=5	k=7	k=1	k=3	k=5	k=7	k=1	k=3	k=5	k=7
db ₂	32×32	63.3	62.1	60.1	57.2	90	87.8	85.8	52.6	82.2	72.0	71.0	70.5
	64×64	92.8	91.1	90.6	86.6	97.6	97.8	97.5	80.5	93.6	93.7	92.5	90.0
	128×128	96.6	95.3	95.3	92.3	98.7	98.7	98.3	90.6	98.7	98.5	97.6	95.2
db ₄	256×256	98.7	98.7	98.7	98.1	99.0	99.0	98.5	92.3	99.2	99.3	99.1	98.9
	32×32	66.4	65.3	65.6	58.5	88.3	88.1	90.1	55.7	75.0	74.3	74.1	73.8
	64×64	89.2	88.6	85.5	80.6	96.6	97.5	95.6	78.6	92.6	92.5	91.7	90.3
db ₆	128×128	96.5	96.7	97.6	93.6	98.6	98.5	97.8	91.3	97.3	97.5	97.2	95.8
	256×256	97.6	98.2	98.5	97.3	98.9	99.0	98.3	95.2	99.3	99.2	99.0	98.7
	32×32	65.8	65.0	62.5	55.3	85.3	85.3	84.8	52.8	74.8	74.3	74.2	73.9
db ₈	64×64	92.2	89.8	88.6	83.6	98.5	98.1	97.7	78.9	92.0	91.3	90.2	88.8
	128×128	96.3	96.8	87.3	93.6	98.9	98.7	98.3	85.6	98.8	98.5	96.6	95.8
	256×256	98.8	98.4	98.4	95.8	99.0	99.0	98.6	88.8	99.3	99.4	99.2	98.2
db ₁₂	32×32	65.5	61.1	58.5	50.5	93.3	90.6	92.3	53.0	75.6	72.5	72.0	71.9
	64×64	91.1	90.3	90.3	81.1	98.6	96.3	95.5	78.8	93.5	92.6	91.7	88.7
	128×128	98.3	98.3	98.2	96.3	98.9	98.1	98.6	88.9	98.7	98.3	97.6	97.0
db ₁₂	256×256	98.5	98.6	97.3	97.3	99.0	98.4	98.8	96.3	99.5	99.3	98.7	97.5
	32×32	72.6	73.5	70.8	67.8	92.1	95.1	94.8	62.1	80.7	80.5	80.0	78.9
	64×64	85.3	81.8	78.1	73.6	97.3	97.0	95.8	71.6	89.5	88.6	85.3	82.5
db ₁₂	128×128	98.3	98.0	98.2	95.6	98.6	98.3	98.7	92.1	98.9	98.8	98.0	95.8
	256×256	98.7	98.5	96.6	92.0	98.9	98.0	98.9	90.8	99.1	99.1	98.3	96.4

6.7 Summary

The approach in this chapter is different from previous two chapters. In this case PD of a texture is defined, which is definitely not unique. Once this PD has been defined, a particular method to find out the orientation of this PD is used. Then the DWT is applied in the principal direction and its various components at different depths are obtained. These components describe the features of the texture, which are used by the k-nn classifier for classification.

The first definition of PD suggested in this chapter is the direction in which there is maximum activity. To find out this PD, we have used DRT and its second derivative. The second definition of PD is the direction of the eigenvector belonging to the maximum eigenvalue. To find out the angle of PD, PCA has been used. The results of the second scheme are better than that of the first scheme. However, it is computationally much heavier than the first one. Both the schemes show better results than that of Khouzani and Zadeh [75] and like them are applicable to anisotropic textures only. Among the two proposed schemes the first scheme using DRT is also gray-scale invariant.

Chapter 7

Conclusions

7.1 Summary of Results

In the field of texture analysis the recent advances in the use of Radon transform, Hidden Markov Models, PCA and Wavelet Transform have been exciting. These techniques provide a level of flexibility and adaptability for texture analysis which has not been fully exploited so far. Therefore, this dissertation shows that these techniques can be extended to handle the case of rotation, translation and gray scale invariant feature extraction for texture analysis. Another important factor is that these techniques, except for one, are equally valid for both isotropic and anisotropic textures.

