Texture Classification Using Texture-based Feature Extraction Algorithms

Shvan Abdullah Saeed

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Engineering

Eastern Mediterranean University
November 2018
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Assoc. Prof. Dr. Ali Hakan Ulusoy
Acting Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science in Computer Engineering.

Prof. Dr. H. Işık Aybay
Chair, Department of Computer Engineering

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Master of Science in Computer Engineering.

Assoc. Prof. Dr. Önsen Toygar
Supervisor

Examining Committee

1. Assoc. Prof. Dr. Duygu Çelik Ertuğrul
2. Assoc. Prof. Dr. Önsen Toygar
3. Asst. Prof. Dr. Mehtap Köse Ulukök

ABSTRACT

Texture is one of the significant characteristics used in identifying objects of interest or regions in an image. Texture is an important characteristic of surface property in visual scenes and is a power cue in visual perception. The real applications of texture classification are remote sensing, medical imaging, industrial inspection and pattern recognition. Texture images are highly affected by rotation and illumination. Extracting texture features that are rotation-invariant and insensitive to illumination with high classification accuracy is still a challenge.

Texture analysis has been a popular area of study in computer vision for decades. In this thesis, six texture-based feature extractors that may perform variously to rotation and illumination are used namely Local Binary Patterns (LBP), Complete Local Binary Patterns (CLBP), Segmentation-based Fractal Texture Analysis (SFTA), Histogram of Oriented Gradients (HOG), Rotation Invariant Histogram of Oriented Gradients (RIHOG) and Haralick feature extractor. They are implemented and tested on three benchmark texture databases, such as The Columbia-Utrecht Database (CUReT), University of Oulu Texture database (OUTex) and Textured Surfaces Database. For feature matching, two classifiers are used namely Naive Bayes and Support Vector Machines (SVM). A comparative study is presented at the end of the experimental evaluations on texture classification.

Keywords: texture classification, feature extraction, texture-based methods

ÖZ

Doku, bir görüntüdeki ilgi alanı veya nesneyi tanımlamak için kullanılan belirgin bir özelliktir. Görsel görünüm için kullanılan yüzey özelliklerinin önemli bir belirleyicisi dokudur ve görsel algıda kullanılan güçlü bir ipucudur. Doku sınıflandırmanın gerçek hayattakiuygulamaları uzaktan algılama, tıbbi görüntüleme, endüstriyel denetim ve örüntü tanımadır. Doku görüntüleri, döndürme ve ışıklandırmadan dolayı fazlaca etkilenirler. Döndürmeye duyarsız ve ışıklandırmadan etkilenmeyen doku özniteliklerinin yüksek bir sınıflandırma yüzdesiyle çıkartılması zor bir işlemdir.

Doku analizi, bilgisayarla görü alanında çalışılan popüler bir alandır. Bu tezde, döndürmeye ve ışıklandırmaya karşı farklı tepkiler veren altıtanedokuya dayalı öznitelik çıkarma yöntemi kullanılmıştır. Bu yöntemler, Yerel İkili Örüntüler (LBP), Tamamlanmış Yerel İkili Örüntüler (CLBP), Bölütlemeye Dayalı Fraktal Doku Analizi (SFTA), Gradient'lere Yönelik Histogramlar (HOG), Döndürmeye Duyarsız Gradient'lere Yönelik Histogramlar (RIHOG) ve Haralick Öznitelik Çıkartıcı yöntemidir. Bu yöntemler üç farklıdoku veritabanı üzerinde uygulanmış ve test edilmiştir. Kullanılan veritabanları Columbia Utrecht Veritabanı (CUReT), Oulu Üniversitesi Doku Veritabanı (OUTex) ve Dokusal Yüzeyler Veritabanı'dır. Öznitelik eşleştirme işlemi için Naïve Bayes ve Destek Vektör Makinesi (SVM) sınıflandırıcıları kullanılmıştır.Deneylerin sonunda, doku sınıflandırma değerlendirilmiş ve karşılaştırmalı bir çalışma sunulmuştur.

Anahtar Kelimeler: doku sınıflandırma, öznitelik çıkarma, dokuya dayalı yöntemler

I dedicate this research to, all the people in my life who touched my heart.

ACKNOWLEDGMENT

First and foremost, I must acknowledge my limitless thanks to my supervisor Assoc. Prof. Dr. Önsen Toygar who has been always helpful during all phases of this research.

I owe a deep debt of gratitude to my family for their generous support they provided me throughout my entire life, this wouldn’t be possible without their unconditional love and support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CLBP	Complete Local Binary Patterns
CUReT	The Columbia-Utrecht Database
HOG	Histogram of Oriented Gradients
LBP	Local Binary Patterns
OUTex	University of Oulu Texture database
RIHOG	Rotation Invariant Histogram of Oriented Gradients
SFTA	Segmentation-based Fractal Texture Analysis
SVM	Support Vector Machines
TTBD	Two-Threshold Binary Decomposition

Chapter 1

INTRODUCTION

1.1 Texture Classification

Texture classification is the procedure of allocating an unknown texture into known texture classes. The texture is one of the properties of the object. Realizing the type of a texture is believed to be an essential task in the human visual system for recognition and clarification of an object. Computer vision scientists try to mimic the human visual system into a computer based application, which after that it can be improved to outperform human beings in more complex and sensitive problems.

Objects have obvious colors and reflection properties but in detail they are made of much more complicated patterns. Humans have a complex visual system, hence it is easy to observe and determine various textures, but when it comes to a computer it is hard to identify these complicated patterns.

The textures are different in formation. Figure 1.1 shows some everyday texture examples. Textures can be hierarchical, if you take two points on the same texture and magnify those two points they will not look similar, that is why multi-scale texture analysis is used, which extracts features from many areas of a texture image. Some textures are characterized by having small basic patterns which are seen repeatedly in the formation of the texture [1].

Humans can easily differentiate between objects but textures can be deceiving, appointing an unknown texture to a known class can be a challenge. Textures have to be classified based on a complex feature space. Each pixel has to be represented by a feature vector, the process is done by obtaining features from the texture images, and then classification process is applied to match features.

Figure 1.1: Some Everyday Texture Examples [2].

After the results from classification process are gained, more work can be done on the results, but the condition of the features must be preserved, hence it is very important and it will have an impact on the outcome of any further processing [2].

1.1.1 Stages of Texture Classification

Texture classification process is divided into two stages: the learning (training) stage and the recognition (testing) stage [3]. In the training stage, feature extractors are applied and features are obtained for every texture. Afterwards depending on their features, each image is grouped under texture name. The features can be unique histograms or empirical distributions which characterize images' textural characteristics, such as structure, brightness, roughness and rotation. During the recognition stage, each and every unknown sample is analyzed, then, each sample is divided based on their features. These features are compared with the features in the training database, in a process called feature matching which follows algorithms of classification. Then, by comparing the features of the new sample with the features of the known classes, the best match is decided. Sometimes this might not be the best decision according to the previously known examples and in such cases the undefined sample can be rejected. Figure 1.2 shows the process of texture classification. Texture classification is one of the most explored research areas in computer vision. There are also many feature extractors, finding the best one is a critical step for achieving an authentic texture classification. Because there are many types of textures in the nature, different features of every texture extractor must be chosen. These features should depend on the nature of texture images. Each texture extractor method should contain different specifications of the image.

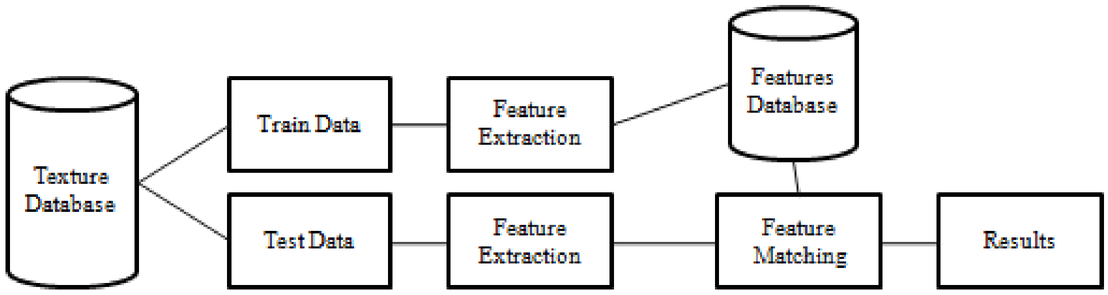


Figure 1.2: The Process of Texture Classification.

1.1.2 Applications of Texture Classification

Textures are one of the key features in computer vision; they are widely used in many computer based applications. Textures are specially used in the applications of computer vision analysis for classification or segmentation of images based on local spatial variations.

Possible areas for using texture-based applications are medical image analysis and biomedical surface inspection. Examples include skin diseases, industrial inspection, studying of satellite images, classification in document analysis, biometric person authentication, scene scanning for robot navigation and texture synthesis for image coding [4].

Comprehensive studies have been done in this field over the past years. Since there are many fields of using the applications, texture extractors have become very complex. However, there are only a few useful applications in industry that humans can really benefit from. The biggest disadvantage in these applications is occurred because of the nature of textures in the real world in which the textures are not homogeneous. Additionally, there are changes in their directions and patterns, which make the applications somehow not reliable.

1.1.3 Challenges

The types of textures come from a very large groups ranging from deterministic textures to statistical textures. Therefore working with textures can be a bit challenging, and many problems are encountered during the process of classification. Some of these problems are explained below.

When an automated texture system tries to scan a woman's mammogram and inspect the breast tissue to check her risks for having breast cancer as shown in Figure 1.3a, one important problem arises. The system has to automatically select the area of mammogram with the breast tissue, so the challenge here is that the automated system first should segment the elements in the mammogram such as muscle, breast tissue and background, so that in the next step it can classify the region of interest.

Another problem is faced when trying to restore digital images whenever there exist some corrupted, lost or unnecessary areas in the image. Those areas have to be replaced with the texture of the background as shown in Figure 1.3b. Another problem is seen when trying to store large databases of fingerprints as in Figure 1.3c, so database could be kept in the most efficient and dynamic aspect. Forming 3D model of a person’s body from the outfit they wear is also a challenge. As shown in Figure 1.3d, all of these problems could be overcome by working on the texture, utilizing its basic elements and finding the textural composition [5].

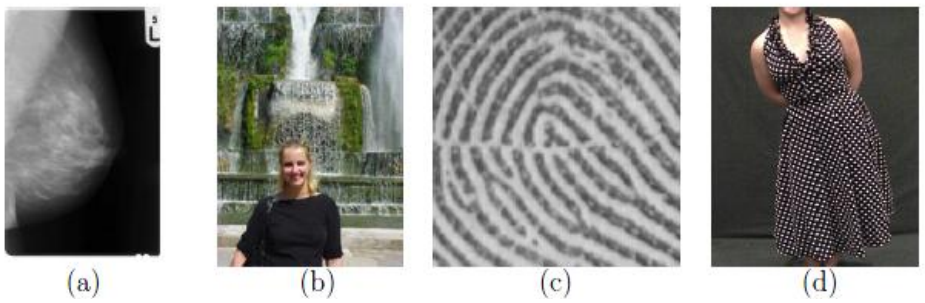


Figure 1.3: Challenges of Texture Classification
(a) Mammogram Image (b) Removing Unwanted Person from Scene
(c) Compressing a Fingerprint Image (d) Shape from Texture.

