Republic of Iraq

Ministry of Higher Education

and Scientific Research

Al-Nahrain University

College of science

Texture Recognition Using Density Distribution Method

A Thesis

Submitted to the College of Science, Al-Nahrain University in Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Science

By

Najma Dera Dhaher

(B.Sc. 2003)

Supervised by

Dr. Loay E. George

June 2011 **1432**

PDF created with pdfFactory trial version www.pdffactory.com

PDF created with pdfFactory trial version www.pdffactory.com

Dedicated to

**My beloved husband ...*

Thank you for your patience and cooperation with me to complete this work, with love and respect

**My father and mother ...*

Dearest people and the cause of my success with wishes for health and longevity

**My Brother ... Sisters ... and Friends*

**My Kids ... Saif and Sama*

**My uncle "Majeed Alsaad" ... with respect*

To every one taught me a letter

Najma

PDF created with pdfFactory trial version www.pdffactory.com

Acknowledgment

Praise is to God for his grace on me to finish this project.

I would like to express my sincere appreciation to my supervisor, Dr. Loay E. George, for his instructions and helpful suggestions throughout the preparation of this thesis, giving me the major steps to go on to explore the subject.

Grateful thanks are due for the Head of Department of Computer Science, Dr. Haitham A. Al-Ani.

Also I wish to thank Dr. Taha S. Bashaga in Computer Science Department for available help and advice.

I wish to thank the staff of the Computer Science Department, Al-Nahrain University for their help.

I would like to acknowledge the warm support of my family for encouraging me to finish this work.

I would like to thank my faithful friends for supporting and giving me advice.

PDF created with pdfFactory trial version www.pdffactory.com

Abstract

The recognition process is an important task in many applications of computer image analysis based on color or texture features. In this research, a texture recognition system has been introduced. The first stage is called the preprocessing stage. In this stage, a number of color transforms have been done on the training images to produce a set of bands (R, G, B, Gray, Y, I, Q, H, S, and V bands). From these bands, two sets of features vectors had been extracted, (i.e., density slicing and Fourier transform). These two techniques were applied in a separated manner.

In density distribution method, each image sample was quantized and converted to segments (areas) of black and white or (0 and 1). To find these areas, a seed filling algorithm was adopted. For these segments, a number of moments values were computed. Also the mean and standard deviation for the determined moments (density features) were computed. In Fourier transform method, each image sample was transformed using quick Fourier transform and the power spectra is divided into rings (Fourier slices), and then the average power of each slice (Fourier features) was determined.

The adopted training set of images consists of seven classes and each class consists of 10 scenes, and each scene is represented by 10 samples. Each class is represented by features vector(s) template in the features space; and it is stored in a table. Then, at feature analysis stage a selection to the best set of features is done. Given the tested image, the system first extracts its features' vector, and then compares the selected features with those stored in the file to find the nearest class using Euclidian distance

PDF created with pdfFactory trial version www.pdffactory.com

measure. The performance of the system was studied, and the effects of each involved parameter (i.e. block size, number of density thresholds and number of Fourier slices) on the recognition accuracy were investigated.

During the evaluation process, it was found that; the best results are obtained from a combination of nine features with the parameters [block length (180) and number of density threshold (5)] in density distribution method, which leads to 94.9% success rate. Also a combination of nine features with the parameters [block length (180) and Fourier slices (8)] in Fourier transform method, which in turn leads to 99.9% success rate.

PDF created with pdfFactory trial version www.pdffactory.com

List of Acronyms

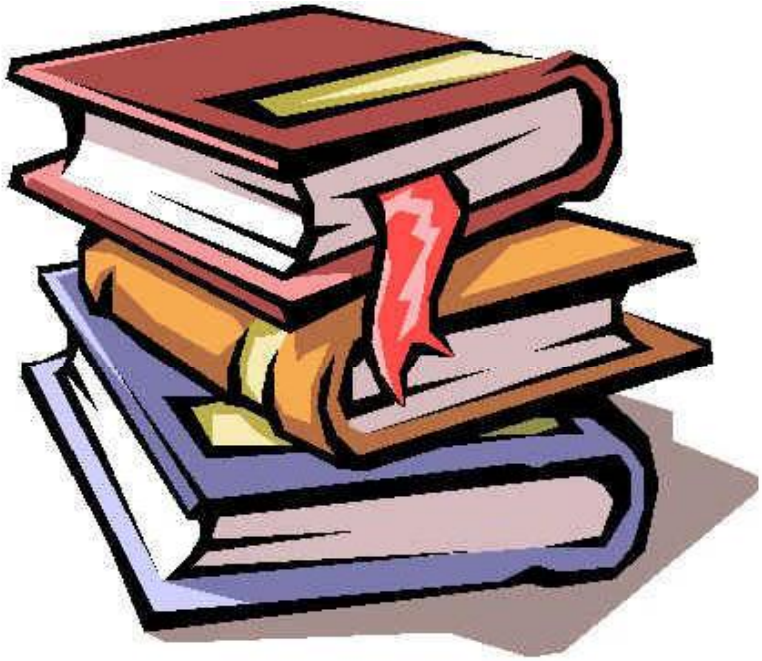
2-D	2 Dimentions
BMP	Bit-Map Images
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DST	Discrete Sine Transform
HSV	Hue Saturation Value
RGB	Red Green Blue
SOM	Self Organizing Map
TRS	Texture Recognition System

PDF created with pdfFactory trial version www.pdffactory.com

Table of Contents

PDF created with pdfFactory trial version www.pdffactory.com

PDF created with pdfFactory trial version www.pdffactory.com



Chapter One

PDF created with pdfFactory trial version www.pdffactory.com

Chapter One
Overview

1.1 Introduction

Pattern recognition techniques are often an important component of intelligent systems and are used for both data pre-processing and decision making. Broadly speaking, pattern recognition is the science that is concerned with the description or classification (recognition) of measurements. The following enumerates a few of the areas where pattern recognition finds its application [Has01]:

1. Computer vision,
2. Artificial intelligence,
3. Seismic analysis,
4. Radar signal classification/analysis,
5. Speech recognition/understanding,
6. Fingerprint identification,
7. Character (letter or number) recognition,
8. Handwriting analysis,
9. Medical diagnosis.

Pattern recognition aims to classify data (patterns) either by utilizing a priori knowledge or using a set of statistical information extracted from the patterns. The patterns to be classified are usually groups of measurements or

PDF created with pdfFactory trial version www.pdffactory.com

observations, defining points in an appropriate multidimensional (feature) space. This is in contrast to pattern matching, where the pattern is rigidly specified [Dud01].

Texture analysis is one of the most important techniques used in the analysis and classification of images presenting repetition or quasi-repetition of fundamental image elements. It is widely used in interpretation and classification of terrain images; radiographic and microscopic cell images, and many other domains of pattern recognition [Har79].

Texture analysis plays an important role in many image processing tasks, ranging from remote sensing to medical image processing, computer vision applications, and natural scenes. A number of texture analysis methods have been proposed in the past decades, but most of them use gray scale images, which represent the amount of visible light at the pixel's position, while ignoring the color information. The performance of such methods can be improved by adding the color information because, besides texture, color is the most important property, especially when dealing with real world images [Abd09].

Texture can be defined as a local statistical pattern of texture primitives in an observer's domain of interest. Texture classification aims to assign texture labels to unknown textures, according to training samples and classification rules. Two major issues are critical for texture classification: the texture classification algorithms and texture feature extraction. The main aim of texture classification is to find a good matching category for a given texture among existing textures. Texture has been analyzed extensively and many texture recognition schemes have been proposed [Abd09]. The important property all have in common is that they constitute an appropriate model for relationships between the adjacent pixels of a neighborhood.

PDF created with pdfFactory trial version www.pdffactory.com

1.2 Problem Definition

Texture is an important characteristic for the analysis of many types of images including natural scenes, remotely sensed data and biomedical modalities. The perception of texture is believed to play an important role in the human visual system for recognition and interpretation [Har79].

Since the textural properties of images appear to carry useful information for discrimination purposes, it is important to extract significant features for texture. One of the most important problems in the field of pattern recognition is the process of finding an appropriate set of feature vectors that represent observations with reduced dimensionality without sacrificing the discrimination power, along with finding the specific feature vector that has the best discrimination power [Tuc93].

Density slicing (also known as pixel-value thresholding) is virtually a process of discretizing the continuously varying pixel values in the input band. Pixel values within a certain gray level range are amalgamated into a single value in the output image. The range of entire pixel values in the input image is reduced to a few categories of values, each corresponding to a unique range of pixel values in the input image. Thus, the potential number of pixel values is considerably reduced in the sliced image. A unique color may then be assigned to each newly created pixel value, converting a gray level image into a pseudo color one. Since the naked human eyes are much more sensitive to variation in color than gray tone, the subtle spatial pattern contained in the input image is much more easily perceived visually in the density-sliced output. In order to produce a meaningful pattern for the phenomenon under study (e.g., concentration levels of silt in near shore water), the thresholds for each discrete category must be carefully selected. Frequently, the histogram of a spectral band is

PDF created with pdfFactory trial version www.pdffactory.com

relied to derive the critical slicing thresholds that should not be overlapping across categories [Gao09].

Signal decomposition has become an important issue in the sphere of signal processing. The Fourier transform is the operation that decomposes a signal into its constituent frequencies. Thus the Fourier transform of a musical chord is a mathematical representation of the amplitudes of the individual notes that make it up. The original signal depends on time, and therefore is called the time domain representation of the signal, whereas the Fourier transform depends on frequency and is called the frequency domain representation of the signal. The term Fourier transform refers both to the frequency domain representation of the signal and the process that transforms the signal to its frequency domain representation [Jam02].

1.3 Literature Survey

Several researches in the field of textural analysis for recognition tasks have been conducted and investigated in the literature. The present survey includes significant previous works related to the thesis objective, among the published works the following were selected:

1. Hee, Wang, and Gilbert (1988), [Hee88], presented a new approach for texture discrimination based on an algorithm that automatically selects the texture features best suited to a particular classification problem. Three sets of texture features have been extracted from the co-occurrence matrix with different displacements, and from Fourier transforms some additional features were extracted. In all, 173-texture features for each (32×32) sample image were input to a statistical classifier. The introduced method had been applied to discriminate 10 Bordatz's textures. The mean recognition rate was

PDF created with pdfFactory trial version www.pdffactory.com

95%. They said that this approach could be used to classify images of various textural properties.

2. Chang, and Kuo (1993), [Cha93], proposed a multiresolution approach based on a modified tree-structured wavelet transform and wavelet packets for texture analysis and classification. The development of this transform is motivated by the observation that a large class of natural textures can be modeled as quasi-periodic signals whose dominant frequencies are located in the middle frequency channels. With this transform, one can zoom into any desired frequency channels for further decomposition. The performance of their method was compared with that of several other methods using the DCT, DST, Pyramid-structured wavelet transform, Gabor filters, and Laws filters.
3. Samawi (1999), [Sam99], proposed a method for the classification of natural-textured image, using neural networks; she tried to get the best neural network architecture could reach the goal of high accuracy texture-image classification. Five different sets of features (statistical, spectral, direct features, or combination of statistical and spectral features) were used to train a number of texture-classification neural networks, and to find out the best feature set that could be used to discriminate texture images and improve the overall performance. She found that training a neural network texture classifier using statistical/spectral features improves the texture classification results.
4. Drimbarean, and Whelan (2002), [Dri02], proposed a method for the classification of color texture images. The main objective is to determine the contribution of color information to classification performance. Three relevant approaches to grayscale texture analysis (namely local linear transforms, Gabor filtering and co-

PDF created with pdfFactory trial version www.pdffactory.com

occurrence) were extended to color images. They were evaluated in a quantitative manner by means of a comparative experiment on a set of color images. Also the effect of using different color spaces was investigated, and the contribution of color and texture features separately and collectively was discussed.

5. Iakovidis, Maroulis, and Flaounas (2004), [Iak04], proposed a new scheme to recognize textural regions for color video analysis. The proposed scheme uses the covariance of 2nd-order statistics on the wavelet domain, between the different color channels of the video frames. These features, named as Color Wavelet Covariance (CWC), were used as color textural descriptors. A Support Vector Machine was chosen for the classification of the CWC feature vectors. Experiments were conducted using both animated VisTex texture mosaics and standard video clips. The estimated average accuracy ranged from 90% to 97%. The results show that the proposed methodology could efficiently be used in various multimedia applications as a complete supervised color texture recognition system.
6. Arivazhagan, and Deivalakshmi (2005), [Ari05], described texture classification using statistical features obtained from discrete wavelet transformed images at different scales, and their results are compared with the results of texture classification using conventional co-occurrence features, derived from original images. They found that, the texture classification using wavelet statistical features outperforms earlier conventional co-occurrence technique, and the success rate is very high when statistical features of three level discrete wavelet transformed images are used.
7. Broek, and Rikxoort (2005), [Bro05], compared 70 different configurations for texture analysis, using four sets of features. For

PDF created with pdfFactory trial version www.pdffactory.com

the configurations they used: (i) a gray value texture descriptor: the co-occurrence matrix and a color texture descriptor: the color correlogram, (ii) six color spaces, and (iii) several quantization schemes. A three classifier combination was used to classify the output of the configurations on the VisTex texture database. The results indicate that the use of a coarse HSV color space quantization can substantially improve texture recognition compared to various other gray and color quantization schemes.

8. Semler, Dettori, and Furst (2005), [Sem05], developed an automated imaging system for classification of tissues in medical images. They used texture analysis for the classification of tissues from CT scans. The approach consists of two steps: automatic extraction of the most discriminative texture features of the regions of interest in the CT medical images and creation of a classifier that will automatically identify the various tissues. A comparative study of wavelets-based texture descriptors from three families of wavelets (Haar, Daubechies, Coiflets), have been coupled, with the implementation of a decision tree classifier based on the Classification and Regression Tree (C&RT) approach was carried on.

1.4 Aim of the Work

The aim of this work is to build a texture recognition system by manipulating the input images and define each one according to one of the predefined set of classes. This recognition is accomplished using the following steps:

1. Extract features sets.
2. Select the best features.

PDF created with pdfFactory trial version www.pdffactory.com

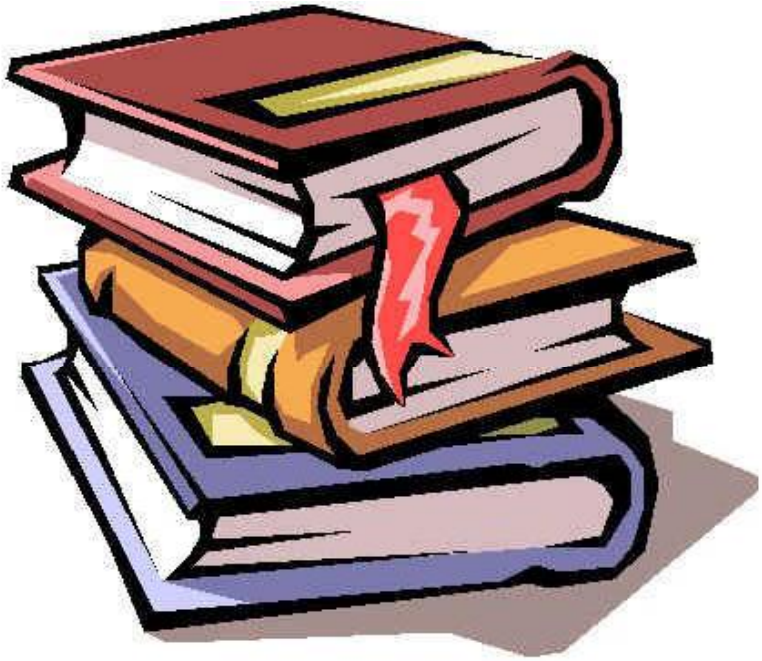
3. Recognize the query image using density slicing method depending on the selected features.

1.5 Thesis Outline

The contents of the remaining individual chapters of this thesis are briefly reviewed:

1. **Chapter Two "Theoretical Background"** introduces the relevant subjects surrounding the texture recognition system, and it illustrates how features are extracted from images. Some basic facts about color models, texture features, density slicing, Fourier transform, and wavelet transform are summarized in.
2. **Chapter Three "Design approach"** covers the proposed texture recognition system for identifying textures using spectral and/or statistical features. This chapter presents flow diagram for the proposed system, the method used to capture the input images, feature extraction (spectral and/or statistical) followed by feature analysis.
3. **Chapter Four "Experimental results"** presents the results of this research work together with the conducted assessments.
4. **Chapter Five "Conclusions and Suggestions**" introduces conclusions based on the experimental results of this work with a list of recommendations for future work.

PDF created with pdfFactory trial version www.pdffactory.com



Chapter Two

PDF created with pdfFactory trial version www.pdffactory.com

Chapter Two
Theoretical Background

2.1 Introduction

Texture analysis is one of the most important techniques used in the analysis and classification of images presenting repetition or quasi-repetition of fundamental image elements. It is widely used in interpretation and classification of terrain images, radiographic and microscopic cell images, and many other domains of pattern recognition. Texture analysis has been a topic of research for the last few decades.

Texture analysis has a wide range of applications. Millions of textural images are created throughout the World Wide Web, digital cameras, different kinds of sensors, medical scanners, etc. Image analysis is based on three main image features: color, shape, and texture. Texture plays an important role in human vision; they provide cues to scene depth and surface orientation. Researchers tend to relate texture elements of varying size to a reasonable 3-D surface [Man96, Bor96].

In a typical pattern recognition or object classification process, the first step is the extraction of features or key properties of objects (i.e. mapping from the real world to the feature space). The next step is classification of objects according to their features (i.e. mapping from the feature space to the classification space). The human brain is an excellent classifier which can successfully classify objects in noisy environments even without significant features.

PDF created with pdfFactory trial version www.pdffactory.com

2.2 Image Analysis

Image analysis is concerned with the extraction of useful measurements, data, or information from an image field by automatic or semiautomatic devices and systems [Pra78]. Image analysis is primarily a data reduction process. Images contain enormous amount of data, typically on the order of hundreds of kilobytes or even megabytes. Often much of this information is not necessary to solve a specific computer - imaging problem, so a primary part of the image analysis is to determine exactly what information is necessary [Umb98]. Dividing the spectrum of techniques in image analysis into three basic areas is conceptually useful. These areas are (1) low-level processing, (2) intermediate-level processing, and (3) high-level processing.

Figure (2.1) illustrates these concepts, with the overlapping dashed lines indicating that clear-cut boundaries between processes do not exist. For example, thresholding may be viewed as preprocessing or segmentation tool, depending on the application [Gon92].

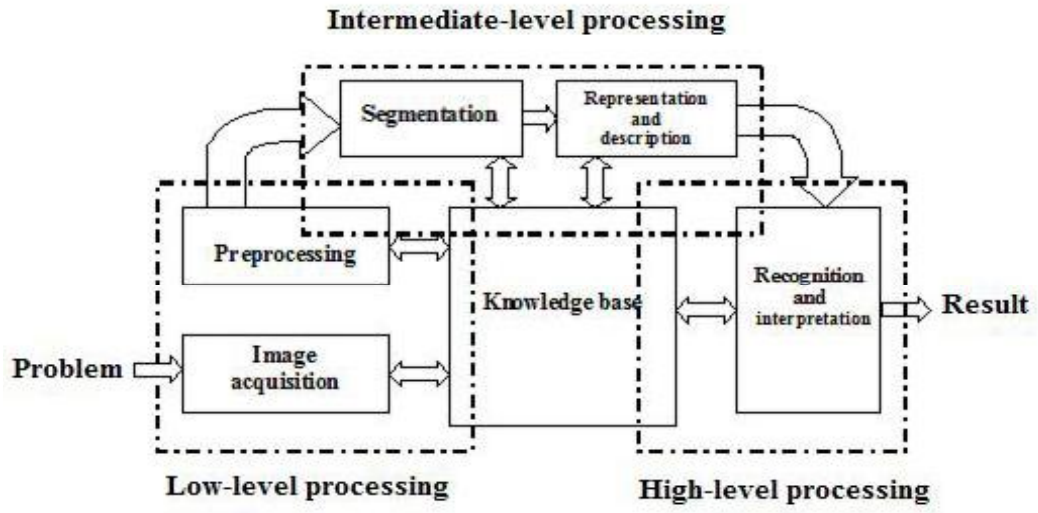


Figure (2.1) Elements of Image Analysis

PDF created with pdfFactory trial version www.pdffactory.com

Low-level processing deals with functions that may be viewed as automatic reactions, requiring no intelligence on the part of the image analysis system [Gon92]. Functions that can be treated as low-level processing are:

1. Image acquisition: It is the stage of acquire a digital image. To do so, it requires an imaging sensor (such as the x-ray) and the capability to digitize the signal produced by the sensor [Gon92].
2. Preprocessing: the preprocessing algorithms, techniques, and operators are used to perform initial processing that makes the primary data reduction and analysis task easier. They include operations related to extracting regions of interest, performing basic algebraic operations on images, enhancing specific features, and reducing data in both resolution and brightness [Umb98].