As we know that in the texture analysis, two important issues are always under consideration. First is the feature extraction or mapping of image samples to n-dimensional points in feature space. Second is the design of classifier that is able to discriminate these feature vectors. It turns out even the optimal Bayesian classifier could come up with misclassification as can be seen from the Bayesian error. One good solution to this problem is to improve upon the methods of feature extraction. This has been the main theme of chapters 4 and 5.

In chapter 4, we have proposed three different ways to capture the texture features. Firstly, we have used the Radon transform but with $s = 0$. This method gives us features that are rotation invariant but not translation and gray scale invariant. These extracted features are used to train one HMM for each texture. This scheme provides us best PCC as 98.25%. Obviously if we use more number of feature vectors for training of HMM, PCC does become better but computationally it becomes very cumbersome. Secondly, we have proposed Radon transform with $s \neq 0$ for feature extraction. The computational level goes higher than the previous case but the features are rotation and translation invariant. Its best PCC is 98.80%. Thirdly, we have proposed feature extraction scheme using variance of Radon transform. These features are rotation, translation and gray scale invariant. The best result has been with PCC as 100%. Comparing it with other existing techniques in the literature, we claim our technique to be the best so far.

In chapter 5 we have introduced the concept of differential Radon transform (DRT) so as to get the feature vectors which are rotation, as well as, gray scale invariant. Again three methods were given to extract the feature vectors. First one is by using DRT only. We have extracted feature vectors by keeping $s = 0$. The feature vectors are rotation and gray scale invariant and are used to train one HMM for each texture. Its best PCC was found to be 96.33%. The second approach used RT and DRT both with $s = 0$, to capture features that are rotation and gray scale invariant. With the same textures as taken before, we get improved results than before. The best PCC is 98.75%. The third approach again uses RT and DRT but with $s \neq 0$. This gives us features that possess translation invariance in addition to rotation and gray scale invariance. After training of HMM the testing and classification gives as PCC 99.10%, this is slightly better than the previous case. The

schemes given in chapter 4 and 5 are equally applicable to isotropic as well as non isotropic textures.

The last scheme, given in chapter 6, is different from the rest. In this part of the thesis the main objective was to define a principal direction of the texture and then to devise a method to find its orientation. Once this preprocessing step is done effectively, then ordinary discrete wavelet transform, which easily provides discriminatory features but is not rotationally invariant, can be used along the orientation of the principal direction. We find the results for PCA case to be the best, but with very high computational price. However, it is applicable to isotropic as well as anisotropic textures. The DRT method is slightly inferior to it but is computationally much lighter. It is applicable to anisotropic textures only. Definitely both the schemes have been tested on 25 textures give better results than that of Khouzani and Zadeh [75].

7.2 Future Directions

In future it may be tried to improve the feature extraction techniques which are not only robust against rotation, translation and illumination but also scaling. Obviously it will be computationally heavier. Therefore, the reduction of the computational cost and complexity could be the further area of interest in this direction.

The robustness of these proposed methods against other types of noises apart from zero mean AWGN can also be looked into by the future research.

One could look into the potential of ridgelets and curvelets for feature extraction which have been invariant to rotation, translation and scaling.

In our work, we have extracted one dimensional feature vectors and used one dimensional HMMs. In future we shall look into extraction of two dimensional features and train two dimensional HMM for robustness and even better result.