It also has to be mentioned that the implication to describe texture can be a puzzle, because it can be separately described based on type of aspiration. Study of texture in computer vision has been divided into 5 main sections namely classification, segmentation, synthesis, compression and shape from texture. To understand the full meaning of texture, its definition will be described in each section accordingly. So a bigger image of term texture could be obtained.

1.2 Texture Segmentation

The idea behind segmentation is to divide an image to separate regions based on their similarities, so an image is segmented to few parts according to similarities in their pixels. Also the process of selection and separating foreground from background is segmentation, the area of interest might be in foreground, for example if there is an image of a leaf it is tried to be classified. First the picture has to be segmented to find the leaf and ignore the background. Then the classification process can begin. An image may have different light and view angles so have to use multiple features for image segmentation because single texture features are not good enough. There are mainly two methods of segmentation as region-based and edge-based segmentation. Firstly, the image have to be searched and then either the edge detection algorithms can be used or special algorithms can be employed to detect objects from image [6].

This process can be a supervised, it means that all the information about intended textures is available and it is stored in the database so it can be used afterwards. The process might be unsupervised, it means that the areas of the image with their pixels have to be somehow processed and the similarities can be found in those pixels.

One image can have more than one successful segmentation based on the method used. So when segmentation is used, the correct segmentation method has to be chosen that fits the experiments. Segmentation is an important preprocessing method before the classification, and in a research study [7], the process of recognition of limbs and important steps to detect human body can be seen.

Most of the classification applications mentioned before depend on segmentation to find the region which is classified. If classification is done without segmentation, no results may be gotten because the picture might contain many useless areas with hundreds of textures. So the segmentation process plays an important role in making the classification process efficient and fast.

The outcome of segmentation is a group of super pixels (segments) which are the main components of the image or they could be the outline of an object, the edge or line that bounds an object. Some features of these segments are similar such as color and intensity. Additionally, different segments differ from each other remarkably.

1.3 Texture Compression and Coding

Nowadays prices of energy and manpower are increasing gradually. In order to be efficient and reduce the cost of saving data, new strategies have to be followed. When it comes to saving samples to texture, the databases are mainly of large sizes, here is when the idea if compression is introduced to texture databases. It decreases database size remarkably.

Mainly there are two types of compression namely lossless compression and lossy compression. The difference is that, in lossless compression all the original bits of file stay intact after the file is uncompressed. This technique is mostly used for text and for important data where losing small parts of data could cause problems. This method can also be used for some cases of texture where images are very sensitive. On the other hand, lossy compression is mostly used for pictures, where similar pixels are stored as one bit which decreases the file size. Lossy compression is widely used for texture compression and in this way, better compression rates are achieved [8]. It is known that textures are also images, but the method used for compression of textures is not the same as images.

Specific algorithms used for compressing textures are designed especially to utilize details of the texture. For example, WSQ compression algorithm is used for compressing fingerprint images. Figure 1.4 shows two images before and after compression. It uses same compression rate as JPEG, but the outcome of WSQ is much more accurate [9]. The use of segmentation process is very important in compression. When you have an image, people tend to focus more on objects in the picture and they neglect the background. Therefore, before compression, segmentation can be used to detect the foreground. Then more details can be used when compressing that area. By using rapid and dynamic compression algorithms, data can easily be stored and used in large databases. If this technique was not available it would have been hard to work with most multimedia applications.

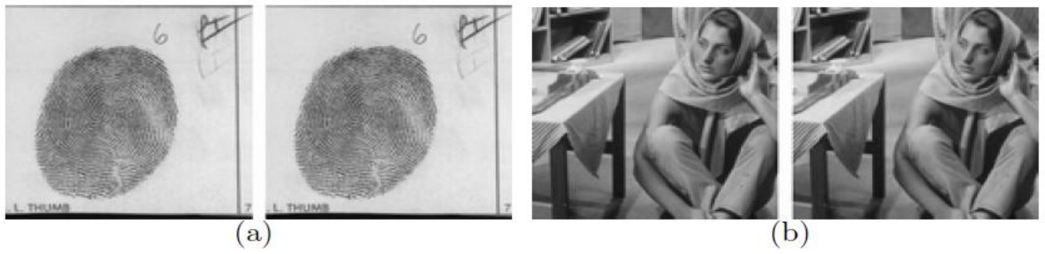


Figure 1.4: Examples of Image Compression (a) Fingerprint Image and (b) Painting Image before and after Compression.

1.4 Shape from Texture

The process of regenerating 3D model of an object from its 2D plane image is called shape from texture. Generation of shape of a dress and a shirt from their images can be seen in Figure 1.5. Textures can help us identify shapes. For the first time in 1950 this method was studied, and the term texture gradient was introduced. It is the parts of a surface with similar textures and areas that look different to human eye which have different angles and deeper surfaces [10].

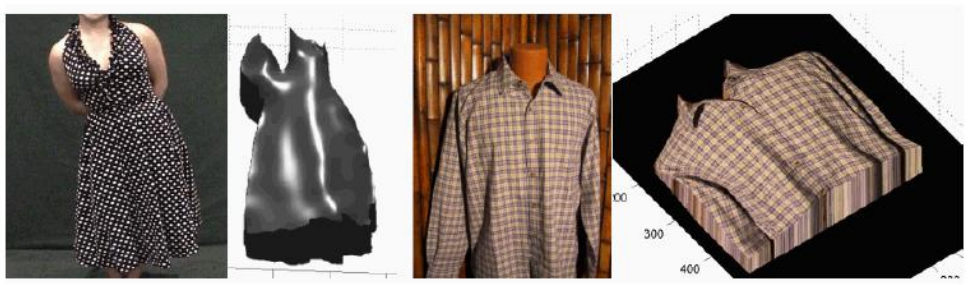


Figure 1.5: Generation of Shape of a Dress and a Shirt from Their Textures [10].

1.5 Texture Synthesis

A small input image can be converted into a larger image by using its structural form. It is like making a model of small image’s structure and putting it together. The aim here is to create an output image similar to the provided image. The output image cannot be identical to the target, but it has to be approximately similar.

A larger image of apples is synthesized from a small sample as shown in Figure 1.6. Synthesizing an image can be done from a specified angle or different light and shadow conditions [11]. Applications of texture synthesis include making high resolution textures by synthesizing images with low resolution. It can also be used with inpainting to cover the unwanted areas in an image.

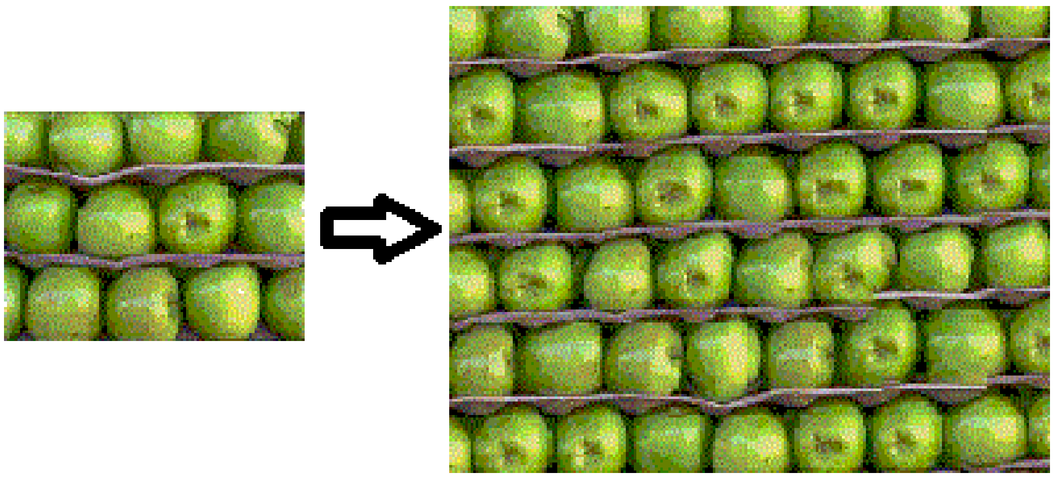


Figure 1.6: Larger Image of Apples Synthesized from a Small Sample.

Chapter 2

LITERATURE REVIEW

2.1 Introduction

Due to the fact that the study of texture is an important field in computer vision, many researchers have considered studying the field of texture. It is remarkable that this field has been in progress during the course of previous decade, and it has seen many outstanding improvements to its foundation. Because of industry demand, the field is very popular.

The inspiration of this research has come from these researches. In this chapter, a sum up of some notable previous researches will be discussed and several main accomplishments and innovations made in the field of texture analysis are emphasized.

2.2 Previous Work

Jain and Farrokhnia [12] proposed a method motivated by the human visual system. The early stage of human visual system is initialized by multi-channel filtering the visual information in the human sight and these channels have a special character of extracting features using Gabor filters which cover the spatial-frequency domain. An organized filter selection scheme is proposed which rebuilds the input images from filtered images. Then the features of textures are gained by transforming filtered images nonlinearly and calculating the amount of energy in each pixel's neighborhood. Then a special algorithm is used to find the mean square-error.

A collection of techniques are presented by Haddon and Boyce [13] for image segmentation and edge detection based on co-occurrence matrices. Matrices of co-occurrence and transformations are reported, the authors pay attention to diversity between typical and atypical image features using co-occurrence matrices as references. Then further work is done on evaluating the matrices and classifying them. The final outcome is a labeled matrix that can classify parts of an image and can be able to find the outstanding edges. Many examples are shown in the research such as: infrared, synthetic, multispectral and temporal. They also let the way for future work which can be used with this work for example hysteresis and relaxation labeling for better results.

The 3-dimensional data obtained from the image function graph is used by Nixon et al. [14] for describing the texture. The image function graph is a crushed surface which has wrinkles in its formation and it is shown like a landscape. Defining the texture using this landscape is done by using 6 texture feature extractors. These extractors are built on some characteristics of real life matter. This method has been tested on large benchmark databases such as Brodatz texture Database. The authors conducted experiments and they have shown very low error rate. Additionally, the same method is tested on VisTex texture Database, which again has shown very well results.

Texture is a key feature for identifying objects and parts of interest in an image. A new area of texture features is researched by He and Wang [15], which is obtained from texture spectrum. In the research, the texture data is more complex regarding the characteristics of texture hence it is gained by collecting information from 8 neighboring pixels instead of one pixel. The simulation is done on the Brodatz natural texture database to find out the precision of this method. Additionally they have compared the new results with few older texture features which use co-occurrence matrices. Good results are shown in the experimental results.

Bigun [16] introduced a new scheme for texture analysis which performs in the best possible manner with the least waste of time and effort. This method relies on dominant local orientation. The Laplacian filter is used to get the textures' characteristics. At every stage of process they calculate the linear symmetry feature. The results obtained are composite because they are made up of two parts: the estimation of local orientation and its confidence measure. They used random pictures selected by a group of people whom had experience in aerial images. According to the results of experiments, it is clearly shown that the method is an efficient way to calculate texture features.

