AUTOMATIC DEFECT DETECTION IN FABRICS USING COMPUTER VISION TECHNIQUES

By

Abubakar Rabiu

A Thesis Submitted to

The Graduate Institute of Science and Engineering

Of

Meliksah University

In partial fulfillment of the requirement for the degree of

Master of Science

In

Electrical and Computer Engineering

August, 2014.

Kayseri, Turkey.

APPROVAL PAGE

This is to certify that I have read the thesis entitled "Automatic defect detection in fabrics using computer vision techniques" by Abubakar Rabiu and that in my opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science in Electrical and Computer Engineering, the Graduate Institute of Science and Engineering, Melikşah University.

August, 2014 Asst. Prof. Dr. Kadir A. Peker

Supervisor

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

August, 2014 Prof. Dr. Murat Uzam

Head of Department

<u>Examining Committee Members</u>

Asst. Prof. Dr. Kadir A. Peker	August, 2014	
Asst. Prof. Dr. Hasan Kitapçı	August, 2014	
Asst. Prof. Dr. Ercan Şevkat	August, 2014	

It is approved that this thesis has been written in compliance with the formatting rules laid down by the Graduate Institute of Science and Engineering.

Prof. Dr. M. H. Keleştemur

Director

August, 2014

AUTOMATIC DEFECT DETECTION IN FABRICS USING COMPUTER VISION TECHNIQUES

Abubakar Rabiu

M.Sc. Thesis- Electrical and Computer Engineering

August, 2014

Supervisor: Asst. Prof. Dr. Kadir A. Peker

ABSTRACT

Quality control in textured web materials is very complex due to their vagueness and ambiguity as such they need real time integrated solutions. Over the years, there has been demand for better approaches at lower costs by the textile industries. This inspection process in computer vision literature becomes a texture analysis problem.

In our research work, we proposed a novel approach “SIFT (scale invariant feature transform) features and Bag-of-Words method with SVM (support vector machines)” to automatically carry out classification on standard regular textures and detection of fabric defects using computer vision techniques. We investigate our novel approach in two experiments and for each experiment we compared our results with that obtained using a standard method used by researchers “Gabor filter with SVM”. In the first experiment we compared the SIFT based method and the Gabor filters method on standard texture classification. And in the second experiment we compared the SIFT based method and the Gabor filters method on fabric defects detection. We selected 10 classes and 20 images per each class from Kylberg Texture Dataset (available online) and used them in the first experiment. While in the second experiment we generated a fabric defects image data set (three different kinds of defects) with 80 images and 4 classes. For each image under inspection, features/descriptors are obtained, processed and classified with SVM. Experiments show that the proposed SIFT-based method gives good results on both texture classification and fabric defect detection.

Keywords: Fabric defect detection, Texture analysis, Local Features, SIFT, Gabor filters, Texture Classification.

BILGISAYARLA GÖRME TEKNIKLERIYLE DOKUMA ÜRÜNLERİNDE OTOMATIK HATA TESPITI

Abubakar Rabiu

Yüksek Lisans Tezi - Elektrik ve Bilgisayar Mühendisliği

Ağustos, 2014

Tez Danışmanı: Asst.Prof.Dr. Kadir A. Peker

ÖZ

Dokuma ürünlerinin kalite kontrolü problemi, içerdiği belirsizlikler nedeniyle oldukça karmaşık bir problemdir ve bütünleşik gerçek zamanlı çözümler gerektirmektedir. Yıllar içinde tekstil endüstrisi tarafından daha iyi ve daha ucuz yaklaşımlara talep süregelmiştir. Bu problem bilgisayarla görme literatüründe, bir doku analizi problemi olarak çalışılmıştır.

Bu tez çalışmasında SIFT (ölçekten bağımsız öznitelik dönüşümü) öznitelikleri, kelime torbası metodu ve SVM (destek vektör makineleri) sınıflandırıcı kullanılarak, önce standart genel doku görüntülerinin sınıflandırması için, sonra da kumaş hatalarının tespiti için kullanılan yeni bir yaklaşım önerilmiştir. Bu yeni yaklaşım iki deney üzerinden, yaygın olarak kullanılan Gabor filtreleri ve SVM tabanlı bir yöntemle karşılaştırılarak incelenmiştir. Birince deneyde, SIFT tabanlı yöntem ile Gabor filtresi metodunu standart doku sınıflandırması üzerinde karşılaştırdık. İkinci deneyde ise bu iki yöntemi kumaş hatası tespiti üzerinde karşılaştırdık. İlk deney için Kylber doku veri setinden 10 sınıf ve her sınıf için 20 resim seçip kullandık. İkinci deney için ise 80 görüntüden ve 4 sınıftan (1 temiz, 3 farklı hata çeşidi) oluşan bir kumaş hata görüntüleri veri seti oluşturduk. Her görüntü için öznitelikler çıkarılıp işlendi ve SVM ile sınıflandırıldı. Yapılan deneyler, önerilen SIFT tabanlı yöntemin hem doku sınıflandırmada hem de kumaş hata tespitinde başarılı sonuçlar verdiğini göstermektedir.

Anahtar Kelimeler: Kumaş hata tespiti, Doku analizi, Yerel Öznitelikler, SIFT, Gabor filtreleri, Doku sınıflandırma.

DEDICATION

This thesis work is dedicated to the family of Alhaji Rabiu Ahmed, whom supported me with prayers and encourages me toward making my study a successful one.

AKCNOWLEDGEMENT

Praise is to Almighty Allah for being my guidance through out the period of my study and for making my research work a successful one.

My warmth gratitude goes to the first governor ever in history to sponsor thousands of Kano indigene youths to pursue their graduate program in England and different European and Asian parts of the world, in person of his Excellency the governor of Kano State Eng. Dr. Rabiu Musa Kwankwaso for sponsoring me to further my education (Graduate program), with a skillful instructors in Meliksah University Turkey. May Allah reward, guide and protect him.

My appreciation goes to my parents for assisting me morally and financially, and for their prayers.

My gratitude goes to my supervisor Ass. Prof. Dr. Kadir A. Peker who always spare enough time with me to analyze tasks and problems during my research and also for his kind advice toward making my research work a successful one.

Not forgetting you, my dear brothers and sisters for your prayers, advice and encouragement.

Finally my regard goes to all my friends and my study mates for their advice and solidarity.

TABLE OF CONTENTS

Contents

LIST OF TABLES

LIST OF FIGURES

Figure:

LIST OF ABBREVIATIONS

TP: True Positive

FP: False Positive

TN: True Negative

FN: False Negative

SVM: Support Vector Machine

SIFT: Scale Invariance Feature Transformation

CHAPTER 1

1.1 INTRODUCTION

Industrial quality control in textured web materials is very complex due to their vagueness and ambiguity as such they need real time integrated solutions. Most of the occurring defects in textile industries are local defects and over the years, there has been need for efficient fabric defect detection and classification approach by the textile industries. And this in computer vision literature becomes a texture analysis problem [10, 14]. Texture analysis is one of the fundamental aspects of human vision where by surfaces and objects are discriminated. Local variation in texture makes localization and classification difficult. While analyzing local defects, spatial arrangement of gray level values in the neighboring pixels is desirable because mean gray level or color differences in small neighborhoods alone are not sufficiently enough [3, 19].

Defects were first analyzed by human vision which is called manual inspection method. This method happens to be very slow, expensive, and inefficient and can only detect few defects [24]. It is shown in figure 1.1 below;

Figure 1.1 Manual inspection methods

In this traditional inspection method, the manufactured fabric material can be inspected in two methods. The first is the process inspection method, where an inspector will monitor the weaving process during production, if there is any change in the process, he will stop the machine and correct the yarn order. In real life, this method is time consuming and it will be very difficult to notice any change in weaving process when the material has a complex pattern. Moreover the inspection environment is not desirable for human inspection. As such, this method is not efficient and thus not commonly used. While in the product inspection method, the fabric is inspected for a defect after it has been produced. The inspection team will unroll the finished fabric on a uniformly illuminated table so that the effect of shadow will be eliminated. A motor move the fabric on the inspection table, when a defect is noticed, the inspector stops the motor and record the defect type, its location and starts the motor again. This method is also time consuming and May leads to defect misclassification due to tiredness. Its advantage over the process inspection is that, it is carry out in a favorable inspection environment.

Since then researchers have been developing machine vision algorithm that can efficiently locates and classifies defects at high speed and low cost. The machine automated inspection method is as shown in figure 1.2 below;

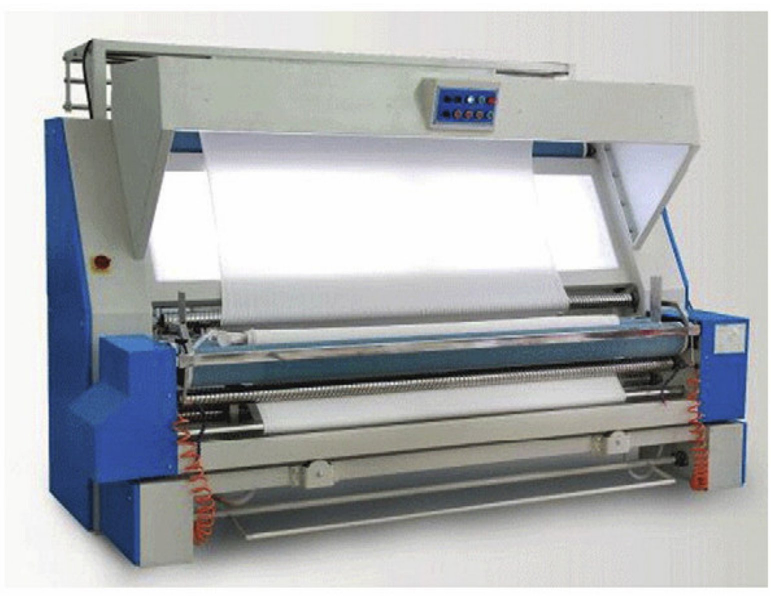


Figure 1.2 Machine automated inspection method

The automated inspection method detects and classifies fabric defects automatically by a written algorithm installed in a computer system. This algorithm is purposely written to perform defect inspection and classification task [28]. The computer system is incorporated to the machine or system together with other supporting components that include camera bank arranged in parallel to the fabric under test, a single or an array of processors, the electrical and mechanical supporting interfaces, a lighting system and the frame grabber. The lightning system provides uniform illumination on the sample under taste, while the camera bank scan through the material for defect as it moves. The frame grabber converts the data from the camera bank to pixels. And these pixels will be feed in to the processor array as an input to the installed algorithm and finally the result is displayed on the computer monitor. The architecture of the automated inspection system is as shown in figure 1.3 below;

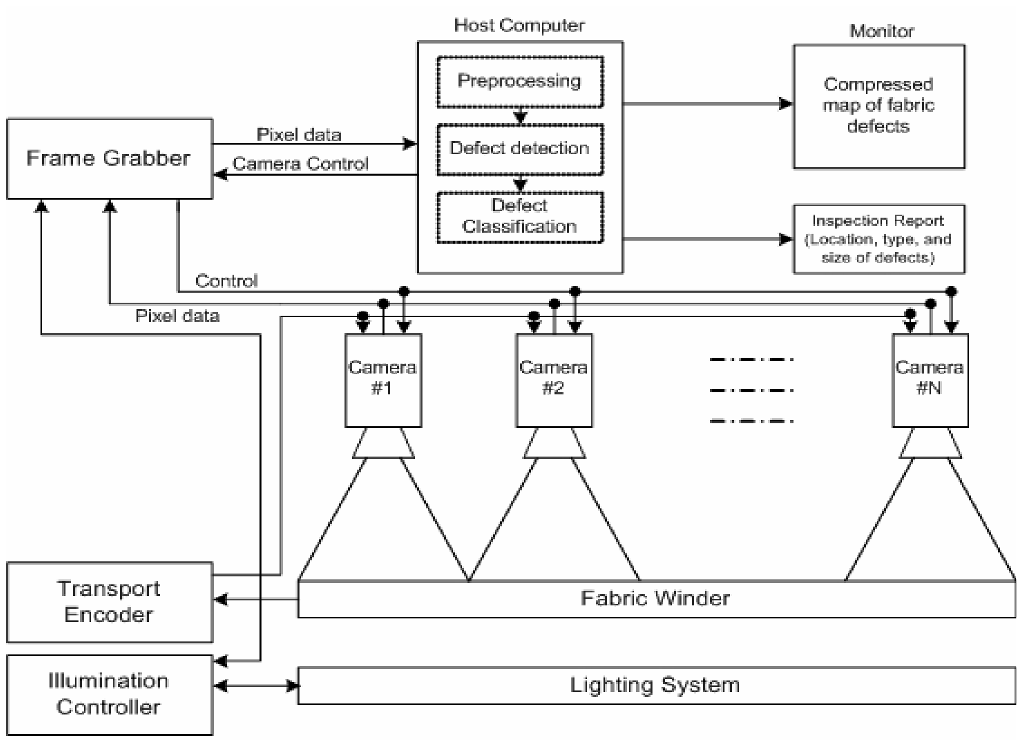


Figure 1.3 Architecture of a typical automated visual inspection system for textile web

Many methods or approaches have been used to tackle this issue, such as the SIFT descriptors and bag of words, Gabor filters, Wavelet filters and Multi scale multi orientation approaches, etc. some of which were efficient and less expensive while some were expensive with a lot of limitations [20, 25, 26]. In chapter two, we will see a review of the approaches carried out by researchers for defect detection and classification.

1.2 MOTIVATION

Manufacturing industries faces a great challenge with quality control of their manufactured fabrics and with their customers need for a better product supply. Even though so many approaches has been done, yet manufacturing industries demands for even better automated inspection methods that will improve quality, increase production and at low cost. And a lot of researchers recommended that using at least two statistical approaches in analyzing fabric defects can improve detection and classification. These motivated me to contribute my own quarter using computer vision techniques to detect and classify fabric defects.

1.3 CONTRIBUTIONS

- A Proposed novel approach to fabric defect detection by using SIFT features and Bag-of-Words method
- We Compared the SIFT based method and the Gabor filters method on a standard texture classification experiment
- We Compared the SIFT based method and the Gabor filters method on a fabric defects detection experiment
- We generated a fabric defects image data set with 80 images and 4 classes

1.4 AIM AND OBJECTIVES

The aim of this research work is to automatically detect defect in fabrics using computer vision techniques.

The objectives are;

- Apply filter bank (Gabor filters) to the fabric under test and compute a set of statistics from it (feature extraction) that are to be use for detection and classification.

- Apply SIFT based algorithm and bag of features to the fabric under test to get descriptors that are invariant to scale, view point and illumination and use them for defect detection and classification.
- Classifying the set of features obtained using Gabor filters, with SVM and the set of features obtained using SIFT base and bag of features, with SVM.
- And finally compare the results obtained using the SIFT based method and Gabor filter with SVM classifier.

1.5 SCOPE

The algorithms developed in both the SIFT based and the Gabor filter methods will extract features/descriptors from samples under inspection, and then train and test these features with SVM classifier to detect and classify fabric defects/textures.

1.6 THESIS ORGANIZATION

After the introduction part which discussed about the fabric inspection methods in chapter 1, chapter 2 shows a literature review on various approaches carried out for defect detection and classification using texture analysis. Experiments carried out using Gabor filters and SIFT based method are explained in chapter 3. Discussion of results obtained follows immediately in chapter 4, while discussion and conclusion are presented in chapter 5. Finally the references used are shown followed by the written codes for detection and classification.

1.7 METHODOLOGY

The SIFT based and the Gabor filter algorithms are developed to carry out fabric defect detection and classification in four stages.