References

- [1] R. C. Gonzalez and R. E. Woods, Digital Image Processing, Prentice Hall, Inc., New Jersey, 2nd edition, 2002.
- [2] W. K. Pratt, O. D. Faugeras and A. Gagalowicz, "Visual discrimination of stochastic texture fields," IEEE Transactions on System, Man and Cybernetics, vol. 8, pp. 796-804, 1978.
- [3] G. R. Cross and A. K. Jain, "Markov Random Field Texture Models," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.5, pp.25-39, 1983.
- [4] J. Zhang and T. Tan, "Brief review of invariant texture analysis methods," Pattern Recognition, vol. 35, pp. 735-747, 2002.
- [5] S. R. Fountain. T. N. Tan, and K. U. Baker, "Comparative study of rotation invariant classification and retrieval of texture images," in Proceedings, British Machine Vision Conference, 1998.
- [6] B. Julesz, Experiments in the visual perception of texture, Scientific software, 232, 2-11, 1975.
- [7] M. Unser. Description statistique de la texture. PhD thesis, EPFL, Lausanne, 1984.
- [8] R. M. Haralick and L. G. Shapiro, Computer and Robot Vision, Volume I, Addison Wesley, 1992.
- [9] A. Rosenfeld and A. Kak, Digital Picture Processing, vol. 1, Academic Press, 1982.
J. Serra, Image Analysis and Mathematical Morphology, Academic Press, 1982.
- [10] M. Levine, Vision in Man and Machine, McGraw-Hill, 1985.
- [11] J. M. Coggins, A framework for texture analysis based on spatial filtering, Phd, Michigan State University, 1982.

-
- [12] H. Tamura, S. Mori, and Y. Yamawaki, "Textural features corresponding to visual perception," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 8, pp. 460-473, 1978.
- [13] J. Sklansky, "Image segmentation and feature extraction," *IEEE Transactions on Systems Man and Cybernetics*, vol. 8, pp. 237-247, 1978.
- [14] IEEE Standard 610.4-1990, *IEEE standard Glossary of Image Processing and Pattern Recognition Terminology*, IEEE Press, New York, 1990.
- [15] M. Tuceryan and A. K. Jain, *Texture Analysis*, World Scientific, 1993.
- [16] R. M. Haralick, "Statistical and structural approaches to texture," *IEEE Proceedings*, vol. 67, no. 5, pp. 786-804, May 1979.
- [17] L. Van Gool, P. Dewaele, and A. Oosterlinck. *Texture analysis anno 1983*. *Comp. Vis. Graph. Im. Process.*, 29: pp.336 - 357, 1985.
- [18] T.R. Reed and J.M.H. du Buf. A review of recent texture segmentation and feature extraction techniques. *CVGIP: Image Understanding*, 57(3):pp. 359 - 372, 1993.
- [19] M. Tuceryan and A.K. Jain, "Texture Analysis," *Handbook of Pattern Recognition and Computer Vision*, pp. 235-276, World Scientific, 1993.
- [20] R.W. Connors & C.A. Harlow, "A theoretical comparison of texture algorithms," *IEEE Transactions on Pattern Analysis and Machine Intelligence* vol. 2, pp. 204-222, 1980.
- [21] A. Rosenfeld and B. S. Lipkin, "Texture synthesis," In *Picture Processing and Psychopictorics*, A Rosenfeld and B. S. Lipkin, eds. Academic Press, New York, 1970.

-
- [22] R. C. Dubes and A. K. Jain, "Random field models in image analysis," J. Appl. Stat. vol. 16, pp. 131-164, 1989.
- [23] A. A. Pentland, "Fractal-based description of natural scenes," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 6, pp. 551-764, 1984.
- [24] J. Mao and A. K. Jain, "Texture classification and segmentation using Multiresolution simultaneous autoregressive models," Pattern Recognition, vol. 25, no. 2, pp. 173-188, 1992
- [25] R. L. Kashyap and A. Khotanzad, "A model-based method for rotation invariant texture classification," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.8, no. 7, pp. 472-481, 1986.
- [26] R. Chellappa, R. L. Kayshap and B. S. Manjanuth, "Model-base texture segmentation and classification," in "Handbook of pattern recognition and vision," pp. 277-310, World Scientific, 1993
- [27] T. Randen, J.H. Husoy, "Filtering for texture classification: A comparative study," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21, no. 4, pp. 291-310, 1999.
- [28] K. I. Laws, "Textured image segmentation", Ph.D. thesis, Deptt. Electrical Engineering, University of Southern California, January 1980.
- [29] A. C. Bovik, M. Clarke and W. S. Geisler, "Multichannel texture analysis using localized spatial filters", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 12, pp. 55-73, 1990.
- [30] A. K. Jain and F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters," Pattern Recognition, vol. 24, pp. 1167-1186, 1991.