One of the most recent studies in the field of medicine is proposed by Ma et al. [17]. The aim is to classify the cells affected by lung cancer. The diagnostics is done based on texture and color. The data is collected from real patients and total of 341 CT images were collected. The experiments are conducted on the collected dataset. They used Support Vector Machine (SVM) algorithm on the available features and taxonomy percentage was calculated using receiver operating characteristic (ROC) curve analysis. Moreover, for color analysis, they used RGB and HSV features. When they combined the texture and color result, they were able to get 95% precision to detect cancer and its sub-types.

In recent years texture classification has been used in gastronomy by Watanabe et al. [18]. The aim is to achieve a soft diet food, and to find out the best way to make this kind of diet food. If the food is cooked in the wrong way, it might lose its nutritive value. In that research, a new way have been improved to make the best soft diet food. The test is done by boiling minced chicken with other food additives. Then the best result was discovered by testing the texture of meat during the cooking. A creep meter was used to determine the texture of meat. Afterwards, resulted food was provided to a nursing home. The process was surveyed by involving 22 elderly people and they achieved more than 100% food intake that research helped to find the best and healthiest way to make soft diet food for elderly people.

Another recent study in the field of agriculture and soil science by Jia et al. [19] showed that the type of soil and its compounds are an important factor for growing crops. In that research, HSI images of soil have been used to classify different kinds of soil, and the total quantity of soil nitrogen (TN) substance. 183 samples of soil were tested by near-infrared hyper spectral imaging system. After analyzing the textures, three methods were used to determine the type of soil and ratio of nitrogen in the soil. Successive Projections Algorithm (SPA), Support Vector Machines (SVM) and Least Squares Regression (LSR) are used in that study. The samples were from three main soil kinds including paddy soil, red soil and seashore saline soil.

Another new study in the field of medicine is introduced by Ardakani and Mohammadi [20] by analyzing the texture of kidney after the transplant operation. In that research, ultrasonic images of kidney are taken and kidney texture is monitored for changes after they were transplanted. The tests were made on 61 patients and a diagnostic system processed the ultrasonic images of their kidneys. 270 texture features were used as descriptors in all ultrasonic images of patients. They used nearest neighbor as a classification method. After getting acceptable results they find out that texture analysis is a good way to characterize the texture of kidney. Moreover, this method can help doctors to find out kidney failure after the transplantation of kidney much faster than using ultrasonic images.

Table 2.1 shows a summary of the previous studies on texture classification using different databases. The table presents various research studies with the details of the systems and the performance obtained in different metrics.

Table 2.1 Summary of Some Previous Classification Researches on Various Databases

Database	Texture classes	Samples per class	Feature extractor	Classification accuracy	Reference
BRODATZ	16	8	LBP/VAR	99.70%	[24]
OUTex_TC_10	24	20	LBP/VAR	90.55%	[24]
OUTex_TC_12	24	20	LBP/VAR	91.60%	[24]
KTHTIPS2b	11	4	LBP	59.10%	[24]
Textured Surfaces	25	40	SFTA	90.80%	[26]
Textured Surfaces	25	40	Haralick	82.40%	[26]
Textured Surfaces	25	40	Gabor	87.70%	[26]
BRODATZ	16	8	HOG	11.53%	[29]
BRODATZ	16	8	RIHOG	95.50%	[29]
DTD	47	120	LBP	36.10%	[30]
DTD	47	120	SIFT	52.30%	[30]
DTD	47	120	VGG-M	68.80%	[30]

Various texture classification experiments were conducted on several texture databases as shown in Table 2.1. Additionally, many feature extraction methods are used for texture classification and the results are encouraging whenever LBP/VAR, SFTA and RIHOG approaches are used. In this thesis, we implemented several efficient feature extractors for texture classification by considering the aforementioned approaches.

Chapter 3

FEATURE EXTRACTION AND CLASSIFICATION

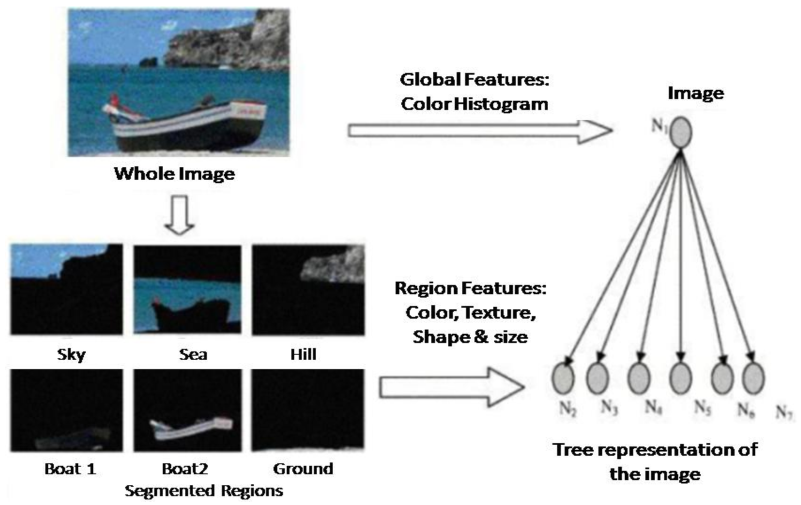
3.1 What is a Feature?

The word “feature” does not have a common definition because the meaning of the word depends on the type of field in which it is used. It can be referred to as an appealing item in an image. In other words, features are distinctive indications and properties of an image. The first step in initializing computer vision algorithms is to extract image features. Algorithms use features as a fundamental basis to perceive the difference between images, and it is used in most of the machine vision applications. The quality of application will depend on the feature detector used. Because of its importance, a lot of attention must be paid to feature extraction [21]. Preprocessing on the images should be performed firstly. The main idea behind preprocessing is to enhance the quality of image, therefore better features can be extracted. Preprocessing may include adjusting the brightness, cropping, rotating, histogram equalization and thresholding of the image. The next step is extraction of features that will be used for classification and recognition of the images. It is not wise to completely depend on one feature extractor because some of them might show good results in several cases and show bad results in another case. Therefore preprocessing will help improve the quality of the application by choosing a good extractor. It can be said that the relationship between quality of features and efficiency of application is directly proportional. Additionally, size of data is another problem. If you have a relatively large dataset, you have to choose a fast and precise feature extractor. If the amount of input data is reduced, better results will be seen because processing a large database requires a lot of memory and it also increases computation time. Features can be divided into the following types [22]:

• General features: Formation of the image texture, color, and shape. It can be sub-divided into the following categories:

- Pixel-level features: Properties of a single pixel, e.g. color, location.
- Local features: Features that are extracted during the process of image segmentation or edge detection.
- Global features: Features that are extracted from the whole image or just a part of the image.

• Domain-specific features: Features which are specific to a special area like iris and fingerprint. Mainly all features can be divided into two levels; low-level features and high-level features. Low-level features are extracted directly from images. However high-level features are originated from low-level features. A unique technique has to be used to combine local and global features. A two-level tree is used to unite features. Global features are put into the first level which is the root of tree, and the local features are stored in child nodes that can be seen in Figure 3.1.


Whole Image
Global Features: Color Histogram
Image
N_1
Region Features: Color, Texture, Shape & size
Sky
Sea
Hill
Boat 1
Boat2
Ground
Segmented Regions
N_2 N_3 N_4 N_5 N_6 N_7
Tree representation of the image
Figure 3.1: Tree Representation of Image Features [22].

3.2 Image Feature Representation

The mechanisms of dividing images and arranging the features are also demanding processes. Separating one image into different regions in order to extract features of each region can be done in three ways: (1) using regular grid approach (2) unsupervised image segmentation and (3) interest point detectors such as “Difference of Gaussians (DoG)” [23]. Implementation of three methods can be seen in Figure 3.2. The original image is demonstrated in Figure 3.2 (a) and Figure 3.2 (b) shows the image segmented by regular grid. On the other hand, Figure 3.2 (c) presents the image segmented by JSEG and Figure 3.2 (d) provides the segmented region by DoG.

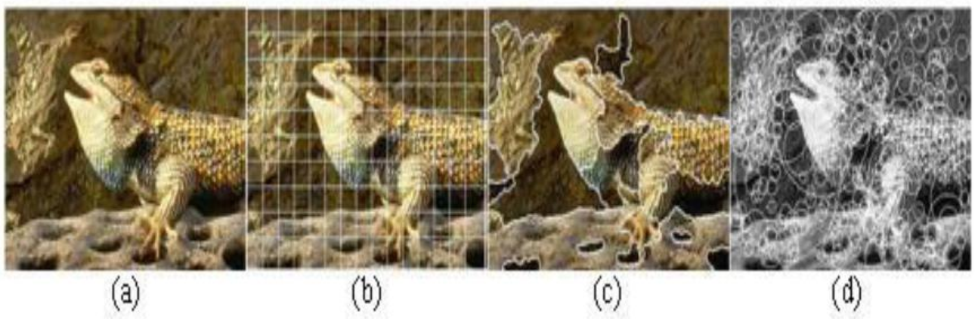


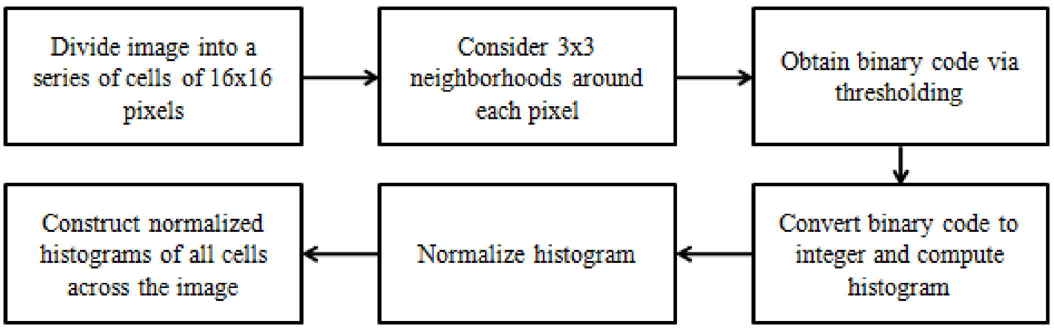
Figure 3.2: (a) Original image and its segmentations using (b) regular grid, (c) Unsupervised Image Segmentation and (d) DoG [23].

3.3 Extraction Methods

In this study six feature extractors are used in the experiments and they will be explained in detail in the following subsections.

3.3.1 Local Binary Patterns (LBP)

One of the well known descriptors in the field of texture analysis is Local Binary Patterns (LBP). LBP with its many variants show promising results when it is used to extract features [24]. This method works by describing every pixel using its neighboring pixels by a binary code. This binary code can be obtained first by applying a set of filters, then changing the result to a binary value. The values in the code are changed with different filters. LBP combines the old statistical and structural methods of texture analysis. One of the advantages of LBP in texture analysis field is that the gray level changes in the images do not affect its results. So if the angle of light changes from one image to another, the LBP code will still have no problem with extracting features. A further advantage of LBP comes with simplicity of its algorithm. This makes it easy to work with LBP under different circumstances. A simple algorithm can be modified easily to work with different kinds of problems stand alone or combining with other types of descriptors. The extraction of features in LBP follows different steps which can be seen in Figure 3.3.