Intermediate-level processing deals with the task of extracting and characterizing components (say, regions) in an image resulting from a low-level process [Gon92]. Intermediate-level processes encompass segmentation and description tasks:

1. Segmentation: is to find regions that represent objects [Umb98].
2. Representation and Description: representing a region involve two choices: (a) representing the region in terms of its external characteristics (its boundary), or (b) representing it in terms of its internal characteristics (the pixels comprising the region). The description of a region is based on the chosen representation. For example, a region may be represented by its boundary, and the boundary described by some of its features such as its length, the orientation of the straight line joining the extreme points, and the number of concavities in the boundary. Generally, an external

PDF created with pdfFactory trial version www.pdffactory.com

representation is chosen when the primary focus is on shape characteristics. An internal representation is selected when the primary focus is on reflectivity [Gon92].

3. Properties, such as color and texture [Gon92].

Finally, *high-level processing* involves recognition and interpretation [Gon92].

2.3 Pattern Recognition System

Pattern recognition is the act of taking in raw data and taking an action based on the category of the pattern [Dud01]. Many image analysis tasks can be successfully performed by the classical pattern recognition system (shown in figure 2.2). Pattern recognition systems are designed to classify an input pattern (like, an image or portion of an image) into one of several categories. For example, in an Earth resources imagery application, a picture of agricultural fields may be broken into small blocks, and each block is classified as crop type, wheat, corn, and cotton [Alb97].

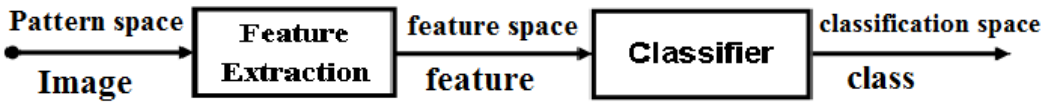


Figure (2.2) Pattern Recognition System [Alb97]

PDF created with pdfFactory trial version www.pdffactory.com

Each input image block may be regarded as a point in a pattern space of all possible amplitude patterns. For an N×N pixel block, each pixel amplitude is quantized to B levels; the dimension of the pattern space B can be enormous even for relatively small blocks and coarse quantization. Therefore it is usually necessary to perform a dimensionality reduction by feature extraction. Typical features include edge points, texture, and Transformation components [Alb97].

2.3.1 Supervised Learning

Supervised learning is a machine learning technique for deducing a function from training data. The training data consist of pairs of input objects (typically vectors), and desired outputs. The output of the function can be a continuous value (called regression), or a class label of the input object (called classification). The task of the supervised learner is to predict the value of the function for any valid input object after having seen a number of training examples (i.e. pairs of input and target output). To achieve this, the learner has to generalize from the presented data to unseen situations in a "reasonable" way [Dud01, Har01].

2.3.2 Unsupervised Learning

In machine learning, unsupervised learning is a class of problems in which one seeks to determine how the data are organized. Many methods employed here are based on data mining methods used to preprocess data. It is distinguished from supervised learning (and reinforcement learning) in that the learner is given only unlabeled examples [Dud01, Har01].

Unsupervised learning is closely related to the problem of density estimation in statistics. However unsupervised learning also encompasses

PDF created with pdfFactory trial version www.pdffactory.com

many other techniques that seek to summarize and explain key features of the data. One form of unsupervised learning is clustering. Among neural network models, the self-organizing map (SOM) and Adaptive resonance theory (ART) are commonly used as unsupervised learning algorithms [Hin99].

2.4 Feature Extraction

The goal in image analysis is to extract the useful information for solving application-based problems. This is done by intelligently reducing the amount of image data including image transforms. Feature extraction can be considered as useful operation for solving computer-image problems [Umb98].

Feature extraction can be defined as (The process of finding a mapping that reduces the dimensionality of the patterns by extracting some numerical measurements from raw input pattern [Set97]).

Feature extraction involves simplifying the amount of resources required to describe a large set of data accurately. When performing analysis of complex data one of the major problems stems from the number of variables involved. Analysis with a large number of variables generally requires a large amount of memory and computation power or a classification algorithm can overfit the training samples and generalizes poorly to new samples. Feature extraction is a general term for methods of constructing combinations of the variables to get around these problems while still describing the data with sufficient accuracy.

Feature extraction can avoid the curse of dimensionality, improve the generalization ability of classifiers, and reduces the computational requirements of the classifier [Mao94].

PDF created with pdfFactory trial version www.pdffactory.com

2.5 Texture

Still there is no distinct definition for "texture", but some references describe texture as:

1. Texture is a property of area. So texture is a contextual property where its definition must involve gray values in spatial neighborhood. The size of this neighborhood depends upon the texture type, or size of the primitives defining texture [Tuc93].
2. Texture features of grey level images are spatial deterministic or stochastic aspects of the grey level distribution [kla04].
3. It is an attribute of a window on the segment.

Figure (2.3) shows some examples of texture images used in this thesis.

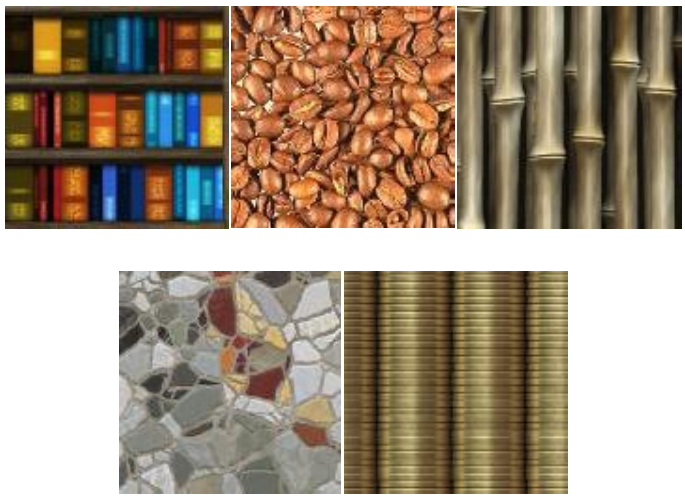


Figure (2.3) Samples of textures

PDF created with pdfFactory trial version www.pdffactory.com

2.6 A Taxonomy of Texture Models

Identifying the perceived qualities of texture in an image is an important step toward building mathematical models for texture. In spite of the fact that there is no general definition of texture, it has number of attributes that are assumed to be true. Among these attributes [Tuc93]:

1. *Texture involves spatial distribution of gray levels*. Thus, 2-D histograms or co-occurrence matrices are reasonable texture analysis tools.
2. *Texture in an image can be perceived at different scales or level of resolution.* For example, consider the texture represented by a set of bricks assembled to form the wall; the interior details in brick are lost. At higher resolution when only a few bricks are in the field of view, the received texture shows the details in the brick.
3. *A region is perceived to have texture when the number of primitive objects in the region is large*. If only a few primitive objects are presented, then a group of countable objects is perceived instead of a textured image. In other words, a texture is presented when significant individual "forms" are not presented.

Uniformity, density, coarseness, roughness, regularity, linearity, directionality, frequency, and phase are important properties used to describe textures. Some of these properties are not independent. For example, frequency is not independent of density, and the direction property is only applicable to directional textures. The fact that the perception of texture has so many different dimensions is an important reason why there is no single method of texture representation that is

PDF created with pdfFactory trial version www.pdffactory.com

adequate for variety of textures. For that reason different types of textures may need different features to represent and classify them [Sam99].

An image texture is a set of metrics calculated in image processing designed to quantify the perceived texture of an image. Image Texture gives us information about the spatial arrangement of color or intensities in an image or selected region of an image. Image textures can be artificially created or found in natural scenes captured in an image. Image textures are one way that can be used to help in Segmentation (image processing) or classification of images [Sha01].

2.7 Texture Segmentation

In computer vision, segmentation refers to the process of partitioning a digital image into multiple segments (i.e., sets of pixels; also known as super pixels). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain visual characteristics. The result of image segmentation is a set of segments that collectively cover the entire image, or a set of contours extracted from the image. Each of the pixels in a region are similar with respect to some characteristic or computed property, such as color, intensity, or texture [Bha87].

The use of image texture can be used as a description for regions into segments. There are two main types of segmentation (methods) based on

PDF created with pdfFactory trial version www.pdffactory.com

image texture; (1) region based and (2) boundary based. Though image texture is not a perfect measure for segmentation it is used along with other measure, such as color, to improve the efficiency of segmentation of an image [Sha01].

1. ***Region Based***

   Attempts to group or cluster pixels based on texture properties together.

2. ***Boundary Based***

   Attempts to group or cluster pixels based on edges between pixels that come from different texture properties.

2.8 Texture-Features Approaches

One of the important approaches for region description is to quantify its *texture* content. Although no formal definition of texture exists, we intuitively view this descriptor as providing a measure of properties such as smoothness, coarseness, and regularity [Gon92].

The three principal approaches used in image processing to describe the textural behavior of regions are: statistical, structural, and spectral.

2.8.1 Statistical Approach

Statistical approaches yield characterizations of textures as smooth, coarse, grainy, and so on [Gon92]. In statistical approaches, a texture is modeled as a random field, and a statistical probability density function model is fitted to the spatial distribution of intensities in the texture. Typically, these approaches measure the interactions between small numbers of pixels. The most common features used in practice are the

PDF created with pdfFactory trial version www.pdffactory.com

measures derived from the spatial gray tone co-occurrence matrix [Ach05].

In statistical texture analysis, texture features are computed from the statistical distribution of observed combinations of intensities at specified positions relative to each other in the image. According to the number of the intensity points (pixels) in each combination, statistics are classified into first-order, second-order and higher-order statistics [Alb95].

2.8.2 Structural Approach

Structural techniques, on the other hand, deal with the arrangement of image primitives, such as the description of texture based on regularly spaced parallel lines [Gon92]. Structural approaches represent texture by well defined primitives (*microtexture*) and a hierarchy of spatial arrangements (*macrotexture*) of these primitives. To describe the texture, one must define the primitives and the placement rules. The choice of a primitive (from a set of primitives) and the probability of the chosen primitive to be placed at a particular location can be a function of location or the neighbor primitives near the location. The advantage of the structural approach is that it provides a good symbolic description of the image; however, this feature is more useful for synthesis than analysis tasks. The abstract descriptions may be weakly defined for natural textures because of the variability of both micro- and macrostructure and no clear distinction may found between them. A powerful tool for structural texture analysis is provided by mathematical morphology [Mat98].

2.8.3 Spectral Approach

PDF created with pdfFactory trial version www.pdffactory.com

Spectral approaches are based on properties of the Fourier spectrum and are used primarily to detect global periodicity in an image by identifying high-energy, narrow peaks in the spectrum [Gon92]. The two-dimensional power spectrum of an image reveals much about the periodicity and directionality of its texture. For instance, an image of coarse texture would have a tendency towards low frequency components in its power spectrum, whereas another image with finer texture would have higher frequency components. Fourier transform based methods usually perform well on textures showing strong periodicity, however their performance deteriorates as the periodicity of texture weakens. Given such performance problems and the high computational complexity of the Fourier transform, the spectral approach is neither a very popular approach among researchers dealing with texture analysis, nor seems to be promising [Kon02].

2.9 Color Image Models

With the color format, a digital image can record and provide more information than the gray scale format image does. Digital acquisition devices (such as scanners and digital cameras) can separate beams of light into three primary colors (red, blue, and green) through the assistance of spectroscopes and filters. In order to record the color information, at least three parameters (e.g., red, blue, and green) are needed to represent a color [Wen04]. Digital image processing is a new and promptly developing field which finds more and more application in various information and technical systems (such as: Radar-tracking, communications, television, astronomy, etc...). There are numerous methods of digital image processing techniques such as: histogram processing, local enhancement, smoothing and sharpening, color

PDF created with pdfFactory trial version www.pdffactory.com

segmentation, a digital image filtration and edge detection. Initially, these methods were designed especially for grey scale image processing. The RGB color model is the standard model used in computer graphics systems, although it is not ideal for all of its applications. Since, the red, green and blue color components are highly correlated, its utilization in many application of digital image processing is not feasible. So, many processing techniques work on the intensity component of an image only. These processes are standard implemented using the HSI or HSV color model [Mah09].

2.9.1 RGB color model

The Red, Green and Blue (RGB) color model is defined as a unit cube with red, green and blue axes, as shown in Figure (2.4). Hence, a color in an RGB color model is represented by a vector with three coordinates. When all three values are set to 0, the corresponding color is black. When all three values are set to 1, the corresponding color is white [Wan01].

Figure (2.4) RGB Cube [Fol95]

2.9.2 YIQ color model

The YIQ color model is designed to refer to the characteristics of the human's visual system (HVS). In the human's visual system, people are

PDF created with pdfFactory trial version www.pdffactory.com

more sensitive to the lightness component than the hue component. So, the YIQ color model is set to separate colors into luminance (Y) and chrominance (I and Q). The YIQ color model is the standard model applied to the signal transmission of color TV sets .The relationship between YIQ and RGB is expressed as:

$$\begin{pmatrix} Y \\ I \\ Q \end{pmatrix} = \begin{pmatrix} 0.299 & 0.587 & 0.114 \\ 0.596 & -0.275 & -0.321 \\ 0.212 & -0.523 & 0.311 \end{pmatrix} \begin{pmatrix} R \\ G \\ B \end{pmatrix} \quad \ldots\ldots\ldots\ldots (2.1)$$

Where Y is the luminance, I and Q indicate the weights of hue. The advantage of the YIQ color model is that we can deal with the luminance component independently [Wen04].

2.9.3 HSV color model

The HSV color model (abbreviation of hue, saturation and value), is one of several color models used by people to select colors from a color wheel or palette. This color system is considerably closer than the RGB system to the way in which humans express or describe color sensations [Gon03]. The perceptual representation of the HSV color space has a conical shape, as shown in figure (2.5).

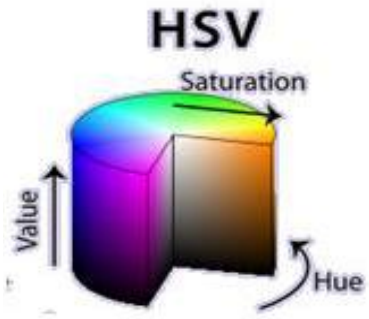


Figure (2.5) HSV color space [Fol95]

PDF created with pdfFactory trial version www.pdffactory.com

The HSV model is computed from RGB model by the following equations [Ach05]:

$$V = \max(R, G, B) \qquad \ldots\ldots\ldots\ldots\ldots\ldots (2.2)$$

$$S = \begin{cases} 0 & if\ V = 0 \\ V - \frac{min(R,G,B)}{V} & if\ V > 0 \end{cases} \qquad \ldots\ldots\ldots\ldots\ldots\ldots (2.3)$$

$$H = \begin{cases} 0 & if\ S = 0 \\ \frac{60 \times (G-B)}{S \times V} & if\ V = R \\ 60 \times \left[2 + \frac{(B-R)}{S \times V}\right] & if\ V = G \\ 60 \times \left[4 + \frac{(R-G)}{S \times V}\right] & if\ V = B \end{cases} \qquad \ldots\ldots\ldots\ldots\ldots\ldots (2.4)$$

2.10 Binary Images

Binary images are those have two quantized values (usually denoted 0 and 1 or 0 and 255), the 0 value represents black and the high value represents white. Binary images are used in many applications since they are the simplest to process, but they are such an impoverished representation of the image information are it use is not always possible. However, they are useful when all the information you need can be provided by the silhouette of the object and when you can obtain the silhouette of that object easily [Fau93]. Among its application domains are:

1. Identifying objects on a conveyor (for example, sorting chocolates!),
2. Identifying orientations of objects, and
3. Interpreting text.

PDF created with pdfFactory trial version www.pdffactory.com

Sometimes the output of other image processing techniques is represented in the form of a binary image, for example, the output of edge detection can be a binary image (edge points and non-edge points). Binary image processing techniques can be useful for subsequent processing of these output images [Fau93].

Binary images are typically obtained by thresholding a grey level image. Pixels with a grey level above the threshold are set to 1 (equivalently 255), whilst the rest are set to 0. This produces a white object on a black background (or vice versa, depending on the relative grey values of the object and the background). Of course, the `negative' of a binary image is also a binary image, simply one in which the pixel values have been reversed [Bur08].

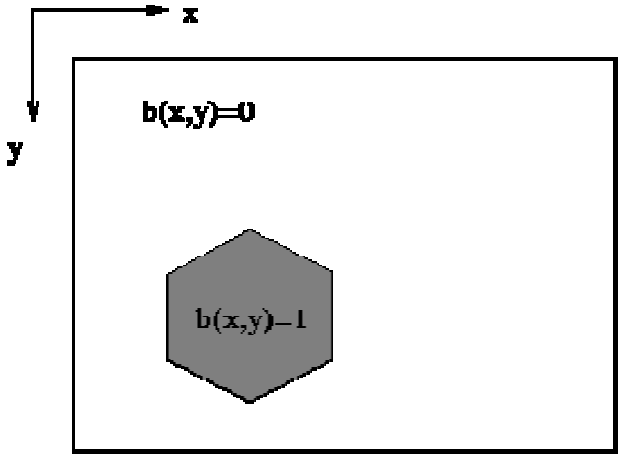


Figure (2.6) A binary image

2.11 Region Filling Operation

The most basic part of image analysis is image segmentation, which is the division of the image into regions that have different properties. For example, finding the text characters and the line graphics in a text image

PDF created with pdfFactory trial version www.pdffactory.com

is part of document image segmentation. Most image segmentation methods have, at their core, two parts:

1. The identification of *seeds*, which are pixels within the image that have a specific property, and
2. A *region-filling* operation that includes constraints to prevent each region from overfilling.

The region-filling constraints can be pre-determined masks, which are typically used in binary image segmentation, and they can also be expressed as a boundary negotiation process where regions expand and meet, thus preventing further expansion [Vin93].

2.12 Density Slicing

Density slicing (sometimes referred to as thresholding) is a digital interpretation method used in analysis of remotely sensed imagery to enhance the information gathered from an individual brightness band. Density slicing is done by dividing the range of brightnesses in a single band into intervals, then assigning each interval to a color [Cam02].

A number of methods may be applied to enhance the visibility of features. These methods are characterized by their mapping functions which relate the output image density to the input image density [Gon92]:

1. **Thresholding:**

   Contrast may be enhanced by thresholding (i.e. by setting all density values below a threshold to black 0 and those above to white 255). This may help to enhance features in an image with little contrast.

PDF created with pdfFactory trial version www.pdffactory.com

2. **Density slicing:**

   A number of thresholds may be set divide the input densities into several ranges. A series of output densities are chosen, one for each range; all densities within a given range are mapped to the corresponding output density.

3. **Contrast stretching and compression:**

   In contrast stretching and compression operations, a continuous function is used to map input densities to output densities. The mapping function is often made up of straight line segments.

4. **Histogram equalization:**

   The distribution of pixel densities may be shown as a histogram with density value plotted on the x-axis and the number of pixels with densities in a narrow range centered on this value on the y-axis. By suitably scaling the y-axis to correspond to the available density range, the histogram may be used as a mapping function for contrast stretching. In this case, the histogram of the output image is approximately uniform (i.e. a horizontal line) indicating an even spread of pixels over all densities.

5. **Color mapping:**

   Features are often more visible if the gray-scale densities are mapped onto a range of colors. Density slicing followed by color mapping produces a color coded counter map.

The input to a thresholding operation is typically a *grayscale* or *color image*. In the simplest implementation, the output is a *binary image* representing the segmentation. Black pixels correspond to background and

PDF created with pdfFactory trial version www.pdffactory.com

white pixels correspond to foreground (or vice versa). In simple implementations, the segmentation is determined by a single parameter known as the *intensity threshold*. In a single pass, each pixel in the images is compared with this threshold. If the *pixel's intensity* is higher than the threshold, the pixel is set to, say, white in the output. If it is less than the threshold, it is set to black. For color images, it may be possible to set different thresholds for each color channel, and so select just those pixels within specified cuboids in RGB space. Another common variant is to set to black all those pixels corresponding to background, but leave foreground pixels at their original color/intensity, so that that information is not lost [Cam02].

2.13 Fourier Transform

The transform of a signal is just another form of representing the signal. It does not change the information content present in the signal. In 19th century, the French mathematician, J. Fourier, showed that any periodic function can be expressed as an infinite sum of periodic complex exponential functions. Many years after this remarkable property of periodic functions was discovered, the ideas were generalized to non-periodic functions, and then to periodic or non-periodic discrete time signals. After this, Fourier transform became a very famous tool for computer calculations [Est01].

The Fourier transform is an important image processing tool which is used to decompose an image into its sine and cosine components. The output of the transformation represents the image in the Fourier or *frequency domain*, while the input image is the *spatial domain* equivalent. The Fourier transform is used in a wide range of applications, such as

PDF created with pdfFactory trial version www.pdffactory.com

image analysis, image filtering, image reconstruction and image compression [Gon92].