-
- [31] L. Jacobson and H. Wechsler, "A paradigm for invariant object recognition of brightness", *Pattern Recognition Letters*, vol. 1, no. 1, pp. 61-68, 1982.
- [32] Joan S. Weszka, Azriel Rosenfeld: An application of texture analysis to materials inspection. *Pattern Recognition* 8(4): 195-200, 1976.
- [33] B. Julesz, "Experiments in the visual perception of texture," *Scient. Am.* 232, pp. 34-43, 1975.
- [34] Niemann, Heinrich : *Pattern Analysis* . Berlin : Springer-Verlag, 1981
- [35] R. Lerski, K. Straughan, L. Shad, D. Boyce, S. Bluml, and I. Zuna, "MR Image Texture Analysis – An Approach to Tissue Characterisation", *Magnetic Resonance Imaging*, 11, 873-887, 1993.
- [36] M. Strzelecki, *Segmentation of Textured Biomedical Images Using Neural Networks*, PhD Thesis, Technical University of Łódź, Poland, 1995.
- [37] K. Valkealathi and E. Oja, "Reduced multidimensional co-occurrence histograms in texture classification," *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 20, no. 1, pp. 90-94, 1998.
- [38] P.C. Chen and T. Pavlidis, "Segmentation by texture using correlation," *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 5, pp. 64-69, January 1983.
- [39] R.M. Haralik, K. Shanmugam, and I. Dinstein, "Textural features for image classification," *IEEE Transactions Systems, Man, and Cybernetics*, vol. 3, no. 6, pp. 610-621, November 1973.

-
- [40] G.R. Cross and A.K. Jain, "Markov random field texture models," IEEE Transactions Pattern Analysis and Machine Intelligence, vol. 5, pp. 25-39, January 1983.
- [41] R. Chellappa and S. Chatterjee, "Classification of texture using Gaussian Markov random fields," IEEE Transactions Acoustics, Speech, and Signal Processing, vol. 33, pp. 959-963, April 1985.
- [42] R. Chellappa and R.L. Kashyap, "Texture synthesis using 2-D non-causal autoregressive models," IEEE Transactions Acoustics, Speech, and Signal Processing, vol. 33, no. 1, pp. 194-203, February 1985.
- [43] J.A. Cadzow, D.M. Wilkes, R.A. Peter II, and X. Li, "Image texture synthesis byanalysis using moving average models," IEEE Transactions Aerospace and Electric Systems, vol. 29, no. 4, pp. 1110-1121, October 1993.
- [44] L.M. Kaplan, "Extended Fractal Analysis for Texture Classification and Segmentation," IEEE Transactions Image Processing, vol. 8, no. 11, pp. 1572-1585, November 1999.
- [45] F. Liu and R.W. Picard, "Periodicity, directionality, and randomness, Wold Features for Image Modeling and Retrieval," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 18, no. 7, July 1996.
- [46] T. Randen, J.H. Huosy, "Filtering for texture classification: A comparative study," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21, no. 4, pp. 291-310, 1999.

-
- [47] M. Unser and M. Eden, "Multiresolution feature extraction and selection for texture segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 717-728, 1990.
- [48] A.C. Bovik, M. Clark, and W.S. Geisler, "Multichannel texture analysis using localised spatial filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 12, pp. 55-73, 1990.
- [49] A. Teuner, O. Pichler, and B.J. Hostica, "Unsupervised texture segmentation of images using tuned matched Gabor filters," *IEEE Transactions on Image Processing*, vol. 4, no. 6, pp. 863-870, June 1995.
- [50] B.S. Manjunath and W.Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, pp. 837-842, 1996.
- [51] S.Z. Li, *Markov Random Field Modeling in Computer Vision*. New York: Springer-Verlag, 2001.
- [52] F.S. Cohen, Z.G. Fan, and M.A. Patel, "Classification of rotated and scaled textured images using Gaussian Markov random field models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, no. 2, pp. 192-202, February 1991.
- [53] J.-L. Chen and A. A. Kundu, "Rotation and gray scale transform invariant texture identification using wavelet decomposition and hidden Markov model," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 2, pp. 208-214, February 1994.