The first part of the algorithm is called the feature extraction stage. Here equal number of defect free and defective samples of equal dimension and the same properties will be given as the input to the algorithm, and the algorithm will read these samples, process them and compute a set of features/descriptors from them.

The second part of the algorithm is known as the global features computing stage. In this stage, a processing task such as features clustering (using k-means), features histogram and mean normalization is carried out on the features/descriptors obtained in the first stage and then the global features are obtained.

The third part of the algorithm is called the training and testing stage. In this stage, the global features obtained in stage two will be divided into set of classes that correspond to the sample types used. Each class will contain equal number of global features. Some part of these features will be use as inputs to an SVM classifier for training, the classifier will learn to discriminate between the classes set as the positive examples class from the other classes (the negative examples class). A one verse all strategy is used so that each class will be trained and tested against others. Then the remaining global features are used for testing. In the testing part, the SVM classifier will attempt to detect and classify defect free samples from non defective ones. And the prediction result is compared with a ground truth data, this data is a matrix consisting of zeros and ones that correspond to the negative and positive entities respectively. And finally some performance parameters are computed and compared for the two texture analysis methods used.

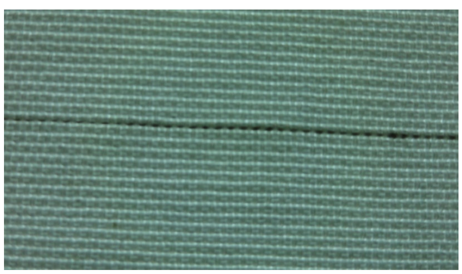
CHAPTER 2

LITERATURE REVIEW

2.1 FABRIC DEFECTS OVERVIEW

Research record shows that fabrics price reduce averagely by 50% due to the presence of defects [26]. The fabric quality is affected by the improper conditioning of yarn and non qualitative raw materials and this result to defects such as hairiness, broken ends, color or width inconsistencies and slubs in the material. Moreover, the weaving machine at times generates weaving irregularities due to change in environmental factors like the temperature or humidity which lead to a fabric defect.

Types of defects that occur on fabric materials are shown in figure 1.4 below;

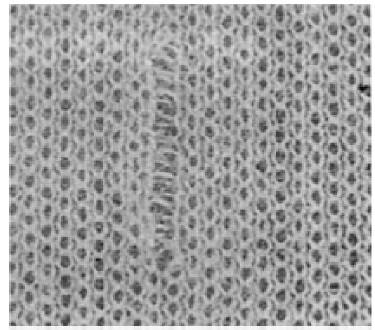


(a) Long bar

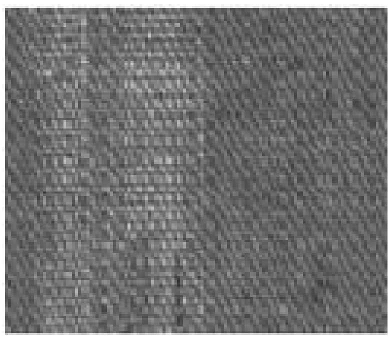
(b) Small hole

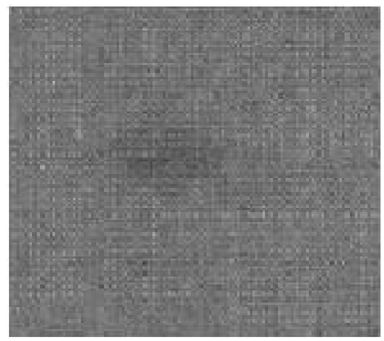
(c) Thin bar

(d) Hairiness



(e) Big hole (f) Dropped stitches





(g) Netting multipliers (h) Oil spot

Figure 2.1 (a) – (h): Types of fabric defects.

The above shown defects are called local defects, because they affect only a small portion of the fabric material.

2.2 TEXTURE AND TEXTURE ANALYSIS

2.2.1 Texture

Texture gives information about the spatial arrangement of colors or intensities in an image or its selected region. And it helps to segment images into regions of interest and classify those regions [26]. The artificial textures are made up of primitive repetition of basic elements or textons while the natural textures are complex in nature and can not be easily segmented as such can be classified by extracting statistical features from their gray tones. These texels can be segmented by thresholding, and their spatial relationship is obtained from a Verona tessellation of the texels [12, 15].

Figure 1.5 below shows some texture types;

(a) Brick (b) Upholstery

(c) Fur (d) Knit

(e) Corduroy (f) Plaid

Figure 2.2 texture types

2.3.2 Texture analysis

Texture analysis is one of the fundamental aspects of human vision whereby surfaces and objects are discriminated. The two major approaches used to analyze image texture in computer graphics are the structured approach and the statistical approach [12].

The structural approach takes an image texture as a set of primitive texels in a regular or repeated pattern, and is good in analyzing artificial textures. To obtain a description of a texture, a characterization of the spatial relationship of its texels is gathered by using Voronoi tessellation of the texels [5, 22]. While the statistical approach sees an image texture as a quantitative measure of the arrangement of intensities in a region. The later approach is easier to compute and is widely used, since natural textures are made of patterns of irregular sub elements [25].

Some of the approaches used in texture analysis include; Edge density, Co-occurrence matrix, Gabor wavelets, Fourier transform and SIFT based. Etc.

2.4 FILTER BANK

Filter banks are used in texture analysis to represent detail information of a texture. They can be use to measure edginess (that is the amount of edges per given region), and are used as a blob detector. Filters used for edge detection include; the Laplacian filter, Sobel and steerable oriented filters etc. Also Gabor filter is used for defect inspection because of their spatial localization in both time domain and frequency domain. Gabor filter is a Gaussian modulated by a complex sinusoid and it can be form in different scale and orientation [13, 17, 29].

The Gabor filters are shown in figure 1.6 below;

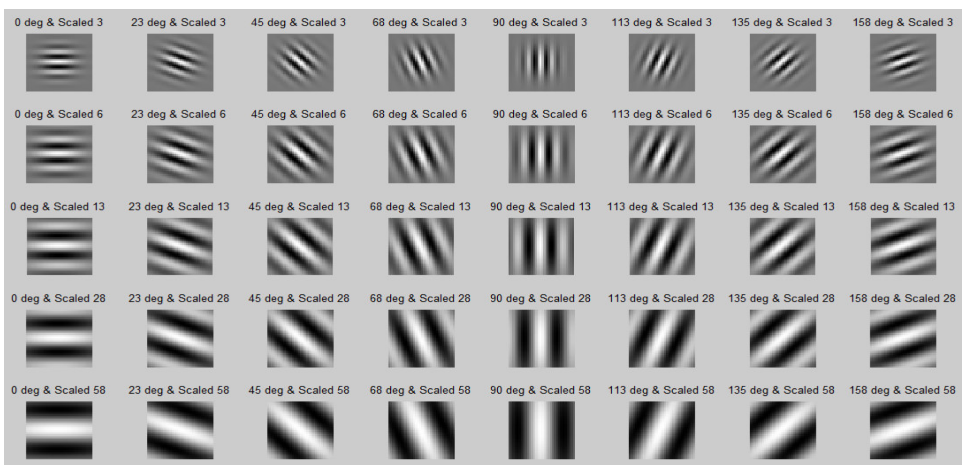


Figure 2.3 Gabor filters

2.5 RELATED WORK

Conci and Proença [1] detected eight kinds of defects in fabrics using fractal dimension estimate of the samples under test, and declared a sample as defective or non defective by looking at the fractal dimension variation. They implemented a modified differential box counting method that reduced computation complexity and enhanced efficiency on a large amount of data. The outcome of the result shows 96% detection with high false alarm and poor localization accuracy.

Zhang and Bresee [2] carried out defect detection using gray level statistics. They divided each image under test into arbitrary blocks and declare an image as defect free or defective by looking at the first order statistics of its blocks. They carefully choose the blocks size using autocorrelation function because, if the size is too large, local regions with defective texture may be lost and if it is too small, similar defect free textures can not be easily distinguish. Even though their method is computationally simple, it failed to detect defects with variable change in second and higher order moment.

Conci and Proença [3] successfully detected fabric defects using Sobel edge detection method and compared their results with those obtained using thresholding and fractal dimension. J. S. Lane [4] have used a set of mask in detecting fabric defects, they transformed the image under inspection into a gradient image and remove noise from the defective pixels by dilating the resultant image. Finally they separated defective pixels from non defective ones by thresholding.

Conners *et al*. [5] used six features obtained using co-occurrence matrix, and detect nine kinds of surface defects in wood. While Tsai *et al*. [6] have carried out fabric defect detection using two features obtained using co-occurrence matrix, and achieved maximum classification rate of 96%. Hung and Chen [7] have used the back propagation neural network with the fuzzy logic technique and they were able to classified eight different kinds of fabric defects

Chan and Pang [8] have identified defects in real fabrics using localized frequency components. Also Tsai and Hu [9] have used discrete Fourier transform to extract Fourier features of the real fabric Defects. They identified four different kinds of fabric defects; missing end, missing pick, broken fabric and stain fabric.

Optimal Gabor filter fabric defects detection has been illustrated in [10]. They form a general inspection system using a bank of symmetric and asymmetric Gabor filters so that randomly varying local defects orientations and scales on textile materials are taking care of. The real part of the Gabor filter has been shown to detect blobs.

Campbell *et al*. [11] have used model-based clustering to detect defects with faint aligned on denim fabrics and declares a fabric as defective by using the Bayesian information criterion. The Bayesian information criterion from the inspected image is estimated after some pre-processing operations such as thresholding, opening, labeling and centroiding. He concluded that the Bayesian information criterion value is a reliable indicator for the presence of defects. Kong *et al*. [12] have used K-mean clustering and a perceptual merging for defect detection on colored random textured images. He conclude that the algorithm perform successfully for colored images when they are not dominated by gray colors.

Escofet et al.1998 [11], performed local defects detection in textile web materials with periodic regular texture. He applied a multi scale and multi orientation Gabor filter scheme to the sample under inspection. The method is robust and has been tested with a variety of fabrics and he finally concluded that the designed algorithm automatically segments defects from regular textures.

Ajay Kumar [2] used a multi-channel filtering method which uses Bernoulli's principle to integrate images from different channels. He used image size and yarn impurities as the key parameters to tune the sensitivity of the algorithm. The sensitivity of the algorithm is higher in conjunction with low spatial sampling and lower in the presence of yarn impurities. Based on the results obtained he concluded that the algorithm is efficient, scalable and automatic for detecting local defects in textured materials.

K. L. Mak and P. Peng[12], have carried out detection and classification of defects using Gabor wavelets with one hidden layer so that basic texture features can be obtain. They analyzed fabric defects of different shapes, sizes, backgrounds and resolusıons. The results obtained shows that the method is automatic, robust and efficient with low false alarm.

Warren J. Jasper, Stephen J. Garnier and Harsh Potlapalli [20], have used adaptive wavelets bases and obtained features on woven fabrics. They successfully classified defects and have concluded that the adapted wavelet basis has higher sensitivity to abrupt changes caused by defects in textures.

J. Escofet, R. Navarro, M.S. Millan, and J. Pladelloreans [15], have proposed a Gabor wavelet network to tackle problem of automated detection of defects in textile materials.

They extract features using the proposed method and based on the features extracted, they designed optimal Gabor filters which consist of real valued Gabor filter and a smoothening filter. They tested the algorithm offline on seventy eight textile images and concluded that the new algorithm is robust, accurate and with low false alarm.

D. Chetverikov and A. Hanbury [22], have analyzed defects in textures based on regularity and local orientation which are assumed to present structural defects viewed as non homogeneous in regularity and orientation fields. The first approach looks for regions of abruptly falling regularity while the second one considers the dominant orientation. Both methods are better applicable to different kinds of patterns. They illustrated two tests to assess and compare the two methods. In the first test, they processed diverse textures individually and searched defects in each pattern. In the second test, they classified the defects into groups of textiles. They concluded that the regularity and local anisotropy can provide a reasonable framework for general approach to detect structural faults. And have recommended combining the two methods for efficiency, since they have different scope.

F. S. Cohen, Z. Fan and S. Attali [17], have carried out defect inspection on textile fabric using visual textural properties to locate various kinds of defects. They used Gaussian Markov random field to model the texture image of defect free fabric and cast as a statistical hypothesis testing problem on the statistics obtained from the model. They partitioned the image into non overlapping windows of square size with each window classified as defective or defect free based on a likelihood ratio test. They generalized that the method is applicable on real fabric samples and it is robust.

J. Wang, R. A. Campbell and R. J. Harwood [26], have proposed a method for detecting all types of textural faults on a carpet. The Gaussian Markov Random Field (GMRF) model is used for modeling the carpet textural surface. The defects occurring on the woven carpets during production process are detected by a line-scan camera and a personal computer. Measures for detecting faults are derived from the GMRF model based on sufficient statistics. They finally conclude that all types of textural faults on a plain carpet can be detected efficiently and can detect faults in colored pattern carpets by using additional techniques.

Y. F. Zhang and R. R. Bresee [22], they proposed a new method for defect detection and classification in knitted fabric using image analysis and neural networks. They carried out the analysis with six different defect types. Statistical procedures and Fourier Transforms were used in extracting features and neural networks were used to detect and classify the defects. They concluded that the method is efficient for detection and classification of most defects especially when the Fourier transforms technique is utilized.

In our research work, we proposed a novel approach “SIFT features and Bag-of-Words” to perform regular textures classification and fabric defects detection, the results are compared with the one obtained using a standard texture analysis method “Gabor filter”. To the best of our knowledge, this is the first time the SIFT based method is used to carry out automatic fabric defects detection using computer vision techniques. By applying each of the method, a number of features or descriptors are obtained from the samples under test and a set of statistics are computed from the features, then their global features are used for classification. For the SIFT based method, a k-means clustering algorithm is applied to the SIFT features obtained, and the frequency count of clusters belonging to each cluster number, gives the SIFT global (Visual vocabulary) which is use for detection and classification. The global SIFT features and the Gabor global features are each independently applied as an input to an SVM classifier for training and classification. And finally their results are compared based on some performance parameters (precision, recall, F1 score and accuracy.) The aim of this research work is to automatically detect and classify fabric defects using computer vision techniques.

CHAPTER 3

DESIGN AND EXPERIMENT

3.1 INTRODUCTION

Experiments are carried out for regular texture classification and fabric defects detection using computer vision techniques. We compared our proposed novel approach "the SIFT features and Bag-of-Words" and a standard approach used by researchers "the Gabor filter method" for regular texture classification and fabric defects detection. These approaches are good in representing and extracting a set of statistical features from textures. And a support vector classifier is used in classifying these features as either belonging to defective or defect free samples or distinguish the regular textures to base on their types. The algorithm written to carry out this task is efficient and robust.

3.2 DESIGN

3.2.1 Samples selection

We selected 10 classes and 20 images per each class from Kylberg Texture Dataset (available online) and used them in the experiment for regular texture classification. And each sample is maintained at a dimension of 250 x 150.While in the experiment for fabric defects detection, we generated a fabric defects image data set (three different kinds of defects) with 80 images and 4 classes. Also out of these eighty fabrics, sixty fabric images are defective and twenty fabric images are clean. And each image is maintained at a dimension of 602 x 602 for analysis efficiency. The data in figure 3.1 below is obtained from a Boyteks company, and it presents information on the kinds of defects that occur during fabrics production. And based on the information presented in the data and for ease of analysis, we choose "hole, oil stain and horizontal bar" among the most occurring kinds of defects and used them for our analysis.