The Discrete Fourier Transform (DFT) is the sampled Fourier transform and therefore does not contain all frequencies forming an image, but only a set of samples which is large enough to fully describe the spatial domain image. The number of frequencies corresponds to the number of pixels in the spatial domain image (i.e. the image in the spatial and Fourier domain are of the same size) [Mar91].

For a square image of size N×N, the two dimensional DFT is given by [Jai89]:

$$F(u,v) = \frac{1}{N^2}\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)e^{-i2\pi(\frac{xu}{N}+\frac{yv}{N})} \qquad \ldots\ldots\ldots\ldots\ldots\ldots (2.5)$$

Where *f(x,y)* is the image in the spatial domain and the exponential term is the basis function corresponding to each point *F(u,v)* in the Fourier space.

In a similar way, the Fourier image can be re-transformed to the spatial domain. The inverse Fourier transform is given by [Jai89]:

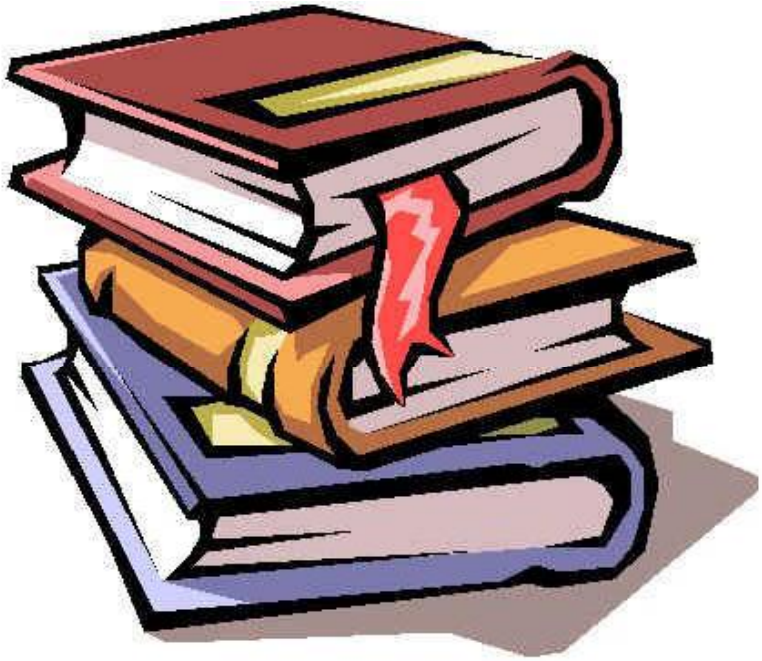
$$f(x,y) = \frac{1}{N^2}\sum_{u=0}^{N-1}\sum_{v=0}^{N-1} F(u,v)e^{i2\pi\left(\frac{xu}{N}+\frac{yv}{N}\right)} \qquad \ldots\ldots\ldots\ldots\ldots\ldots (2.6)$$

The lack of time information is a serious weakness of Fourier Transform. That is why Fourier transform is not suitable if the signal has time varying frequency (i.e., the signal is non-stationary) [Est01].

PDF created with pdfFactory trial version www.pdffactory.com

The Fourier transform produces a complex number valued output image which can be displayed as two images, either as the real and imaginary part or as magnitude and phase. For a given signal, the power spectrum gives a plot of the portion of a signal's power (energy per unit time) falling within given frequency bins. The most common way of generating a power spectrum is by using a DFT [Pre92]. In image processing, it is often than the magnitude of the Fourier transform is displayed exclusively. It contains most of the information of the geometric structure of the spatial domain image. However, if we want to re-transform the Fourier image into the correct spatial domain after some processing in the frequency domain, we must make sure to preserve both magnitude and phase of the Fourier image [Gon92].

PDF created with pdfFactory trial version www.pdffactory.com



Chapter Three

PDF created with pdfFactory trial version www.pdffactory.com

Chapter Three
Design and Implementation

3.1 Introduction

This chapter is concerned with the description of the design and implementation of the introduced Texture Recognition System (TRS). The system includes the modules: *preprocessing, feature extraction, feature analysis* and *matching* module. Two sets of features had been used in this research, density based features and Fourier based features. Density based features computed as follows: after performing an algorithm called (seed filling algorithm) on each input image to find the areas of black patches, a set of moment values was computed for all of these patches. Fourier based features computed by first dividing the Fourier spectrum for each input image into slices, and then the average power of each slice is determined. For implementation, the programming language "Microsoft Visual Basic 06" had been used to establish the required program of this project.

3.2 The Proposed (TRS) Layout

The proposed (TRS) consists of two units: The first one is the enrollment unit; as an output of this unit, a set of features' vectors are extracted from known texture images, this set is passed through feature analysis and evaluation to find the optimal values set of features, and then they are saved in a database as features templates.

The collection of training texture images consists of 70 images belong to 7 classes, 10 images for each class. The same number of feature vectors have

PDF created with pdfFactory trial version www.pdffactory.com

been extracted, analyzed and saved in the database. The block diagram for the training (Enrollment) phase is shown in figure (3.1).

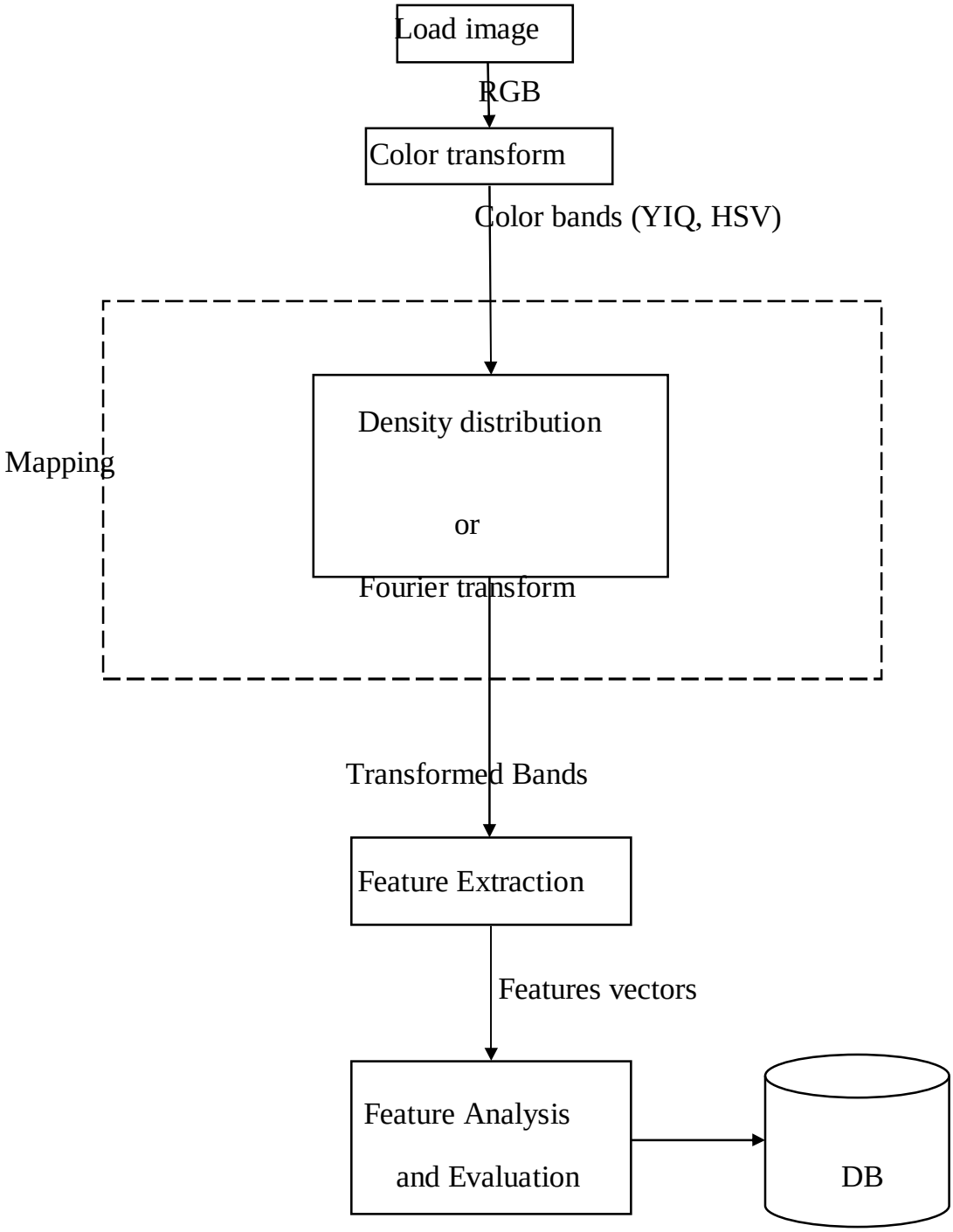


Figure (3.1) The diagram of enrollment phase

PDF created with pdfFactory trial version www.pdffactory.com

In the second unit (recognition phase), another set of test samples had been used to investigate the efficiency of the established recognition system. In this phase, all test images are passed through the feature extraction module to extract their features' vectors, and each vector is passed through a matching module to recognize its class. Finally, this class index was checked with the actual class index of the input image to find whether the class decision is correct or not. The result of this check is used to determine the recognition success rate of the system. The block diagram for the recognition phase is shown in figure (3.2).

In both system operation phases, the same preprocessing module is included and also the same feature extraction module is applied. Mainly, the system is composed of the following main modules:

1. Bitmap image loading.
2. Pre-processing.
3. Feature extraction.
4. Feature analysis.
5. Evaluation and training (in training phase only).
6. Recognition/Matching (in recognition phase only).

3.3 Image Loading

The textural image data may be artificial or natural; they could obtain from a real world application. The so-called Brodatz textures are probably the most widely used image data in texture analysis research projects.

In our TRS, the input image has bitmap format with color pixel resolution 24 bit/pixel. Seven classes of textured image were used. They are: bookshelves, food, bamboo, broken mosaic tiles, quantum weaves, twill weaves, and coin stack.

PDF created with pdfFactory trial version www.pdffactory.com

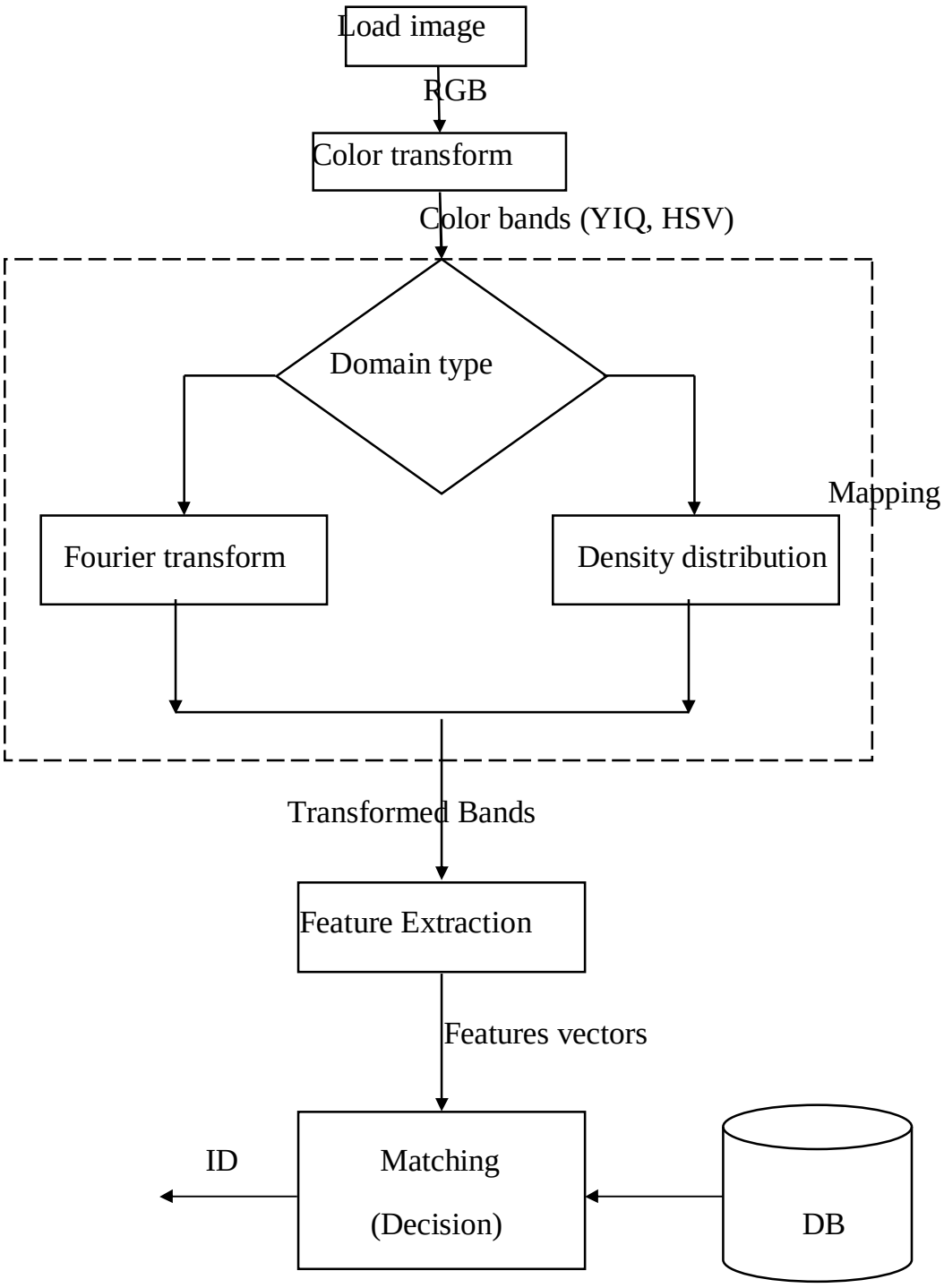


Figure (3.2) The diagram of recognition phase

PDF created with pdfFactory trial version www.pdffactory.com

When loading bitmap data, first the content of the file header is read. Then, the image header information is read, and finally the bitmap pixel data is loaded. Any 24-BMP file consists of the data structure presented in appendix E.

In Bmp file format, the lines of Bitmap pixels are stored in inverse order, starting from the bottom line and stop at the end of the upper line in image. Therefore, when reading from or writing into a file, the first image data should belong to the last (bottom) line in the image, and the last data set should assigned to the first (upper) line. Since the image is stored in inverted order, each set of RGB values is arranged in backward order; starting with blue, green, and ends with red. In this stage, the color image data is loaded and put into three (red, green, blue) arrays, as illustrated in Algorithm (3.1). All images used in this research project have size (256*256) pixel.

Algorithm (3.1) Read BMP Image

Goal: *Read 24 bit/pixel BMP image file*
Input: *ImgFileName // image file name*
Output: *Wid,Hgt // images width and height*
Red (0 to Wid-1,0 to Hgt-1) // Red component of image
Grn (0 to Wid-1,0 to Hgt-1) // Green component of image
Blu (0 to Wid-1,0 to Hgt-1) // Blue component of image
Gry (0 to Wid-1,0 to Hgt-1) // Gray component of image

Step1: ***Get*** *from ImgFileName the BMPH // BMPH is the BMP Header*
Get *image's width and height values from its header*
Set *Wid ← BMPH.Width*
Set *Hgt ←BMPH.Height*
Step2: *Check image pixel resolution*
If *BMPH.BitPlane=24* ***then***
Set *DataSize ←BMPH.FileSize – BMPHSize*
Get ImgFileName, Img(DataSize–1)//Img contain the image's data
Set *I ← 0*

PDF created with pdfFactory trial version www.pdffactory.com

```
        For all X,Y Do {where 0≤X≤Wid-1,0≤Y≤Hgt-1}
          Set Red(X,Y) ← Img(I)
          Set Grn(X,Y) ← Img(I+1)
          Set Blu(X,Y) ← Img(I+2)
          Set Gry(X,Y) ← (Red(X,Y)+Grn(X,Y)+Blu(X,Y))/3
          Increment I by 3
        End For
        Returen (Wid, Hgt, Red, Grn, Blu,Gry)
      Else
        Display massage "This Image Bitplane is not 24"
      End If
Step3: End.
```

3.4 Pre-processing

In this stage, some processes are applied on the image data to make the primary data reduction and analysis easier. In this project, the established texture analysis system consists of two steps for pre-processing stage:

3.4.1 Color Transform

To get color features, two color models have been utilized, they are YIQ and HSV. After splitting the image data into its three principal color components (Red, Green and Blue); these color models are mapped to another color space in order to get more descriptive representation. Each color model was implemented by coding the associated transformation equations that are listed in section (2.9). Algorithm (3.2) illustrates the steps of transformation from RGB to YIQ color model.

PDF created with pdfFactory trial version www.pdffactory.com

Algorithm (3.2) RGB to YIQ Transform

Goal: *convert image from RGB to YIQ color model.*
Input: *Wid,Hgt*
Red (0 to Wid-1,0 to Hgt-1)
Grn (0 to Wid-1,0 to Hgt-1)
Blu (0 to Wid-1,0 to Hgt-1)
Output: *Yc(0 to Wid-1,0 to Hgt-1) // Yc component of image*
I (0 to Wid-1,0 to Hgt-1) // I component of image
Q(0 to Wid-1,0 to Hgt-1) // Q component of image

Step1: *Convert each RGB pixel value into its corresponding YIQ value*
For *all X,Y* ***Do*** *{where 0≤X≤Wid-1, 0≤Y≤Hgt-1}*
Set *Yc(X,Y)← 0.299*Red(X,Y) +0.587*Grn(X,Y)+0.114*Blu(X,Y)*
Set *I(X,Y)← 0.596*Red(X,Y) + 0.274*Grn(X,Y) + 0.322*Blu(X,Y)*
Set *Q(X,Y)← 0.211*Red(X,Y) + 0.523*Grn(X,Y)+0.312*Blu(X,Y)*
End For
Return *(Yc,I,Q)*
Step2:End.

Algorithm (3.3) illustrates the steps of transformation from RGB to HSV color model.

Algorithm (3.3) RGB to HSV Transform

Goal: *convert image from RGB to HSV color model.*
Input: *Wid,Hgt*
Red (0 to Wid-1,0 to Hgt-1)
Grn (0 to Wid-1,0 to Hgt-1)
Blu (0 to Wid-1,0 to Hgt-1)
Output: *H(0 to Wid-1,0 to Hgt-1) // Hue component of image*
S (0 to Wid-1,0 to Hgt-1) // Saturation component of image
V(0 to Wid-1,0 to Hgt-1) // Value component of image

Step1: *Convert each RGB pixel value into its corresponding HSI value*

PDF created with pdfFactory trial version www.pdffactory.com

```
    For all X,Y Do {where 0≤X≤Wid-1, 0≤Y≤Hgt-1}
      Set R ← Red(X,Y)
      Set G ← Grn(X,Y)
      Set B ← Blu(X,Y)
      //calculate V
      Set V(X,Y) ←max (R+G+B)
      //calculate S
      If V(X,Y)=0 Then Set S(X,Y) ← 0
      Else  Set S(X,Y) ←V(X,Y) - [min(R,G,B) / V(X,Y)]
      //calculate H
      If S(X,Y)=0 Then Set H(X,Y) ← 0
      Else If V(X,Y)=R Then Set H(X,Y) ←60*(G-B)/S(X,Y)*V(X,Y)
      Else If V(X,Y)=G Then
           Set H(X,Y) ←60*[2+[(B-R)/S(X,Y)*V(X,Y)]]
     Else If V(X,Y)=G Then
          Set H(X,Y) ←60*[4+[(R-G)/S(X,Y)*V(X,Y)]]
      End If
     If H(X,Y)<0 Then Set H(X,Y) ← H(X,Y) + 360
     End For
    Return (H,S,V)
Step2: End.
```

3.4.2 Partitioning

Image data are often limited in terms of the number of original source images available; hence in order to increase the amount of data and to cover the features localization (variation), the images are divided into sub-images, either by overlapping or disjoint of a particular window size. In this project, an overlapped partitioning was adopted to partition each input image to N sub-images. This process is done after specifying the length of the sub-image (block) and the color domain (i.e., Red, Green, Blue, Gray, Y, I, Q, H, S, V) to work with, as illustrated in algorithm (3.4).