-
- [54] H. Deng and D. A. Clausi, "Gaussian MRF rotation-invariant features for image classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 7, July 2004.
- [55] T. Chang and C.-C.J. Kuo, "Texture analysis and classification with tree-structured wavelet domain," *IEEE Transactions on Image Processing*, vol. 2, no. 10, pp. 429-441, October 1993.
- [56] M. Unser, "Texture classification and segmentation using wavelet frames," *IEEE Transactions on Image Processing*, vol. 4, no. 11, pp. 1549-1560, November 1995.
- [57] G. Van de Wouwer, P. Scheunders, and D. Van Dyck, "Statistical texture characterization from discrete wavelet representation," *IEEE Transactions on Image Processing*, vol. 8, no. 4, pp. 592-598, April 1999.
- [58] D. Charalampidis and T. Kasparis, "Wavelet-based rotational invariant roughness features for texture classification and segmentation," *IEEE Transactions on Image Processing*, vol. 11, no. 8, pp. 825-837, August 2002.
- [59] C.-M. Pun and M.-C. Lee, "Log-polar wavelet energy signatures for rotation and scale invariant texture classification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 590-603, May 2003.
- [60] K. Jafari-Khouzani and H. Soltanian-Zadeh, "Rotation-invariant Multiresolution texture analysis using Radon and wavelet transforms," *IEEE Transactions on Image Processing*, vol. 14, no. 6, pp. 783-759, June 2005.
- [61] X. Liu and D.L. Wang, "Texture classification using spectral histograms," *IEEE Transactions on Image Processing*, vol. 12, no. 6, pp. 661-670, 2003.

-
- [62] R. Porter and N. Canagarajah, "Robust rotation-invariant texture classification: wavelet, Gabor filter, and GMRF based schemes," *IEE Proceedings, Vision Image Signal Processing*, vol. 144, no. 3, pp. 180-188, June 1997.
- [63] G.M. Haley and B.S. Manjunath, "Rotation-invariant texture classification using a complete space-frequency model," *IEEE Transactions Image Processing*, vol. 8, no. 2, pp. 255-269, February 1999.
- [64] P. Campisi, A. Neri, G. Panci, and G. Scarano, "Robust rotation-invariant texture classification using a model based approach," *IEEE Transactions on Image Processing*, vol. 13, no. 6, June 2004.
- [65] W.-R. Wu and S.-C. Wei, "Rotation and gray-scale transform-invariant texture classification using spiral resampling, subband decomposition, and hidden Markov model," *IEEE Transactions on Image Processing*, vol. 5, no. 10, pp. 1423–1434, October 1996.
- [66] M. N. Do and M. Vetterli, "Rotation invariant characterization and retrieval using steerable wavelet-domain hidden Markov models," *IEEE Transactions, Multimedia*, vol. 4, no. 4, pp. 517–526, December 2002.
- [67] R. Mester, "Orientation estimation: conventional techniques and a new non-differential approach," in *Proc. 10th European Signal Proc. Conf.*, vol.2, pp. 921-924, 2000.
- [68] J. Bigun, G.H. Granlund, and J. Wiklund, "Multidimensional orientation estimation with applications to texture analysis and optical flow," *IEEE Transactions, Pattern Anal. Machine Intelligence* vol. 13, no. 8, pp. 775-790, August. 1991.