Divide image into a series of cells of 16x16 pixels
Consider 3x3 neighborhoods around each pixel
Obtain binary code via thresholding
Convert binary code to integer and compute histogram
Normalize histogram
Construct normalized histograms of all cells across the image
Figure: 3.3 Flowchart of LBP [24].

LBP was introduced by Ojala et al. [24] as a strong way to extract features hence it is not dependent on the gray level and it can be obtained from a pixel by using its neighborhood. LBP works by making labels from pixels of images. If there is a 3x3 matrix of pixel values, it will threshold each pixel with the pixel at location (2,2) pixel which is the center pixel. When the result is obtained, it is converted to a binary number as shown in Figure 3.4. Therefore 2^8 =256 different values are obtained by histogram and then these values can be used for classification of the texture. In LBP, there are also two notations as (P, R). P is the number of sampling points and R is the radius of the circle which P points lay on it. By changing these values, the size of neighborhoods can be changed. For instance, some feature classifiers will perform better for smaller neighborhoods, other classifiers might be contrary.

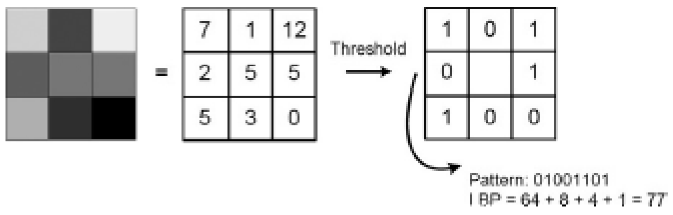


Figure: 3.4 Basic LBP Operator [24].

3.3.2 Complete Local Binary Patterns (CLBP)

Complete Local Binary Patterns (CLBP) [25] is a variant of LBP. It uses three measurement units for extracting features from an image. The first one is CLBP-Sign (CLBP-S) which represents the local difference between the gray levels. The second unit is CLBP-Magnitude (CLBP-M) which represents the magnitude or known as slope descriptor. Finally the last unit is CLBP-Center (CLBP- C) which stores the gray level value of central pixel. The calculation is performed as follows:

$$\mathrm{CLBPS}_{p,r}. = \sum_{n=0}^{p-1} \mathrm{s}(\mathrm{g_n} - \mathrm{g_c})2^{\mathrm{n}} \tag{3.1}$$

$$\mathrm{CLBP\_M}_{p,r}. = \sum_{n=0}^{p-1} \mathrm{s}(\mathrm{D_n} - \mathrm{T})2^{\mathrm{n}} \tag{3.2}$$

$$\mathrm{CLBP\_C}_{p,r}. = \mathrm{S}(\mathrm{g_c} - \mathrm{g_n}) \tag{3.3}$$

where

$$\mathrm{D_n} = \mathrm{g_n} - \mathrm{g_c} \tag{3.4}$$

$$\mathrm{T} = \frac{1}{N}\sum_{i=0}^{n-1} \frac{1}{\mathrm{p}} \sum_{n=0}^{p-1}(\mathrm{g_n} - \mathrm{g}_c) \tag{3.5}$$

$$\mathrm{G}_N = \frac{1}{n}\sum_{i=0}^{N-1} \mathrm{g_i} \tag{3.6}$$

The area representation in CLBP is the combination of value of central pixel (CLBP-C) and a Local Difference Sign-Magnitude Transform (LDSMT). The pixel in the center refers to the value of gray level. The LDSMT consists of two other values which are CLBPS and CLBPM. Combining both CLBP-Sign and CLBP-Magnitude is done by using the following equations:

$$d_r = s_r * m_r a \begin{cases} s\,(d_r) \\ m_r = |d_r| \end{cases} \tag{3.7}$$

where s_r is the sign component, m_r is the magnitude component ands_p is determined as follows:

$$s_p \begin{cases} 1, d_r \geq 0 \\ -1, d_r < 0 \end{cases} \tag{3.8}$$

where the local difference vector d_r=[$d_{0,\ldots\ldots}$ d_{r-1}] which can be constructed from the following vectors:

sign vector s_r=[$s_{0,\ldots\ldots}$ s_{r-1}] and (3.9)

magnitude vector m_r =[$m_{0,\ldots\ldots}$ m_{r-1}] (3.10)

During implementation of experiments in this research, it was seen that CLBP gives much better results than LBP and the overall accuracy was enhanced. The difference between LBP and CLBP is that LBP descriptor uses only the sign component for matching features. In many cases sign component is not enough to construct the matching process. On the other hand, CLBP uses native structure magnitudes alongside the sign component. This makes it a robust feature extractor and increases the algorithm precision. Therefore, relatively more accurate results were seen with CLBP [25].

3.3.3 Segmentation-based Fractal Texture Analysis (SFTA)

Segmentation-based Fractal Texture Analysis method is proposed by Costa et al. [26] with the aim of increasing the speed of analyzing textures and extracting features efficiently. SFTA extraction is done in two steps. In the first step, the gray level image should be analyzed into a set of binary images (input images). The images are analyzed by Two-Threshold Binary Decomposition (TTBD). Fractal dimension and mean gray level are computed from border pixels of each image as seen in Figure 3.5.

TTBD starts by calculating threshold standards for images and this is done by choosing gray level values with equal spaces. Then two threshold values are applied on images by the following equation:

$$I_b(x,y) = \begin{cases} 1, \text{if } t_l < I\,(x,y) \le t_u \\ 0, \text{otherwise} \end{cases} \tag{3.11}$$

where t_l is upper threshold value and t_u is the lower value.

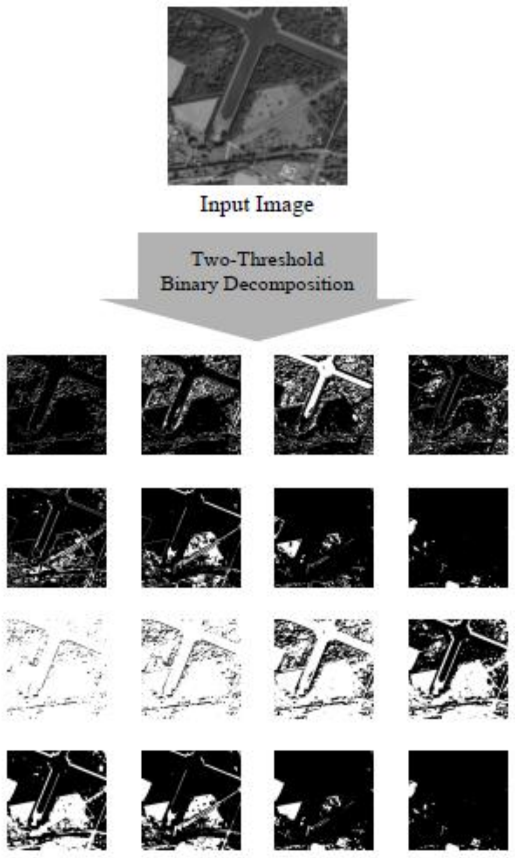


Figure 3.5: Decomposition of a Satellite Image using TTBD Algorithm [26].

The reason why two threshold values are used is because of segmentation of images. Because using single value is a challenging work, especially for the textures with middle range gray levels. This can be seen clearly in Figure 3.6 as a texture of terrazzo stone. Using two threshold values helps to extract all of the texture information successfully.

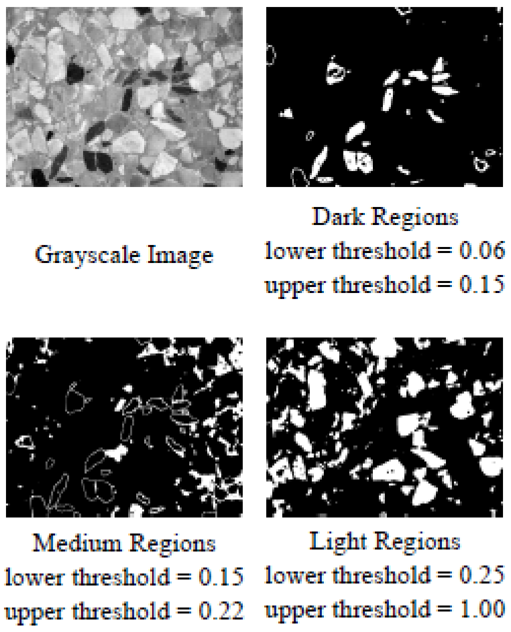


Figure: 3.6 Textures of Terrazzo Stone [26].

After the process of TTBD, the SFTA feature vector is calculated by combining size of images, mean gray level and fractal dimension of border pixels denoted by △(x,y). SFTA feature vector is calculated by using the following equation:

$$\Delta(x,y) = \begin{cases} 1 \text{ if } \exists (x',y') \in N_8\,[(x,y)]: \\ \qquad I_b(x',y') = 0 \\ \qquad I_b(x',y') = 1, \\ 0, \text{ otherwise} \end{cases} \tag{3.12}$$

where $N_8\,[(x,y)]$ is the set of pixels that are 8-connected to $[(x,y)]$.

Figure 3.7 illustrates SFTA extraction algorithm. First, the input image is separated into a series of binary images by TTBD algorithm, then SFTA feature vector is constructed.

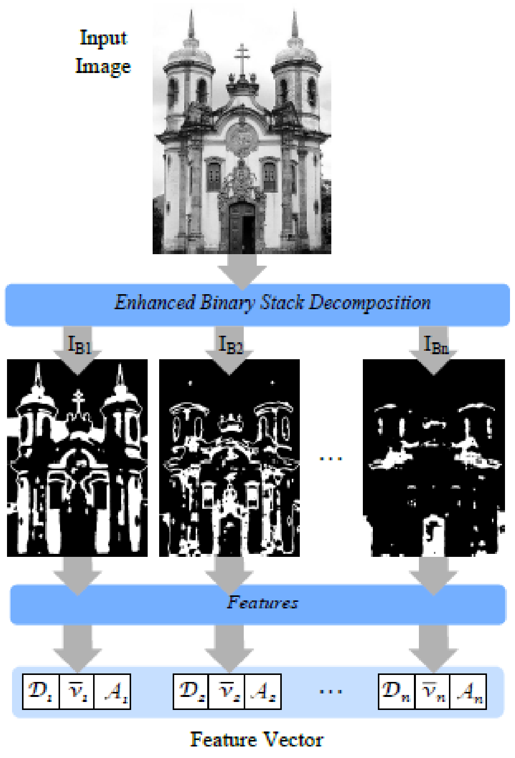


Figure 3.7: SFTA Extraction Algorithm [26].

3.3.4 Histogram of Oriented Gradients (HOG)

Another feature extraction method that is used for the feature extraction of images is Histogram of Oriented Gradients **(**HOG) [27, 28]. It is a solid feature detector that extracts features from all parts of image. HOG uses the data from histogram of gradients to form the shape of objects. Histogram of an image is the graphical illustration of the digital image. It shows how many pixels belong to the same gray level value.

The calculation of HOG is a three step process: first gradient of the image is calculated in this step by adding 1x3 filter (-1 0 1) to the horizontal axis and $(-1\ 0\ 1)^T$ to the vertical axis of the image. After that, the orientations of the images are bound. The image is partitioned into smaller and constant blocks. The third step is histogram generation, which is the process of plotting the orientation histogram and appointing each value of histogram for every tonal value. Then, the final histogram is obtained as seen in Figure 3.8.