PDF created with pdfFactory trial version www.pdffactory.com

Algorithm (3.4) Partition

Goal: *Get N Blocks (subimages) from the input image.*
Input: *BlkLen // is the block size*
DType // is the Domain Type
Output: *Blk(0 to BlkLen-1,0 to BlkLen-1) // Blk is the sub-image*

Step1: *Check the Domain to get the sub-image from it*
Select Case *DType*
Case *1: //Red domain*
For *all X,Y* ***Do*** *{ where 0≤X≤BlkLen-1, 0≤Y≤BlkLen-1}*
Set *Blk(X,Y) ← Red(X,Y)*
End For
Case *2: //Green domain*
For *all X,Y* ***Do*** *{ where 0≤X≤BlkLen-1, 0≤Y≤BlkLen-1}*
Set *Blk(X,Y) ← Grn(X,Y)*
End For
Case *3: //Blue domain*
For *all X,Y* ***Do*** *{ where 0≤X≤BlkLen-1, 0≤Y≤BlkLen-1}*
Set *Blk(X,Y) ← Blu(X,Y)*
End For
Case *4: //Gray domain*
For *all X,Y* ***Do*** *{ where 0≤X≤BlkLen-1, 0≤Y≤BlkLen-1}*
Set *Blk(X,Y) ← Gry(X,Y)*
End For
Case *5: //Yc domain*
For *all X,Y* ***Do*** *{ where 0≤X≤BlkLen-1, 0≤Y≤BlkLen-1}*
Set *Blk(X,Y) ← Yc(X,Y)*
End For
Case *6: //I domain*
For *all X,Y* ***Do*** *{ where 0≤X≤BlkLen-1, 0≤Y≤BlkLen-1}*
Set *Blk(X,Y) ← I(X,Y)*
End For
Case *7: //Q domain*
For *all X,Y* ***Do*** *{ where 0≤X≤BlkLen-1, 0≤Y≤BlkLen-1}*
Set *Blk(X,Y) ← Q(X,Y)*
End For

PDF created with pdfFactory trial version www.pdffactory.com

```
        Case 8: //H domain
            For all X,Y Do { where 0≤X≤BlkLen-1, 0≤Y≤BlkLen-1}
              Set Blk(X,Y) ← H(X,Y)
            End For
        Case 9: //S domain
            For all X,Y Do { where 0≤X≤BlkLen-1, 0≤Y≤BlkLen-1}
              Set Blk(X,Y) ← S(X,Y)
            End For
        Case 10: //V domain
            For all X,Y Do { where 0≤X≤BlkLen-1, 0≤Y≤BlkLen-1}
              Set Blk(X,Y) ← V(X,Y)
            End For
      End Select
      Return (Blk)
Step2: End.
```

3.5 Feature Extraction

In this stage, the representation of an image was reduced to a small amount of elements (called features) carrying enough discriminating information (i.e. data reduction). Textural feature extraction is applied on each sample (sub-image). The computation of these features is done by the following two modules:

1. Density based feature extraction module.
2. Fourier based feature extraction module.

3.5.1 Density Based Features

To extract the set of density based features, the following steps are performed:

1. The first step in computing density features is to take each input image sample and compute its equivalent histogram. The underlying principle of histogram equalization is straightforward

PDF created with pdfFactory trial version www.pdffactory.com

and simple, it is assumed that each level in the displayed image should contain an approximately equal number of pixel values, so that the histogram of these displayed values is almost uniform (though not all 256 colors are necessarily occupied). The objective of the histogram equalization is to spread the range of pixel values present in the input image over the full range of the display device. From the histogram array, a lookup table had been computed to make the quantization process easier. Quantization is the procedure of constraining something from a large (i.e. continuous) set of values; such as the real numbers to a discrete set (such as the integers). Algorithm (3.5) illustrates the implemented steps for computing the histogram for each block and then quantizes it.

Algorithm (3.5) Histogram equalization and quantization

Goal: *convert the input block values to a uniform values (0 to 255).*
Input: *Blk // is the input sub-image*
BlkLen // is the window size
NT // is the number of density thresholds.
Output: *QuBlk // is the quantized block*

Step1: *Compute histogram array*
For *all I* ***Do*** *{ where 0≤ I ≤ 255 }*
Set *His(I) ← 0*
End For
For *all X,Y* ***Do*** *{ where 0≤X≤BlkLen-1, 0≤Y≤BlkLen-1}*
Set *I ← Blk(X,Y)*
Set *His(I) ← His(I)+1*
End For
For *all I* ***Do*** *{ where 0≤ I ≤ 255 }*
Set *His(I) ← His(I)+His(I-1)*
End For
Step2: *Compute lookup table*
For *all I* ***Do*** *{ where 0 ≤ I ≤ 255}*

PDF created with pdfFactory trial version www.pdffactory.com

Set LookTbl(I) ← *His(I)*$*(NT-1)/(BlkLen*BlkLen)$
End For
Step3: *Compute the quantized block*
For *all X,Y* ***Do*** *{ where 0≤X≤BlkLen-1, 0≤Y≤BlkLen-1}*
Set *QuBlk(X,Y)* ← *LookTbl(CInt(Blk(X,Y)))*
End For
Return *(QuBlk)*
Step4: End.

2. For each quantized image sample find the areas of black patches. To find these areas, a seed filling algorithm was adopted. The first step in this algorithm is the computations of the center point coordinates and then register this center point into a temporary (scan) array; and then start checking its 4-neighbors, if any of the four tested points is found black then register this point in the array and convert the value of the detected black point to white. Then, some steps are applied on the next point listed in array till reaching the last point listed in the scan array and no new found point is added to the array. Algorithm (3.6) illustrates the implemented steps to fill the quantized image sample by using seed filling algorithm. After finding the areas of black patches, the following moment values are computed:

$$MX_2(K) = \frac{1}{L_K}\sum_{i=1}^{L_K}(X_K(i) - X_{CK})^2 \qquad \ldots\ldots\ldots\ldots (3.1)$$

$$MY_2(K) = \frac{1}{L_K}\sum_{i=1}^{L_K}(Y_K(i) - Y_{CK})^2 \qquad \ldots\ldots\ldots\ldots\ldots (3.2)$$

$$MX_3(K) = \frac{1}{L_K}\sum_{i=1}^{L_K}|X_K(i) - X_{CK}|^3 \qquad \ldots\ldots\ldots\ldots (3.3)$$

PDF created with pdfFactory trial version www.pdffactory.com

$$MY_3(K) = \frac{1}{L_K}\sum_{i=1}^{L_K}|Y_K(i) - Y_{CK}|^3 \quad \ldots\ldots\ldots\ldots (3.4)$$

$$MX_1Y_2(K) = \sum_{i=1}^{L_K}(X_K(i) - X_{CK})(Y_K(i) - Y_{CK})^2 \quad \ldots\ldots (3.5)$$

$$MX_2Y_1(K) = \sum_{i=1}^{L_K}(X_K(i) - X_{CK})^2(Y_K(i) - Y_{CK}) \quad \ldots\ldots (3.6)$$

$$Mm(K) = L_K \quad \ldots\ldots\ldots (3.7)$$

Where:

K is the segment (patch) number

$X_K(i)$ is the x-coordinate of i^{th} point belong to K^{th} segment

X_{CK} is the x-coordinate of the center of K^{th} segment

$Y_K(i)$ is the y-coordinate of i^{th} point belong to K^{th} segment

Y_{CK} is the y-coordinate of the center of K^{th} segment

L_K is the number of points belongs to K^{th} segment

Algorithm (3.6) Seed Filling Algorithm

Goal: *find the areas of black patches.*
Input: *QuBlk // is the quantized block*
BlkLen // is the block size
V // is a density threshold value.
Output: *K // is the number of patches areas*
Mx2,Mx3,My2,My3,Mx1y2,Mx2y1,Mm // moments used in density features computation

---

Step1: For *all X,Y* ***Do*** *{ where 0≤X≤BlkLen-1, 0≤Y≤BlkLen-1}*
If *QuBlk(X,Y)=V* ***Then***
Set *LL ← 200 :* ***Set*** *M ← 0*

PDF created with pdfFactory trial version www.pdffactory.com

```
Set L ← 0 :  Set Bx(L) ← X
Set By(L) ← Y : Set QuBlk(X,Y) ← 0
While M≤L
    Set Xx ← Bx(M) : Set Yy ← By(M)
    If Xx  > 0 Then
      Set Xp ← Xx-1
      If QuBlk(Xp,Yy)=V Then
        If  L ≥ LL Then
          Set LL← LL+200
          Set L← L+1: Set Bx(L)← Xp
          Set By(L)← Yy :  Set QuBlk(Xp,Yy) ← 0
        End If
      End If
    End If
    If  Yy < BlkLen-1 Then
      Set Yp ← Yy+1
      If QuBlk(Xx,Yp)=V Then
        If  L ≥ LL Then  Set LL← LL+200
          Set L← L+1: Set Bx(L)← Xx
          Set By(L)← Yp : Set QuBlk(Xx,Yp) ← 0
        End If
      End If
    End If
    If Xx  < BlkLen − 1 Then
      Set Xp ← Xx+1
      If QuBlk(Xp,Yy)=V Then
        If  L ≥ LL Then  Set LL← LL+200
          Set L← L+1: Set Bx(L)← Xp
          Set By(L)← Yy : Set QuBlk(Xp,Yy) ← 0
        End If
      End If
    End If
    If Yy  >0 Then
      Set Yp ← Yy-1
      If QuBlk(Xx,Yp)=V Then
        If  L ≥ LL Then  Set LL← LL+200
          Set L← L+1 : Set Bx(L)← Xx
          Set By(L)← Yp : Set QuBlk(Xx,Yp) ← 0
        End If
      End If
    End If
    Set M ← M+1
```


PDF created with pdfFactory trial version www.pdffactory.com

```
            Wend
            Set sum ← 0 : Set sum1 ← 0
            For all J Do  { where 0≤J≤L}
               Set sum ← sum+ Bx(J)
               Set sum1 ← sum1+ By(J)
            End For
            Set Xc ← sum / (L+1) : Set Yc ← sum1 / (L+1)
            Set sum ← 0 : Set sum1 ← 0
            Set sum2 ← 0 : Set sum3 ← 0
            Set sum4 ← 0 : Set sum5 ← 0
            For all J Do  { where 0≤J≤L}
               Set sum ← sum+ (Bx(J) - Xc) ^ 2
               Set sum1 ← sum1+(By(J) – Yc) ^ 2
               Set sum2 ← sum2+Abs (Bx(J) - Xc) ^ 3
               Set sum3 ← sum3+Abs(By(J) – Yc) ^ 3
               Set sum4 ← sum4+ (Bx(J) - Xc) ^ 2 * (By(J) – Yc)
               Set sum5 ← sum5+(Bx(J) - Xc) * (By(J) – Yc) ^ 2
            End For
            Set Mx2(K) ← sum / (L+1)
            Set My2(K) ← sum1 / (L+1)
            Set Mx3(K) ← sum2 / (L+1)
            Set My3(K) ← sum3 / (L+1)
            Set Mx2y1(K) ← sum4 / (L+1)
            Set Mx1y2(K) ← sum5 / (L+1)
            Set Mm(K) ← L
            Set K ← K+1
         End If
      End For
      Return (K, Mx2, Mx3,My2,My3,Mx1y2,Mx2y1,Mm)
Step2: End.
```

3. After performing the above two steps, then the set of density features can be extracted. The mean and standard deviation for the distribution of extracted black areas are calculated (i.e. *MX2, MX3, MY2, MY3, MX1Y2, MX2Y1, and Mm*) according to the following equations:

PDF created with pdfFactory trial version www.pdffactory.com

$$MeanZ = \frac{1}{K}\sum_{i=1}^{K} Z(K) \qquad \ldots\ldots\ldots\ldots (3.8)$$

$$StdZ = \sqrt{\frac{1}{K}\sum_{i=1}^{K} (Z(K) - MeanZ)^2} \qquad \ldots\ldots\ldots\ldots (3.9)$$

Where Z is either *MX2, MX3, MY2, MY3, MX1Y2, MX2Y1, or Mm.*

Algorithm (3.7) illustrates the implemented steps to compute the mean and standard deviation values for these areas.

Algorithm (3.7) Density Features Extraction

Goal: *find the areas of black patches and compute density features.*
Input: *QuBlk // is the quantized block*
BlkLen // is the block size
NT // is the number of density thresholds
Output: *density features.*

Step1:
For *all V* ***Do*** *{ where 1≤V≤NT-1 }*
Call *Seed-filling (QuBlk,BlkLen,V,K)*
Calculate the Mean and Stdiv for all moments return from Seed-filling
End For
Return *(MeanX2,StdX2, MeanX3, StdX3, MeanY2, StdY2, MeanY3, StdY3, MeanX1Y2, StdX1Y2, MeanX2Y1, StdX2Y1, MeamMm, StdMm)*
Step2: End.

PDF created with pdfFactory trial version www.pdffactory.com

3.5.2 Fourier Based Features

To extract the set of Fourier based features, first the Fourier spectrum for each image sample is divided into slices (as shown in figure 3.3), and the average power of each slice is determined using the following equations:

$$\boldsymbol{Power(x,y) = \sqrt{F_r{}^2(x,y) + F_I{}^2(x,y)}} \qquad \ldots\ldots\ldots (\mathbf{3.10})$$

Where:

F_r is the real part, and

F_I is the imaginary part.

$$\boldsymbol{\overline{Power}(I) = \frac{1}{n(I)} \sum_{x,y \in S(I)} Power(x,y)} \qquad \ldots\ldots\ldots\ldots (\mathbf{3.11})$$

Where:

$\overline{Power}(I)$ is the average power for I^{th} slice (region),

$S(I) = \left\{(x,y) | I \times d \leq \sqrt{x^2 + y^2} \leq (I+1)d - 1\right\}$

$is\ the\ i^{th}\ slice\ region$

$n(I)$ = length ($S(I)$), i.e. the no. of FT coefficients belong to $S(I)$

Algorithm (3.8) illustrates the implemented steps for extracting Fourier features, considering that these features had been extracted after performing a quick Fourier transform on all of the input images classes.

PDF created with pdfFactory trial version www.pdffactory.com

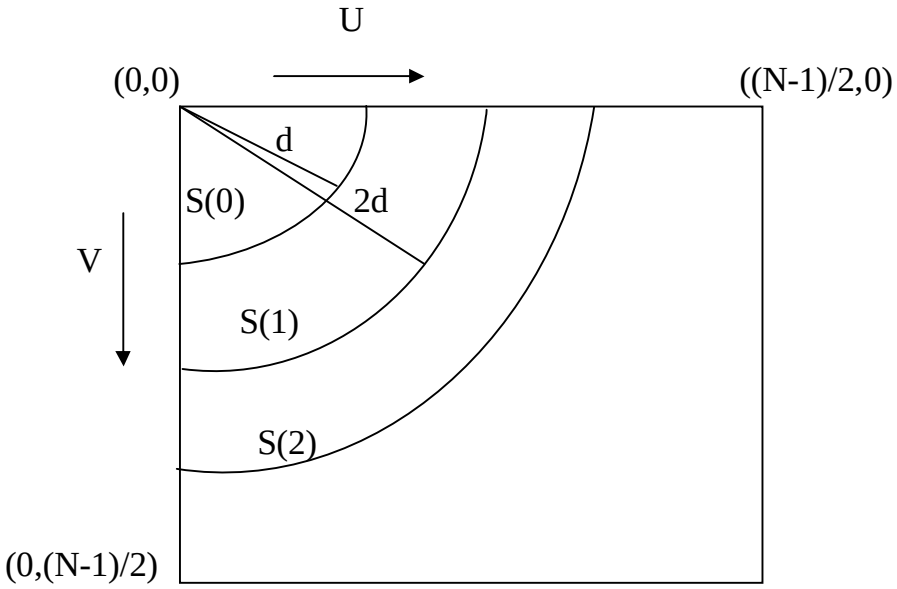


Figure (3.3) Fourier spectrum divided into slices

Algorithm (3.8) Fourier Features Extraction

Goal: *Extract Fourier features.*
Input: *TrBlk // is the transformed (real part) block*
TiBlk // is the transformed (imaginary part) block
BlkLen // is the block size
Nc // is the number of Fourier slices
Output: *FPowr // is the array containing Fourier power features.*

Step1: *Initialization*
Set *Stp ← BlkLen/Sqr(2)/Nc*
Set *S ← BlkLen\2*
Set *S1 ← S-1*
Set *S2 ← S*S*
Step2:For *all U,V* ***Do*** *{ where 0≤ U ≤BlkLen-1, 0≤ V ≤BlkLen-1 }*
Set *Powr(U,V) ← Sqr(TrBlk(U,V)^2 + TiBlk(U,V)^2)*
End For
For *all I* ***Do*** *{ where 0 ≤ I≤ Nc-1}*
Set *DenNo(I) ← 0 :* ***Set*** *DenPow(I) ← 0*
End For

PDF created with pdfFactory trial version www.pdffactory.com

```
        For all U Do  { where 1 ≤ U≤ S1}
            Set I ← Int(U/Stp)
            Set DenNo(I) ← DenNo(I)+2
            Set DenPow(I) ← DenPow(I)+Powr(U,0)+Powr(BlkLen-U,0)
        End For
       For all V Do  { where 1 ≤ V≤ S1}
            Set I ← Int(V/Stp)
            Set DenNo(I) ← DenNo(I)+2
            Set DenPow(I) ← DenPow(I)+Powr(0,V)+Powr(0,BlkLen-V)
        End For
       For all U Do  { where 1 ≤ U≤ S1}
            Set I ← Int(Sqr(U*U+S2/Stp)
            Set DenNo(I) ← DenNo(I)+2
            Set DenPow(I) ← DenPow(I)+Powr(U,S)+Powr(BlkLen-U,S)
        End For
        For all V Do  { where 1 ≤ V≤ S1}
            Set I ← Int(Sqr(S2+V*V)/Stp)
            Set DenNo(I) ← DenNo(I)+2
            Set DenPow(I) ← DenPow(I)+Powr(S,V)+Powr(S,BlkLen-V)
       End For
       Set I ← Int(S/Stp)
       Set DenNo(I) ← DenNo(I)+2
       Set DenPow(I) ← DenPow(I)+Powr(S,0)+Powr(0,S)
       Set DenNo(Nc-1) ← DenNo(Nc-1)+1
       Set DenPow(Nc-1) ← DenPow(Nc-1)+Powr(S,S)
      For all V Do  { where 1 ≤ V≤ S1}
           Set Vv ← BlkLen-V
           For all U Do  { where 1 ≤ U≤ S1}
                Set Uu ← BlkLen-U
                Set I ← Int(Sqr(U*U+V*V)/Stp)
                If  I > Nc-1 Then I=Nc-1
                Set DenNo(I) ← DenNo(I)+4
                Set DenPow(I) ← DenPow(I)+Powr(U,V)+Powr(Uu,V)
                                  + Powr(U,Vv)+Powr(Uu,Vv)
          End For
         For all I Do  { where 0 ≤ I≤ Nc-1}
            Set FPowr(I) ← DenPow(I) / DenNo(I)
        End For
       Return FPowr( )
Step3: End.
```


PDF created with pdfFactory trial version www.pdffactory.com

3.6 Feature Analysis

The matching process is done by measuring the minimum of the overall distance between a selected set of some extracted features. Before starting the matching process, the template feature vector for each class should be determined and registered in a database file. To compute the template vectors, the mean and standard deviation vectors for the feature vectors extracted from different image samples for each class are determined. From the values of the standard deviation vector elements, the variability of the features could be assessed.

The mean and standard deviation vectors are determined using the following equations [The03]:

$$\overline{F_{ip}} = \frac{1}{n}\sum_{j=1}^{n} f_{ijp} \qquad \ldots\ldots\ldots\ldots (3.12)$$

$$\sigma_{ip} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(f_{ijp} - \overline{F_{jp}})^2} \qquad \ldots\ldots\ldots\ldots (3.13)$$

Where:

f_{ijp} is the value of the i^{th} feature extracted from j^{th} sample image of the p picture.

N is the number of sample images taken for p picture.

$\overline{F_{ip}}$ is the mean value of the i^{th} feature for p picture.

σ_{ip} is the standard deviation of the i^{th} feature for p picture.

Also the features variability for each class are determined as the ratio between the standard deviation value by the mean value for each class.

PDF created with pdfFactory trial version www.pdffactory.com

$$CRate(i) = \frac{\sigma_i}{m_i} \quad \ldots\ldots\ldots\ldots\ldots (3.14)$$

Where:

CRate(i) is the rate of the i^{th} class, where $1 \leq i \leq Nclass$

σ_i is the standard deviation of the i^{th} class

m_i is the mean of the i^{th} class

Algorithm (3.9) illustrates the implemented steps for determining the mean, standard deviation and degree of variability for each class.