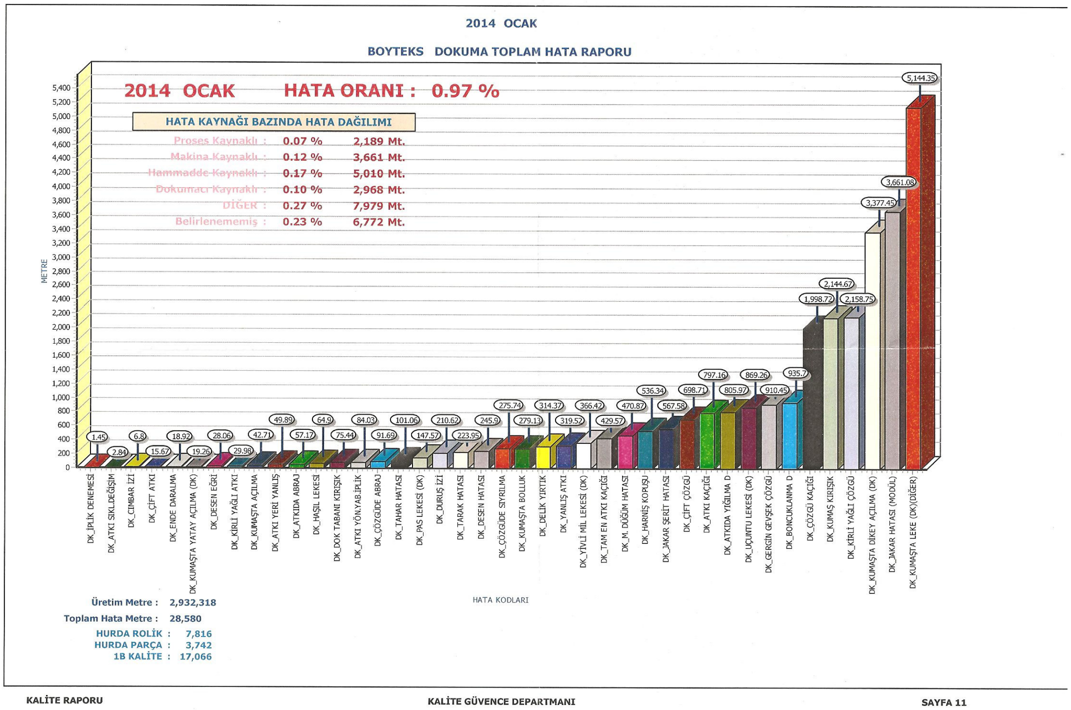


Figure 3.1a types of fabric defects

We cropped the most occurring kinds of defects as shown in figure 3.1b;

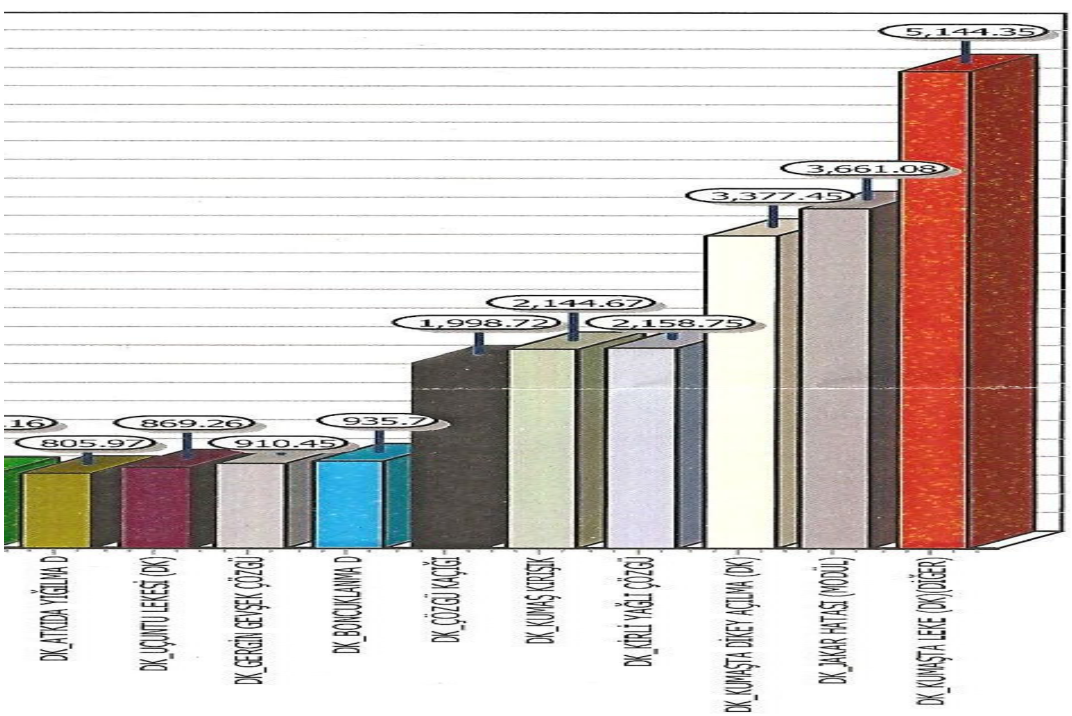


Figure 3.1b: Ten most occurring kinds of defects from Boyteks company.

3.2.2 Method selection

The SIFT based method and Gabor filter method are used in this analysis and are explained below;

SIFT features and Bag-of-Words and the Gabor filter methods are chosen in this analysis because; SIFT based method detects salient locations in an image and extract descriptors from them that are yet invariant to changes in scale, view point and illumination. And these 128-dimensional descriptors are good in representing textures.

The Gabor filter is used because of their spatial localization in both time domain and frequency domain. Gabor filter is a Gaussian modulated by a complex sinusoid. Its impulse response is shown below.

$$h(x, y) = \frac{1}{2 * \sigma x \sigma y} \exp\left[\left(-\frac{1}{2}\right) * \left(\frac{x^2}{\sigma x^2} + \frac{y^2}{\sigma y^2}\right)\right] * exp(2\pi j \mu x) \qquad \ldots\ldots\ldots\ldots\ldots\ldots \textbf{3.1}$$

Where (σx) and (σy) define the Gaussian envelope along the *x* and *y* axes. If $(\sigma x)=(\sigma y)$, the Gabor filter is circularly symmetric; otherwise, it is asymmetric. μ Represents the Gabor filter's radial center frequency and determines its location in the frequency domain. The real part and imaginary part of Gabor filter are expressed as

$$he(x, y) = \frac{1}{2 * \sigma x \sigma y} \exp\left[\left(-\frac{1}{2}\right) * \left(\frac{x^2}{\sigma x^2} + \frac{y^2}{\sigma y^2}\right)\right] * \cos 2\pi \mu x \qquad \ldots\ldots\ldots\ldots\ldots\ldots \textbf{3.1.1}$$

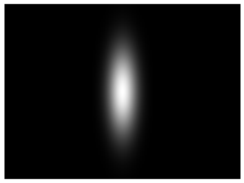
$$ho(x, y) = \frac{1}{2 * \sigma x \sigma y} \exp\left[\left(-\frac{1}{2}\right) * \left(\frac{x^2}{\sigma x^2} + \frac{y^2}{\sigma y^2}\right)\right] * \sin 2\pi \mu x \qquad \ldots\ldots\ldots\ldots\ldots\ldots \textbf{3.1.2}$$

The real part of Gabor filter, is an even function called even symmetric Gabor filter, while the imaginary part, is an odd function called odd symmetric Gabor filter. The difference of phase between them is 90°.

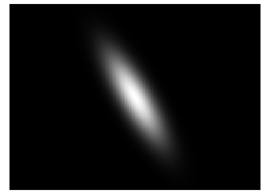
3.2.3 Gabor filters selection

A set of Gabor filters at a scale of three and nine orientations (that is; theta = 0: pi/8: pi) are applied to each of the samples under test. Then a set of statistics that is; mean and standard deviation are computed from the combine response of these filters per each sample, which give N x 18 features. These N x18 features are scaled and mean normalized which give our global features for classification.

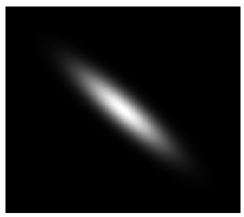
Figure 3.1 shown below, are the set of filters used in this analysis;



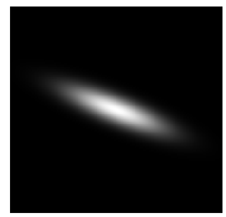
(a) 0 degree filter



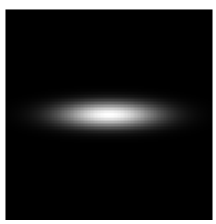
(b) 45 degree filter



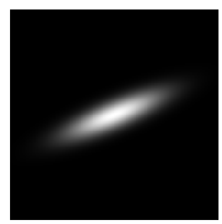
(c) 90 degree filter



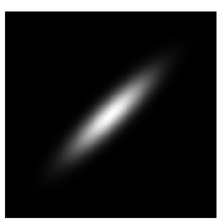
(d) 135 degree filter



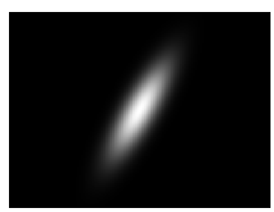
(e) 180 degree filter



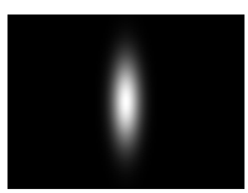
(f) 225 degree filter



(g) 270 degree filter



(h) 315 degree filter



(i) 360 degree filter

Figure 3.2 (a) – (i) Filters used for the defect detection.

These filters are formed by convolving three concentric symmetric Gaussians shown below;

$$a = \frac{\cos\theta^2}{2} * \frac{\delta x^2}{2} + \frac{\sin\theta^2}{2} * \frac{\delta y^2}{2} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \mathbf{3.3.1}$$

$$b = -\frac{\sin(2*\theta)}{4*\delta x^2} + \frac{\sin(2*\theta)}{4*\delta x^2} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \mathbf{3.3.2}$$

$$c = \frac{\sin\theta^2}{2*\delta x^2} + \frac{\cos\theta^2}{2*\delta x^2} \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \mathbf{3.3.3}$$

$$z = A * e^{\uparrow}((-(a*(〚X-x)〛^{\uparrow}2 + 2*b*(X-x)*(Y-y) - c*(〚Y-y)〛^{\uparrow}2)) \ldots\ldots \ldots\ldots\ldots\ldots\ldots\ldots \mathbf{3.4}$$

Where, $(\delta x) = 4$, $= (4 * \delta x)$ and A= $A = 2 * (\delta x) * (\delta y)$

3.2.4 Classifier selection: support vector machine (SVM)

Support vector machine is used as the classifier for this analysis, and one verse all strategy is employed for training and testing. The support vector machine is used in this experiment for so many reasons as follows; 1) it learned from the given samples before classifying them with any other class of samples, 2) it is a large margin classifier because, it chooses the best possible hyper plane separating the negative and the positive support vectors, 3) it does not stuck in the local minima, because its algorithm is convex in nature and 4) it has been used for classification purpose by so many researchers successfully and efficiently.

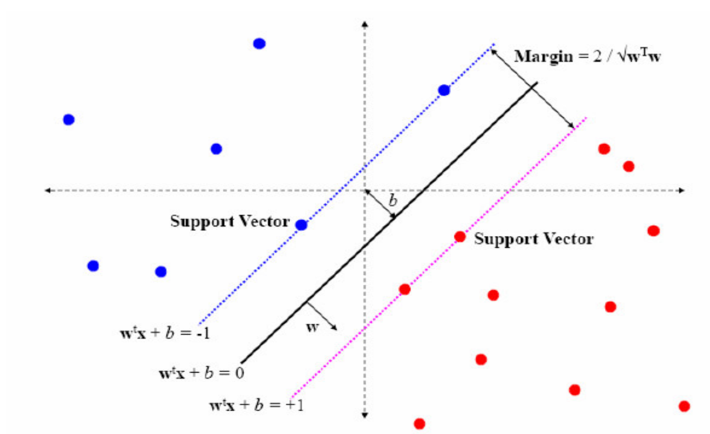


Figure 3.3 SVM classifier

The global descriptors to be use for training and classification are classified into C classes, depending on the different kind of defects to be analyzed. For analysis on six different kinds of defects, the classes should be seven with the seventh class containing defect free samples. Each class consists of equal number of features. For each class taking as the positive class, one third of its features are used for training and the remaining features are used for testing. Furthermore, the remaining classes are taking as the negative classes. And from each of them, one quarter is used for training and the remaining is used for testing. Concurrently, a ground truth matrix is developed with ones representing the position of the positive training examples and zeros representing the position of negative training examples. Similarly a classification ground truth is developed, with ones representing the position of the positive testing examples and zeros representing the position of negative testing examples. The outcome of the prediction is then compared with the ground truth in order to compute the performance of the classification. The whole procedure in this paragraph is repeated for all the ten classes, with each class playing a role of positive class and the others negative classes.

3.3 EXPERIMENT

Two experiments are carried out; the first experiment is on classification of standard regular textures while the second experiment is on fabric defect detection and classification using texture analysis. Two methods are employed; SIFT based model and the Gabor filter, which are good in representing texture and extracting a set of statistical features from them. A support vector classifier classifies these features as either belonging to defective or defect free samples.

3.3.1 Procedure

A number of defect free and defective fabric samples are collected and each sample is maintained at a dimension of 250 x 150 for analysis simplicity. Having acquired the needed samples to be inspected, they are then feed in to the written algorithms for analysis. There are two methods employed in this analysis; the SIFT based method and the Gabor filter method. Below shows the full procedure for each method and classification of the result obtained using SVM classifier.

3.3.1.1 SIFT-based method with SVM

(A) Feature extraction

The SIFT descriptor produced by David Lowe 2004 is used in this analysis and it is run using Matlab R2013a. When the algorithm is run on the total number of samples to be inspected, it takes each sample finds it key points and divide them into 16 x 16 windows with each window containing eight bins. A 128-dimensional descriptor is then computed per each window and saved in a matrix. Concurrently, an index matrix is generated by keeping track of the location of each descriptor computed. All the set of descriptors obtained per each sample are saved in the same matrix, similarly their corresponding indexes are saved in a separate matrix. The number of descriptors per image/sample depend upon the number of key points found by the SIFT algorithm on each sample and these descriptors represent our local features for analysis.

A k-means clustering algorithm is then applied to these local features into specified K number of clusters; to minimize the sum of squared Euclidean distances between point *xi* and their nearest cluster centers *mk.* And the clustering algorithm is as shown below:

$$D(X, M) = \sum_{cluster\ k} \sum_{point\ i\ in\ cluster\ k} (x_i - m_k) \qquad \text{………………………………}\mathbf{3.5}$$

- Randomly initialize K cluster centers.
- Iterate until convergence.
- Each data point is assigned to the nearest cluster center.
- Each cluster center is calculated as the mean of all points assigned to it.
- Each cluster center produced by k-means becomes a visual word.
- A vector quantization of these feature vectors gives our visual vocabulary (that is; N x K dimensional features per each sample descriptors. N is the total number of samples.) And this visual vocabulary also called SIFT global descriptors is use as input to our classifier for classification.

(B) Classification with support vector machine (SVM)

A support vector machine classifier is used for training and classification of these global features to declare a sample as being defective or non defective. The global features are first divided into C number of classes in which each class contains features belonging to a specific kind of defective samples or defect free samples.