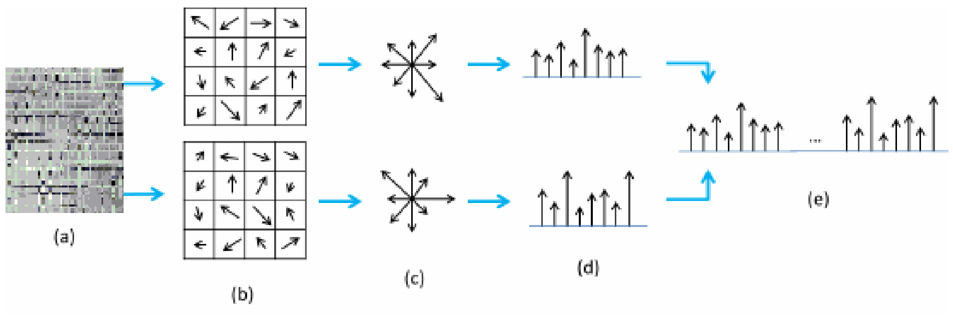


Figure 3.8: Steps of Calculation of HOG Features [27].

One problem arises here as the value of gradient changes. When the orientation of the object is changed, in the experiments, HOG's outcome was the lowest of all because of its weakness against rotation. It can be seen in Figure 3.9 how the histogram of the image changes with the change of orientation of the book in the image.

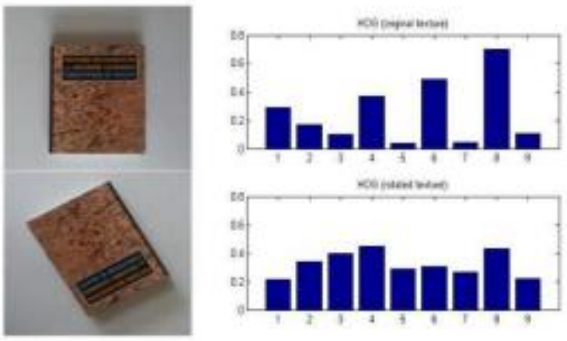


Figure 3.9 Images of a Book and Their Histograms Before and After Rotation [27].

Figure 3.9 shows that the book's image is not changed, only its orientation is changed slightly. This happens because the direction of illumination is changed which leads to change in the value of the tone for each pixel. But when it comes to texture, there is always the risk of changing orientation, because in nature the pattern of textures, are rarely symmetrical and orientation is not constant. In order to solve this problem, RIHOG (Rotation Invariant Histogram of Oriented Gradients) approach is used, which is an important version of HOG.

3.3.5 Rotation Invariant Histogram of Oriented Gradients (RIHOG)

Rotation Invariant Histogram of Oriented Gradients **(**RIHOG) [29] is a variant of HOG descriptor. It basically imitates HOG algorithm with the advantage of robustness against rotation. This is achieved by storing each pixel's neighborhood data so that when the image is rotated, the pixels information is stored according to other neighboring pixels. Then, by plotting the histogram which is rotation invariant, RIHOG turns out to be a vigorous feature extractor which is not affected by rotation as seen in Figure 3.10.

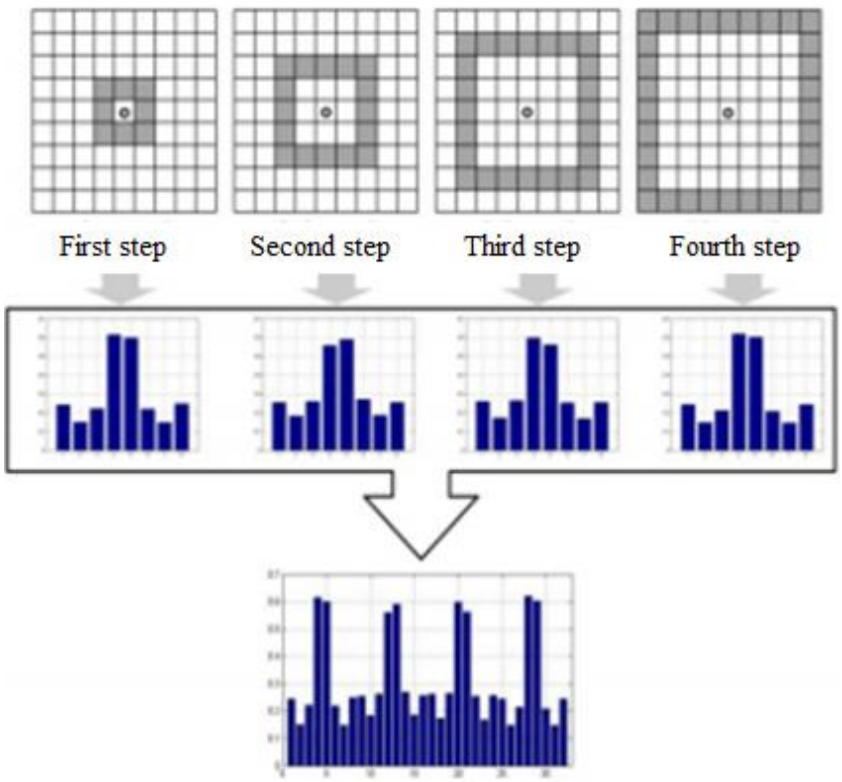
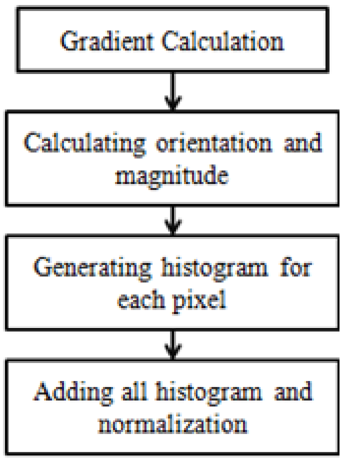


Figure 3.10: Neighboring Steps of Histogram Generation [29].

The process of generating RIHOG features is initialized in the same manner as HOG. It is done by using the same filter as HOG. The difference is that in HOG only the information about the magnitude is used for generating the histogram, but in RIHOG, both magnitude and orientation information are used. After the magnitude and orientation results for all pixels of the image are obtained, the information is forwarded to plot the histogram. When the difference between the orientation of a pixel and its neighboring pixel is calculated, that is called orientation information of a pixel. Therefore, orientation and magnitude information for each pixel is stored and histogram is plotted. Afterwards all histograms are combined for all pixels and results are normalized. Figure 3.11 illustrates steps of this process.


Gradient Calculation
Calculating orientation and magnitude
Generating histogram for each pixel
Adding all histogram and normalization
Figure 3.11 Steps of Calculating RIHOG Features.

Comparing HOG and RIHOG in experiments, the results were improved significantly up to 23% in some cases of this experiment. This shows the importance of the rotation invariant feature extractors in the field of texture classification.

3.3.6 Haralick Features

Haralick features are extracted from Gray Level Co-occurrence Matrix (GLCM) [30] which is a popular way for image feature extraction. GLCM with its straightforward operation gives outstanding results and it stores the occurrence of distinct types of gray level pixels in a system of tables. That is why Haralick has become a suitable feature extractor among many researchers. From the name of Co-occurrence matrix, it is understood that the matrix is devoted to frequency of occurrence. The matrix uses tables to store how often each pixel is followed by another pixel of the same type. This can vary at a given distance, **d.** In algorithm of GLCM, four different matrices for each direction P are used. On the other hand, P0 is used for calculation of horizontally neighboring pixels. P90, is used for vertical neighborhood and both P45 and P135 are used for diagonal neighborhood in both directions as shown in Figure 3.12.

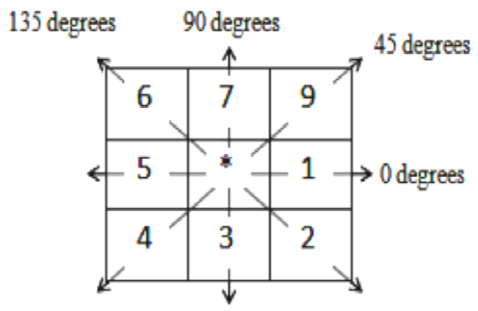
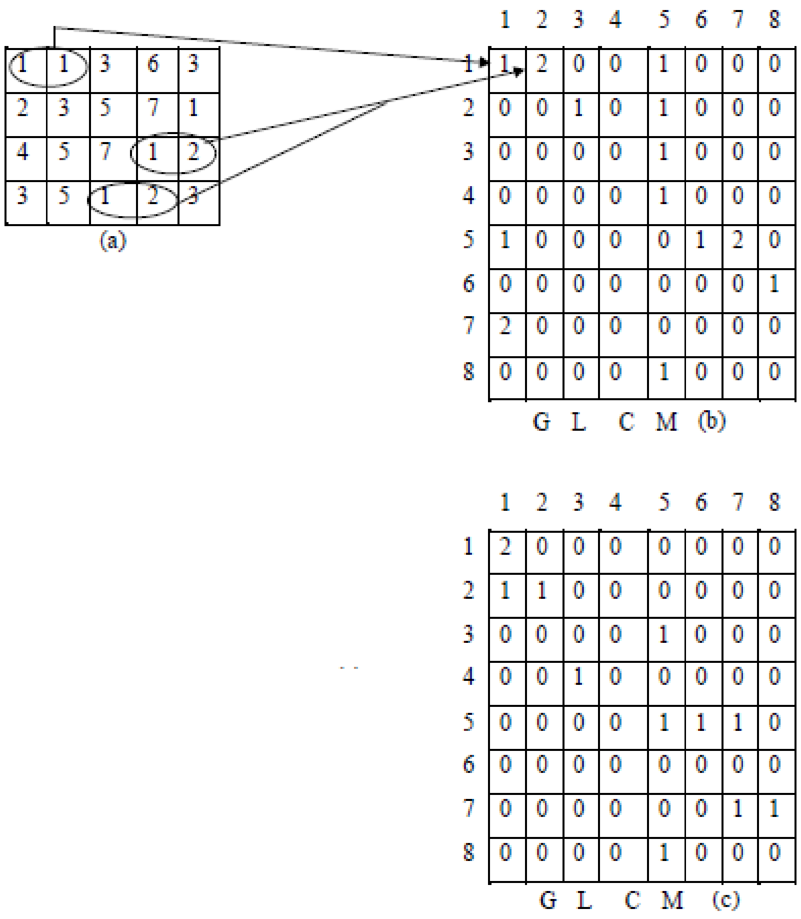


Figure 3.12: Calculating GLCM in 4 Directions [30].

In general, Haralick extracts features from images and GLCM uses Co-occurrence matrix to analyze the textures. GLCM selects each pixel and its neighboring pixel and then checks their value and stores how often each set of pixels are repeated in the matrix. Figure 3.13 summarizes the calculation of GLCM.