Algorithm (3.9)Computation of mean, standard deviation and Rate vectors

---

Goal: *compute mean, standard deviation and Rate vectors for each class*
Input: *feat //is the array of extracted features*
nclass // is the number of classes
npict // is the number of pictures per class
nsamp// is the number of samples per picture
nband // is the number of bands
nfeat // is the number of features in feat array
Output: *CAvg // is the class mean vector*
Cstd // is the class standard deviation vector
CRate // is the class rate

Step1: ***For*** *all iclass* ***Do*** *{where 1≤ $iclass \leq nclass$}*
For *all ipict* ***Do*** *{where 1≤ $ipict \leq npict$}*
For *all iband* ***Do*** *{where 1≤ $iband \leq nband$}*
For *all ifeat* ***Do*** *{where 1≤ $ifeat \leq nfeat$}*
For *all isamp* ***Do*** *{where 1≤ $isamp \leq nsamp$}*
Set *CAvg(iclass,iband,ifeat) ← CAvg(iclass,iband,ifeat)+ feat(ipict,isamp,iband,ifeat)*
Set *CStd(iclass,iband,ifeat) ← CStd(iclass,iband,ifeat)+ feat(ipict,isamp,iband,ifeat)^2*
End For : End For
End For : End For
For *all iband* ***Do*** *{where 1≤ $iband \leq nband$}*
For *all ifeat* ***Do*** *{where 1≤ $ifeat \leq nfeat$}*

PDF created with pdfFactory trial version www.pdffactory.com

```
        Set CAvg(iclass,iband,ifeat) ← CAvg(iclass,iband,ifeat)/
                                       (npict*nsamp)
        Set CStd(iclass,iband,ifeat) ← Sqr(CStd(iclass,iband,ifeat)/
                                  (npict*nsamp)- CAvg(iclass,iband,ifeat) ^2)
        Set CRate(iclass,iband,ifeat)← CStd(iclass,iband,ifeat)/
                                       CAvg(iclass,iband,ifeat)
      End For : End For : End For
   Return  CAvg, CStd, CRate.
Step2:End.
```

After computing the template feature vectors and feature variability, an indicator called "Ratio1" was determined for each feature using the following equation:

$$R_1 = \sum_{i=1}^{Nclass} CRate(i) \qquad \ldots\ldots\ldots\ldots (3.15)$$

Algorithm (3.10) illustrates the implemented steps for determining the Ratio1 (R_1) for each class features.

Algorithm (3.10)Computation of Ratio1

Goal: *compute Ratio1 for each class features*
Input: *CRate //is the array of class rate values*
nclass // is the number of classes
nband // is the number of bands
nfeat // is the number of features
Output: *R1 // is the Ratio1 vector*

PDF created with pdfFactory trial version www.pdffactory.com

Step1: ***For*** *all ifeat* ***Do*** *{where 1≤ $ifeat \leq nfeat$}*
For *all iband* ***Do*** *{where 1≤ $iband \leq nband$}*
Set *sum ← 0*
For *all iclass* ***Do*** *{where 1≤ $iclass \leq nclass$}*
Set *sum ← sum + CRate(iclass,iband,ifeat)*
End For
Set *R1(iband,ifeat)← sum*
End For
End For
Return *R1.*
Step2:End.

After computing the overall variability indicator Ratio1, another variability indicator called "Ratio2" was determined for each class features using the following equation:

$$R_2 = \frac{\sum_{i=1}^{Nclass} \sigma_i}{\sum_{i=1}^{Nclass-1} \sum_{j=i+1}^{Nclass} |mi - mj|} \quad \ldots\ldots\ldots\ldots\ldots\ldots (3.16)$$

Where:

σ_i is the standard deviation of the feature for i^{th} class

m_i is the mean of the feature for i^{th} class

m_j is the mean of the feature for j^{th} class

Algorithm (3.11) illustrates the implemented steps for determining the indicator "Ratio2 (R_2)" for all class features.

Algorithm (3.11)Computation of Ratio2

Goal: *compute Ratio2 for each class features*
Input: *CAvg //is the array of classes mean values*

PDF created with pdfFactory trial version www.pdffactory.com

CStd, //is the array of classes standard deviation values
nclass // is the number of classes
nband // is the number of bands
nfeat // is the number of features

Output: *R2 // is the Ratio2 vector*

Step1: For *all ifeat* ***Do*** *{where 1≤ $ifeat \leq nfeat$}*
For *all iband* ***Do*** *{where 1≤ $iband \leq nband$}*
Set *sum1 ← 0*
Set *sum2 ← 0*
For *all iclass* ***Do*** *{where 1≤ $iclass \leq nclass$}*
Set *sum1 ← sum1 + CStd(iclass,iband,ifeat)*
End For
For *all i* ***Do*** *{where 1≤ $i \leq nclass - 1$}*
For *all j* ***Do*** *{where $i + 1 \leq j \leq nclass$}*
Set *sum2 ← sum2 + Abs(CAvg(i,iband,ifeat)*
- CAvg(j,iband,ifeat))
End For
End For
Set *R2(iband,ifeat)← sum1/sum2*
End For
End For
Return *R2.*
Step2:End.

The total number of the taken image samples in this research work is 700 samples (i.e. 7 classes each with 10 pictures; and 10 samples taken for each picture). Instead of using an array with 4 indexes (i.e., feat), it is reduced to be an array with 3 indexes (newfeat) for programming simplicity. Where (newfeat) is the mapped array from (feat) array. The new indicator "Ratio3" is easily determined using the following equation:

$$R_3 = \frac{\sum_{i=1}^{Nclass} \sigma_i}{\sum_{j=1}^{newsamp} \sigma_j} \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.17)$$

PDF created with pdfFactory trial version www.pdffactory.com

Where:

σ_i is the standard deviation of the i^{th} class

σ_j is the standard deviation of the j^{th} sample

newsamp is the total number of samples (i.e. 700 samples)

Algorithm (3.12) illustrates the implemented steps for determining the Ratio3 (R_3) for all class features.

Algorithm (3.12)Computation of Ratio3

Goal: *compute R3 for each class feature.*
Input: *newfeat //is the samples vector*
nclass // is the number of classes
newsamp// is the total number of samples
nband // is the number of bands
nfeat // is the number of features
Output: *R3 // is the Ratio3 vector*

Step1:*compute the standard deviation for newsamp features*
For *all iband* ***Do*** *{where 1≤ iband ≤ nband}*
For *all ifeat* ***Do*** *{where 1≤ ifeat ≤ nfeat}*
Set *sum1 ← 0* **:** ***Set*** *sum2 ← 0*
For *all n* ***Do*** *{where 1≤ n ≤ newsamp}*
Set *sum1←sum1+newfeat(n,iband,ifeat)*
Set *sum2←sum2+newfeat(n,iband,ifeat)^2*
End For
Set *sum1←sum1 / newsamp*
Set *sum2←sum2 /newsamp – sum1*sum1*
Set *SampStd(iband,ifeat)←Sqr(sum2)*
End For
End For
Step2:*compute R3*
For *all iband* ***Do*** *{where 1≤ iband ≤ nband}*
For *all ifeat* ***Do*** *{where 1≤ ifeat ≤ nfeat}*
Set *sum ← 0*
For *all iclass* ***Do*** *{where 1≤ iclass ≤ nclass}*
Set *sum ←sum+CStd(iclass,iband,ifeat)*
End For

PDF created with pdfFactory trial version www.pdffactory.com

***Set** R3(iband,ifeat)←sum / SampStd(iband,ifeat)*
End For
End For
***Return** R3.*
Step3:End.

From the previous three vectors (i.e. R_1, R_2 and R_3), a unique vector called (*mutualfeat*) had been determined by founding the mutual features between them and adding it to (*mutualfeat*) vector. This can be done by sorting the three Ratio vectors ascending sort according to ratio value, and searching for the mutual features between them. Before searching to the mutual features, all of R_1, R_2 and R_3 files had been converted to a corresponding records containing the same three values (band, feat, R) in these files and sorting them ascending sort.

Algorithm (3.13) illustrates the implemented steps for sorting the Rec1 array after filling it from Ratio1 vector (R_1). The same steps had been performed to the remaining two vectors (R_2, R_3).

Algorithm (3.13)Fill and sort Rec1

Goal: *fill and sort the new record ascending sort*
Input: *FR1 //is a file containing R1 vector values*
nfeat // is the number of features
Output: *Rec1 // is a new sorting record contain FR1 values*

PDF created with pdfFactory trial version www.pdffactory.com

```
Step1:fill Rec1 from FR1 file
        Open FR1 for input as #1
        Set n ← nfeat*10
        For all i Do  {where 1≤ i ≤ n}
           Input #1, Rec1(i).band, Rec1(i).feat, Rec1(i).R
        End For
Step2:sort Rec1 ascending sort
        Do
           Set flag ← True
          For all i Do  {where 1≤ i ≤ n − 1}
              Set  j ←i + 1
              If Rec1(i).R > Rec1(j).R then
                 Set temp ← Rec1(i)
                 Set Rec1(i) ← Rec1(j)
                 Set Rec1(j) ← temp
                 Set flag ← False
             End If
          End For
          For all i Do  {where n − 1 ≤ i ≤ 1  step − 1}
              Set  j ←i + 1
              If Rec1(i).R > Rec1(j).R then
                 Set temp ← Rec1(i)
                 Set Rec1(i) ← Rec1(j)
                 Set Rec1(j) ← temp
                 Set flag ← False
              End If
          End For
       Loop Until flag← True
      Return  Rec1.
Step3:End.
```

After determining the new three sorted records (Rec1, Rec2 and Rec3), a specified number of features (NewF) had been taken from these records to search the mutual features (i.e. not all values in these three records should be taken in consideration when searching the mutual features because it's a time and memory cost operation). In this work, the

PDF created with pdfFactory trial version www.pdffactory.com

specified number of fields that had been taken from each record is 250 fields.

Algorithm (3.14) illustrates the implemented steps for determining the *mutualfeat* vector.

Algorithm (3.14) Compute mutual features vector

Goal: *compute the mutual features between Rec1, Rec2 and Rec3*
Input: *Rec1 //is a record containing Ratio1 values*
Rec2 //is a record containing Ratio2 values
Rec3 //is a record containing Ratio3 values
NewF // is the number of specified features
Output: *mutualfeat // is a mutual features vector*

```
Step1:Set NewF ← 250 : Set mutual ← 0
      For all i Do {where 1≤ i ≤ NewF}
        Set k ← Rec1(i).band : Set j ← Rec1(i).feat : Set L ← 0
        For all m Do {where 1≤ m ≤ NewF}
          If Rec2(m).feat = j and Rec2(m).band = k then
            Set L ← L+1 : Set m ← NewF
          End If
        End For
        If L>0 then
          For all m Do {where 1≤ m ≤ NewF}
            If Rec3(m).feat = j and Rec3(m).band = k then
              Set L ← L+1 : Set m ← NewF
            End If
          End For
        End If
        If L=2 then
          Set mutualfeat(0,mutual) ← k
          Set mutualfeat(0,mutual) ← j
          Set mutual ← mutual+1
        End If
      End For
      Return mutualfeat.
Step2:End.
```


PDF created with pdfFactory trial version www.pdffactory.com

The adopted mechanism to handle the feature analysis task was aimed to find out the lowest possible combinations of features that can lead to good texture recognition results. The size of features combination was gradually increased and at each test step the best sets of features which can give the highest recognition rate found using heuristic search. In this work, the sets of features combinations was started with single feature and gradually expanded to a pair of features, and then to a set consist of triple and so on. It is found that combinations of nine features could lead to acceptable levels of recognition rates. For matching purpose, the Euclidian distance measure was adopted. The relative Euclidian measure could be expressed mathematically as follows [The03]:

$$d(t,f) = \sum_{i=1}^{n} \left(\frac{t_i - f_i}{\sigma_i} \right)^2 \qquad \ldots\ldots\ldots\ldots\ldots\ldots (3.18)$$

Where:

d is the distance measure value

t is the template feature vector

f is the extracted feature vector for the unknown image

n is the number of used features in the matching process (best features)

Algorithm (3.15) illustrates the implemented steps to perform the exhaustive search for different possible combinations of two features. The features pair that lead to maximum *PassNo* value, is recorded as a best features pair that could be expanded to hold a third feature. After finding the best triple combination of features, it is expanded to hold another

PDF created with pdfFactory trial version www.pdffactory.com

feature, and so on until we reach the combination of features that have optimal recognition rate.

Algorithm (3.15) Combination of two features

Goal: *compute the distance using 2 features combination.*
Input: *mutualfeat // is a mutual features vector*
mutual //is a number of features in mutualfeat vector
newsamp// is the total number of samples
newfeat //is the samples features vector
nclass // is the number of classes
CAvg //is the array of classes mean values
CStd, //is the array of classes standard deviation values
Output: *PassNo // is the array of each combination success rate*
Iband1// the first recorded band
Ifeat1// the first recorded feature

```
Step1:For all i Do {where 1≤ i ≤ mutual − 1}
        Set iband1 ← mutualfeat(0,i): Set ifeat1 ← mutualfeat(1,i)
        For all j Do {where i+1≤ j ≤ mutual}
          Set iband2 ← mutualfeat(0,j): Set ifeat2 ← mutualfeat(1,j)
          Set PassNo(i,j) ← 0
          For all isamp Do {where 1≤ isamp ≤ newsamp}
            Set Cls ← ((isamp-1)\100)+1
           For all iclass Do {where 1≤ iclass ≤ nclass}
             Set Distance(iclass) ← Abs((CAvg(iclass,iband1,ifeat1)-
              Newfeat(isamp,iband1,ifeat1))/CStd(iclass,iband1,ifeat1))+
                                    Abs((CAvg(iclass,iband2,ifeat2)-
              Newfeat(isamp,iband2,ifeat2))/CStd(iclass,iband2,ifeat2))
           End For
           Set min ← Distance(1): Set Clsmin ← 1
          For all k Do {where 2≤ k ≤ nclass}
            If Distance(k) < min then
              Set min ← Distance(k)
              Set Clsmin ← k
            End If
          End For
          If Clsmin = Cls then
            Set PassNo(i,j) ← PassNo(i,j)+1
          End If
        End For
```


PDF created with pdfFactory trial version www.pdffactory.com

```
    End For
  End For
Step2: Search the first recorded feature(have max PassNo)
      Set max ← 0
      For all i Do  {where 1≤ i ≤ mutual − 1}
        For all j Do  {where i+1≤ j ≤ mutual}
          If PassNo(i, j) > max then
            Set max ← PassNo(i, j)
            Set IBand1 ← FMutual(0, i)
            Set  IFeat1 ← FMutual(1, i)
          End If
        End For
    End For
    Return  PassNo, IBand1, IFeat1
Step3:End.
```

Algorithm (3.16) illustrates the implemented steps to perform the exhaustive search for different possible combinations of three features. The same steps had been performed to implement the combination of four, five, six, seven, eight, and nine features.

PDF created with pdfFactory trial version www.pdffactory.com

Algorithm (3.16) Combination of three features

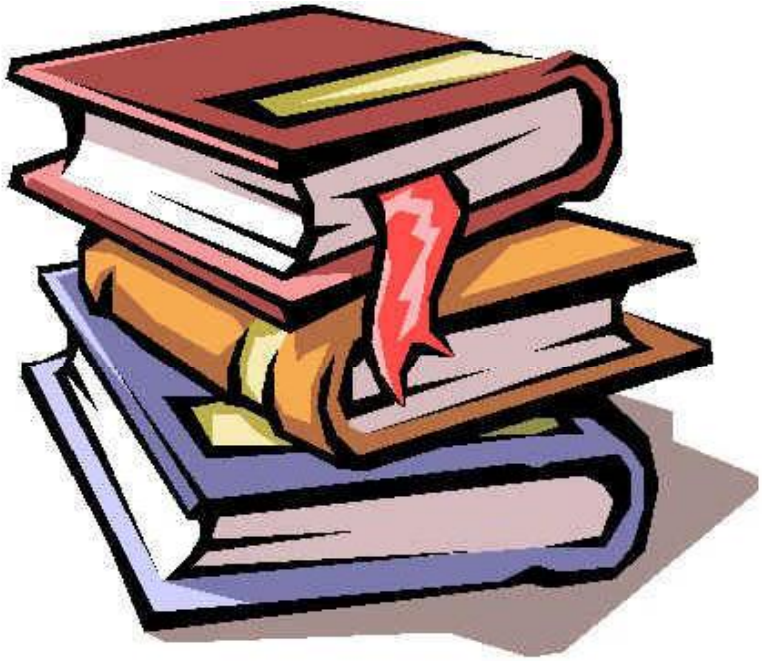
Goal: *compute the distance using 2 features combination.*
Input: *mutualfeat // is a mutual features vector*
mutual //is a number of features in mutualfeat vector
newsamp// is the total number of samples
newfeat //is the samples features vector
nclass // is the number of classes
CAvg //is the array of classes mean values
CStd, //is the array of classes standard deviation values
Iband1// the first recorded band
Ifeat1// the first recorded feature
Output: *PassNo // is the array of each combination success rate*
Iband2// the second recorded band
Ifeat2// the second recorded feature

Step1:For *all i* ***Do*** *{where* $1 \leq i \leq mutual - 1$*}*
Set *iband2 ← mutualfeat(0,i):* ***Set*** *ifeat2 ← mutualfeat(1,i)*
For *all j* ***Do*** *{where* $i+1 \leq j \leq mutual$*}*
Set *iband3 ← mutualfeat(0,j):* ***Set*** *ifeat3 ← mutualfeat(1,j)*
Set *PassNo(i,j) ← 0*
For *all isamp* ***Do*** *{where* $1 \leq isamp \leq newsamp$*}*
Set *Cls ← ((isamp-1)\100)+1*
For *all iclass* ***Do*** *{where* $1 \leq iclass \leq nclass$*}*
Set *Distance(iclass) ← Abs((CAvg(iclass,iband1,ifeat1)-*
Newfeat(isamp,iband1,ifeat1))/CStd(iclass,iband1,ifeat1))+
Abs((CAvg(iclass,iband2,ifeat2)-
Newfeat(isamp,iband2,ifeat2))/CStd(iclass,iband2,ifeat2))+
Abs((CAvg(iclass,iband3,ifeat3)-
Newfeat(isamp,iband3,ifeat3))/CStd(iclass,iband3,ifeat3))
End For
Set *min ← Distance(1):* ***Set*** *Clsmin ← 1*
For *all k* ***Do*** *{where* $2 \leq k \leq nclass$*}*
If *Distance(k) < min* ***then***
Set *min ← Distance(k)*
Set *Clsmin ← k*
End If
End For

PDF created with pdfFactory trial version www.pdffactory.com

```
            If Clsmin = Cls then
               Set PassNo(i,j) ← PassNo(i,j)+1
            End If
         End For
      End For
   End For
Step2: Search the second recorded feature(have max PassNo)
      Set max ← 0
      For all i Do  {where 1≤ i ≤ mutual − 1}
         For all j Do  {where i+1≤ j ≤ mutual}
            If PassNo(i, j) > max then
               Set max ← PassNo(i, j)
               Set IBand2 ← FMutual(0, i)
               Set  IFeat2 ← FMutual(1, i)
            End If
         End For
      End For
      Return  PassNo, IBand2, IFeat2
Step3:End.
```


PDF created with pdfFactory trial version www.pdffactory.com



Chapter Four

PDF created with pdfFactory trial version www.pdffactory.com

Chapter Four
Tests and Results

4.1 Introduction

In this chapter, the results of some conducted tests are presented to evaluate the performance of the established texture recognition system (TRS), whose structure was given in previous chapter. The effects of different involved system parameters on the overall system performance will be explored.

4.2 Test Strategy

To test the performance of the established system, first the system was trained using 70 images (belong to 7 classes) each of size (256*256), and color resolution 24 bit per pixel as a training data. To test the effect of any parameter, the values of other parameters were set fixed to specific values while the values of the considered parameter are varied. This testing mechanism was repeated over all tested parameters.

At first, the feature vectors for all training set of images are extracted and analyzed to test the efficiency of the recognition system. Then, as a second testing phase, the system performance was tested using another set of images which are not part of the training set.

The main stages of the established TRS are: feature extraction and recognition (i.e. decision making) using two sets of features (i.e., density distribution and Fourier transform).

PDF created with pdfFactory trial version www.pdffactory.com

The recognition based on density distribution is involved with two parameters. (Namely: block length, and number of density thresholds). Fourier transform is involved with two parameters. (Namely: block length, and number of Fourier slices). These parameters have considerable effects on the discrimination power of the extracted feature vector.

4.3 Test Materials (Images Samples)

In this research, the collected samples of texture images are classified into two classes:

4.3.1 Training Samples

The use of large amounts of data in training phase improves the probability of getting more precise knowledge about the behaviors of the representative features; this will make the training results more stable. The overall collected images are belong to seven classes of texture:

1. The first class is the Bookshelves texture, it is consist of 10 images; and 10 samples (cut outs) are taken from each image. Figure (4.1) presents these 10 images.
2. The second class is the Food texture, it is also consist of 10 images; and 10 samples are taken from each image. Figure (4.2) presents these images.
3. The third class is the Bamboo texture, it is consist of 10 images; and 10 samples are taken from each image. Figure (4.3) presents these 10 images.
4. The fourth class is the Broken mosaic texture, it is consist of 10 images; and 10 cut outs are taken from each image. Figure (4.4) presents these 10 images.