One verses all strategy is employed for training and testing. Each class consists of equal number of features. For each class taking as the positive class, one third of its features are used for training and the remaining features are used for testing. Conversely, the remaining classes are taking as the negative classes. And from each of them, one quarter of their features is used for training and the remaining is used for testing. Concurrently, a training ground truth matrix is developed with ones representing the position of the positive training examples and zeros representing the position of negative training examples. Similarly a classification ground truth is developed, with ones representing the position of the positive testing examples and zeros representing the position of negative testing examples. The outcome of the prediction is then compared with the ground truth in order to compute some performance parameters which indicate the efficiency of the method. The whole procedure in this paragraph is repeated for all the ten classes, with each class playing a role of positive class and the others negative classes.

The following four terms; true positive (TP), true negative (TN), false positive (FP) and false negative (FN) are obtained by comparing the result of the prediction with the ground truth. True positive (TP), is the total number of positive examples classify as positive, true negative (TN) is the total number of negative examples classify as negative, false positive (FP) is the total number of negative examples misclassified correctly or classified as positive and false negative (FN) is the total number of positive examples misclassified correctly or classified as negative. The numerical values of these four terms are used in computing the performance parameters (precision, recall, f1-score and accuracy) of the algorithm to declare the effectiveness of the classification. And are computed as shown below;

1. **Precision:** This gives the percentage of number of positive examples that are predicted correctly but it says nothing on number of those that are not classified correctly. It is obtained using this formula;

   Precision = TP/(TP + FN)………………………………………………..**3.6.1**

2. **Recall:** Gives the percentage of number of relevant examples predicted correctly but it says nothing on those that are misclassified. It is obtained using this formula;

   Recall = TP/ (TP+TN)…………………………………………………….**3.6.2**

3. **F1-score:** this uniformly weights the precision and recall. And it is given by

   F1-score = 2*Precision*Recall/ (Precision + Recall)………………………..**3.6.3**

4. **Accuracy:** Gives the classification efficiency. And is obtained using the formula

   Accuracy = (TP + TN)/ (TP + FN + TN + FP)……………………………**3.6.4**

3.3.1.2 Gabor filter based method with SVM

(A) Feature extraction

A set of Gabor filters at a scale of three and nine orientations (that is; theta = 0: pi/8: pi) are used in this classification and are applied to each sample under test. Then a set of statistics; mean and standard deviation are compute from the combine response of the whole filters applied per each sample. The same thing is done for the remaining samples and these give N x 18 features, N is the total number of samples under inspection. These N x 18 features are scaled and normalized, which are then use as inputs to our SVM classifier for training and testing. The same procedure explained under classification in the SIFT based method above is applied to these N x 18 Gabor features.

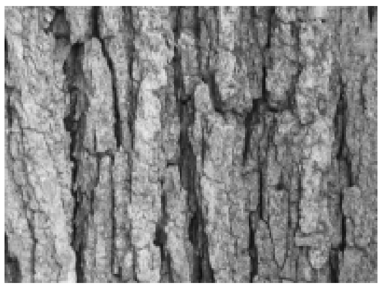
Finally the SIFT based method and the Gabor Filter method, are compared for robustness and effectiveness based on their computed performance parameters.

(B) Classification with SVM

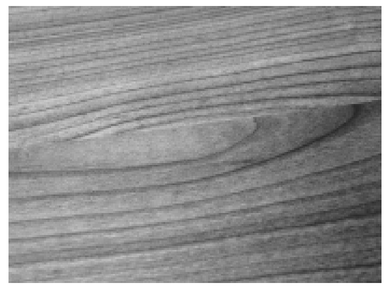
The procedure is the same as the one described for SIFT based classification with SVM in 3.3.1.1 (B).

3.4 EXPERIMENTS WITH 200 STANDARD REGULAR TEXTURES

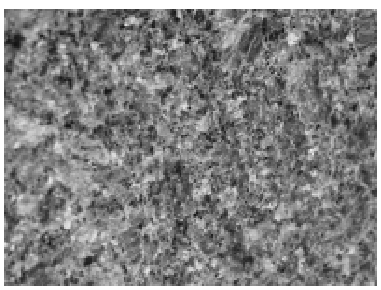
The algorithm of our proposed method mentioned above is tested on two hundred standard regular textures obtained from ten different classes. Each class has twenty standard regular textures and each sample is maintained at a dimension of 200 x 150 for analysis simplicity. The algorithm performs well, which presented a better result with SIFT based method. This is so because of the nature of the textures used, they are of different orientation. The ten classes of the standard regular textures used in experiment one, are shown in figure 3.3 below;



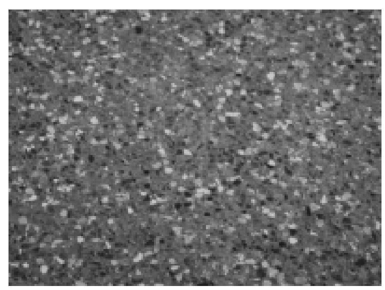
(a) Bark



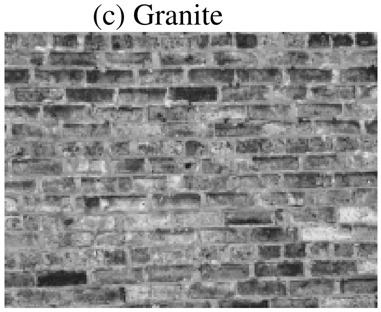
(b) Wood



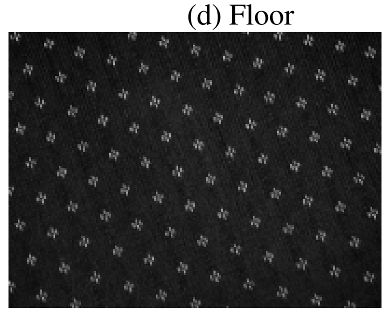
(c) Granite



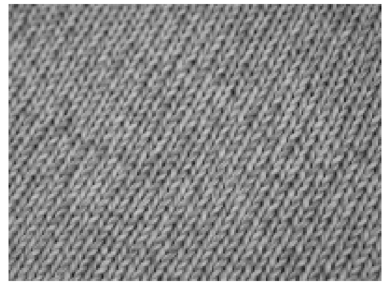
(d) Floor



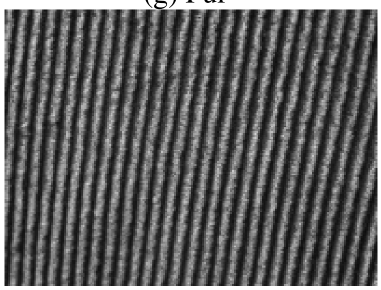
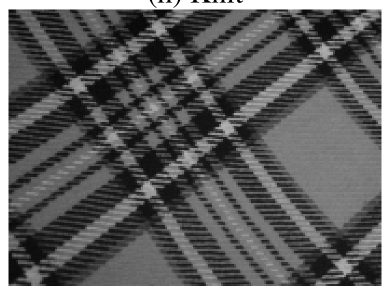
(e) Brick



(f) Upholstery

(g) Fur (h) Knit

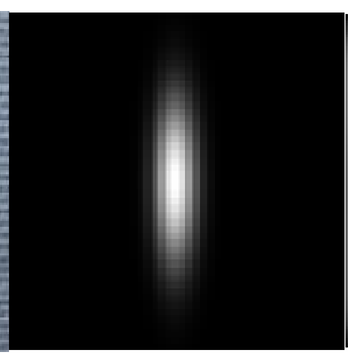
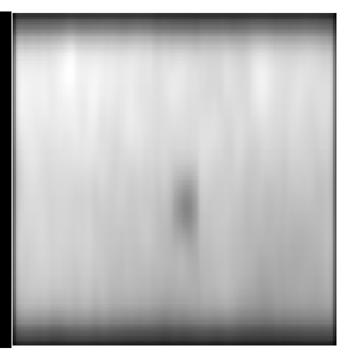
(i) Corduroy (j) Plaid

Figure 3.3 (a) – (j): Types of textures.

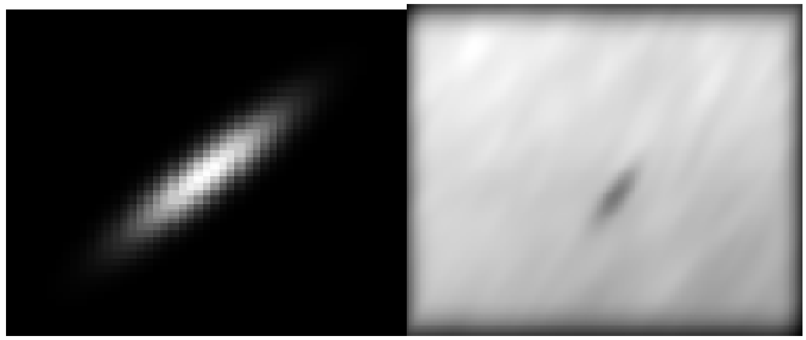
3.4.1 Classification of standard regular textures using Gabor filters

The 200 x 18 features obtained from the response of the set of Gabor filters are divided into ten classes with each class carrying 20 x 18 dimensional features. The classification procedure mentioned above is then applied to these 200 x 18 features.

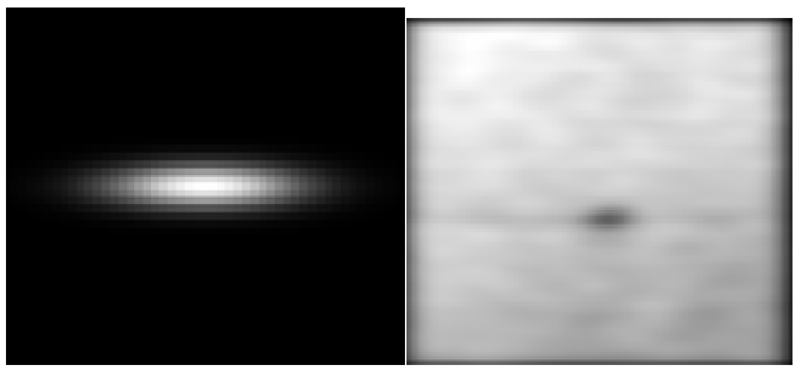
Below shows the original image, Gabor filters applied and the filtered images;

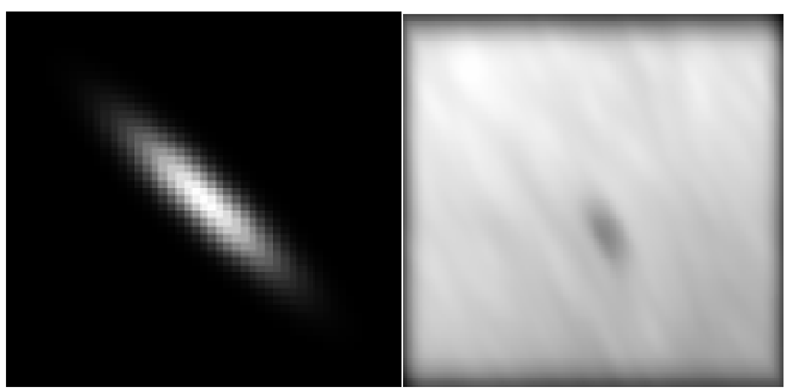
(a) image (b) 0 degrees filter (c) filtered image



(d) 270 degrees filter (e) filtered image



(f) 180 degrees filter (g) filtered image



(h) 45 degrees filter (i) filtered image

Figure 3.4 some filters with their response

And the following performance measures were obtained base on the out come of the prediction:

With class 1 as positive example class and class two to ten as negative examples class;

TP = 3; TN = 111; FP = 24; FN = 2

Precision = TP/ (TP + FP) = 3/ (3 + 24) = 0.1111 = 11.11%

Recall = TP/ (TP +FN) = 3/ (3 + 2) = 0.60 = 60%

F1-score = 2*Precision*Recall/ (Precision + Recall) = 2*0.1111*0.60/ (0.1111 + 0.60) = 0.1875 = 18.75%

Accuracy = TP + TN/ (TP + FN +TN + FP) = 3 + 111/ (3 + 2 + 111 + 24) = 81.43%

Similarly with class two as positive examples class and class 1 and three to ten as negative examples class;

TP = 4, TN = 114, FP = 21, FN = 1

Precision = TP/ (TP + FP) = 4/ (4 + 21) = 0.16 = 16%

Recall = TP/ (TP +FN) = 4/ (4 + 1) = 0.80 = 80

F1-score = 2*Precision*Recall/ (Precision + Recall) = 2*0.16*0.80/ (0.16 + 0.80) = 0.2667 = 26.67%

Accuracy = TP + TN/ (TP + FN +TN + FP) = 4 + 114/ (4 + 1 + 114 + 21) = 84.29%

The same is repeated for the rest of the classification up to when class ten is used as the positive examples class and the other nine classes as the negative examples class. And the performance measures for classification ten is as shown below;

TP = 1, TN = 123, FP = 12, FN = 4

Precision = TP/ (TP + FP) = 1/ (1 + 12) = 0.0769 = 7.69%

Recall = TP/ (TP +FN) = 1/ (1 + 4) = 0.20 = 20%

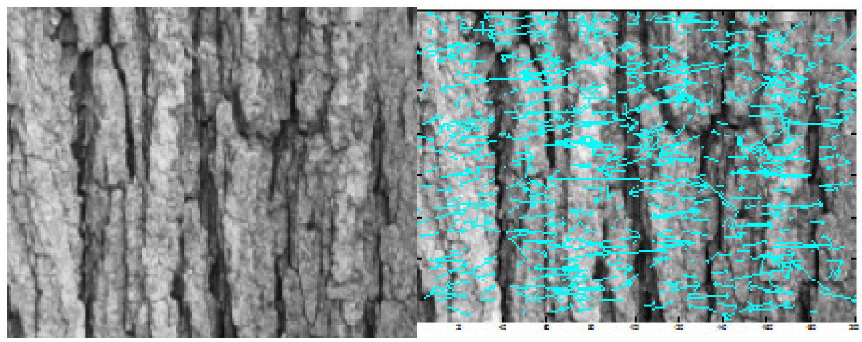
F1-score = 2*Precision*Recall/ (Precision + Recall) = 2*7.69*20/ (7.69 + 20) = 11.11%

Accuracy = TP + TN/ (TP + FN +TN + FP) = 1 + 123/ (1 + 4 + 123 + 12) = 88.57%

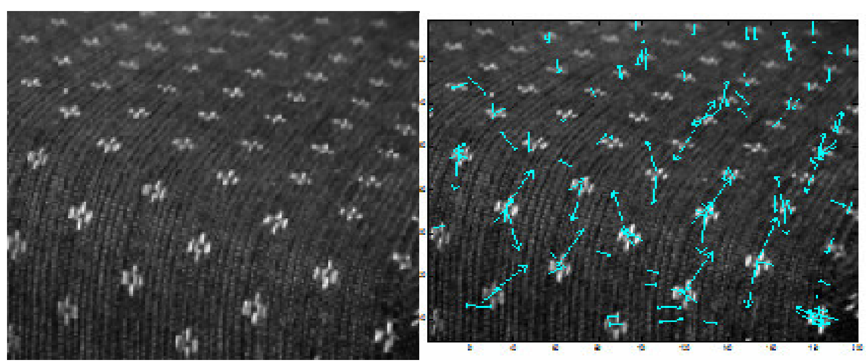
Detail classification result for the ten classes is shown in table 3.1.