1	1	3	6	3
2	3	5	7	1
4	5	7	1	2
3	5	1	2	3

(a)

	1	2	3	4	5	6	7	8
1	1	2	0	0	1	0	0	0
2	0	0	1	0	1	0	0	0
3	0	0	0	0	1	0	0	0
4	0	0	0	0	1	0	0	0
5	1	0	0	0	0	1	2	0
6	0	0	0	0	0	0	0	1
7	2	0	0	0	0	0	0	0
8	0	0	0	0	1	0	0	0

G L C M (b)

	1	2	3	4	5	6	7	8
1	2	0	0	0	0	0	0	0
2	1	1	0	0	0	0	0	0
3	0	0	0	0	1	0	0	0
4	0	0	1	0	0	0	0	0
5	0	0	0	0	1	1	1	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	1	1
8	0	0	0	0	1	0	0	0

G L C M (c)

Figure 3.13: Calculating GLCM [30] (a) Original image
(b) GLCM with angle of 0 degrees (c) GLCM with angle of 45 degrees.

3.4 Feature Classifiers

Humans can classify objects in their lives easily. For example, when a human sees a car, they can say at a glance it is a car because they have learnt in the past the concept of cars. So, human's brain quickly decides on the class of object that they see. The same method applies to computer vision in order to distinguish between classes. First of all, features of a specific class must be learnt then it can distinguish between classes. So, feature extractors extract the main properties of every class and store them. Afterwards feature classifiers compare the input image with the stored properties of each class and then it decides the class of the image [31]. There are different types of classifiers and they differ in the way they work. In the experiments of this thesis, two different classifiers have been used for classification process namely Naive Bayes classifier and Support Vector Machines (SVM).

3.4.1 Naive Bayes Classifier

Naive Bayes classifier belongs to a family of Bayesian classifiers which are probabilistic classifiers. They use probability to decide on the results. In this case, it gives the probability of a texture belonging to a specific class. It uses Bayes' theories. The name Naive comes from the mechanism of its work which handles each attribute independently and assumes that their values do not affect each other, which is a conditional independence mechanism. Regardless of being a simple classifier, Naive Bayes classifier in some cases can get better results than complex classifiers [32].

3.4.2 Support Vector Machines (SVM)

Support Vector Machines belong to a group of machine learning techniques. The work of machine learning can be compared to human learning usage of tools. For example, it is known that we can use a pen for writing and a knife for cutting. Support Vector Machines are supervised learning mechanisms with their special algorithms that show how to handle the train set of data. When there are features from more than one set of train data, the training algorithm tries to differentiate between them and appoint input texture image to previously known features and afterwards to its corresponding class. SVM tries to find a decision boundary among classes. It can achieve this by finding the gaps between two classes in the feature space [33]. The performance of SVM is directly connected with the quality of features, so when different feature extractors are used, the performance of SVM also changes accordingly. SVM is sensitive to noise and can not perform very well when images contain noise. Another problem with SVM is that the algorithm is very time consuming and when a large database is used, it takes a long time to process the data and to get results. Regardless the disadvantages in the experiments of this thesis the best results were achieved using the SVM classifier when compared to Naive Bayes classifier.

Chapter 4

EXPERIMENTS AND RESULTS

Various experiments are conducted on publicly available texture databases in order to obtain the texture classification performance using several feature extractors. The following subsections give information about these texture databases. Then the experimental setup and different types of experiments are explained. Finally, discussion on experimental results are presented.

4.1 Description of Databases

In the experiments, three benchmark databases namely the CUReT, OUTex and Textured Surfaces databases were used and a brief description of each database is given in the following subsections.

4.1.1 The Columbia-Utrecht Database (CUReT)

Researchers at Columbia University and Utrecht University have done a joint work to construct this database. CUReT database [31] includes 61 different material classes. Some samples of CUReT database are shown in Figure 4.1.

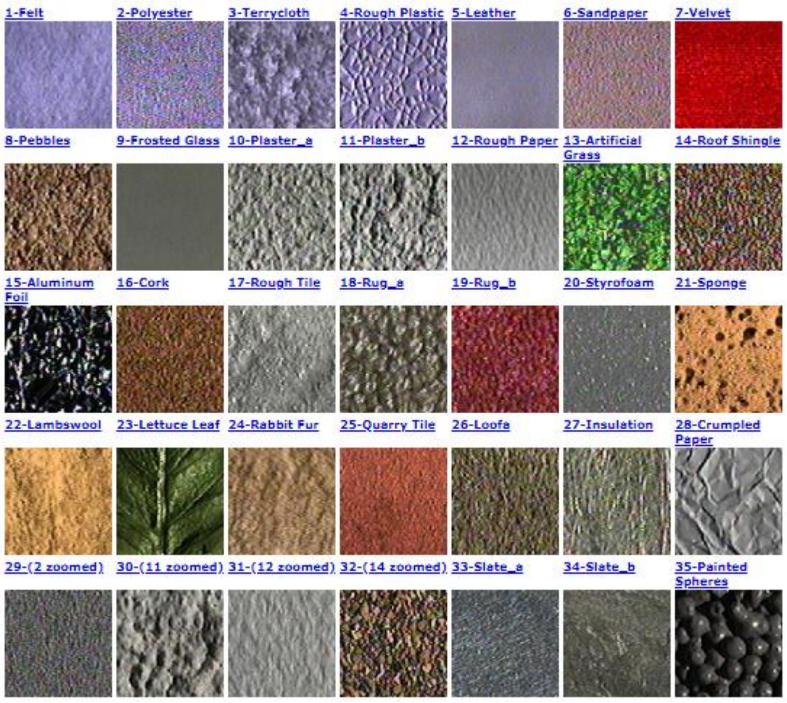


Figure 4.1: Samples from CUReT Database [31].

It includes natural textures such as fur, wood, etc. and textures from manmade materials like artificial grass, aluminum foil, etc. Each of these materials have 205 samples of 512 x 512 pixel images which are taken from different angles and have different illumination sources. In recent years, the CUReT database have become a benchmark and it is a popular database in the field of texture classification. In the experiments, images are cropped to 200 x 200 pixels from the center and only 92 images per class are chosen based on the angle of view and illumination, The images that only include texture without any background are selected and other images that do not include enough texture foreground are ignored. Original images are RGB color images. All selected images are then converted to gray scale images and then histogram equalization is applied. Afterwards the intensity is normalized to zero mean and unit standard deviation. Total images used are 5612 images and they are divided into 2806 images for train set and 2806 images for test set in which 46 images per class are randomly selected.

4.1.2 OUTex Database

OUTex stands for University of Oulu Texture database. This database contains many images from natural textures and surface textures. OUTex database [32] is considered to be a framework for evaluation of texture classification and segmentation algorithms. All images have been taken in the laboratory under controlled environment. The datasets used in experiments are OUTex_TC_10 and OUTex_TC_12in which each of them contains 24 classes of different textures and 180 images per class. Figure 4.2 shows samples of OUTex database.

OUTex database includes different subsets such as OUTex_TC_10 and OUTex_TC_12. OUTex_TC_10 subset includes images that have six spatial resolutions and nine rotation angles. But in OUTex_TC_12, images have three different illumination angles and six spatial resolutions and nine rotation angles.

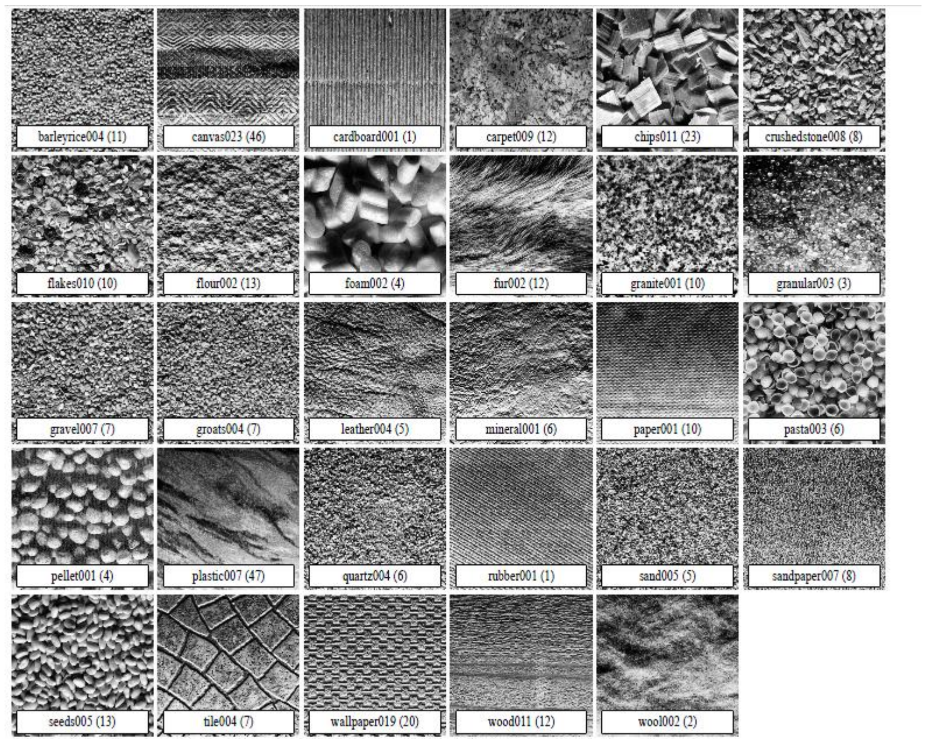


Figure 4.2: Samples from OUTex Database [32].

Having many illumination angles and texture rotation angles makes this database different from other databases. Therefore, it is a strong feature of this database when it comes to classification. All the images are perfectly cropped to 538x746 pixels and pre-divided to 2160 images for training and 2160 images for testing.

4.1.3 Textured Surfaces Database

Textured Surfaces Database [33] belongs to the natural texture images category. It means that the natural objects from real world are dominant in this database and the database contains textures from 25 different surfaces. Each texture has 40 different images of 640x480 pixel images with different angles and different focus level as shown in Figure 4.3. The illumination settings of images are not controlled and it contains many textures like water, wood, wool and marble, and some three-dimensional objects like fur and gravel. In the experiments, a total of 1000 images are used, equally divided to 500 images for testing and 500 images for training. No modifications were applied to the images and original images were used in the experiments. Hence the size of this database is relatively small compared with other two databases used, consequently the experiments conducted on this database were run faster than the experiments conducted on other large databases.

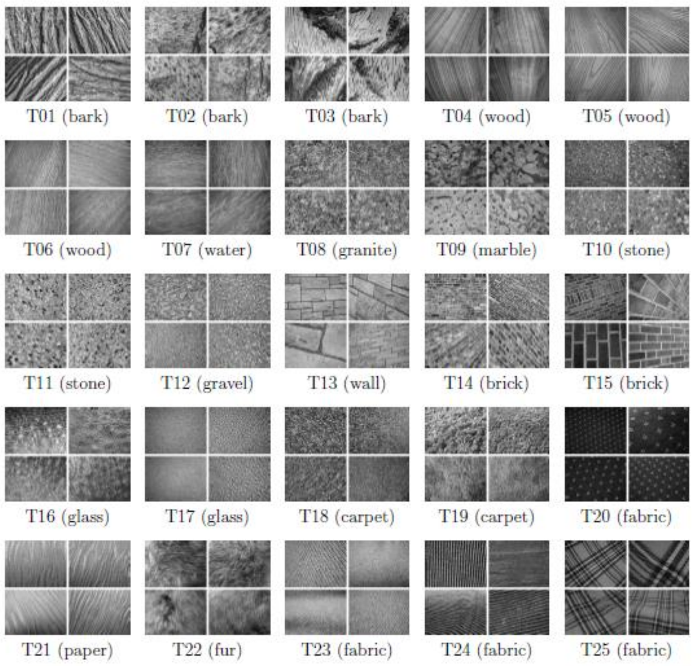


Figure 4.3: Samples from Textured Surfaces Database [33].