-
- [69] D.V.S. Chandra, "Target orientation estimation using Fourier energy spectrum," *IEEE Transactions , Aerosp. Electron. Syst.*, vol. 34, no. 3, pp. 1009-1012, July 1998.
- [70] E. Magli, L. Lo Presti, and G. Olmo, "Pattern detection and compression algorithm based on the joint wavelet and RT," in *Proc. IEEE 13th Int. Conf. Digital Signal Processing*, vol. 2, pp. 559-562, 1997.
- [71] A.L. Warrick and P.A. Delaney, "Detection of linear features using a localized RT with a wavelet filter," in *Proc. ICASSP*, vol. 4, pp. 2769-2772, 1997.
- [72] V.F. Leavers, "Use of the two-dimensional RT to generate a taxonomy of shape for the characterization of abrasive powder particles," *IEEE Transactions Pattern Analysis Machine Intelligence*, vol. 22, no. 12, pp. 1411-1423, December 2000.
- [73] M.N. Do and M. Vetterli, "The finite ridgelet transform for image representation," *IEEE Transactions Image Processing*, vol. 12, no. 1, pp. 16-28, January. 2003.
- [74] Jean-Luc Starck, E.J. Candes, and D.L. Donoho, "The curvelet transform for image denoising," *IEEE Transactions Image Processing*, vol. 11, no. 6, pp. 670-684, June 2002.
- [75] K. Jafari-Khouzani and H. Soltanian-Zadeh, "Radon transform orientation estimation for rotation invariant texture analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 6, June 2005.
- [76]. Lindsay I Smith, "A tutorial on Principal Components Analysis," February 26, 2002.
- [77] S. Pittner, J. Schneid, and C.W. Ueberhuber, *Wavelet Literature Survey*, Technical University of Vienna, Vienna, Austria, 1993.

-
- [78] A. Laine, J. Fan, "Texture classification by wavelet packet signatures," IEEE Transactions on Pattern Anal. Machine Intell., Vol. 15, pp. 1186-1191, 1993.
- [79] S. Mallat, A wavelet Tour of Signal Processing. New York: Academic, 1999.
- [80] V. Vapnik. Estimation of Dependences Based on Empirical Data. Springer-Verlag, New York, 1982.
- [81] A. K. Jain, R. P. W. Duin, and J. Mao. "Statistical pattern recognition: A review," IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(1):4-37, 2000.
- [82] K.S. Fu, Syntactic Pattern Recognition and Applications. Englewood Cliffs, N.J.: Prentice-Hall, 1982.
- [83] T. Pavlidis, Structural Pattern Recognition, New York: Springer-Verlag, 1977.
- [84] Duda, R.O., Hart, P.E. & Stork, D.G. Pattern Classification, 2nd Edition, John Wiley & Sons, New York, 2001.
- [85] R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis, New York: John Wiley & Sons, 1973.
- [86] F. Rosenblatt. Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms. Spartan, Washington DC, 1962.
- [87] J. Hertz, A. Krogh, and R.G. Palmer. Introduction to the Theory of Neural Computation, volume 1 of Lecture Notes. Addison Wesley, Redwood City, 9th edition, August 1991.
- [88] C.M. Bishop. Neural Networks for Pattern Recognition. Clarendon Press, Oxford Press, New York, 2nd edition, 1995.
- [89] B.D. Ripley. Pattern Recognition and Neural Networks. University, Cambridge, 1996.

-
- [90] C.G. Looney. Pattern recognition using neural networks. Oxford University Press, New York, 1997.
- [91] D.E. Rumelhart, R. Durbin, R. Golden, and Y. Chauvin. Backpropagation: the basic theory. In *Backpropagation: Theory, Architectures, and Applications*, pp. 1-34, Lawrence Erlbaum, Hillsdale, NJ, 1995.
- [92] K. Fukunaga. *Introduction to Statistical Pattern Recognition*. Academic Press, London, 2nd edition, 1990.
- [93] P.A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach*, Prentice Hall, Englewood Cliffs, New Jersey, 1982.
- [94] D. Hull, "Improving text retrieval for the routing problem using latent semantic indexing," *Proc. 17th ACM-SIGIR Conf.*, pp. 282-291, 1994.
- [95] W. Niblack, R. Barber, W. Equitz, M. Flickner, E. Glasman, D. Petkovic, and P. Yanker, "The QBIC project: Querying images by content using color, texture and shape," *Proc. SPIE Conf. 1908 on Storage and Retrieval for Image and Video Databases*, vol. 1908, pp. 173-187, February 1993.
- [96] K.V.R. Kanth, D. Agrawal, and A. Singh, "Dimensionality reduction for similarity searching in dynamic databases," *Proc. ACM SIGMOD Int'l Conf. Management of Data*, 1998.
- [97] Michie, D.J. Spiegelhalter, and C.C. Taylor, editors. *Machine Learning, Neural and Statistical Classification*, Hertfordshire, Ellis Horwood, 1994.
- [98] S. Livens, P. Scheunders, G. Van de Wouwer, and D. Van Dyck, "Wavelets for texture analysis, an overview," in *Proc. IEE Int'l Conf. Image Processing and its Applications*, Dublin, Ireland, pp. 581-585, July 1997.

