PREFACE

First of all, I am grateful to my thesis supervisor Prof. Dr. Muhittin Gökmen for privileging me to work with him. His guidance and support throughout this study made things easier for me. I would also like to thank him for leading me into this interesting and important subject.

I would like to express my gratitude to Binnur Kurt to benefit from his wisdom on the subject.

Finally, I want to thank my family for their continuous support and encouragement. This could not be true without them.

May, 2006 **Özcan Çavuş**

CONTENTS

ABBREVATIONS

MRTD	: Machine Readable Travel Documents
SVM	: Support Vector Machine
RKHS	: Reproducing Kernel Hilbert Space
RBF	: Radial Basis Function
PCA	: Principal Component Analysis
FLDA	: Fisher Linear Discriminant Analysis
KFDA	: Kernel Fisher Discriminant Analysis

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

$\tilde{k}(x,y)$	: Cosine kernel
L_p	: Primal lagrange
x^t	: Class element
α^t	: Lagrange multiplier
ξ^t	: Slack variable
μ^t	: Lagrange parameter
C	: Penalty factor
$Ø$	: Nonlinear mapping
$k(x,y)$	: Kernel function
S_B	: Between class scatter matrix
S_W	: Within class scatter matrix
m_i	: Mean matrix of class i
m	: Mean matrix of all classes
λ	: Eigenvalue
N_i	: Number of elements in class i
Ω	: Covariance Matrix
c	: Number of classes
Λ	: Diagonal matrix of eigenvalues
$\tilde{x}_i$	: Projected sample x
F	: Feature space
$S_B^{ø}$	: Between class scatter matrix in feature space F
$S_W^{ø}$	: Within class scatter matrix in feature space F
$m_i^{ø}$	: Mean matrix of class i in feature space F
K_B	: New between class scatter in feature space F
K_W	: New within class scatter in feature space F
N	: Number of pixels of an image
M	: Number of training instances

FACE RECOGNITION WITH KERNEL FISHER DISCRIMINANT ANALYSIS

SUMMARY

Security has become very important for people in today's world. The widely used security mechanisms like passwords and pins may not be enough secure especially considering the high cracking and stealing probability. This situation makes researchers to research more reliable and user-friendly security mechanisms. A kind of biometric technique called face recognition has started to be used for security purposes in the last years. Since it is natural, non-intrusive and easy-to-use it stands out among other security techniques. In this thesis, a face recognition technique named Kernel Fisher Discriminant Analysis (KFDA) is considered. Nowadays, kernel methods have become more popular for machine learning tasks such as classification and regression. Kernel methods are a new class of pattern analysis algorithms which can operate on very general types of data and can detect very general types of relations. With kernel methods, Fisher's discriminant which gives impressive results under different lighting conditions and different face expressions becomes more appropriate for face recognition. Instead of only using linear discriminant information, kernels give us the chance to look for non-linear directions. In this study, two methods are implemented. As a linear algorithm, Fisherface classification and as a non-linear algorithm KFDA are implemented. The performance and the comparison of these two algorithms in different databases can be seen in this thesis.

Keywords: Face Recognition, Fisher Linear Discriminant Analysis, Fisherface, Kernel Fisher Discriminant Analysis.

KERNEL FISHER AYRIŞTIRMA ANALİZİ YÖNTEMİ İLE YÜZ TANIMA

ÖZET

Günümüz dünyasında güvenlik insanlar için gün geçtikçe daha çok önem kazanmaktadır. Hali hazırda yaygın bir şekilde kullanılan şifre ve pin gibi güvenlik mekanizmaları, gerek insanlar tarafından unutulma gerekse başkaları tarafından ele geçirilme ihtimalleri olduğundan güvenlik zaafı yaratmaktadırlar. Bu durum araştırmacıları daha güvenli, doğal ve kullanışlı mekanizmalar araştırmaya yöneltmiştir. Gerek kişiye özgü olması gerekse uygulama esnasında insanlarla birebir etkileşim zorunluluğu olmaması, bir biometrik tekniği olan yüz tanımayı yeni bir güvenlik mekanizması olarak diğer yöntemlere göre öne çıkarmaktadır. Bu tezde, bir yüz tanıma tekniği olan Kernel Fisher Ayrıştırma Analizi (KFDA) incelenmiştir. Günümüzde, kernel metodları sınıflandırma ve regresyon gibi makine öğrenme işlerinde giderek popüler hale gelmektedir. Kernel metodları genel veri tipleri üzerinde çalışabilen ve bunlar arasındaki genel ilişkileri açığa çıkaran yeni tip örüntü analizi algoritmalarıdır. Kernel metodları ile beraber farklı aydınlatma koşullarında ve değişik yüz ifadelerinde oldukça etkileyici sonuçlar veren Fisher ayrıştırması, yüz tanıma için daha uygun hale gelmiştir. Sadece lineer ayrıştırma bilgisi kullanmak yerine kernel metodları bize lineer olmayan yönleri de dikkate alma olanağı sağlamıştır. Bu çalışmada iki tip yüz tanıma metodu gerçeklenmiştir. Lineer sınıflandırma algoritması olarak Fisher Yüzleri ve lineer olmayan sınıflandırma algoritması olarak Kernel Fisher Ayrıştırma Analizi algoritması gerçeklenmiştir. İki yöntemin farklı veritabanlarındaki performans değerleri ve karşılaştırma sonuçları da tez içinde verilmiştir.