3.4.2 Classification of standard regular textures using SIFT based method

The same classification procedure described above is applied to the 200 x 100 SIFT features obtained using the SIFT algorithm. Figure 3.5 below shows images with their descriptors;



(a) Bark (b) descriptors



(c) Upholstery (d) descriptors

Figure 3.5: Example of applying the SIFT algorithm on our data set

And the following performance measures are obtained base on the out come of the prediction:

With class 1 as positive example class and class two to ten as negative examples class;

TP = 5; TN = 126; FP = 9; FN = 0

Precision = TP/ (TP + FP) = 5/ (5 + 9) = 35.71%

Recall = TP/ (TP +FN) = 5/ (5 + 0) = 100%

F1-score = 2*Precision*Recall/ (Precision + Recall) = 2*0.3571*1/ (0.3571 + 1) = 52.63%

Accuracy = TP + TN/ (TP + FN +TN + FP) = 5 + 126/ (5 + 0 + 126 + 9) = 93.57%

Similarly with class two as positive examples class and class 1 and three to ten as negative examples class;

TP = 5, TN = 122, FP = 13, FN = 0

Precision = TP/ (TP + FP) = 5/ (5 + 13) = 27.78%

Recall = TP/ (TP +FN) = 5/ (5 + 0) = 100%

F1-score = 2*Precision*Recall/ (Precision + Recall) = 2*27.78*100/ (27.78 + 100) = 43.48%

Accuracy = TP + TN/ (TP + FN +TN + FP) = 5 + 122/ (5 + 0 + 122 + 13) = 90.71%

The same is repeated for the rest of the eight classifications up to when class ten is used as the positive examples class and the other nine classes as the negative examples class. And the performance measures for classification ten is as shown below;

TP = 5, TN = 135, FP = 0, FN = 0

Precision = TP/ (TP + FP) = 5/ (5 + 0) = 100%

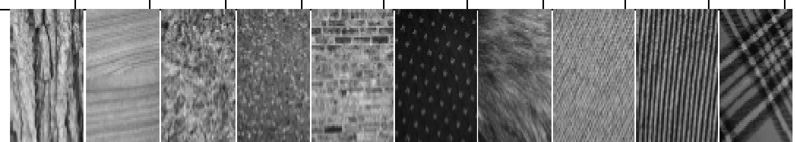
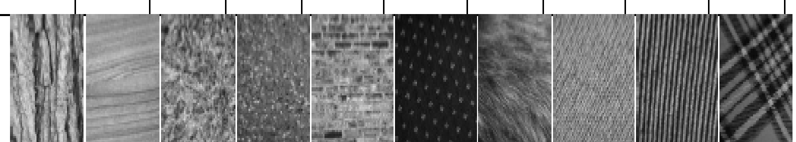
Recall = TP/ (TP +FN) = 5/ (5 + 0) = 100%

F1-score = 2*Precision*Recall/ (Precision + Recall) = 2*1*1/ (1 + 1) = 1 = 100%

Accuracy = TP + TN/ (TP + FN +TN + FP) = 5 + 135/ (5 + 0 + 135 + 0) = 1 =100%

Detail classification results for both the Gabor filters and the SIFT based methods, is as shown in table 3.1 below:

Table 3.1: Global Gabor features with SVM and Global SIFT features with SVM result

Total images used: 200; Number of classes: 10; Global features each: 200 x 100 Train_(+) = 15; Train_(-) = 45; Test_(+) = 5; Test_(-) = 135 for each classification										
Positive class others negative	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8	Class9	Class10
Image type										
Image name	Bark1	Wood1	Granite	Floor1	Brick1	Upholst ery	Fur	Knit	Corduro y	Plaid
ANALYSIS: GLOBAL GABOR FEATURES VS SVM										
TP	3	4	2	1	3	5	4	3	3	1
TN	111	114	132	122	115	134	102	122	134	123
FP	24	21	3	13	20	1	33	13	1	12
FN	2	1	3	4	2	0	1	2	2	4
Precision (%)	11.11	16.00	40.00	7.14	13.04	83.33	10.81	18.75	75.00	7.69
Recall (%)	60.00	80.00	40.00	20.00	60.00	100.00	80.00	60.00	60.00	20.00
F1-score (%)	18.75	26.67	40.00	10.53	21.43	90.91	19.05	28.57	66.67	11.11
Accuracy (%)	81.43	84.29	95.71	87.86	84.29	99.29	75.71	89.29	97.86	88.57
ANALYSIS: GLOBAL SIFT FEATURES VS SVM										
Positive class others negative	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8	Class9	Class10
Image type										
Image name	Bark1	Wood1	Granite	Floor1	Brick1	Upholst ery	Fur	Knit	Corduro y	Plaid
TP	5	5	4	5	5	5	5	5	3	5
TN	126	122	134	134	133	119	127	129	135	135
FP	9	13	1	1	2	16	8	6	0	0
FN	0	0	1	0	0	0	0	0	2	0
Precision (%)	35.71	27.78	80.00	83.33	71.43	23.81	38.46	45.45	100.00	100.00
Recall (%)	100.00	100.00	80.00	100.00	100.00	100.00	100.00	100.00	60.00	100.00
F1-score (%)	52.63	43.48	80.00	90.91	83.33	38.46	55.56	62.50	75.00	100.00
Accuracy (%)	93.57	90.71	98.57	99.29	98.57	88.57	94.24	95.71	98.57	100.00

3.5 EXPERIMEN WITH 80 SAMPLES FOR FABRIC DEFECT DETECTION

The algorithms of our proposed methods are tested on eighty fabric samples, in which twenty samples are defect free and the remaining sixty samples consist of twenty holes, twenty stains and twenty horizontal bars defective fabrics. Each sample is maintained at a dimension of 602 x 602 for analysis simplicity. The same procedure is followed, as used in the experiment with the 200 standard regular textures, only that 200 clusters are used in the k-means instead of 100 clusters. The algorithms perform well, which also presented a better result with SIFT based method.

Figure 3.6 below shows some of the fabric images used in our analysis;

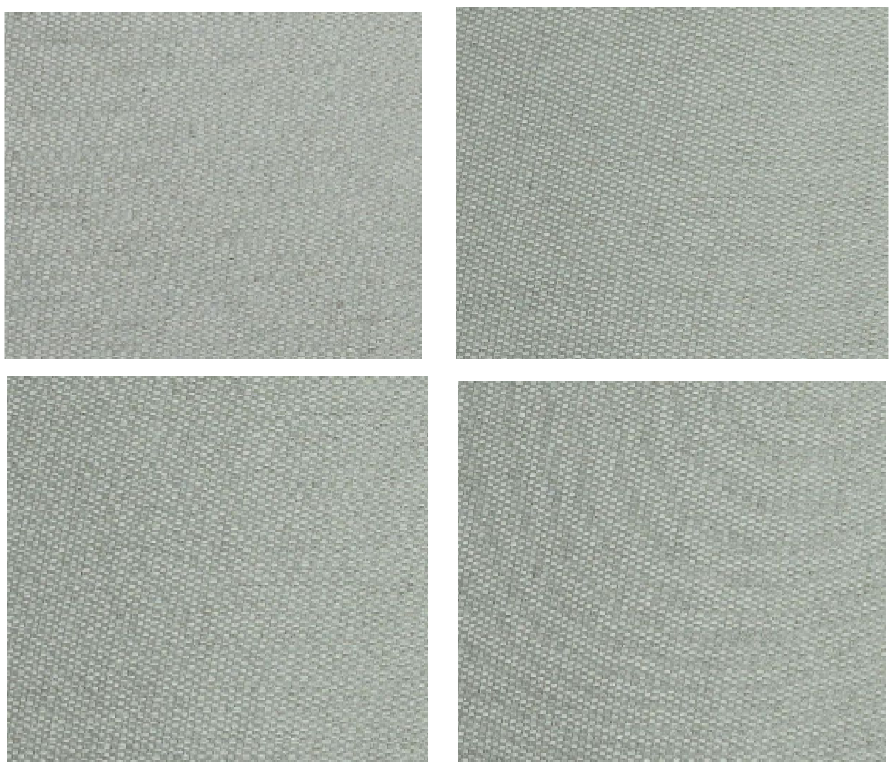
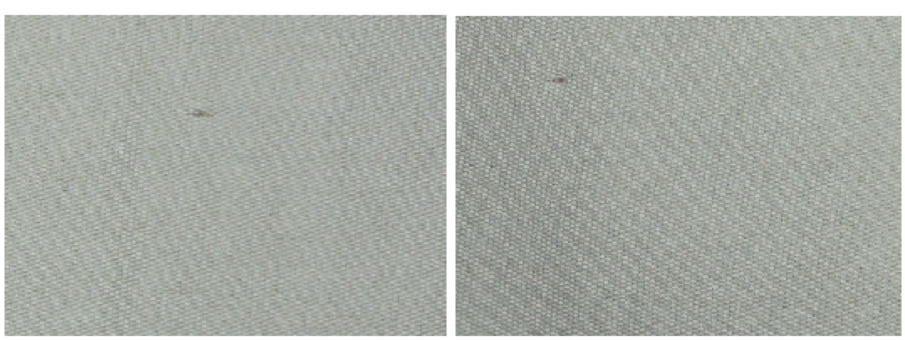


Figure 3.6a: clean images



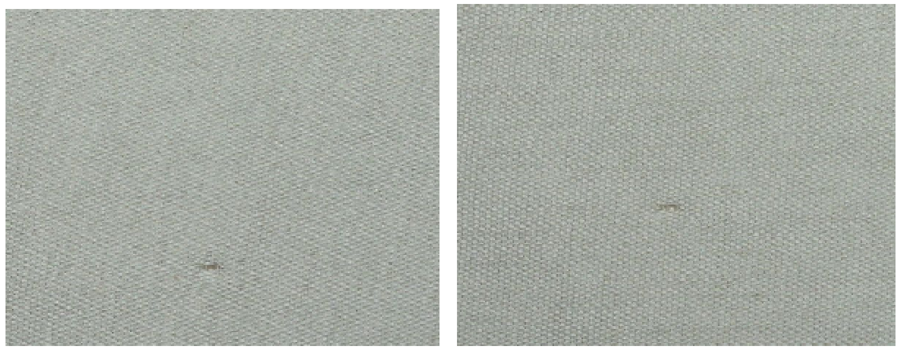


Figure 3.6b: Fabrics with Holes defect

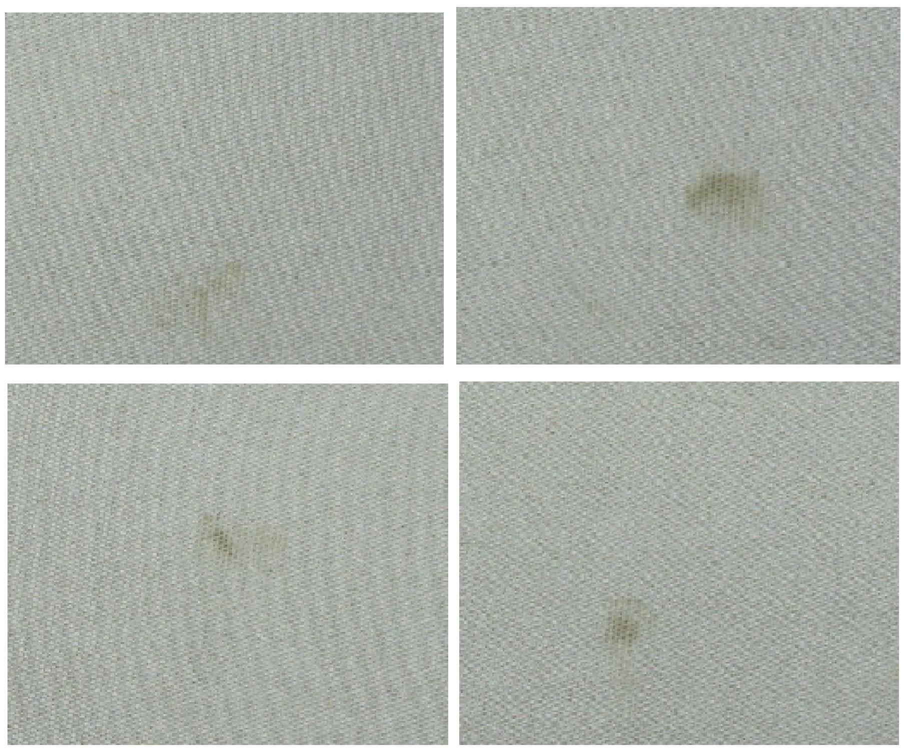


Figure 3.6c: Fabrics with oil stain defects

Figure 3.6d: Fabrics with Horizontal cuts

The experiments on fabric defects detection using the two methods are shown below;

3.5.1 Fabric defects detection and classification using SIFT based method

The same classification procedure described in (B) above is applied to the 80 x 200 SIFT features obtained using the SIFT algorithm, and the following performance measures are obtained base on the out come of the prediction:

With clean samples as the positive example class and the remaining examples as negative examples class;

TP = 5/5; FN = 0/5; TN = 45/45; FP = 0/45

Precision = TP/ (TP + FP) = 5/ (5 + 0) = 100%

Recall = TP/ (TP +FN) = 5/ (5 + 0) = 100%

F1-score = 2*Precision*Recall/ (Precision + Recall) = 2*1*1/ (1 + 1) = 100%

Accuracy = TP + TN/ (TP + FN +TN + FP) = 5 + 45/ (5 + 0 + 45 + 0) = 100%

Similarly with holes samples as the positive examples class and the rest as the negative examples class;

TP = 5/5; FN = 0/5; TN = 36/45; FP = 9/45

Precision = TP/ (TP + FP) = 5/ (5 + 9) = 35.71%

Recall = TP/ (TP +FN) = 5/ (5 + 0) = 100%

F1-score = 2*Precision*Recall/ (Precision + Recall) = 2*0.3571*1/ (0.3571 + 1) = 52.62%

Accuracy = TP + TN/ (TP + FN +TN + FP) = 5 + 9/ (5 + 0 + 36 + 9) = 28%

The same is repeated for stain samples up to when horizontal bar samples are used as the positive examples class and the remaining classes as the negative examples class. And the performance measures for the horizontal bar samples is as shown below;

TP = 4/5; FN = 1/5; TN = 28/45; FP = 17/45

Precision = TP/ (TP + FP) = 4/ (4 + 17) = 19.04%

Recall = TP/ (TP +FN) = 4/ (4 + 1) = 8%

F1-score = 2*Precision*Recall/ (Precision + Recall) = 2*0.1904*0.08/ (0.1904 + 0.08) = 11.26%

Accuracy = TP + TN/ (TP + FN +TN + FP) = 4 + 28/ (4 + 1 + 28 + 17) = 64%

3.5.2 Fabric defects detection and classification using Gabor filters

The 80 x 18 features obtained from the response of the set of Gabor filters are divided into four classes with each class carrying 20 x 18 dimensional examples. The classification procedure mentioned in (B) above is then applied to these 80 x 18 features. And the following performance measures are obtained base on the out come of the prediction:

With clean samples as the positive example class and the remaining examples as negative examples class;

TP = 5/5; FN = 0/5; TN = 12/45; FP = 33/45

Precision = TP/ (TP + FP) = 5 (5 + 33) = 13.15%

Recall = TP/ (TP +FN) = 5/ (5 + 0) = 100%

F1-score = 2*Precision*Recall/ (Precision + Recall) = 2*0.1315*1/ (0.1315 + 1) = 23.24%

Accuracy = TP + TN/ (TP + FN +TN + FP) = 5 + 12/ (5 + 0 + 12 + 33) = 34%

Similarly with holes samples as the positive examples class and the rest as the negative examples class;

TP = 4/5; FN = 1/5; TN = 8/45; FP = 37/45

Precision = TP/ (TP + FP) = 4/ (4 + 37) = 9.756%

Recall = TP/ (TP +FN) = 4/ (4 + 1) = 80%

F1-score = 2*Precision*Recall/ (Precision + Recall) = 2*0.09756*0.8/ (0.09756 + 0.8) = 17.39%

Accuracy = TP + TN/ (TP + FN +TN + FP) = 4 + 1/ (4 + 1 + 8 + 37) = 10%

The same is repeated for stain samples up to when horizontal bar samples are used as the positive examples class and the remaining classes as the negative examples class. And the performance measures for the horizontal bar samples is as shown below;

TP = 3/5; FN = 2/5; TN = 14/45; FP = 31/45

Precision = TP/ (TP + FP) = 3/ (3 + 31) = 8.823%

Recall = TP/ (TP +FN) = 3/ (3 + 2) = 60%

F1-score = 2*Precision*Recall/ (Precision + Recall) = 2*0.08823*0.6/ (0.08823 + 0.6) = 15.38%

Accuracy = TP + TN/ (TP + FN +TN + FP) = 3 + 14/ (3 + 2 + 14 + 31) = 34%

Detail explanation of the classification results for both the Gabor filters and the SIFT based method, is as shown in chapter four.