PDF created with pdfFactory trial version www.pdffactory.com

5. The fourth class is the Twill weaves texture, it is consist of 10 images; and 10 cut outs are taken from each image. Figure (4.5) presents these 10 images.
6. The fourth class is the Quantum weaves texture, it is consist of 10 images; and 10 cut outs are taken from each image. Figure (4.6) presents these 10 images.
7. The last class is the Coin stack texture, it is also consist of 10 images; and 10 samples are taken from each image. Figure (4.7) presents these images.

Table (C.1) presents feature vectors of the templates for these seven classes using density distribution method. Table (D.1) presents feature vectors of the templates for these seven classes using Fourier transform.

PDF created with pdfFactory trial version www.pdffactory.com

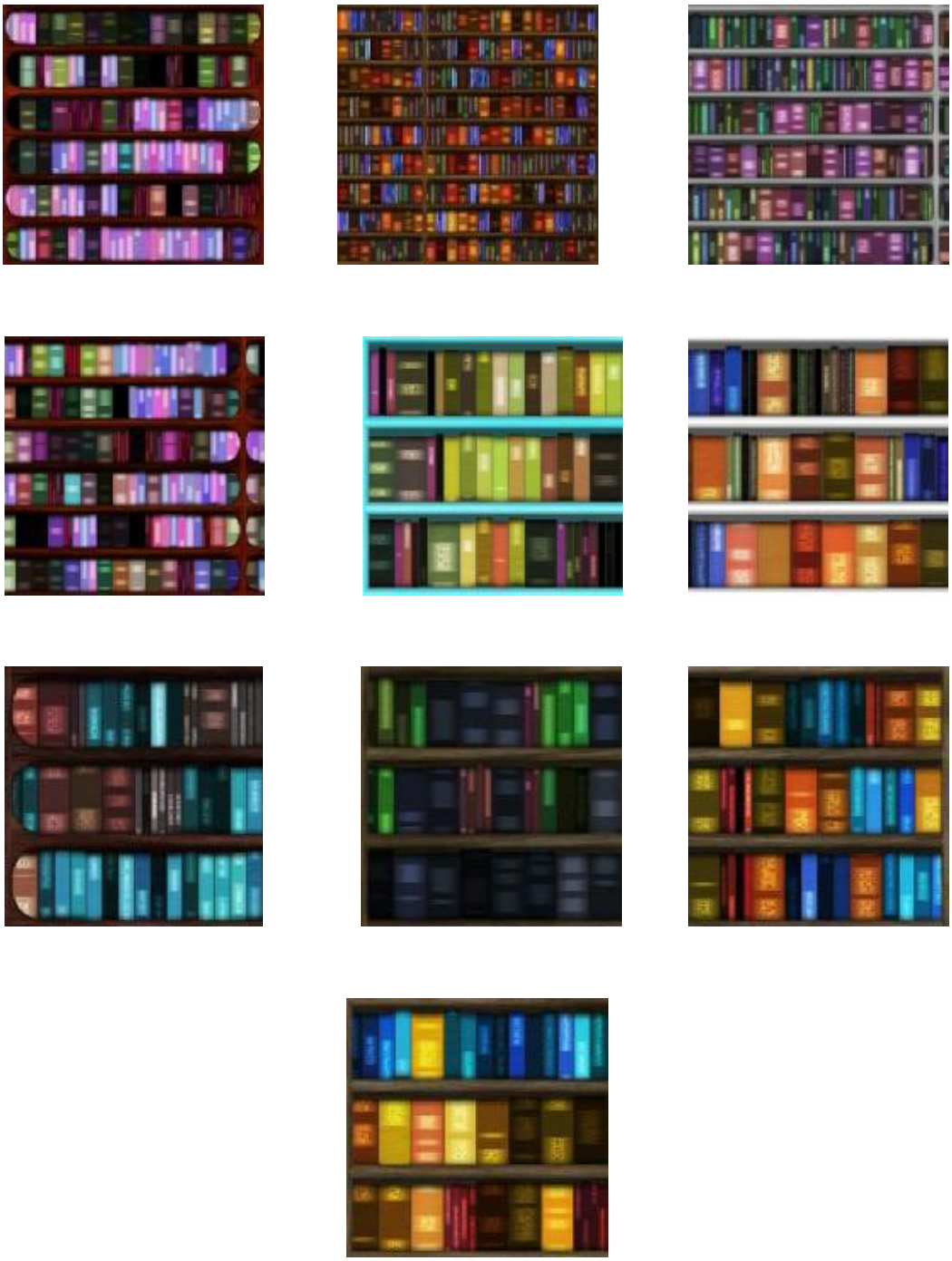


Figure (4.1) Class-1 images (Bookshelves)

PDF created with pdfFactory trial version www.pdffactory.com

Figure (4.2) Class-2 images (Food)

PDF created with pdfFactory trial version www.pdffactory.com

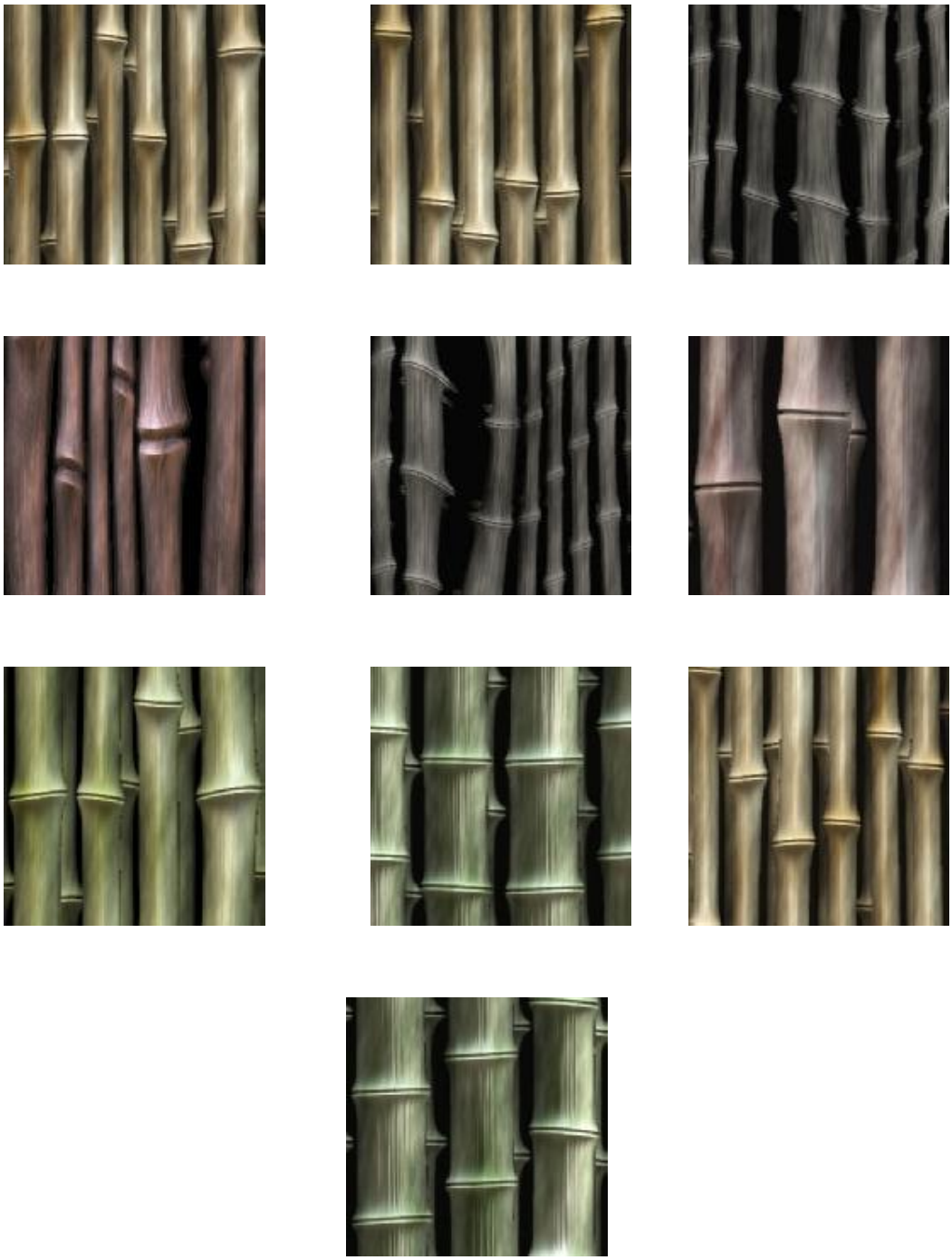


Figure (4.3) Class-3 images (Bamboo)

PDF created with pdfFactory trial version www.pdffactory.com

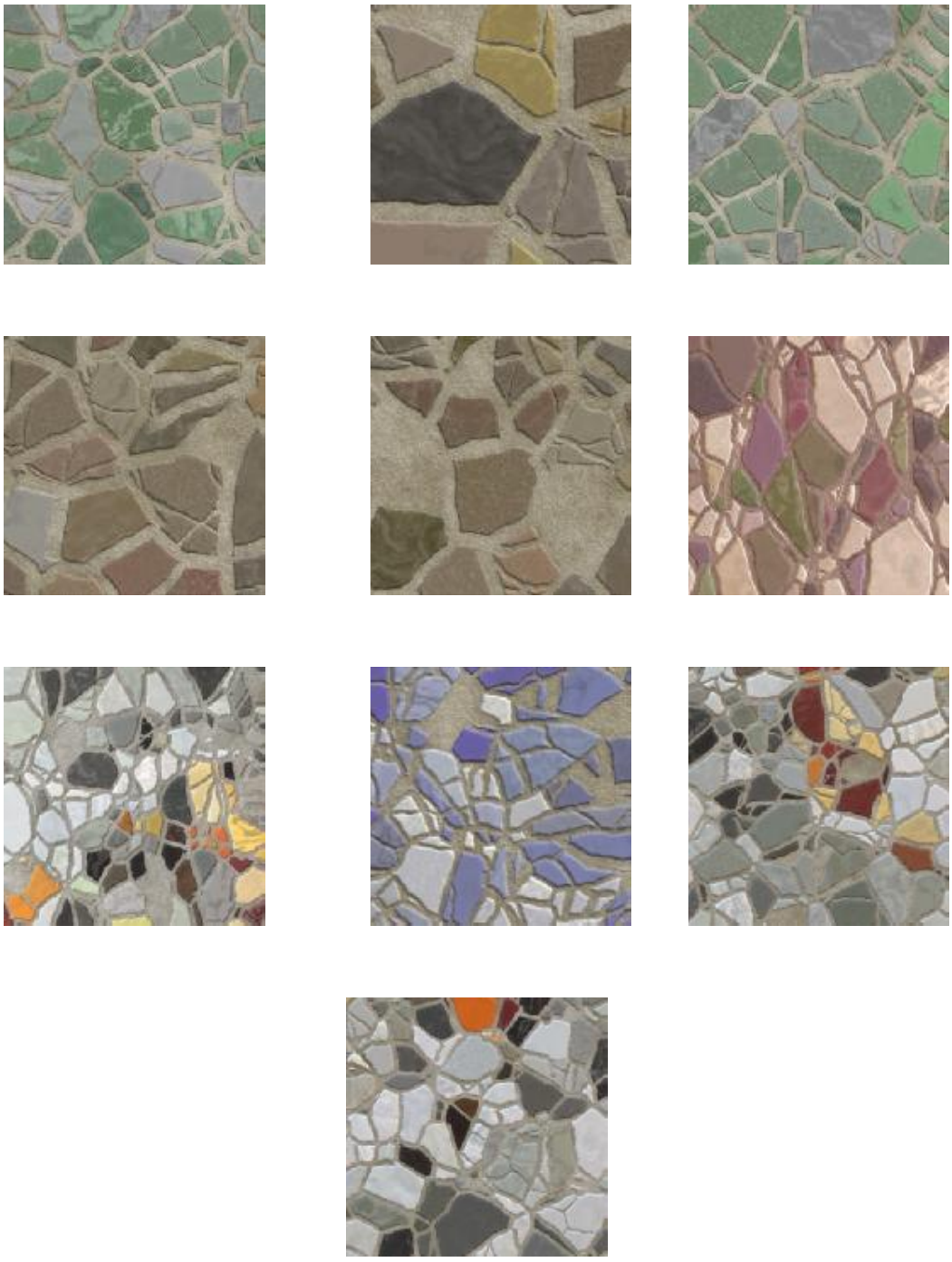


Figure (4.4) Class-4 images (Broken mosaic)

PDF created with pdfFactory trial version www.pdffactory.com

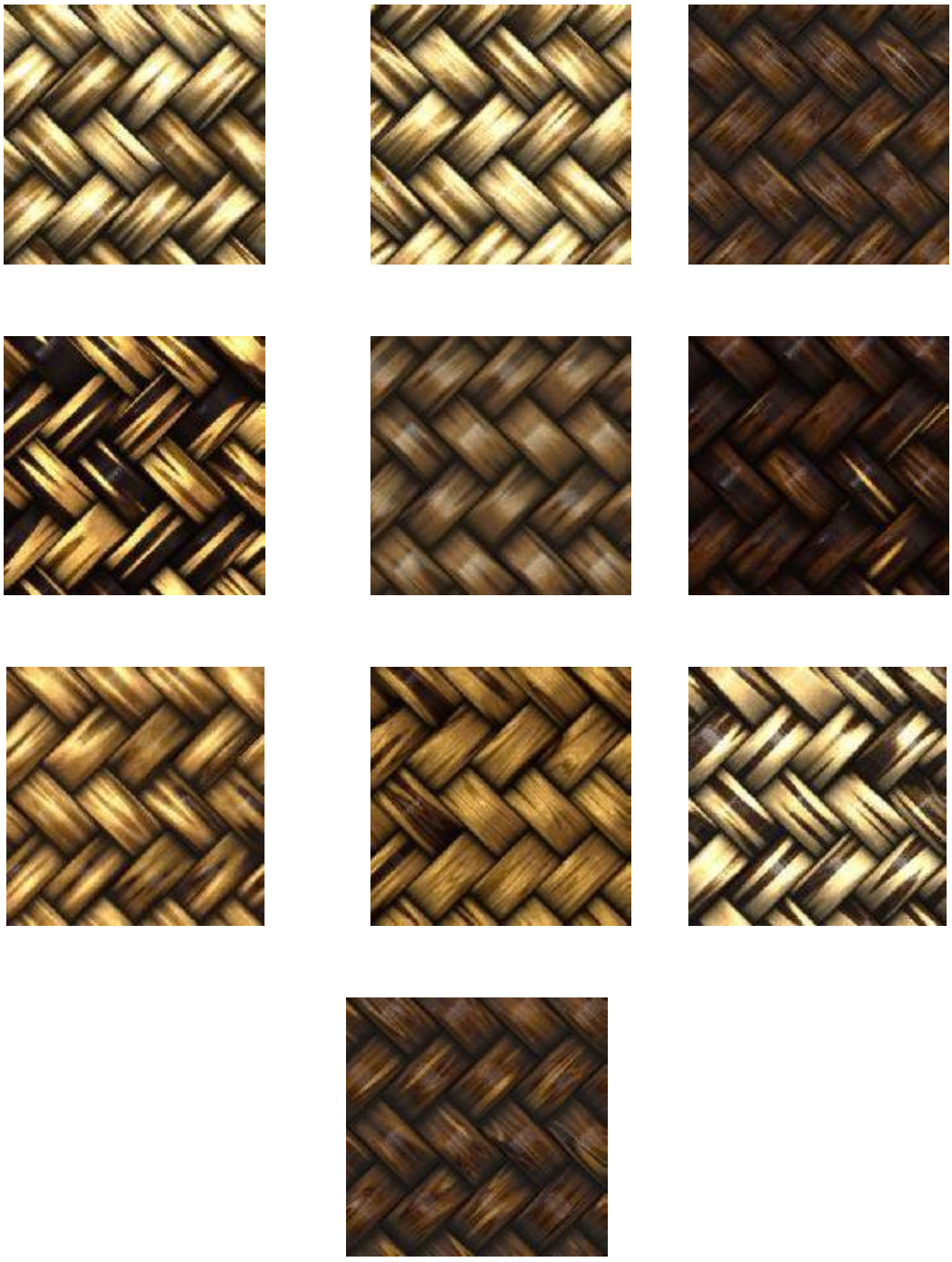


Figure (4.5) Class-5 images (Twill weaves)

PDF created with pdfFactory trial version www.pdffactory.com

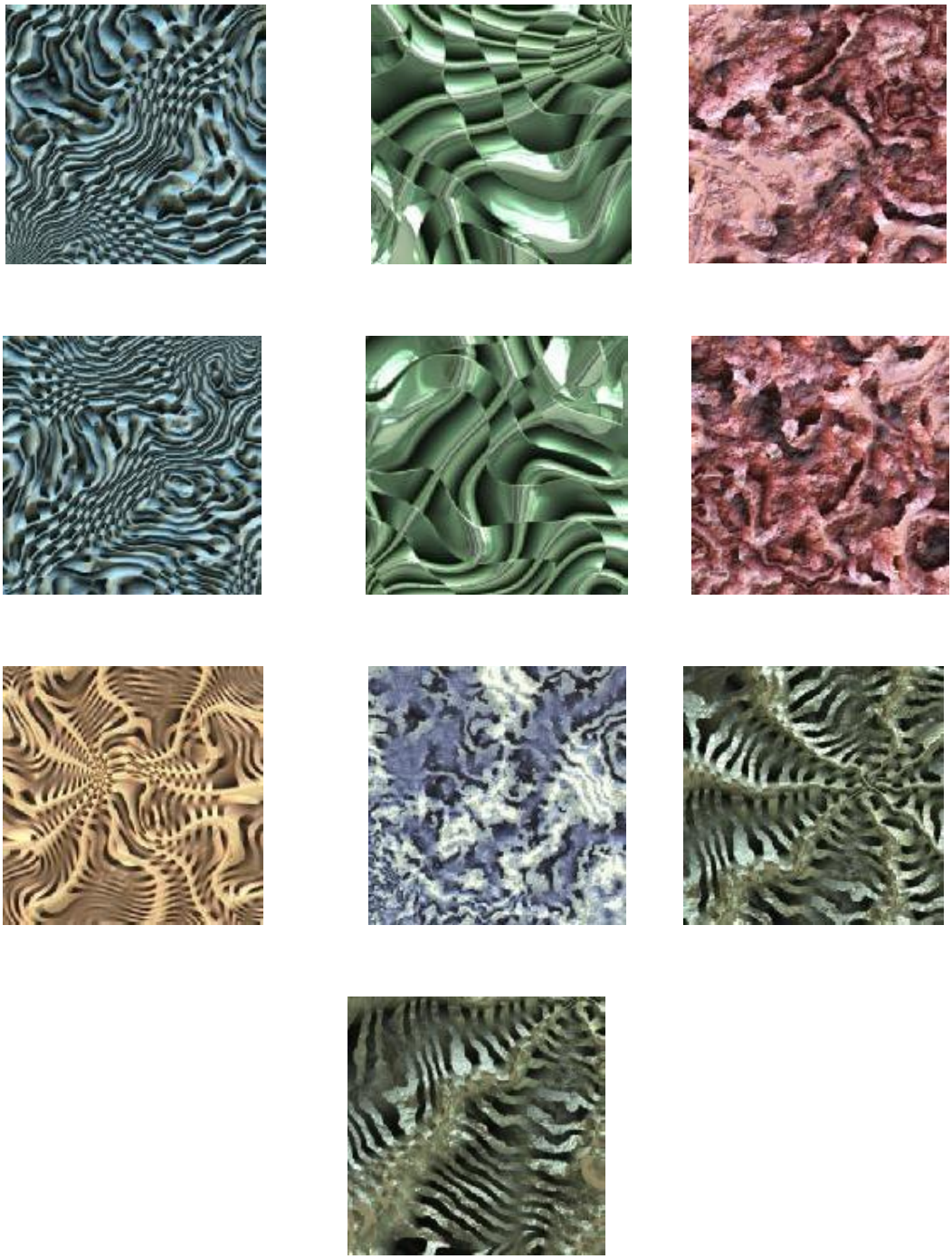


Figure (4.6) Class-6 images (Quantum weaves)