4.2 Experimental Methodology

The experiments were implemented using Matlab 2017a using a computer with Intel i7 processor and 6 GB of RAM. To give an illustration to the experiments, they are mainly divided into three main parts. In the first part, 50% of the images are used for train set and the remaining 50% are used for test set. In the second part, 60% of the images are used for train set and 40% are used for test set. In the last experiment, only 20 classes were chosen from each dataset, and they were equally divided into train and test sets. Accuracy of many texture classification methods are compared and analyzed in different conditions. Finally the results are coupled with two classification methods in which the best results are shown in the next subsections.

4.3 Experiments Using Different Databases

The first set of experiments are presented below that are conducted on different databases. All feature extractors are run over each database at a time. Then, features of all train sets are extracted and feature matching is done by Naive Bayes and Support Vector Machine classifiers, separately.

4.3.1 Experimental Setup I

All feature extractors are tested on each database one by one. Moreover, all the images were divided equally so that 50% of images are used for training and the remaining 50% are used for testing.

4.3.1.1 Experiments Using CUReT Database

In this experiment, all 61 classes from CUReT dataset were selected and each class has 92 samples. Total number of samples equals 5612 which are divided to train and test samples evenly. The results can be seen in Table 4.1.

Table 4.1: Recognition Rates Using the Experimental Setup Ion CUReT database

	Feature Extractor	Classifier: Naive Bayes	Classifier: SVM
1	LBP	79.34%	85.47%
2	CLBP	94.17%	97.28%
3	HOG	43.19%	46.58%
4	RIHOG	90.84%	92.31%
5	SFTA	74.77%	81.40%
6	Haralick	75.88%	77.56%

It can be seen that CLBP with SVM classifier has the best result, and in case of RIHOG, results are acceptable and after that LBP and SFTA also have moderate results. Overall results obtained from SVM are much more accurate than Naive Bayes results.

4.3.1.2 Experiments using OUTex database

OUTex_TC_10 and OUTex_TC_12 datasets have 24 classes. Each class contains 180 samples with total 4320 images for each dataset, divided to train and test subsets equally. Both datasets have less classes and less samples than CUReT database. The results can be seen in Table 4.2 and Table 4.3.

Table 4.2: Recognition Rates Using the Experimental Setup Ion OUTex_TC_10 dataset

	Feature Extractor	Classifier: Naive Bayes	Classifier: SVM
1	LBP	84.81%	89.56%
2	CLBP	96.56%	97.85%
3	HOG	21.17%	21.37%
4	RIHOG	94.45%	95.35%
5	SFTA	84.97%	85.26%
6	Haralick	79.34%	82.50%

Hence OUTex_TC_10 has 37 less classes than the CUReT and each class has 180 samples that increase the train samples for OUTex dataset. This leads to more accurate results as seen in Table 4.2. It is clearly shown that CLBP gives the best results over all other feature extractors. The only result which decreased in this dataset is HOG results and that is because the samples of each class from OUTex dataset have six resolutions and nine rotation angles and that is a weakness point for HOG. But in RIHOG, the problem is solved and 74% improvement is seen in case of SVM classifier.

Table 4.3: Recognition Rates Using the Experimental Setup Ion OUTex_TC_12 dataset

	Feature Extractor	Classifier: Naive Bayes	Classifier: SVM
1	LBP	65.46%	70.72%
2	CLBP	89.21%	90.30%
3	HOG	19.88%	20.37%
4	RIHOG	81.24%	81.35%
5	SFTA	83.40%	89.10%
6	Haralick	76.24%	78.31%

Furthermore in case of OUTex_TC_12 dataset as seen in Table 4.3 a slight decrease in all results in comparison with OUTex_TC_10 is observed. This is becauseOUTex_TC_12 dataset has 3 different illumination angles. With this in mind, CLBP results drop from 97.85% to 90.30% that is more than 7% decrease. Only the SFTA results with SVM classifier were improved, and this is because SFTA changes the gray level image to a set of binary images for input, and this makes the illumination angles to be normalized.

4.3.1.3 Experiments using Textured Surfaces Database

Textured Surfaces database was divided according to 50/50 strategy for train and test samples. In this way the database has a limited number of only 40 images per class that makes the test and train small sets compared to the other databases. Total of 1000 images are divided into 25 classes. In Table 4.4, the recognition results on this database are presented.

Table 4.4: Recognition Rates Using the Experimental Setup Ion Textured Surfaces database

	Feature Extractor	Classifier: Naive Bayes	Classifier: SVM
1	LBP	60.20%	67.85%
2	CLBP	88.35%	90.15%
3	HOG	49.27%	48.85%
4	RIHOG	72.76%	72.35%
5	SFTA	73.87%	88.24%
6	Haralick	66.46%	74.72%

Due to the fact that the Textured Surfaces database has small number of test and train sets, it makes the running time of feature extraction and feature matching very fast. However the train samples are only 20 images and it is not a sufficient number to get more accurate results, yet CLBP and SFTA gives promising results even with limited train sets.

4.3.2 Experimental Setup II

In all cases of experimental setup II, the train set is increased by 10% compared to the experimental setup I reaching 60% for train set and 40% for testing set. Consequently, a slight improvement is seen in all results.

4.3.2.1 Experiments using CUReT Database

In this experiment all 61 classes from CUReT dataset were selected and each class has 92 samples. Total number of samples equals 5612 which is divided to 60% train that makes 3376 samples and 40% test that makes 2236 samples. The results can be seen in Table 4.5.

Table 4.5: Recognition Rates Using the Experimental Setup II on CUReT Database

	Feature Extractor	Classifier: Naive Bayes	Classifier: SVM
1	LBP	81.45%	87.34%
2	CLBP	97.46%	98.83%
3	HOG	45.37%	48.64%
4	RIHOG	91.34%	92.79%
5	SFTA	75.14%	83.10%
6	Haralick	75.97%	77.83%

In comparison with experimental setup I, overall improvement for all methods is around 1.5% and the reason of the improvements in the results is the increase in the train set. 98.83% accuracy is achieved in case of CLBP which is almost faultless.

4.3.2.2 Experiments using OUTex database

OUTex_TC_10 and OUTex_TC_10 datasets have 24 classes. Each class contains 180 samples with total 4320 images for each dataset, divided by 60% which makes 2592 samples for training set and 1728 samples for testing. The results can be seen in Table 4.6 and Table 4.7.

Table 4.6: Recognition Rates Using the Experimental Setup II on OUTex_TC_10 Dataset

	Feature Extractor	Classifier: Naive Bayes	Classifier: SVM
1	LBP	87.01%	90.88%
2	CLBP	98.07%	98.79%
3	HOG	25.49%	23.76%
4	RIHOG	95.89%	97.10%
5	SFTA	88.19%	86.74%
6	Haralick	80.37%	83.07%

Considering the results in experiment set I in Table 4.2, the results of experiment set II in Table 4.6 have been improved by nearly 2%. Both CLBP and RIHOG have outstanding results. Moreover all other results are tolerable excluding HOG results which are the least accurate in consequence of various rotation angles of samples.

Table 4.7: Recognition Rates Using the Experimental Setup II onOUTex_TC_12 Dataset

	Feature Extractor	Classifier: Naive Bayes	Classifier: SVM
1	LBP	66.78%	71.20%
2	CLBP	90.84%	92.12%
3	HOG	20.17%	20.87%
4	RIHOG	82.01%	81.93%
5	SFTA	84.14%	90.46%
6	Haralick	77.03%	78.86%

Regardless of the fact that the number of images in the train set are increased in this experiment, the results are shown in Table 4.7. They are still not as good as the results of experiment using OUTex_TC_10 dataset as shown in Table 4.6.

On the other hand, comparing with the results of the same dataset of OUTex_TC_12 dataset in experiment set I, as shown in Table 4.3, a superficial increase in overall results is seen, such as 2% increase in CLBP accuracy and 1% improvement in SFTA accuracy.

4.3.2.3 Experiments using Textured Surfaces Database

In this experiment, the Textured Surfaces database is divided by using 60% of the images for training set and 40% for testing set. 600 samples for training and 400 samples for testing are used for all 25 classes. The results of the experiment are shown in Table 4.8.

Table 4.8 Recognition Rates Using the Experimental Setup II on Textured Surfaces Database

	Feature Extractor	Classifier: Naive Bayes	Classifier: SVM
1	LBP	63.11%	70.21%
2	CLBP	90.74%	92.89%
3	HOG	51.05%	50.95%
4	RIHOG	73.13%	73.44%
5	SFTA	76.40%	91.08%
6	Haralick	66.91%	75.19%

As shown in Table 4.8, CLBP and SFTA give fair accuracy and other methods have somehow acceptable results. However the results using this database have not reached the accuracy of CUReT Database as in Table 4.5 nor of OUTex Dataset as in Table 4.6.

4.3.3 Experimental setup III

In this experimental setup, 20 classes are randomly chosen from all databases, namely the CUReT database, OUTex Database and Textured Surfaces. The images are divided into 50% training and 50% testing sets.

4.3.3.1 Experiments using CUReT database

In CUReT database 20 classes were selected randomly out of 61 classes and each class has 92 samples.

Table 4.9: Recognition Rates Using the Experimental Setup III on CUReT Database.

	Feature Extractor	Classifier: Naive Bayes	Classifier: SVM
1	LBP	84.79%	89.17%
2	CLBP	98.76%	99.58%
3	HOG	46.91%	50.37%
4	RIHOG	94.27%	96.81%
5	SFTA	79.31%	85.73%
6	Haralick	80.71%	82.29%

In Table 4.9 the results are presented. CLBP has achieved accuracy of 99.58% which is an outstanding result. After that RIHOG shows 96.81% accuracy. Overall accuracy is improved by 4% in comparison with the results in experiment I as shown in Table 4.1. The reason of this big improvement is because, 41 classes were neglected and only 20 were used. Moreover computation time was also decreased relatively.

4.3.3.2 Experiments using OUTex database

In these experiments only 20 classes out of 24 classes were selected randomly. All samples were divided evenly for training and testing sets.

Table 4.10: Recognition Rates Using the Experimental Setup III on OUTex_TC_10 Dataset

	Feature Extractor	Classifier: Naive Bayes	Classifier: SVM
1	LBP	85.17%	90.03%
2	CLBP	96.91%	98.14%
3	HOG	21.83%	22.12%
4	RIHOG	95.23%	95.86%
5	SFTA	85.33%	85.67%
6	Haralick	80.24%	83.16%

A slight improvement is seen in the results of Table 4.10 compared with the results of experimental setup I as in Table 4.2. Hence the difference between the data is not a lot since 20 classes are selected out of 24 classes.