CHAPTER 4

RESULTS ANALYSIS AND COMPARISON

4.1 INTRODUCTION

After conducting experiments for defect detection in fabric materials using SIFT base with SVM and Gabor filter with SVM in chapter three, the results are discussed in this chapter. Two experiments were conducted; the first was carried out on two hundred standard regular textures consisting of ten different classes and the second was carried out on upholstery fabric with four different kinds of defects on it. The SIFT based and the Gabor filter methods are then compared based on their performance parameters.

4.2 EXPERIMENT 1

The results obtained using each method on the standard regular textures are explained below;

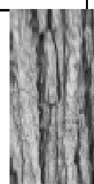
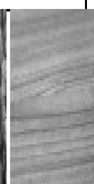
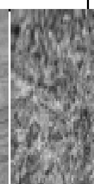
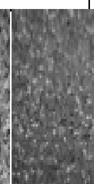
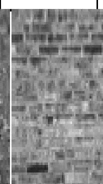
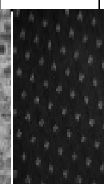
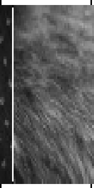
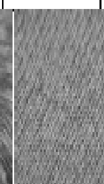
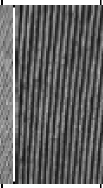
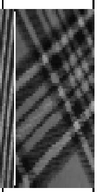
4.2.1 SIFT based method with SVM

SIFT based is a powerful algorithm used in texture analysis to carry out detection and classification of fabric defects. It has the advantage of detecting salient key points that are invariant to changes in scale, view point and illumination and compute a 128-dimensional descriptor from them which is use for the classification task.

This experiment is carried out on 200 standard regular textures, and the result shows a better performance with the SIFT algorithm. This happened because of the nature of the textures used. All textures belonging to a particular class are of the same scale but different orientation, and SIFT being a powerful algorithm that is invariant to view point made the classification performed best.

The table 4.1 below shows the classification result obtained using SIFT based method with SVM;

Table 4.1: GLOBAL SIFT VS SVM

ANALYSIS: GLOBAL SIFT FEATURES VS SVM										
Positive class others negative	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8	Class9	Class10
Image type										
Image name	Bark1	Wood1	Granite	Floor1	Brick1	Upholst ery	Fur	Knit	Corduro y	Plaid
TP	5	5	4	5	5	5	5	5	3	5
TN	126	122	134	134	133	119	127	129	135	135
FP	9	13	1	1	2	16	8	6	0	0
FN	0	0	1	0	0	0	0	0	2	0
Precision (%)	35.71	27.78	80.00	83.33	71.43	23.81	38.46	45.45	100.00	100.00
Recall (%)	100.00	100.00	80.00	100.00	100.00	100.00	100.00	100.00	60.00	100.00
F1-score (%)	52.63	43.48	80.00	90.91	83.33	38.46	55.56	62.50	75.00	100.00
Accuracy (%)	93.57	90.71	98.57	99.29	98.57	88.57	94.24	95.71	98.57	100.00

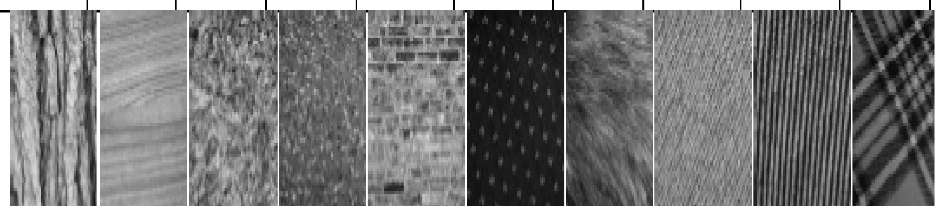
From table 4.1 shown above, it can be seen that out of five total positive examples used for testing in the classification, the classifier is able to retrieved; the whole five positives in class 1,2,4,5,6,7 and 8; four positives retrieved in class 3; three positives are retrieved in class 9. Also the numeric values of the true negatives obtained shows that most of the negative examples used for testing are retrieved. Moreover, the performance measures displayed a good values, with maximum f1 score and accuracy of 100% each.. This really shows that the proposed novel approach was able to discriminate and classify the ten classes of regular textures used. Therefore the algorithm is effective, automatic and robust.

4.2.2 Gabor filter method with SVM

The Gabor filter algorithm also performs well on the two hundred standard regular textures. Even though the textures from a given class have different orientation, but the property of a Gabor filter of having maximum joint localization both in time domain and the frequency domain made the algorithm to performs well.

The classification results for this experiment are shown in table 4.2 below;

Table 4.2: Gabor filter method with SVM

Total images used: 200; Number of classes: 10; Global features each: 200 x 100 Train_(+) = 15; Train_(-) = 45; Test_(+) = 5; Test_(-) = 135 for each classification										
Positive class others negative	Class1	Class2	Class3	Class4	Class5	Class6	Class7	Class8	Class9	Class10
Image type										
Image name	Bark1	Wood1	Granite	Floor1	Brick1	Upholst ery	Fur	Knit	Corduro y	Plaid
ANALYSIS: GLOBAL GABOR FEATURES VS SVM										
TP	3	4	2	1	3	5	4	3	3	1
TN	111	114	132	122	115	134	102	122	134	123
FP	24	21	3	13	20	1	33	13	1	12
FN	2	1	3	4	2	0	1	2	2	4
Precision (%)	11.11	16.00	40.00	7.14	13.04	83.33	10.81	18.75	75.00	7.69
Recall (%)	60.00	80.00	40.00	20.00	60.00	100.00	80.00	60.00	60.00	20.00
F1-score (%)	18.75	26.67	40.00	10.53	21.43	90.91	19.05	28.57	66.67	11.11
Accuracy (%)	81.43	84.29	95.71	87.86	84.29	99.29	75.71	89.29	97.86	88.57

From table 4.2 shown above, it can be seen that out of the five total positive examples used for testing in the classification, the classifier is able to retrieved; the whole five positives in class 6; four positives retrieved in class 2 and 7; three positives are retrieved in class 1, 5, 8 and 9; two positives in class 3 and one positive in class 4 and 10. Moreover, the f1 score and accuracy values presented are less effective compare to the results obtained with our proposed method. Even though few of the classes got accuracy of at least 95% but the maximum classification accuracy is 99.29% and at least half of the regular texture classes used are retrieved. These show that the algorithm is also robust and effective.

4.3 EXPERIMENT 2

Under this experiment, the two written algorithms are tested on upholstery fabric with three different kinds of defects on it. Both the two methods perform well and their result is discussed below;

4.3.1 SIFT based method with SVM

The tables below show the classification result obtained after applying the SIFT algorithm to the defective fabric under inspection;

Table 4.3: TP = 5/5; FN = 0/5; TN = 45/45; FP = 0/45

SIFT TEST FOR CLEAN			
	TP1(1)	FN1(0)	OUT OF
CLEAN	5	0	5
TOTAL	5	0	5
	FP1(1)	TN1(0)	OUT OF
HOLE	0	15	15
STAİN	0	15	15
HORIZONTAL BAR	0	15	15

Table 4.4: TP = 5/5; FN = 0/5; TN = 36/45; FP = 9/45

SIFT TEST FOR HOLE			
	TP2(1)	FN2(0)	OUT OF
HOLE	5	0	5
TOTAL	5	0	5
	FP2(1)	TN2(0)	OUT OF
CLEAN	0	15	15
STAİN	0	15	15
HORIZONTAL BAR	9	6	15

Table 4.5: TP = 4/5; FN = 1/5; TN = 35/45; FP = 10/45

SIFT TEST FOR STAİN			
	TP3(1)	FN3(0)	OUT OF
STAİN	4	1	5
TOTAL	4	1	5
	FP3(1)	TN3(0)	OUT OF
HOLE	6	9	15
CLEAN	0	15	15
HORIZONTAL BAR	4	11	15

Table 4.6: TP = 4/5; FN = 1/5; TN = 28/45; FP = 17/45

SIFT TEST FOR HORIZONTAL BAR			
	TP4(1)	FN4(0)	OUT OF
HORIZONTAL BAR	4	1	5
TOTAL	4	1	5
	FP4(1)	TN4(0)	OUT OF
HOLE	9	6	15
CLEAN	0	15	15
STAIN	8	7	15

Table 4.7: Performance measures for SIFT on fabric classification

Experiment with SIFT based method on fabric defects detection				
Measures	Accuracy (%)	F1 score (%)	Precision (%)	Recall (%)
Clean	100.00	100.00	100.00	100.00
Hole	82.00	52.63	35.71	100.00
Stain	78.00	42.11	28.57	80.00
Horizontal bar	64.00	30.77	19.05	80.00

From table 4.3 to table 4.7 shown above, it can be seen that out of five total positive examples used for testing in the classification, the classifier is able to retrieve; the whole five positives in table 4.3 and table 4.4; four positives were retrieved in table 4.5

and table 4.6. And at least 6 out of the 15 negative examples used per each class for testing are predicted. Moreover, perfect classification results are obtained in test for clean and least classification test for horizontal bar. Even though four positives and one miss are obtained in both test for stain and horizontal bar, we can conclude that horizontal bar is the most difficult to find because it has the least number of percentage accuracy. Also in binary classification, F1 score measures a test's accuracy and horizontal bar has the least number of f1 score.

4.3.2 Gabor filter method with SVM

The tables below shows the classification result obtained after applying the Gabor filter algorithm to the defective fabric under inspection;

Table 4.8: TP = 5/5; FN = 0/5; TN = 12/45; FP = 33/45

GABOR TEST FOR CLEAN			
	TP1(1)	FN1(0)	OUT OF
CLEAN	5	0	5
TOTAL	5	0	5
	FP1(1)	TN1(0)	OUT OF
HOLE	11	4	15
STAİN	10	5	15
HORIZONTAL BAR	12	3	15

Table 4.9: TP = 3/5; FN = 2/5; TN = 13/45; FP = 32/45

GABOR TEST FOR HOLE			
	TP2(1)	FN2(0)	OUT OF
HOLE	3	2	5
TOTAL	3	2	5
	FP2(1)	TN2(0)	OUT OF
CLEAN	9	6	15
STAİN	10	5	15
HORIZONTAL BAR	13	2	15

Table 4.10: TP = 4/5; FN = 1/5; TN = 8/45; FP = 37/45

GABOR TEST FOR STAİN			
	TP3(1)	FN3(0)	OUT OF
STAİN	4	1	5
TOTAL	4	1	5
	FP3(1)	TN3(0)	OUT OF
HOLE	12	3	15
CLEAN	13	2	15
HORIZONTAL BAR	12	3	15

Table 4.11: TP = 3/5; FN = 2/5; TN = 14/45; FP = 31/45

GABOR TEST FOR HORIZONTAL BAR			
	TP4(1)	FN4(0)	OUT OF
HORIZONTAL BAR	3	2	5
TOTAL	3	2	5
	FP4(1)	TN4(0)	OUT OF
HOLE	12	3	15
CLEAN	8	7	15
STAIN	11	4	15

Table 4.12: Performance measures for Gabor filters on fabric classification

Experiment with Gabor filter method on fabric defects detection				
Measures	Accuracy (%)	F1 score (%)	Precision (%)	Recall (%)
Clean	42.00	25.64	14.71	100.00
Hole	66.00	10.53	7.14	20.00
Stain	36.00	23.81	13.51	100.00
Horizontal bar	30.00	14.63	8.33	60.00

From table 4.8 shown above, it can be seen that out of the five positive examples used for testing in the classification, the classifier was able to retrieve the whole five positives and at least 3 out of the 15 negative examples per each class for testing were recalled. Four positives were retrieved in table 4.9, three positives were retrieved in table 4.10 and table 4.11 with at least 2 negative examples used for testing recalled. Since three positives and two misses are obtained in both test for stain and horizontal bar, and the horizontal bar has the minimum accuracy and f1 score, we can conclude that horizontal bar is the most difficult to find.

4.4 COMPARING THE SIFT BASED METHOD VERSES THE GABOR FILTER METHOD

Our proposed approach "SIFT features and Bag-of-Words" is compared with a standard method used in texture analysis "Gabor filter method" base on the following performance measures; maximum accuracy, average accuracy, maximum F1 Score and average F1 Score. And they are obtained from the results of our two experiments. F1 score is a measure of a test's accuracy in statistical analysis of binary classification. And an F1 score of 1 (100%) means a best test accuracy while an F1 score of o (0%) means poor test accuracy. Therefore we are going to compare the two methods based on the experiment carried out on regular textures classification and the experiment carried out on fabric defects detection.

Table 4.13 below shows the performance measures obtained using each method in both the two experiments.

Table 4.13: comparison between our proposed novel approach "SIFT features and Bag-of-Words" with a standard approach "Gabor filter method"

Measures experiments	SIFT-based on regular textures	Gabor filters on regular textures	SIFT based on fabric defects	Gabor filters on fabric defects
Max. Accuracy (%)	100.00	99.29	100.00	66.00
Average Accuracy (%)	86.78	88.43	58.50	43.50
Max. F1 score (%)	100.00	90.91	100.00	25.64
Average F1 score (%)	59.19	33.37	33.88	18.65

From table 4.13 shown above, the following comparisons can be obtained;

- In the experiment for the classification of regular textures, the SIFT based method has the highest percentage of maximum accuracy and maximum F1 score compared to the Gabor filter method. Also it has the highest percentage of average F1 score while the Gabor filter method has the highest average accuracy.
- In the experiment on fabric defects detection, the SIFT based method has the highest percentage on all the four performance measures.

Therefore, base on the above comparisons it can be seen that our proposed method has the best performance measure values. And since F1 score measures test's accuracy and our proposed method has got the highest percentage in both the two experiments conducted, we can conclude that the proposed novel approach "SIFT features and Bag-of-Words" is the best, robust, automatic and efficient.

CHAPTER 5

DISCUSSION AND CONCLUSION

In our research work, we proposed a novel approach “SIFT features and Bag-of-Words” to perform regular textures classification and fabric defects detection, We Compared the SIFT based method and the Gabor filters method on a standard texture classification experiment. We then generated a fabric defects image data set with 80 images and 4 classes and Compared the SIFT based method and the Gabor filters method on a fabric defects detection experiment. Based on the results obtained, it can be seen that our proposed method performs well in both of the experiments. All of the textures are classified efficiently, also all the fabric defects are efficiently discriminated with a good recall. The SIFT based provides useful features regardless of the scale, viewpoint and illumination of the region and the Gabor filter have a maximum joint localization both in time and frequency domain. Therefore, test images with the same kinds of defects on them but of different orientation can be inspected and classified accurately.

The SIFT based model has higher feature extraction speed than the Gabor filter method

In Gabor filter method, features are obtained from more general region while features from well define region are obtained in the SIFT based method.

Large numbers of features are extracted using the Gabor filter method than the SIFT based.