PDF created with pdfFactory trial version www.pdffactory.com

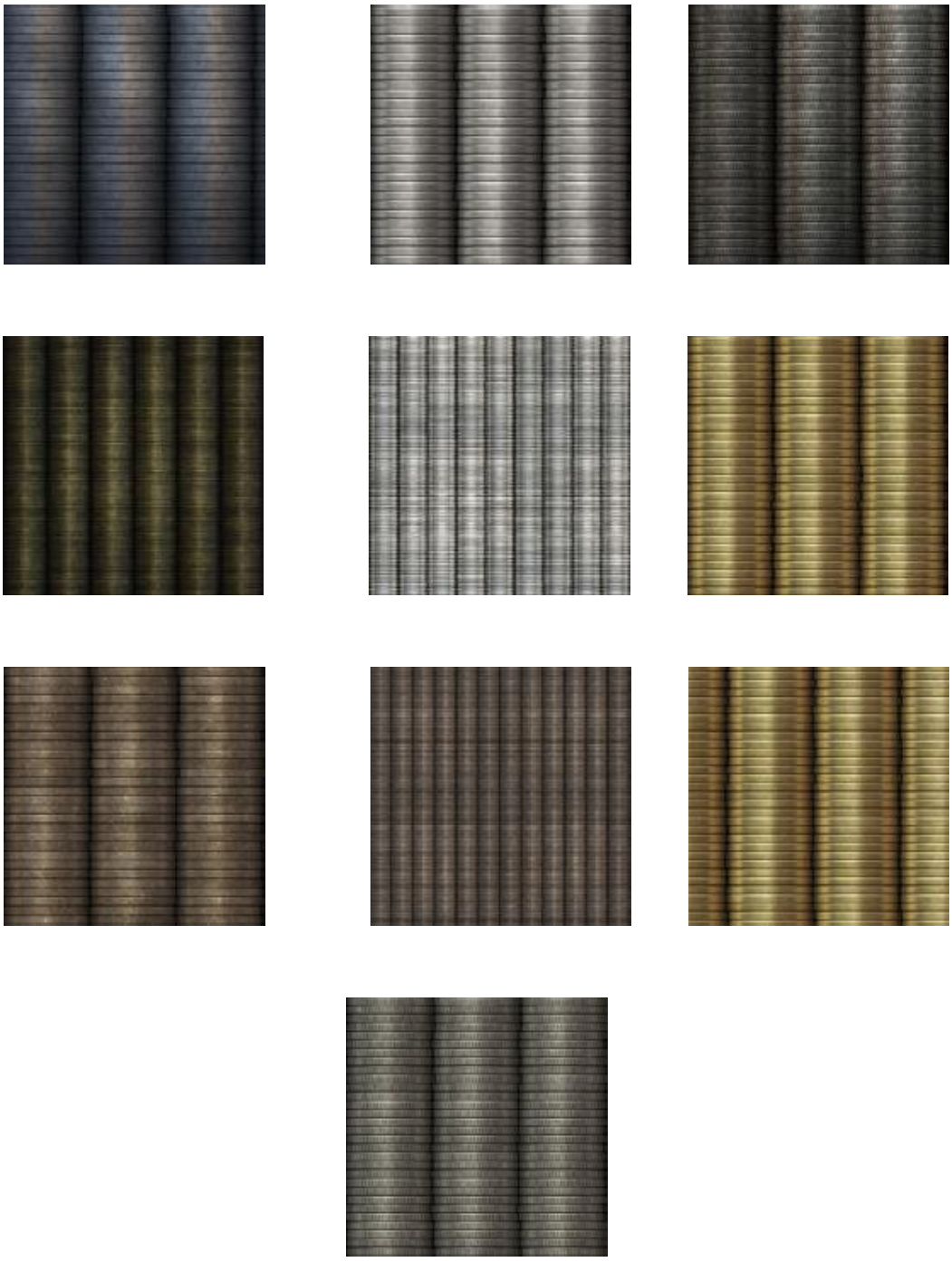


Figure (4.7) Class-7 images (Coin stack)

PDF created with pdfFactory trial version www.pdffactory.com

4.3.2 Test Samples

Once the TRS is properly trained, it is tested using a set of test images. The taken test set consists of 16 images; and 10 samples are taken from each image. In this test, the assessed class index for each sample is compared with its actual class index to determine the recognition accuracy of the system. Figure (4.8) presents some of the tested images used in this project.

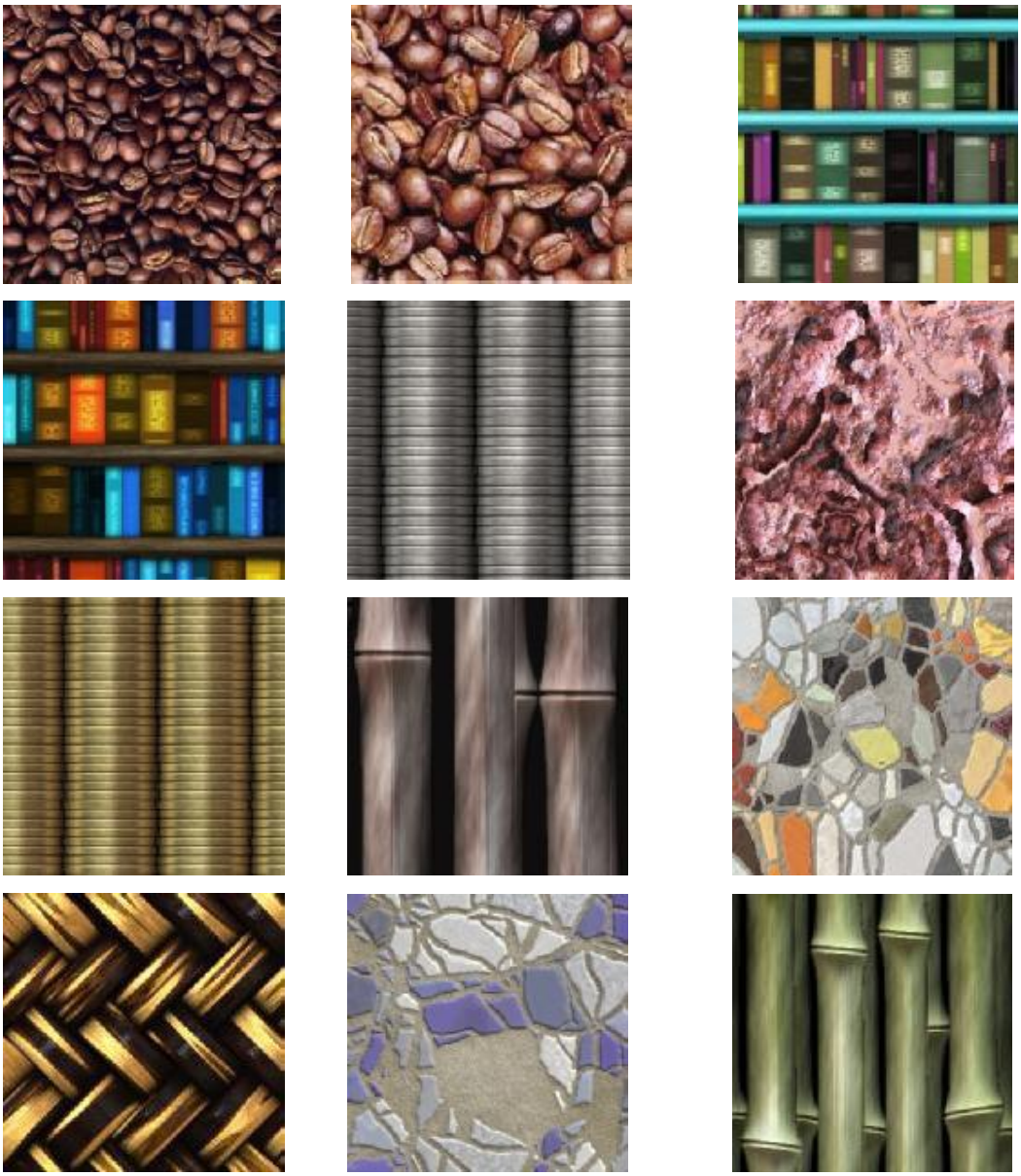


Figure (4.8) Testing Images

PDF created with pdfFactory trial version www.pdffactory.com

4.4 Testing Results

Various tests have been performed to find the best values for the density parameters (i.e., block length and density thresholds) and Fourier parameters (i.e., block length and Fourier slices). If the value of block size is too small, then the analyzed area within the block will greatly affected by the unstable local properties of the image signal. While if the block size is too large, then, it may not be possible highlight the impact of small details on the features. There is no analytical method for finding the optimal block size; it is usually assessed using trail mechanism (i.e. trying different block length values and monitoring the system performance). Same procedure had been done for the second parameter (i.e., density thresholds / Fourier slices), it suitable value is assessed by monitoring the impact of its variation of system performance.

The environment used to conduct the tests is a single PC with Celeron (M) Processor (1.40 GHz), and 256 MB of RAM.

4.4.1 The Effects of Block-length and Density-thresholds on the System Success Rate

The following test results reflect the system behavior when the block length and number of density thresholds are varied:

1. In this set of test results, the determined system success rate tested by setting the block length (BlkLen) to (100) and trying four different values for density threshold (NT=3 , 4, 5, and 6). The max success rate that had been assessed in this test is 84.4 % when (NT=3). Table (4.1) presents the success rate values for this test.

PDF created with pdfFactory trial version www.pdffactory.com

Table (4.1) The success rate values for BlkLen=100

Combined Features.	No. of Density thresholds			
	3	4	5	6
1 feat.	43.0%	45.9%	49.1%	47.3%
2 feat.	63.7%	68.4%	64.3%	65.3%
3 feat.	70.1%	72.9%	69.1%	71.4%
4 feat.	75.7%	74.6%	73.4%	75.3%
5 feat.	77.3%	76.7%	76.3%	78.4%
6 feat.	80.7%	79.1%	78.7%	79.6%
7 feat.	81.7%	80.9%	79.7%	80.9%
8 feat.	83.4%	81.7%	81.0%	82.4%
9 feat.	84.4%	83.1%	81.3%	83.0%

2. In this set of test results, the determined system success rate tested by setting the block length (BlkLen) to (130) and trying four different values for density threshold (NT=3 ,4 , 5, and 6). The max success rate that had been assessed in this test is 87.1 % when (NT=6). Table (4.2) presents the success rate values for this test.

Table (4.2) The success rate values for BlkLen=130

Combined Features.	No. of Density thresholds			
	3	4	5	6
1 feat.	44.3%	48.0%	51.0%	50.7%
2 feat.	66.4%	67.1%	69.3%	72.3%
3 feat.	73.4%	73.6%	72.7%	75.4%
4 feat.	80.4%	77.7%	75.7%	79.9%
5 feat.	82.6%	80.4%	78.7%	82.0%
6 feat.	83.6%	83.3%	80.9%	84.3%
7 feat.	83.4%	84.1%	81.9%	85.4%
8 feat.	84.9%	85.0%	83.0%	86.3%
9 feat.	85.4%	86.0%	84.9%	87.1%

PDF created with pdfFactory trial version www.pdffactory.com

3. In this set of test results, the determined system success rate tested by setting the block length (BlkLen) to (160) and trying four different values for density threshold (NT=3 ,4 , 5, and 6). The max success rate that had been assessed in this test is 91.4 % when (NT=4). Table (4.3) presents the success rate values for this test.

Table (4.3) The success rate values for BlkLen=160

Combined Features.	No. of Density thresholds			
	3	4	5	6
1 feat.	49.6%	46.7%	52.4%	53.3%
2 feat.	70.3%	68.7%	72.1%	71.1%
3 feat.	75.1%	72.3%	77.4%	79.4%
4 feat.	81.9%	78.6%	82.9%	82.6%
5 feat.	84.1%	83.1%	84.9%	85.1%
6 feat.	85.6%	87.9%	87.4%	87.1%
7 feat.	86.6%	89.4%	89.1%	88.3%
8 feat.	87.4%	91.0%	90.7%	89.1%
9 feat.	87.0%	91.4%	91.1%	90.1%

4. In this set of test results, the determined system success rate tested by setting the block length (BlkLen) to (180) and trying four different values for density threshold (NT=3 ,4 , 5, and 6). The max success rate that had been assessed in this test is 94.9 % when (NT=5). Table (4.4) presents the success rate values for this test.

PDF created with pdfFactory trial version www.pdffactory.com

Table (4.4) The success rate values for BlkLen=180

Combined Features.	No. of Density thresholds			
	3	4	5	6
1 feat.	48.9%	51.3%	59.7%	54.1%
2 feat.	68.1%	69.1%	72.9%	70.9%
3 feat.	73.9%	76.3%	82.7%	78.9%
4 feat.	79.3%	81.1%	85.7%	81.7%
5 feat.	82.4%	84.9%	89.4%	85.4%
6 feat.	85.4%	86.4%	92.1%	87.9%
7 feat.	86.4%	88.0%	93.7%	88.9%
8 feat.	87.4%	88.9%	93.6%	89.0%
9 feat.	87.7%	90.1%	94.9%	91.0%

4.4.2 The Effects of Block-length and Fourier-slices on the System Success Rate

The following test results reflect the system behavior when the block length and number of Fourier slices are varied:

1. In this set of test results, the determined system success rate tested by setting the block length (BlkLen) to (100) and trying three different values for Fourier slices (NS=4, 8, and 10). The max success rate that had been assessed in this test is 97.7 % when (NS=10). Table (4.5) presents the success rate values for this test.

PDF created with pdfFactory trial version www.pdffactory.com

Table (4.5) The success rate values for BlkLen=100

Combined Features.	No. of Fourier slices		
	4	8	10
1 feat.	54.4%	51.7%	52.9%
2 feat.	79.4%	79.7%	75.9%
3 feat.	89.1%	93.0%	88.9%
4 feat.	92.6%	95.0%	93.3%
5 feat.	94.7%	95.7%	94.7%
6 feat.	95.1%	97.3%	96.1%
7 feat.	96.1%	97.3%	97.0%
8 feat.	96.4%	97.3%	97.3%
9 feat.	96.4%	97.4%	97.7%

2. In this set of test results, the determined system success rate tested by setting the block length (BlkLen) to (130) and trying three different values for Fourier slices (NS=4, 8, and 10). The max success rate that had been assessed in this test is 98.7 % when (NS=10). Table (4.6) presents the success rate values for this test.

Table (4.6) The success rate values for BlkLen=130

Combined Features.	No. of Fourier slices		
	4	8	10
1 feat.	56.3%	59.4%	55.0%
2 feat.	84.1%	83.1%	79.4%
3 feat.	92.3%	93.7%	92.7%
4 feat.	94.4%	96.4%	95.9%
5 feat.	96.9%	96.9%	97.3%
6 feat.	97.6%	97.3%	98.0%
7 feat.	97.9%	97.4%	98.3%
8 feat.	97.9%	97.6%	98.7%
9 feat.	97.7%	97.9%	98.7%

PDF created with pdfFactory trial version www.pdffactory.com

3. In this set of test results, the determined system success rate tested by setting the block length (BlkLen) to (160) and trying three different values for Fourier slices (NS=4, 8, and 10). The max success rate that had been assessed in this test is 99.6 % when (NS=8). Table (4.7) presents the success rate values for this test.

Table (4.7) The success rate values for BlkLen=160

Combined Features.	No. of Fourier slices		
	4	8	10
1 feat.	59.0%	58.1%	56.7%
2 feat.	86.6%	83.9%	83.0%
3 feat.	93.0%	94.9%	93.0%
4 feat.	94.6%	97.3%	95.3%
5 feat.	96.1%	98.4%	96.0%
6 feat.	97.9%	98.7%	96.2%
7 feat.	98.4%	99.1%	97.6%
8 feat.	98.6%	99.4%	98.0%
9 feat.	98.6%	99.6%	97.9%

4. In this set of test results, the determined system success rate tested by setting the block length (BlkLen) to (180) and trying three different values for Fourier slices (NS=4, 8, and 10). The max success rate that had been assessed in this test is 99.9% when (NS=8). Table (4.8) presents the success rate values for this test.

PDF created with pdfFactory trial version www.pdffactory.com

Table (4.8) The success rate values for BlkLen=180

Combined Features.	No. of Fourier slices		
	4	8	10
1 feat.	58.3%	61.1%	57.4%
2 feat.	85.4%	86.7%	84.9%
3 feat.	93.3%	96.3%	93.7%
4 feat.	95.1%	97.7%	95.9%
5 feat.	96.4%	98.7%	97.7%
6 feat.	97.0%	99.4%	98.3%
7 feat.	98.0%	99.4%	98.6%
8 feat.	97.9%	99.7%	98.6%
9 feat.	98.0%	99.9%	98.6%

As seen from the previous tests results, the max success rate that had been assessed using density distribution is 94.9 % when (BlkLen=180) and (DT=5) using the combination of nine features. These features are: (band 1, feat 49), (band 9, feat 50), (band 10, feat 39), (band 1, feat 14), (band 3, feat 10), (band 9, feat 14), (band 10, feat 43), (band 3, feat 53),and (band 8, feat 5). While, the max success rate that had been assessed using Fourier transform is 99.9 % when (BlkLen=180) and (NS=8) using the combination of nine features. These features are: (band 10, feat 0), (band 9, feat 7), (band 6, feat 4), (band 1, feat 7), (band 1, feat 5), (band 8, feat 1), (band 7, feat 5), (band 6, feat 6),and (band 1, feat 6). This set of features (best features) that assess the max success rate had been stored in a vector to be used in the recognition phase.

The last stage in the established system is the feature analysis and evaluation stage. The relative Euclidian distance measure had been used to determine the degree of similarity between the template vectors for each class and the features vector for the unknown image (test image).

PDF created with pdfFactory trial version www.pdffactory.com

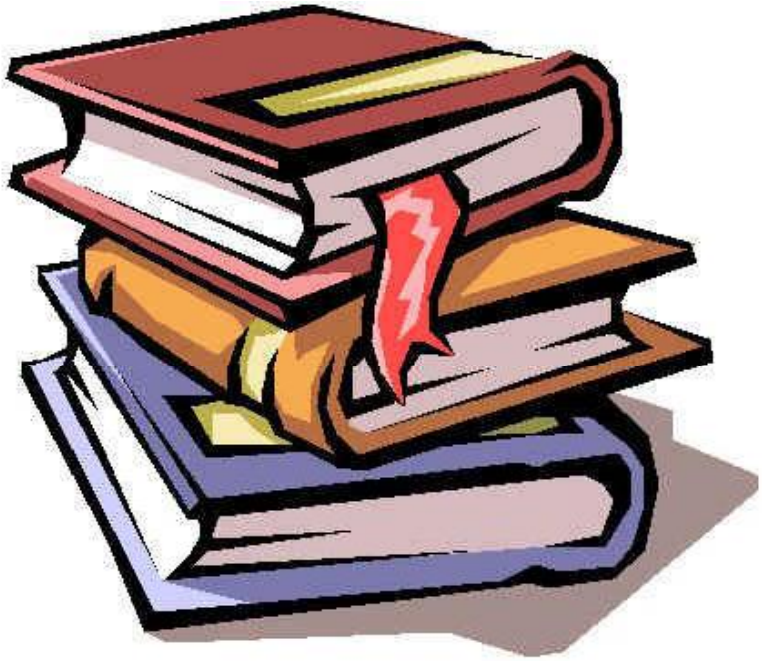
4.5 Calculation of Execution Time

Several performance tests have been applied. In this paragraph, the difference in execution time between the two methods (i.e., density distribution and Fourier transform) is listed. For the considered seven classes (i.e., Bookshelves, Food, Bamboo, Broken mosaic, Twill weaves, Quantum weaves, Coin stack) the attained maximum runtime value when using density distribution method was (48.10938 sec.) , and (1048.383 sec.) for the Fourier transform method. Table (4.9) presents the execution time values (in second) for these seven classes.

Table (4.9) The execution time values

Class	Execution time (second)	
	Density slicing	Fourier transform
Bookshelves	48.10938	991.8203
Food	39.75	1034.75
Bamboo	35.47656	1048.383
Broken mosaic	40.28125	1026.141
Twill weaves	42.89844	959.3984
Quantum weaves	44.24219	1016.234
Coin stack	35.39453	972.9609

PDF created with pdfFactory trial version www.pdffactory.com



Chapter Five

PDF created with pdfFactory trial version www.pdffactory.com

Chapter Five
Conclusions and Suggestions

5.1 Introduction

In this chapter, a list of remarks is presented; these remarks are derived from the investigation of the test results shown in chapter four. Also, some suggestions for a future work are given.

5.2 Conclusions

1. The block size has an effect on the success rate. A small block size leads to less success rate and vice versa.
2. The number of density thresholds has significant effect on the success rate, especially when the number of used features is taken high. The increase in number of thresholds decreases the recognition efficiency.
3. The experimental results showed that the best success rate (94.9%) had been achieved with the parameters set [block size (180) and density threshold (5), with the combination of nine features] using density distribution method.
4. The experimental results showed that the best success rate (99.9%) had been achieved with the parameters set [block size (180) and Fourier slices (8), with the combination of nine features] using Fourier transform method.