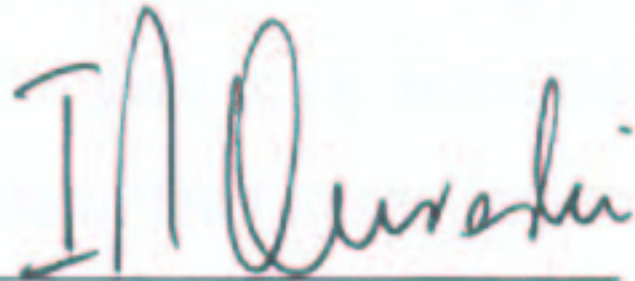
-
- [99] M. Unser, "Texture classification and segmentation using wavelet frames IEEE Transactions Image Proc., vol. 4, pp. 1549-1560, 1995.
- [100] J. R. Smith and S. F. Chang, "Transform features for texture classification and discrimination in large image databases," in Proc. IEEE ICIP, 1994.
- [101] D. Charalampidis and T. Kasparis, "Wavelet-based rotational invariant roughness features for texture classification and segmentation," IEEE Transactions Image Processing, vol. 11, no. 8, pp. 825-837, August 2002
- [102] R. Manthalkar, P.K. Biswas, and B.N. Chatterji, "Rotation and scale invariant texture features using discrete wavelet packet transform," Pattern Recognition Letters, vol. 24, pp. 2455-2462, 2003
- [103] J. Mao and A. K. Jain, "Texture classification and segmentation using Multiresolution simultaneous autoregressive models," Pattern Recognition ,vol. 25, no. 2, pp. 173–188, 1992.
- [104] W.-R. Wu and S.-C. Wei, "Rotation and Gray-Scale Transform-Invariant Texture Classification Using Spiral Resampling, Subband Decomposition, and Hidden Markov Model," IEEE Transactions Image Processing, vol. 5, no. 10, pp. 1423-1434, October 1996.
- [105] L. R. Rabiner, "A tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," IEEE Proceedings, vol. 77, No. 2, February 1989
- [106] T. Ojala, M. Pietikainen, and T. Maenpaa, "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns," IEEE Transactions Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 971-987, July 2002.

-
- [107] T. K. Moon, "The expectation-maximization algorithm," *IEEE Signal Processing magazines*, pp. 47-60, 1996.
- [108] I. Daubechies. *Ten Lectures on Wavelets*. Capital City Press, Montpelier, Vermont, 1992.
- [109] S. Mallat, *A wavelet Tour of Signal Processing*. New York: Academic, 1999.
- [110] H. Greenspan, R. Goodman, and P. Perona. "Rotation invariant texture recognition using a steerable pyramid," In *Int'l. Conf. on Pattern Recognition*, pp. 162-167, 1994.
- [111] J. Kovacevic and M. Vetterli. "Non-separable multidimensional perfect reconstruction filter banks and wavelet bases for R^n ," *IEEE Transactions Inform. Theory*, 38(2): pp. 533-555, 1993.
- [112] A. Cohen and I. Daubechies. Non-separable bidimensional wavelets bases. *Revista Matematica Iberoamericana*, 9(1):pp. 51-137, 1993.
- [113] B. S. Manjunath and R. Chellappa, "A note on unsupervised texture segmentation," *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 13, no. 5, pp. 478-483, May 1991.
- [114] B. S. Majunath and W. Y. Ma, "Texture features for browsing and retrieval of image data," *IEEE Transactions Pattern Analysis and Machine Intelligence*, vol. 18, no. 8, pp. 837-842, August 1996.
- [115] S.K. Nayar K.J. Dana, B. van Ginneken and J.J. Koenderink. "Reflectance and texture of real world surfaces," In *ACM Transactions on Graphics*, volume 18, pages 1-34, January 1999.
- [116] P. Brodatz, *Texture—A Photographic Album for Artists and Designers*, New York: Reinhold, 1968.