SIFT based method performs better on both the two experiments, most especially on the image samples belonging to the same class but different orientation (like in the first experiment) than the Gabor filter method. Our proposed method does not require any changes in the written algorithms for a new data sets, it works well on data sets with different dimension or larger number. any new set of data can be tested efficiently.

The proposed method is efficient, automatic, robust and time saving. And combining more than two statistical approaches together is recommended for more efficient classification.

REFERENCES

1. Conci and C. B. Proença a fractal image analysis system for fabric inspection based on box-counting method, Computer Networks and ISDN Systems, vol. 30, pp. 1887-1895, 1998.
2. Y. F. Zhang and R. R. Bresee fabric defect detection and classification using image analysis, text. Res. J. 65 (1995) 1-9.
3. Conci and C. B. Proença "computer vision approach to textile inspection" Text. Res. J. vol. 70, pp. 347-350, Apr. 2000.
4. J. S. Lane, "Textile fabric inspection system," US Patent No. 5,774,177, Jun. 1998.
5. R. W. Conners, C. W. McMillan, K. Lin, and R. E. Vasquez-Espinosa, "Identifying and locating surface defects in wood: Part of an automated lumber processing system," IEEE Trans. Patt. Anal. Machine Intell., no. 6, pp. 573-583, Nov. 1983.
6. I.Tsai, C. Lin, and J. Lin, "Applying an artificial neural network to pattern recognition in fabric defects," Text. Res. J., vol. 65, pp. 123-130, Mar. 1995.
7. Hung and I.-C Chen "neural-Fuzzy classification for fabric defects" Text. Res. J. Vol. 71(3), pp.220-224, (2001).
8. H. Chan and G. Pang, "Fabric defect detection by Fourier analysis," IEEE Trans. Ind. Appl., vol. 36, pp. 1267-1276, Sep/Oct. 2000.
9. S. Tsai and M. C. Hu, "Automated inspection of fabric defects using an artificial neural networks," Text. Res. J., vol. 66, pp. 474-482, Jul. 1996.
10. Kumar and G. Pang, "Defect detection in textured materials using Gabor Filters," IEEE Trans. Ind. Appl., vol. 38, no. 2, pp. 425-440, Mar. 2002.

11. J. G. Campbell, C. Fraley, F. Murtagh, and A. E. Raftery, “Linear flaw detection in woven textiles using model-based clustering,” Technical Report No. 314, Dept. of Statistics, University of Washington, Seattle, pp. 1-15, Jul. 1996.

12. K. Y. Kong, J. Kittler, M. Petrou, and I. Ng, “Chromato-Structural approach towards surface defect detection in random textured images,“ Proc. SPIE 2183, pp. 193-204, Feb. 1994.

13. Wikipedia, http://en.wikipedia.org/wiki/Gaborfilter.

14. Kumar, "Computer-Vision-Based Fabric Defect Detection: A Survey," IEEE Trans. Ind. Electron., vol. 55, no. I, pp. 348-363, Jan. 2008.

15. Ajay Kumar, Grantham K.H. Pang, “Defect detection in textured materials using Gabor filters” IEEE, vol. 38, no. 2, March/April 2002.

16. J. Escofet, R. Navarro, M.S. Millan, and J. Pladelloreans, “Detection of local defects in textiles webs using Gabor filters,” Opt. Eng., vol. 37, no. 8, pp. 2297-2307, Aug. 1998.

17. Yi Yang and Shawn Newsam, “Comparing SIFT descriptors and Gabor texture features for classification of remote sensed imagery” ICIP 2008.

18. D.F. Dunn and W.E. Higgins, “Optimal Gabor filters for texture segmentation,” IEEE Trans. Image.Processing, vol. 7, pp. 947-964, 1995.

19. K. Y. Song, M. Petrou, and J. Kittler, “Texture crack detection,” Machine Vision & Appl., vol. 8, pp. 63-76, 1995.

20. A. Kumar and G. Pang, “Fabric defect segmentation using multichannel blob detectors,” Opt. Eng., vol.39, no. 12, pp. 3176-3190, Dec. 2000.

21. A. Kumar and H. C. Shen, “Texture inspection for defects using neural networks and support vector machines,” Proc. Intl. Conf. Image Process., ICIP-2002, pp. 353-356, Rochester, New York, Sep. 2002.

22. F. S. Cohen and Z. Fan, “Rotation and scale invariant texture classification,” Proc. IEEE Conf. Robotics and Automation, vol. 3, pp. 1394-1399, Apr. 1988.

23. Forsyth and Ponce, “Computer Vision” Text book, A modern approach, 2003.

24. R. T. Chin and C. A. Harlow, “Automated visual inspection: A survey,” IEEE Trans. Patt. Anal. Machine Intell., vol. 6, pp. 557-573, Jun. 1982.

25. A. Serdaroglu, A. Ertuzun, A. Ercil, "Defect detection in textile fabric images using wavelet transforms and independent component analysis", Pattern recognition & image analysis, vol.16, no.1, pp.61-64, 2006

26. T. Mäenpää, M. Turtinen, and M. Pietikäinen, "Real-time surface inspection by texture," Real-Time Imaging, vol. 9, pp. 289-296, 2003.

27. R. M. Harlick, "Statistical and Structural approaches to texture," Proc. IEEE, vol. 67, no. 5, May 1979.

28. G. Pang, R. Wong, H. Liu, T. Kwan, and A. Kumar, "CAVIS: A low-cost fabric defect inspection machine based on Machine Vision," Proc. Asian Textile Conf., pp. 11-16, The Hong Kong Polytechnic University, Hong Kong, Aug. 2001.

29. Jain, A.K., Farrokhnia, F.: Unsupervised Texture Segmentation Using Gabor Filters. In: Proc. of the IEEE Conf. on Syst. Man and Cybern. pp. 14-19 (1990).

APPENDIX

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
Name:                Abubakar Rabiu
Student number:      50011219
Department:          Electrical and Computer Engineering
Thesis tlitle:       Automatic defect detection in fabrics using computer
vision techniques
University:          Meliksah University Turkey
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%

                              %%%%%%%%%%%
                              SOURCE CODE
                              %%%%%%%%%%%

                         %%%%%%%%%%%%%%%%%%%%%
                         COMPUTE SIFT FEATURES
                         %%%%%%%%%%%%%%%%%%%%%

NUMIMG = 80;

D = zeros(1, 128);
imx = [0];
for q=1:NUMIMG
    image = imread(['AR' num2str(q) '.pgm']);
    [imag, descriptors, locs] = SIFT(['AR' num2str(q) '.pgm']);
    D = [D; descriptors];
    imx = [imx; q*ones(size(descriptors,1),1)];
end

D = D(2:end, :);
imx = imx(2:end);
save 'SIFTtMatrix_AR80' D imx

                           %%%%%%%%%%%%%%%%%%
                           SIFT VECTORIZATiON
                           %%%%%%%%%%%%%%%%%%

NUMCLUSTERS = 200;
 [ix, centroids] = kmeans(D, NUMCLUSTERS);
numim = max(imx);                  % tell us how many images are there.
G= zeros(numim, NUMCLUSTERS);
for imno = 1:numim                 %  numim is the highest number of
images
    useix = find(imx==imno);       % this get the descriptr index of an
                                 image with num imno and save in useix
    a = ix(useix);                 % This get all clustrs that belong to
an
                                 image n save to a
    for i=1:NUMCLUSTERS
```

```
G(imno, i) = sum(a==i);        % this take all the clusters of an
image, take
                               the 1st cluster sum all similar n save
                  in G
                               %continue until the last cluster and
                        will
                                 %be numim X numclusters matrix. which
is
                                 %later converted to column vector i.e
                                 %G = (numim X numclusters,1)

    end
end


mu = mean(G);
st = std(G);
Gn = G - repmat(mu, size(G,1), 1);
Gn = Gn./repmat(st, size(G,1), 1);

                        %%%%%%%%%%%%%%%%%%%%%%%
                        SIFT TRAIN AND CLASSiFY
                        %%%%%%%%%%%%%%%%%%%%%%%

%Data for 80 images

c1 = Gn(1:20,:); c2 = Gn(21:40,:); c3 = Gn(41:60,:); c4 = Gn(61:80,:);

                            %%%%%%%%%%%%%%%
                            INITILIZATIONS
                            %%%%%%%%%%%%%%%

oneslimit = 20;                % Maximum number of positive examples to
be
                               use for training and testing.
numones_train = 15;          % Maximum number of positive examples for
                                training, from a choosing positive
                        class.
aa = 1:numones_train;        % Range of positive examples to be use for
                               training from a choosing positive class.
numones_val = 5;             % Maximum number of positive examples for
                               testing, from a choosing positive class.
hh = numones_train +1;       % Starting point of positive examples to
be
                               use for testing.

jj = 15;                     % Total number of negative examples from
all
                                negative classes, for training (i.e
                        5*3).
bb = 1:5;                    % Range of negative examples per negative
                               class, for training.
kk = 45;                    % Total number of negative examples from
all
                               negative classes, for testing (i.e
                        15*3).
maxi = max(bb)+1;            % Starting point of negative examples to
be
```

```
                             use for testing (i.e max number of negative
                            examples used
                              % for training per class + 1).
limit_range1 = kk - 25;
mk = kk - 10;
range1 = (numones_val + 1):limit_range1;
range2 = (limit_range1 + 1):mk;
range3 = (mk + 1):(kk + numones_val);



            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            TRAINING_CLASSIFICATION STAGE FOR FOUR CLASSES
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% CLASS1(C1)(CLEAN)

X1_train = [c1(aa,:); c2(bb,:); c3(bb,:); c4(bb,:)];
Y1_train = [ones(numones_train,1); zeros(jj,1)];
X1_val = [c1(hh:oneslimit,:); c2(maxi:end,:); c3(maxi:end,:);
c4(maxi:end,:)];
Y1_val = [ones(numones_val,1); zeros(kk,1)];

SVMStruct_1 = svmtrain(X1_train,Y1_train);
predict_out1 = svmclassify(SVMStruct_1,X1_val);

tp1 = sum((predict_out1 == 1) & (Y1_val == 1));
fp1 = sum((predict_out1 == 1) & (Y1_val == 0));
fn1 = sum((predict_out1 == 0) & (Y1_val == 1));
tn1=  sum((predict_out1 == 0) & (Y1_val == 0));

%Confusion Matrix 1

tp1_clean = sum((predict_out1(1:numones_val) == 1) &
(Y1_val(1:numones_val) == 1));
fn1_clean = sum((predict_out1(1:numones_val) == 0) &
(Y1_val(1:numones_val) == 1));
tn1_hole=  sum((predict_out1(range1) == 0) & (Y1_val(range1) == 0));
fp1_hole = sum((predict_out1(range1) == 1) & (Y1_val(range1) == 0));
tn1_stain=  sum((predict_out1(range2) == 0) & (Y1_val(range2) == 0));
fp1_stain = sum((predict_out1(range2) == 1) & (Y1_val(range2) == 0));
tn1_horizonbar=  sum((predict_out1(range3) == 0) & (Y1_val(range3) ==
0));
fp1_horizonbar = sum((predict_out1(range3) == 1) & (Y1_val(range3) ==
0));


prec1= tp1/(tp1 + fp1);
rec1= tp1/(tp1 + fn1);
F1_1=  (2* prec1* rec1)/(prec1 + rec1);
accuracy1 = (tp1 + tn1)/(tp1 + tn1 + fp1 + fn1);

% CLASS1(C2)(HOLE)

X2_train = [c2(aa,:); c1(bb,:); c3(bb,:); c4(bb,:)];
Y2_train = [ones(numones_train,1); zeros(jj,1)];
X2_val = [c2(hh:oneslimit,:); c1(maxi:end,:); c3(maxi:end,:);
c4(maxi:end,:)];
```

```
Y2_val = [ones(numones_val,1); zeros(kk,1)];

SVMStruct_2 = svmtrain(X2_train,Y2_train);
predict_out2 = svmclassify(SVMStruct_2,X2_val);

tp2 = sum((predict_out2 == 1) & (Y2_val == 1));
fp2 = sum((predict_out2 == 1) & (Y2_val == 0));
fn2 = sum((predict_out2 == 0) & (Y2_val == 1));
tn2=  sum((predict_out2 == 0) & (Y2_val == 0));

%Confusion Matrix 2

tp2_hole = sum((predict_out2(1:numones_val) == 1) &
(Y2_val(1:numones_val) == 1));
fn2_hole = sum((predict_out2(1:numones_val) == 0) &
(Y2_val(1:numones_val) == 1));
tn2_clean=  sum((predict_out2(range1) == 0) & (Y2_val(range1) == 0));
fp2_clean = sum((predict_out2(range1) == 1) & (Y2_val(range1) == 0));
tn2_stain=  sum((predict_out2(range2) == 0) & (Y2_val(range2) == 0));
fp2_stain = sum((predict_out2(range2) == 1) & (Y2_val(range2) == 0));
tn2_horizonbar=  sum((predict_out2(range3) == 0) & (Y2_val(range3) ==
0));
fp2_horizonbar = sum((predict_out2(range3) == 1) & (Y2_val(range3) ==
0));


prec2= tp2/(tp2 + fp2);
rec2= tp2/(tp2 + fn2);
F1_2=  (2* prec2* rec2)/(prec2 + rec2);
accuracy2 = (tp2 + tn2)/(tp2 + tn2 + fp2 + fn2);

% CLASS1(C3)(STAiN)

X3_train = [c3(aa,:); c2(bb,:); c1(bb,:); c4(bb,:)];
Y3_train = [ones(numones_train,1); zeros(jj,1)];
X3_val = [c3(hh:oneslimit,:); c2(maxi:end,:); c1(maxi:end,:);
c4(maxi:end,:)];
Y3_val = [ones(numones_val,1); zeros(kk,1)];

SVMStruct_3 = svmtrain(X3_train,Y3_train);
predict_out3 = svmclassify(SVMStruct_3,X3_val);

tp3 = sum((predict_out3 == 1) & (Y3_val == 1));
fp3 = sum((predict_out3 == 1) & (Y3_val == 0));
fn3 = sum((predict_out3 == 0) & (Y3_val == 1));
tn3=  sum((predict_out3 == 0) & (Y3_val == 0));

%Confusion Matrix 3

tp3_stain = sum((predict_out3(1:numones_val) == 1) &
(Y3_val(1:numones_val) == 1));
fn3_stain =sum((predict_out3(1:numones_val) == 0) &
(Y3_val(1:numones_val) == 1));
tn3_hole=  sum((predict_out3(range1) == 0) & (Y3_val(range1) == 0));
fp3_hole = sum((predict_out3(range1) == 1) & (Y3_val(range1) == 0));
tn3_clean=  sum((predict_out3(range2) == 0) & (Y3_val(range2) == 0));
fp3_clean = sum((predict_out3(range2) == 1) & (Y3_val(range2) == 0));
```

```
tn3_horizonbar=  sum((predict_out3(range3) == 0) & (Y3_val(range3) ==
0));
fp3_horizonbar = sum((predict_out3(range3) == 1) & (Y3_val(range3) ==
0));


prec3= tp3/(tp3 + fp3);
rec3= tp3/(tp3 + fn3);
F1_3=  (2* prec3* rec3)/(prec3 + rec3);
accuracy3 = (tp3 + tn3)/(tp3 + tn3 + fp3 + fn3);

% CLASS1(C4)(Horizontal bar)

X4_train = [c4(aa,:); c2(bb,:); c3(bb,:); c1(bb,:)];
Y4_train = [ones(numones_train,1); zeros(jj,1)];
X4_val = [c4(hh:oneslimit,:); c2(maxi:end,:); c3(maxi:end,:);
c1(maxi:end,:)];
Y4_val = [ones(numones_val,1); zeros(kk,1)];