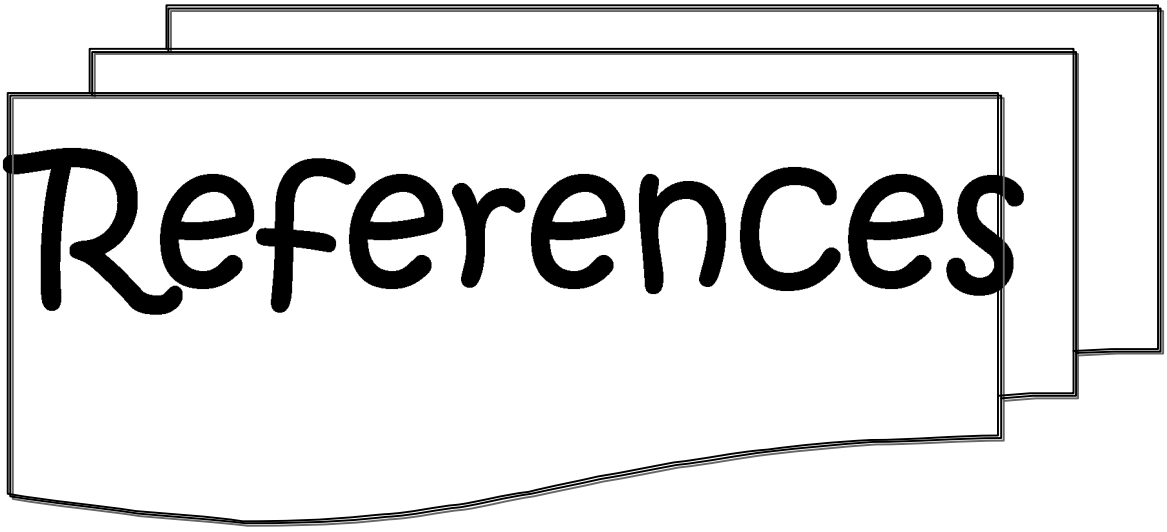
PDF created with pdfFactory trial version www.pdffactory.com

5. For the same block size, the density distribution method takes time less than (i.e., faster than) Fourier transform time (21 times). The attained max runtime value in density distribution method was (48.10938 sec.), while the attained max runtime value in Fourier transform method was (1048.383 sec.).

5.3 Suggestions for Future Works

1. Using the proposed system to recognize other types of images (like, cancer images).
2. Using other slicing methods (like, fans) to partition the Fourier spectral.
3. Using 2^{nd} order statistical parameters to describe the behavior of patches (or areas distributions).
4. Instead of adopting the patches area, other kind of shape attributes could be used to get addition set of descriptive parameters.
5. Try to utilize the quick Fourier transform technique to handle the long time problem that facing the Fourier transform based method.
6. Consider larger database (more than 7 classes) to insure the recognition accuracy.

PDF created with pdfFactory trial version www.pdffactory.com



PDF created with pdfFactory trial version www.pdffactory.com

References

[Abd09] Abdulkadir, S., "Color Texture Classification Using Wavelet Transform and Neural Network Ensembles", The Arabian Journal for Science and Engineering, Vol. 34, No. 2B, PP. 491-502, Turkey, 2009.

[Ach05] Acharya, T., and Ray, A.K., "Image Processing: Principles and Applications", John Wiley & Sons, Inc., Hoboken, New Jersey, 2005.

[Alb95] Albregtsen, F., "Statistical Texture Measures Computed From Gray Level Run Length Matrices", Image Processing Laboratory, Department of Informatics, Oslo University, 1995.

[Alb97] Alberto Del Bimbo, "Image Analysis and Processing", Vol. 2, Springer Inc., Berlin, ISBN 3-540-63508-4, 1997.

[Ari05] Arivazhagan, S., and Deivalakshmi, S., "Texture Classification Using Wavelet Statistical Features", Department of Electronics & Communication Engineering, Mepco Schlenk Engineering Collage, Sivakasi, 2005.

[Bha87] Bhanu, B., and Parvin, B. A., "Segmentation of Natural Scenes", Pattern Recognition, Vol. 2, No. 5, PP. 487-496, 1987.

PDF created with pdfFactory trial version www.pdffactory.com

[Bor96] Bordatz, P., "Texture, A Photographic Album for Artists and Designers", Dover Publication, New York, USA, 1996.

[Bro05] Broek, E. L. V., and Rikxoort, E. M. V., "Evaluation of Color Representation For Texture Analysis: A Comparative Study", Radbond University, Nijmegen, 2005.

[Bur08] Burge, M. J., and Burger, W., "Digital Image Processing: An Algorithmic Introduction Using Java", Springer, 2008.

[Cam02] Campbell, "Introduction to Remote Sensing", 3rd Edition, Taylor & Francis, ISBN 0-415-28294-2, London, 2002.

[Cha93] Chang, T., and Jay kuo, C. C., "Texture Analysis and Classification With Tree-Structured Wavelet Transform", IEEE Transactions on Image Processing, Vol. 2, No. 4, PP. 429-441, 1993.

[Dri02] Drimbarean, A., and Whelan, P. F., "Color Texture Analysis: A Comparative Study", Dublin City University, 2002.

[Dud01] Duda, R. O., Hart, P. E., Stork, D. G., "Pattern Classification", 2nd Edition, Wiley, New York, ISBN 0-471-05669-3, 2001.

[Est01] Estes, M., "The Discrete Wavelet Transform", Digital Signal Processing, Spring, 2001.

PDF created with pdfFactory trial version www.pdffactory.com

[Fau93] Faugeras, O., "Three-Dimensional Computer Vision: A Geometric Viewpoint", MIT Press, 1993.

[Fol95] Foley, J. D., "Color Appearance Models", 2nd Edition, Redwood City, CA, Addison-Wesley, ISBN 0-201-84840-6, 1995.

[Gao09] Gao, J., "**Digital Analysis of Remotely Sensed Imagery**", McGraw-Hill, New York, 2009.

[Gon03] Gonzalez, R. C., Woods, R. E., and Eddins, S. L., "Digital Image Processing Using Matlab", John Wiley & Sons, Inc, 2003.

[Gon92] Gonzalez, R. C., and Woods, R. E., "Digital Image Processing", 1st Edition, Addison-Wesley, 1992.

[Har01] Hart, P. E., Duda, R. O., and Stork, D. G., " Unsupervised Learning and Clustering", Chapter 10 in Pattern Classification, 2nd Edition, Wiley, New York, ISBN 0-471-05669-3, 2001.

[Har79] Haralick, R. M., "Statistical and Structural Approaches to Texture", Proc. IEEE, Vol. 67, PP. 786-804, 1979.

[Has01] Alhassani, M. D., "Design of A Fingerprint Recognition System Using Wavelet Transformation", M.Sc. Thesis, College of Science, Al-Nahrain University, Baghdad, 2001.

PDF created with pdfFactory trial version www.pdffactory.com

[Hee88] Hee, D. C., Wang, L., and Gilbert, J., "Texture Discrimination Based on Optimal Utilization of Texture Features", Pattern Recognition, Vol. 21, No. 2, PP. 141-146, 1988.

[Hin99] Hinton, G., and Sejnowski, T. J., " Unsupervised Learning: Foundations of Neural Computation", MIT Press, ISBN 0-262-58168-X, 1999.

[Iak04] Iakovidis, D. K., Maroulis, D. E., and Flaounas, I. N., "Color Texture Recognition in Video Sequences Using Wavelet Covariance Features and Support Vector Machines", Real-Time Systems & Image Analysis Group, Department of Informatics & Telecommunications, University of Athens, Greece, 2004.

[Jai89] Jain, A., "Fundamentals of Digital Image Processing", Prentice-Hall, PP. 15-20, 1989.

[Jam02] James, J.F., "A Student's Guide to Fourier Transforms", 2nd Edition, Cambridge University Press, New York, ISBN 0-521-00428-4, 2002.

[Kla04] Klaus, D. T., "Features From Images", Department of Simulation & Graphics, Faculty of Computer Science, University of Magdeburg, Germany, 2004.

[Kon02] Konak, E. S., "A Content-Based Image Retrieval System for Texture and Color Queries", M.Sc. Thesis, Bilkent

PDF created with pdfFactory trial version www.pdffactory.com

University, 2002.

[Mah09] Mahdi, A., "Acceleration of Image Processing Using New Color Model", American Journal of Applied Sciences, Department of Mechatronics, Faculty of Engineering Technology, Al-Balqa'a Applied University, ISSN 1546-9239, Jordan, 2009.

[Man96] Manjunath, B. S., and Ma, W. Y., "Texture Features for Browsing and Retrieval of Image Data", IEEE Transactions on Pattern Analysis & Machine Intelligence, Vol. 8, No. 18, PP. 837-842, 1996.

[Mao94] Mao, J., and Jain, A.K., "Neural Network and Pattern Recognition", Comp. Intell.: Imitating Life, J. M. Zurada, R. J. Marks, C. J. Robinson (Eds.), IEEE Inc., PP. 194-212, 1994.

[Mar91] Marion, A., "An Introduction to Image Processing", Chapman & Hall, Chapter 9, 1991.

[Mat98] Materka, A., and Strzelecki, M., "Texture Analysis Method – A Review", Technical University, of Lodz, Institute of Electronics, Cost B11 Report, Brussels, 1998.

[Pra78] Pratt, William K., "Digital Image Processing", A Wiley-Interscience Publication, New York, 1978.

[Pre92] Press, W. H., Flannery, B. P., Teukolsky, S. A., and Vetterling, W. T., "Power Spectra Estimation Using the

PDF created with pdfFactory trial version www.pdffactory.com

FFT", 2nd Edition, Cambridge, England, Cambridge University Press, PP. 542-551, 1992.

[Sam99] Samawi, V. W., "An Investigation into the Use of Neural Networks in Texture Classification", Ph.D. Thesis, College of Science, Al-Nahrain University, Baghdad, 1999.

[Sem05] Semler, L., Dettori, L., and Furst, J., "Wavelet Based Texture Classification of Tissues in Computed Tomography", DePaul University, USA, 2005.

[Set97] Setiono, R., and Lui, H., "Neural Network Feature Selector", IEEE Trans. On Neural Network, Vol. 8, PP. 654-662, 1997.

[Sha01] Shapiro, L. G., and Stockman, G. C., "Computer Vision", Upper Saddle River, Prentice-Hall, 2001.

[The03] Theodoridis, S., and Koutroutmbas, K., "Pattern Recognition", 2, Elsevier, USA, 2003.

[Tuc93] Tuceryam, M., and Jain A. K., "Texture Analysis", Chapter 2.1 in handbook of Patten Recognition & Computer Vision, C. H. Chen, L. F. Pau, and P.S.P. Wang (eds.), World Scientific Publishing Company, PP. 235-276, 1993.

[Umb98] Umbangh, S. E., "Computer Vision and Image Processing", Prentice-Hall, Inc., USA, 1998.

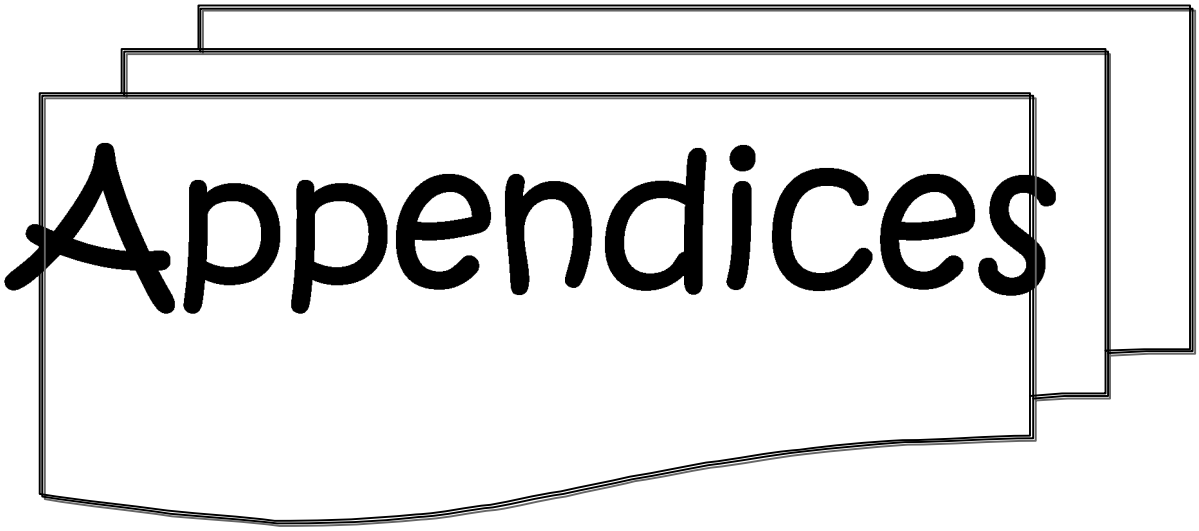
PDF created with pdfFactory trial version www.pdffactory.com

[Vin93] Vincent, L., "Morphological Grayscale Reconstruction in Image Analysis: Applications and Efficient Algorithms", IEEE Trans. on Image Processing, Vol. 2, No. 2, PP. 176-2001, April, 1993.

[Wan01] Wang, S., "A Robust CBIR Approach Using Local Color Histograms", M.Sc. Thesis, Department of Computing Science, University of Alberta, Canada, 2001.

[Wen04] Wen, C. Y., and Chou, C. M., "Color Image Models and its Applications to Document Examination", Forensic Science Journal, Vol. 3, No. 1, PP. 24-28, 2004.

PDF created with pdfFactory trial version www.pdffactory.com



PDF created with pdfFactory trial version www.pdffactory.com

Appendix A

Bands

The following table illustrates each band name with its corresponding number.

Table A-1 Bands names and their corresponding numbers.

Band name	Band no.
Red	1
Green	2
Blue	3
Gray	4
Y	5
I	6
Q	7
H	8
S	9
V	10

PDF created with pdfFactory trial version www.pdffactory.com

Appendix B

Features

1. Density features

When using density distribution to extract a feature set, there is 14 features vectors, each of them has NT-1 elements (i,e., NT is the number of density thresholds). The following table illustrates these vectors assuming that the number of density thresholds is 5.

Table B-1 features names and their corresponding numbers.

Feature name	Feature no.
Meanx2(1)	1
Meanx2(2)	2
Meanx2(3)	3
Meanx2(4)	4
Stdx2(1)	5
Stdx2(2)	6
Stdx2(3)	7
Stdx2(4)	8
Meany2(1)	9
Meany2(2)	10
Meany2(3)	11
Meany2(4)	12
Stdy2(1)	13
Stdy2(2)	14
Stdy2(3)	15
Stdy2(4)	16
Meanx3(1)	17
Meanx3(2)	18
Meanx3(3)	19
Meanx3(4)	20
Stdx3(1)	21
Stdx3(2)	22

PDF created with pdfFactory trial version www.pdffactory.com

Stdx3(3)	**23**
Stdx3(4)	**24**
Meany3(1)	**25**
Meany3(2)	**26**
Meany3(3)	**27**
Meany3(4)	**28**
Stdy3(1)	**29**
Stdy3(2)	**30**
Stdy3(3)	**31**
Stdy3(4)	**32**
Meanx1y2(1)	**33**
Meanx1y2(2)	**34**
Meanx1y2(3)	**35**
Meanx1y2(4)	**36**
Stdx1y2(1)	**37**
Stdx1y2(2)	**38**
Stdx1y2(3)	**39**
Stdx1y2(4)	**40**
Meanx2y1(1)	**41**
Meanx2y1(2)	**42**
Meanx2y1(3)	**43**
Meanx2y1(4)	**44**
Stdx2y1(1)	**45**
Stdx2y1(2)	**46**
Stdx2y1(3)	**47**
Stdx2y1(4)	**48**
MeanMm(1)	**49**
MeanMm(2)	**50**
MeanMm(3)	**51**
MeanMm(4)	**52**
StdMm(1)	**53**
StdMm(2)	**54**
StdMm(3)	**55**
StdMm(4)	**56**

PDF created with pdfFactory trial version www.pdffactory.com

2. Fourier Features

There is only one feature vector for Fourier transform method, named (Powr (NS)) where NS is the number of slices of Fourier transform. So, if NS=4 then the no. of features for each band=4 (i.e., Powr(1), Powr(2), Powr(3), and Powr(4)).

PDF created with pdfFactory trial version www.pdffactory.com

Appendix E

Bitmap File Structure

Any 24-BMP file consists of the following data structure:

A. The Bitmap File Header: it is a structure, its length for bmp file format is 14-byte, and its fields are arranged as showing in table (E.1).

Table (E.1) Bitmap file header

Start (offset)	Size (bytes)	Name	Purpose
1	2	Type	Must always be set to 'BM' to declare that this is a .bmp file.
3	4	Size	Specifies the size of the file in bytes.
7	2	Reserved1	Must always be set to zero.
9	2	Reserved2	Must always be set to zero.
11	4	OffsetBits	Specifies the offset from the beginning of the file to the bitmap data.

B. The Bitmap Info. Header: it is a structure consists of 11 fields, with total length equal 40 byte. Table (E.2) shows the contents of this structure, some of them represent the image attributes.

PDF created with pdfFactory trial version www.pdffactory.com

Table (E.2) Bitmap info. header

Start (offset)	Size (bytes)	Name	Purpose
15	4	Size	Specifies the size of the bitmapinfoheader structure, in bytes.
19	4	Width	Specifies the width of the image, in pixels.
23	4	Height	Specifies the height of the image, in pixels.
27	2	Planes	Specifies the number of planes of the target device, must be set to zero.
29	2	BitCount	Specifies the number of bits per pixel.
31	4	Compression	Specifies the type of compression, usually set to zero (no compression).
35	4	SizeImage	Specifies the size of the image data, in bytes. If there is no compression, it is valid to set this member to zero.
39	4	XPelsPerMeter	Specifies the horizontal pixels per meter on the designated target device, usually set to zero.
43	4	YPelsPerMeter	Specifies the vertical pixels per meter on the designated target device, usually set to zero.
47	4	ColorsUsed	Specifies the number of colors used in the bitmap, if set to zero the number of colors is calculated using the biBitCount member.
51	4	ColorsImportant	Specifies the number of color that are important for the bitmap, if set to zero, all colors are important.

PDF created with pdfFactory trial version www.pdffactory.com

الخلاصة

تعتبر عملية التمييز واحدة من العمليات المهمة في العديد من تطبيقات الحاسوب لغرض تمييز الصور اعتمادا على اللون او التركيب النسيجي. ان النظام المقترح هو نظام التعرف على الصور ذات التركيب النسيجي. المرحلة الاولى هي مرحلة التحضيرات. في هذه المرحلة تم القيام بعملية التحويل اللوني على صور التدريب لانتاج عدد من الحزم (R, G, B, Gray, Y, I, Q, H, S, V) ومن هذه الحزم يتم استخراج متجهات الخصائص.

تستند مرحلة الاستخراج على تقنيتين: كثافة التقطيع وتحويل فورير. وقد تم تطبيق كل من هاتين التقنيتين بشكل منفصل. في طريقة توزيع الكثافة ، تم تحويل الصورة الى نوعين من المناطق (شرائح بيضاء و سوداء) أو (0 و 1). ولتحديد كل منطقة ، تم اعتماد خوارزمية ملء البذرة (seed filling) وبعدها جرى حساب عدد من العزوم (moments) لكل من هذه المناطق. كما تم حساب نسبة الانحراف والمعدل لهذه العزوم (moments) على نطاق الصورة بأكملها.

في طريقة تحويل فورير، تم تحويل عينة الصورة الى المجال الترددي ومن ثم جرى تقسيمها إلى حلقات (شرائح فورير), ولكل شريحة تم حساب متوسط الطاقة (power). وبعد استخراج الخصائص لكل عينة مدروسة يتم خزنها في قاعدة البيانات, وقد تم جمع الخصائص المحسوبة في متجه واحد ولكل عينة. وقد شملت الاختبارات المنفذة استخدام الصور لسبعة فئات, ولكل فئة تم اخذ 10 مشاهد, ولكل مشهد تم اخذ 10 عينات. وخلال الاختبارات تم استخراج متجهات الخصائص لعينات الاختبار ومن ثم تمت مقارنة الميزات المستخرجة مع تلك المخزونة في الملف للبحث عن اقرب تشابه باستخدام مقياس الفرق الاقليدي (EDM).

خلال عملية التقييم, وجد ان افضل النتائج التي تم الحصول عليها جاءت من مزج تسعة خصائص ولعينات ذات حجم 180*180 بكسل وعدد عتبات الكثافة (5) في طريقة توزيع الكثافة، الأمر الذي ادى إلى نسبة نجاح 94،9 ٪ . كذلك تم الحصول على افضل النتائج من مزج تسعة خصائص ولعينات ذات حجم 180*180 بكسل مقسمة الى (8) شرائح فورير في طريقة تحويل فورير ، والذي بدوره ادى الى نسبة النجاح 99،9 ٪.

PDF created with pdfFactory trial version www.pdffactory.com

جمهورية العراق

وزارة التعليم العالي والبحث العلمي

جامعة النهرين

كلية العلوم

تمييز النسيج باستخدام طريقة توزيع الكثافة

رسالة

مقدمة الى كلية العلوم في جامعة النهرين كجزء من متطلبات نيل درجة الماجستير في علوم الحاسبات

من قبل

نجمة درع ظاهر

(بكالوريوس جامعة النهرين 2003)

اشراف

د. لؤي أدور جورج

حزيران 2011 **1432**

PDF created with pdfFactory trial version www.pdffactory.com