Certificate

It is certified that the research work contained in this dissertation has been carried out under the supervision of Dr. Ijaz Mansoor Qureshi, at Muhammad Ali Jinnah University, Islamabad Campus. It is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

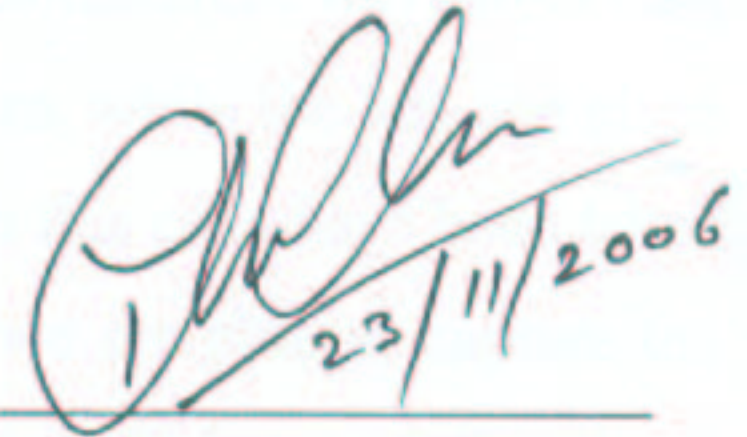
Signature: _____



Supervisor:

Professor Dr. Ijaz Mansoor Qureshi
Department of Electronics Engineering
Muhammad Ali Jinnah University

Signature: _____



Co-supervisor:

Dr. T. A. Cheema
Assistant Professor
Department of Electronics Engineering
Muhammad Ali Jinnah University

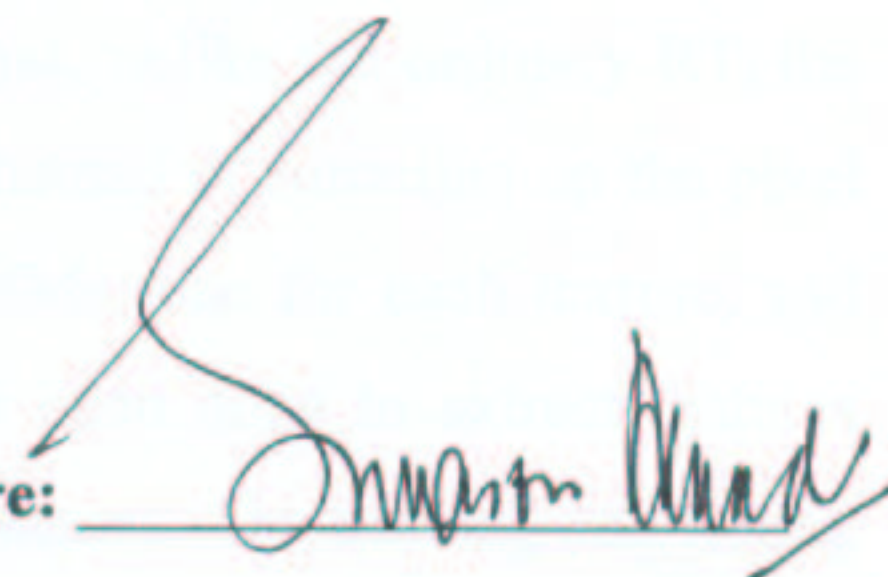
Signature: _____



Prof. Dr. Mohammad Mansoor Ahmed

Head of Department
Department of Electronics Engineering

Signature: _____



Prof. Dr. Mohammad Mansoor Ahmed

Dean
Faculty of Engineering and Sciences