SVMStruct_4 = svmtrain(X4_train,Y4_train);
predict_out4 = svmclassify(SVMStruct_4,X4_val);

tp4 = sum((predict_out4 == 1) & (Y4_val == 1));
fp4 = sum((predict_out4 == 1) & (Y4_val == 0));
fn4 = sum((predict_out4 == 0) & (Y4_val == 1));
tn4=  sum((predict_out4 == 0) & (Y4_val == 0));

%Confusion Matrix 4

tp4_horizonbar = sum((predict_out4(1:numones_val) == 1) &
(Y4_val(1:numones_val) == 1));
fn4_horizonbar =sum((predict_out4(1:numones_val) == 0) &
(Y4_val(1:numones_val) == 1));
tn4_hole=  sum((predict_out4(range1) == 0) & (Y4_val(range1) == 0));
fp4_hole = sum((predict_out4(range1) == 1) & (Y4_val(range1) == 0));
tn4_stain=  sum((predict_out4(range2) == 0) & (Y4_val(range2) == 0));
fp4_stain = sum((predict_out4(range2) == 1) & (Y4_val(range2) == 0));
tn4_clean=  sum((predict_out4(range3) == 0) & (Y4_val(range3) == 0));
fp4_clean = sum((predict_out4(range3) == 1) & (Y4_val(range3) == 0));


prec4 = tp4/(tp4 + fp4);
rec4 = tp4/(tp4 + fn4);
F1_4 = (2* prec4* rec4)/(prec4 + rec4);
accuracy4 = (tp4 + tn4)/(tp4 + tn4 + fp4 + fn4);


                          %%%%%%%%%%%%%%%%%%%
                          FUNCTION: mytexture
                          %%%%%%%%%%%%%%%%%%%

function feat = mytexture(img)
im = im2double(img);
x0 = 0; y0 = 0;
sigma_x = 40;    % former sigma_x value is 6. 40
sigma_y = 3*sigma_x;    %former sigma_x value is 4. 3
 A = 2*pi*sigma_x*sigma_y;
 feat = [0 0];
```

```
numfeat = 0;

for theta = 0:pi/8:pi,
    a = cos(theta)^2/2/sigma_x^2 + sin(theta)^2/2/sigma_y^2;
    b = -sin(2*theta)/4/sigma_x^2 + sin(2*theta)/4/sigma_y^2 ;
    c = sin(theta)^2/2/sigma_x^2 + cos(theta)^2/2/sigma_y^2;
    www= max(sigma_x,sigma_y);
  [X, Y] = meshgrid(-3*www:1:3*www, -3*www:1:3*www);
    Z = A*exp( - (a*(X-x0).^2 + 2*b*(X-x0).*(Y-y0) + c*(Y-y0).^2));
    outimg = imfilter(im,Z);
    outimg = outimg.^2;
    %outimg = abs(outimg);
     numfeat = numfeat + 1;
    feat(numfeat) = mean2(outimg);
    numfeat = numfeat + 1;
    feat(numfeat) = std2(outimg);
    figure, imshow(Z,[])
    %figure, imshow((outimg),[])
   % figure,imshow(img)
end
end

                              %%%%%%%%%%
                               Main code
                              %%%%%%%%%%

for q=1:80;
    image = imread(['AR' num2str(q) '.jpg']);
    img = image;

        eval(['X' num2str(q) '= mytexture(img)']);



end

X =
[X1;X2;X3;X4;X5;X6;X7;X8;X9;X10;X11;X12;X13;X14;X15;X16;X17;X18;X19;X2
0;
X21;X22;X23;X24;X25;X26;X27;X28;X29;X30;
X31;X32;X33;X34;X35;X36;X37;X38;X39;X40;
X41;X42;X43;X44;X45;X46;X47;X48;X49;X50;
X51;X52;X53;X54;X55;X56;X57;X58;X59;X60;
X61;X62;X63;X64;X65;X66;X67;X68;X69;X70;
X71;X72;X73;X74;X75;X76;X77;X78;X79;X80];

imdxx = [1*ones(size(X1)); 2*ones(size(X2));
3*ones(size(X3));4*ones(size(X4));5*ones(size(X5));6*ones(size(X6));7*
ones(size(X7));8*ones(size(X8));9*ones(size(X9));10*ones(size(X10));11
*ones(size(X11));12*ones(size(X12));13*ones(size(X13));14*ones(size(X1
4));15*ones(size(X15));16*ones(size(X16));17*ones(size(X17));18*ones(s
ize(X18));19*ones(size(X19));20*ones(size(X20));21*ones(size(X21));22*
ones(size(X22));23*ones(size(X23));24*ones(size(X24));25*ones(size(X25
));26*ones(size(X26));27*ones(size(X27));28*ones(size(X28));29*ones(si
ze(X29));30*ones(size(X30));31*ones(size(X31));32*ones(size(X32));33*o
nes(size(X33));34*ones(size(X34));35*ones(size(X35));36*ones(size(X36)
);37*ones(size(X37));38*ones(size(X38));39*ones(size(X39));40*ones(siz
e(X40));41*ones(size(X41));42*ones(size(X42));43*ones(size(X43));44*on
```

```
es(size(X44));45*ones(size(X45));46*ones(size(X46));47*ones(size(X47))
;48*ones(size(X48));49*ones(size(X49));50*ones(size(X50));51*ones(size
(X51));52*ones(size(X52));53*ones(size(X53));54*ones(size(X54));55*one
s(size(X55));56*ones(size(X56));57*ones(size(X57));58*ones(size(X58));
59*ones(size(X59));60*ones(size(X60));61*ones(size(X61));62*ones(size(
X62));63*ones(size(X63));64*ones(size(X64));65*ones(size(X65));66*ones
(size(X66));67*ones(size(X67));68*ones(size(X68));69*ones(size(X69));7
0*ones(size(X70));71*ones(size(X71));72*ones(size(X72));73*ones(size(X
73));74*ones(size(X74));75*ones(size(X75));76*ones(size(X76));77*ones(
size(X77));78*ones(size(X78));79*ones(size(X79));80*ones(size(X80))];

                        %%%%%%%%%%%%%%%%%%%%%%%%
                        GABOR TRAiN AND CLASSiFY
                        %%%%%%%%%%%%%%%%%%%%%%%%


c1 = X(1:20,:); c2 = X(21:40,:); c3 = X(41:60,:); c4 = X(61:80,:);

                             %%%%%%%%%%%%%%%
                             INITILIZATIONS
                             %%%%%%%%%%%%%%%

oneslimit = 20;             % Maximum number of positive examples to
be
                              use for training and testing.
numones_train = 15;         % Maximum number of positive examples for
                              training, from a choosing positive
                        class.
aa = 1:numones_train;       % Range of positive examples to be use for
                              training from a choosing positive class.
numones_val = 5;            % Maximum number of positive examples for
                              testing, from a choosing positive class.
hh = numones_train +1;      % Starting point of positive examples to
be
                              use for testing.

jj = 15;                    % Total number of negative examples from
all
                              negative classes, for training (i.e
                        5*3).
bb = 1:5;                   % Range of negative examples per negative
                              class, for training.
kk = 45;                   % Total number of negative examples from
all
                              negative classes, for testing (i.e
                        15*3).
maxi = max(bb)+1;           % Starting point of negative examples to
be
                              use for testing (i.e max number of
                        negative
                               examples used
                            % for training per class + 1).
limit_range1 = kk - 25;
mk = kk - 10;
range1 = (numones_val + 1):limit_range1;
range2 = (limit_range1 + 1):mk;
range3 = (mk + 1):(kk + numones_val);
```

```
TRAINING_CLASSIFICATION STAGE FOR FOUR CLASSES

% CLASS1(C1)(CLEAN)

X1_train = [c1(aa,:); c2(bb,:); c3(bb,:); c4(bb,:)];
Y1_train = [ones(numones_train,1); zeros(jj,1)];
X1_val = [c1(hh:oneslimit,:); c2(maxi:end,:); c3(maxi:end,:);
c4(maxi:end,:)];
Y1_val = [ones(numones_val,1); zeros(kk,1)];

SVMStruct_1 = svmtrain(X1_train,Y1_train);
predict_out1 = svmclassify(SVMStruct_1,X1_val);

tp1 = sum((predict_out1 == 1) & (Y1_val == 1));
fp1 = sum((predict_out1 == 1) & (Y1_val == 0));
fn1 = sum((predict_out1 == 0) & (Y1_val == 1));
tn1=  sum((predict_out1 == 0) & (Y1_val == 0));

%Confusion Matrix 1

tp1_clean = sum((predict_out1(1:numones_val) == 1) &
(Y1_val(1:numones_val) == 1));
fn1_clean = sum((predict_out1(1:numones_val) == 0) &
(Y1_val(1:numones_val) == 1));
tn1_hole=  sum((predict_out1(range1) == 0) & (Y1_val(range1) == 0));
fp1_hole = sum((predict_out1(range1) == 1) & (Y1_val(range1) == 0));
tn1_stain=  sum((predict_out1(range2) == 0) & (Y1_val(range2) == 0));
fp1_stain = sum((predict_out1(range2) == 1) & (Y1_val(range2) == 0));
tn1_horizonbar=  sum((predict_out1(range3) == 0) & (Y1_val(range3) ==
0));
fp1_horizonbar = sum((predict_out1(range3) == 1) & (Y1_val(range3) ==
0));


prec1= tp1/(tp1 + fp1);
rec1= tp1/(tp1 + fn1);
F1_1=  (2* prec1* rec1)/(prec1 + rec1);
accuracy1 = (tp1 + tn1)/(tp1 + tn1 + fp1 + fn1);

% CLASS1(C2)(HOLE)

X2_train = [c2(aa,:); c1(bb,:); c3(bb,:); c4(bb,:)];
Y2_train = [ones(numones_train,1); zeros(jj,1)];
X2_val = [c2(hh:oneslimit,:); c1(maxi:end,:); c3(maxi:end,:);
c4(maxi:end,:)];
Y2_val = [ones(numones_val,1); zeros(kk,1)];

SVMStruct_2 = svmtrain(X2_train,Y2_train);
predict_out2 = svmclassify(SVMStruct_2,X2_val);

tp2 = sum((predict_out2 == 1) & (Y2_val == 1));
fp2 = sum((predict_out2 == 1) & (Y2_val == 0));
fn2 = sum((predict_out2 == 0) & (Y2_val == 1));
tn2=  sum((predict_out2 == 0) & (Y2_val == 0));

%Confusion Matrix 2
```

```
tp2_hole = sum((predict_out2(1:numones_val) == 1) &
(Y2_val(1:numones_val) == 1));
fn2_hole = sum((predict_out2(1:numones_val) == 0) &
(Y2_val(1:numones_val) == 1));
tn2_clean=  sum((predict_out2(range1) == 0) & (Y2_val(range1) == 0));
fp2_clean = sum((predict_out2(range1) == 1) & (Y2_val(range1) == 0));
tn2_stain=  sum((predict_out2(range2) == 0) & (Y2_val(range2) == 0));
fp2_stain = sum((predict_out2(range2) == 1) & (Y2_val(range2) == 0));
tn2_horizonbar=  sum((predict_out2(range3) == 0) & (Y2_val(range3) ==
0));
fp2_horizonbar = sum((predict_out2(range3) == 1) & (Y2_val(range3) ==
0));


prec2= tp2/(tp2 + fp2);
rec2= tp2/(tp2 + fn2);
F1_2=  (2* prec2* rec2)/(prec2 + rec2);
accuracy2 = (tp2 + tn2)/(tp2 + tn2 + fp2 + fn2);

% CLASS1(C3)(STAiN)

X3_train = [c3(aa,:); c2(bb,:); c1(bb,:); c4(bb,:)];
Y3_train = [ones(numones_train,1); zeros(jj,1)];
X3_val = [c3(hh:oneslimit,:); c2(maxi:end,:); c1(maxi:end,:);
c4(maxi:end,:)];
Y3_val = [ones(numones_val,1); zeros(kk,1)];

SVMStruct_3 = svmtrain(X3_train,Y3_train);
predict_out3 = svmclassify(SVMStruct_3,X3_val);

tp3 = sum((predict_out3 == 1) & (Y3_val == 1));
fp3 = sum((predict_out3 == 1) & (Y3_val == 0));
fn3 = sum((predict_out3 == 0) & (Y3_val == 1));
tn3=  sum((predict_out3 == 0) & (Y3_val == 0));

%Confusion Matrix 3

tp3_stain = sum((predict_out3(1:numones_val) == 1) &
(Y3_val(1:numones_val) == 1));
fn3_stain =sum((predict_out3(1:numones_val) == 0) &
(Y3_val(1:numones_val) == 1));
tn3_hole=  sum((predict_out3(range1) == 0) & (Y3_val(range1) == 0));
fp3_hole = sum((predict_out3(range1) == 1) & (Y3_val(range1) == 0));
tn3_clean=  sum((predict_out3(range2) == 0) & (Y3_val(range2) == 0));
fp3_clean = sum((predict_out3(range2) == 1) & (Y3_val(range2) == 0));
tn3_horizonbar=  sum((predict_out3(range3) == 0) & (Y3_val(range3) ==
0));
fp3_horizonbar = sum((predict_out3(range3) == 1) & (Y3_val(range3) ==
0));


prec3= tp3/(tp3 + fp3);
rec3= tp3/(tp3 + fn3);
F1_3=  (2* prec3* rec3)/(prec3 + rec3);
accuracy3 = (tp3 + tn3)/(tp3 + tn3 + fp3 + fn3);

% CLASS1(C4)(Horizontal bar)
```

```
X4_train = [c4(aa,:); c2(bb,:); c3(bb,:); c1(bb,:)];
Y4_train = [ones(numones_train,1); zeros(jj,1)];
X4_val = [c4(hh:oneslimit,:); c2(maxi:end,:); c3(maxi:end,:);
c1(maxi:end,:)];
Y4_val = [ones(numones_val,1); zeros(kk,1)];

SVMStruct_4 = svmtrain(X4_train,Y4_train);
predict_out4 = svmclassify(SVMStruct_4,X4_val);

tp4 = sum((predict_out4 == 1) & (Y4_val == 1));
fp4 = sum((predict_out4 == 1) & (Y4_val == 0));
fn4 = sum((predict_out4 == 0) & (Y4_val == 1));
tn4=  sum((predict_out4 == 0) & (Y4_val == 0));

%Confusion Matrix 4

tp4_horizonbar = sum((predict_out4(1:numones_val) == 1) &
(Y4_val(1:numones_val) == 1));
fn4_horizonbar =sum((predict_out4(1:numones_val) == 0) &
(Y4_val(1:numones_val) == 1));
tn4_hole=  sum((predict_out4(range1) == 0) & (Y4_val(range1) == 0));
fp4_hole = sum((predict_out4(range1) == 1) & (Y4_val(range1) == 0));
tn4_stain=  sum((predict_out4(range2) == 0) & (Y4_val(range2) == 0));
fp4_stain = sum((predict_out4(range2) == 1) & (Y4_val(range2) == 0));
tn4_clean=  sum((predict_out4(range3) == 0) & (Y4_val(range3) == 0));
fp4_clean = sum((predict_out4(range3) == 1) & (Y4_val(range3) == 0));

prec4 = tp4/(tp4 + fp4);
rec4 = tp4/(tp4 + fn4);
F1_4 = (2* prec4* rec4)/(prec4 + rec4);
accuracy4 = (tp4 + tn4)/(tp4 + tn4 + fp4 + fn4);
```