Table 4.11: Recognition Rates Using the Experimental Setup III on OUTex_TC_12 Dataset

	Feature Extractor	Classifier: Naive Bayes	Classifier: SVM
1	LBP	65.94%	71.14%
2	CLBP	89.94%	90.87%
3	HOG	20.56%	20.88%
4	RIHOG	82.10%	82.79%
5	SFTA	83.91%	89.93%
6	Haralick	76.37%	78.51%

As seen in Table 4.11, the results are improved around 1%. CLBP and SFTA got notable accuracies using SVM classifier.

4.3.3.3 Experiments using Textured Surfaces database

In this experiment only 20 classes out of 25 classes were selected. Samples were divided evenly for train and test sets.

Table 4.12: Recognition Rates Using the Experimental Setup III on Textured Surfaces Database

	Feature Extractor	Classifier: Naive Bayes	Classifier: SVM
1	LBP	60.91%	68.37%
2	CLBP	88.97%	91.10%
3	HOG	49.89%	49.24%
4	RIHOG	72.88%	73.08%
5	SFTA	74.15%	88.93%
6	Haralick	67.13%	74.97%

Overall experimental results shows that 1% improvement is seen for CLBP results as shown in Table 4.12. Compared to the results of experiment I, as shown in Table 4.4, in general, results were not affected immensely. Because, there was not a large change on the database since only 5 classes were neglected and only 50% of the data was used for training and the remaining 50% for testing.

4.3.4 Discussion on Experimental Results

The experimental evaluations on texture classification were conducted on CUReT, OUTex and Textures Surfaces databases. The first experimental setup was constructed using 50% of the images foe training and 50% of the images for testing on all databases. Then the ratio of train-test images was changed and 60% of the images were used for training and 40% of images were involved in testing. In the last experimental setup, 20 randomly selected classes were used in the experiments instead of using all classes in order to have same number of classes for all databases.

The results of the experiments show that in all experimental setups, the best accuracies were obtained using CLBP feature extractor and SVM classifier. Additionally, RIHOG feature extractor was also efficient for texture classification compared to other feature extractors. Moreover, SFTA feature extractor was another powerful method which achieved good results on most of the datasets for texture classification. In general, it can be stated that CLBP feature extractor is the most powerful feature extractor compared to the other approaches used in this thesis, because, CLBP uses three measurement units for extracting features from an image. The first one is CLBP-Sign which represents the local difference between the gray levels. The second unit is CLBP-Magnitude which represents the magnitude or known as slope descriptor. Finally the last unit is CLBP-Center which stores the gray level value of central pixel. Therefore, the aforementioned three measurement units make CLBP the most effective and powerful feature extractor for texture classification among the other feature extractors used in this thesis.

Chapter 5

CONCLUSION

In this thesis, a comprehensive analysis of texture classification was done by performance analysis of six feature extraction methods namely Local Binary Patterns (LBP), Complete Local Binary Patterns (CLBP), Segmentation-based Fractal Texture Analysis (SFTA), Histogram of Oriented Gradients (HOG), Rotation Invariant Histogram of Oriented Gradients (RIHOG) and Haralick feature extractor. Additionally, two feature classifiers were used such as Naive Bayes and Support Vector Machines (SVM). The experiments were done on three benchmark databases namely The Columbia-Utrecht Database (CUReT), University of Oulu Texture database (OUTex) and Textured Surfaces Database. In this work it is found out that firstly, CLBP describes the texture more precisely and CLBP texture recognition accuracy is the best in all cases compared to other feature extractors. Secondly, SFTA had given good results faster than the other methods. The difference between HOG and its rotation invariant version RIHOG is tested and it is found that HOG gives bad results because it is not rotation invariant. However, in case of RIHOG, the results were improved notably. Overall performance using CLBP feature extractor and SVM classifier was the best on all three databases.

In order to get real benefits from texture classification, there has to be a well-designed application for real world problems. So the challenge here is to move the work from controlled databases in the lab to uncontrolled databases and unsupervised algorithms in real world. There are some new databases in which their samples are taken in the wild in uncontrolled conditions such as Describable Textures Dataset (DTD). Another dispute is developing a real-time texture classification application which requires the usage of state-of-the-art tools, robust feature extractors and fast feature matching classifiers.

Further work can be done in the future by using more benchmark texture databases with wide variety of classes and more class samples that have several illumination angles and rotation variations. Moreover, another motivating future research is to add more subjects to the experiments and complicated tests can be done on the new databases. In order to obtain a futuristic application and go beyond the boundaries, deep learning techniques may also be involved in this field.

REFERENCES

[1] K. Brady, "A Probabilistic Framework for Adaptive Texture Description", Ph.D, University of Nice-Sophia Antipolis, 2003.

[2] J. Beck, "Textural segmentation, second-order statistics, and textural elements", Biological Cybernetics, vol. 48, no. 2, pp. 125-130, 1983.

[3] G. Caenen and L. Van Gool, "Maximum response filters for texture analysis", in the IEEE Workshop on Perceptual Organization in Computer Vision, Washington, DC, 2004.

[4] M. Chantler, G. McGunnigle, A. Penirschke and M. Petrou, "Estimating lighting direction and classifying textures", in British Machine Vision Conference, Cardiff, UK, 2002, pp. 737-746.

[5] D. Chetverikov and A. Hanbury, "Finding defects in texture using regularity and local orientation", Pattern Recognition, vol. 35, no. 10, pp. 2165-2180, 2002.

[6] P. Felzenszwalb and D. Huttenlocher, "Efficient Graph-Based Image Segmentation", International Journal of Computer Vision, vol. 59, no. 2, pp. 167-181, 2004.

[7] G. Mori, X. Ren, A. Efros, and J. Malik, “Recovering human body configurations: Combining segmentation and recognition”. in IEEE Conference on Computer Vision and Pattern Recognition II, Washington, DC., 2004, pp 326-333

[8] J. Bradley, C. Brislawn and T. Hopper, "The FBI wavelet/scalar quantization standard for grayscale fingerprint image compression", in SPIE Conference on Visual Information Processing II, Orlando, Florida, 1993, pp. 293-304.

[9] X. Ran and N. Farvardin, "Adaptive DCT image coding on a three-component image model", in International Conference on Acoustics, Speech, and Signal Processing, volume 3, San Francisco, California., 1992, pp. 201-204.

[10] A. Lobay and D. Forsyth, "Recovering shape and irradiance maps from rich dense texton fields", in IEEE Conference on Computer Vision and Pattern Recognition, volume 1, Washington, DC, 2004, pp. 400-406.

[11] A. Zalesny and L. Van Gool, "A compact model for viewpoint dependent texture synthesis", in European Workshop on 3D Structure from Multiple Images of Large- Scale Environments, Dublin, Ireland, 2000, pp. 124-143.

[12] A. Jain and F. Farrokhnia, "Unsupervised texture segmentation using Gabor filters", Pattern Recognition, vol. 24, no. 12, pp. 1167-1186, 1991.

[13] J. Haddon and J. Boyce, "Co-occurrence matrices for image analysis", Electronics & Communications Engineering Journal, vol. 5, no. 2, p. 71, 1993.

[14] Y. Chen, M. Nixon and D. Thomas, "Statistical geometrical features for texture classification", Pattern Recognition, vol. 28, no. 4, pp. 537-552, 1995.

[15] D. He and L. Wang, "Texture features based on texture spectrum", Pattern Recognition, vol. 24, no. 5, pp. 391-399, 1991.

[16] J. Bigun, "Frequency and orientation selective texture measures using linear symmetry and Laplacian pyramid", in SPIE Conference, on Visual Communications and Image Processing, Lausanne, 1990, pp. 1319–1331.

[17] Y. Ma, W. Feng, Z. Wu, M. Liu, F. Zhang, Z. Liang, C. Cui, J. Huang, X. Li and X. Guo, "Intra-tumoural heterogeneity characterization through texture and colour analysis for differentiation of non-small cell lung carcinoma subtypes", Physics in Medicine & Biology, vol. 63, no. 16, p. 165018, 2018.

[18] E. Watanabe, M. Maeno, J. Kayashita, K. Miyamoto And M. Kogirima, "Cooking Methods for a Soft Diet Using Chicken Based on Food Texture Analysis", Journal of Nutritional Science and Vitaminology, vol. 63, no. 4, pp. 256-262, 2017.

[19] S. Jia, H. Li, Y. Wang, R. Tong and Q. Li, "Hyperspectral Imaging Analysis for the Classification of Soil Types and the Determination of Soil Total Nitrogen", Sensors, vol. 17, no. 10, p. 2252, 2017.

[20] A. Ardakani, A. Mohammadi, B. Najafabad and J. Abolghasemi, "Assessment of Kidney Function After Allograft Transplantation by Texture Analysis", Iranian journal of kidney diseases, vol. 11, no. 2, pp. 157-164, 2017.

[21] O. Due Trier, A. Jain and T. Taxt, "Feature extraction methods for character recognition - A survey", Pattern Recognition, vol. 29, no. 4, pp. 641-662, 1996.

[22] D. Lisin, M. Mattar, M. Blaschko, E. Learned-Miller and M. Benfield, "Combining Local and Global Image Features for Object Class Recognition", 2005.

[23] Y. Deng and B. Manjunath, "Unsupervised segmentation of color-texture regions in images and video", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 23, no. 8, pp. 800-810, 2001.

[24] L. Liu, P. Fieguth2 and G. Kuang, "Generalized Local Binary Patterns for Texture Classification", Image and Vision Computing, vol. 30, no. 2, pp. 86-99, 2011.

[25] Z. Guo, L. Zhang and D. Zhang, "A Completed Modeling of Local Binary Pattern Operator for Texture Classification", IEEE Transactions on Image Processing, vol. 19, no. 6, pp. 1657-1663, 2010.

[26] F. Alceu, H. Gabriel and T. Agma, "An Efficient Algorithm for Fractal Analysis of Textures", in Conference on Graphics, Patterns and Images (SIBGRAPI), Ouro Preto, MG, Brazil, 2012.

[27] N.Dalal and B. Triggs, "Histograms of oriented gradients for human detection Computer Vision and Pattern Recognition", in IEEE Computer Society Conference, 2005, pp. 886-893.

[28] M. Farmanbar and Ö. Toygar, "Spoof detection on face and palm print biometrics", Signal, Image and Video Processing, vol. 11, no. 7, pp. 1253-1260, 2017.

[29] M. Cheon, W. Lee, C. Hyun and M. Park, "Rotation Invariant Histogram of Oriented Gradients", International Journal of Fuzzy Logic and Intelligent Systems, vol. 11, no. 4, pp. 293-298, 2011.

[30] M. Cimpoi, S. Maji, I. Kokkinos and A. Vedaldi, "Deep Filter Banks for Texture Recognition, Description, and Segmentation", International Journal of Computer Vision, vol. 118, no. 1, pp. 65-94, 2016.

[31] R. Duda, and P. Hart, Pattern classification and scene analysis. Menlo Park, Calif.: Wiley and Sons, Inc, 1973.

[32] P. Domingos, and M. Pazzani, “Beyond independence: conditions for the optimality of the simple Bayesian classifier” in ICML Conference, 1996.

[33] D. Tian, X. F. Zhao and Z. Shi, “Support vector machine with mixture of kernels for image classification” in ICIIP Conference, pp. 67-75, 2012.