Anahtar Kelimeler: Yüz Tanıma, Fisher Lineer Ayrıştırma Analizi, Fisher Yüzleri, Kernel Fisher Ayrıştırma Analizi

1. INTRODUCTION

Face recognition is a common process for human being, which is performed at many times in a day. The researches in this area were started in 1960's with a semi-automated system. In this study, researchers marked the photographs to locate the major features such as eyes, noses, mouths, and ears. After that, the distances and ratios between these major points are calculated and compared with the reference data, so the recognition performance is evaluated. Starting from 1960's, researches have developed many face recognition algorithms. However, since the success rate of the algorithms are not at the desired level, researchers continue to work on this area rigorously.

Nowadays, passwords and pins are used in many areas for security purposes. However, stealing pins or cracking passwords of computer users cause many serious problems for people in daily life. In order to prevent this kind of actions, researchers make many studies to find reliable and user-friendly mechanism which is used for identification and verification of a person.

Biometrics is an alternative to such kind of traditional approaches for the automatic identification of a person. The oldest biometric technique which is used nowadays is fingerprint recognition. This technique was first used in China in 14^{th} century for parents to distinguish their children from those of others. In 1890's, fingerprint recognition started to be used in Europe for identifying the convicted criminals. Nowadays, the currently biometric technique called iris recognition is used in many areas such as for the identification of frequent air travelers. Today in some airports, the identification is made by iris recognition instead of using passports. On the other hand, these techniques are inconvenient due to the necessity of interaction with the individual who is to be identified or authenticated. But, the biometrics technique which is called face recognition can be applied without a necessity of interaction with the individual. This is one of the most important reasons why researchers have an increased interest to face recognition in the last decade.

Face recognition has many advantages to the other biometric techniques. First of all, it is non-intrusive, natural, and easy to use. In a study which considers the compatibility of six biometric techniques (face, finger, eye, signature, hand, and voice) with machine readable travel documents (MRTD) [1], facial features scored the highest percentage of compatibility, see Figure 1.1. In this study enrolment, renewal, machine requirements, and public perception were taken as the parameters. However, because of the unreliability of facial features, face recognition should not be considered as the most reliable biometric technique.

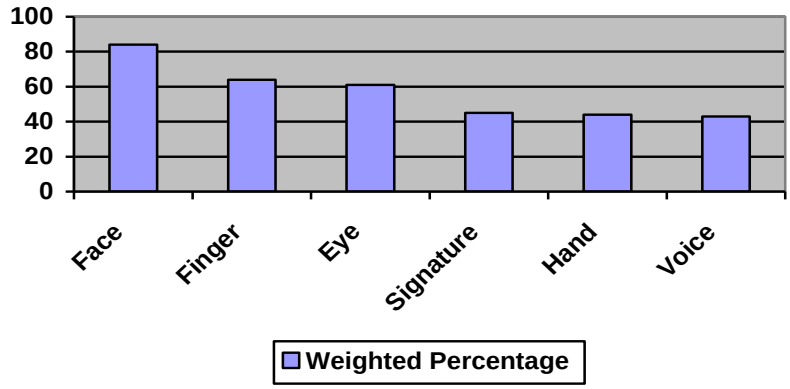


Figure 1.1: Comparison of machine readable travel documents (MRTD) compatibility with six biometric techniques: face, finger, eye, signature, hand, voice.

After the September 11th 2001, the face recognition systems have become important not only for researchers but also for the people who are interested in security.

By the way, face recognition can be used in many areas other than security oriented applications like customized human-computer interaction and computer entertainment. Customized human-computer interaction applications will be in use in cars, buildings, and etc. in the near future. The interest in face recognition and the amount of applications will probably increase even more in the future.

1.1 Thesis Overview

In this thesis, Kernel Fisher Discriminant Analysis (KFDA) algorithm is compared with Fisher Linear Discriminant Analysis (FLDA) algorithm. In part 2, we give a brief description about the kernel methods. Support vector machines (SVMs), soft margins and errors, kernel functions are described in this part. In part 3, Principle Component Analysis (PCA), discriminant analysis methods like FLDA and KFDA

are explained. In part 4, we share the experimental results of our study. In the conclusion, we conclude the work that is done in this thesis and present a discussion of possible ideas to future work.

2. KERNEL METHODS

2.1 Support Vector Machines

Support vector machines (SVMs) can be used in many areas like face recognition, text categorization, and data mining. SVMs use the kernel trick to apply linear classification techniques to non-linear classification problems. In the beginning, we will first look at the application of SVMs to the simplest case of binary classification.

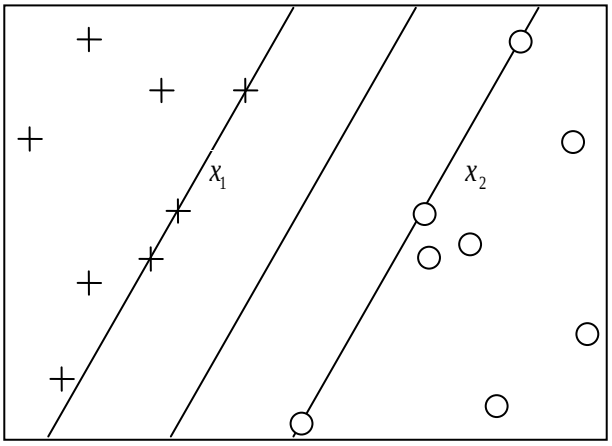


Figure 2.1: The perpendicular distance between the separating hyperplane and a hyperplane through the closest points (the support vectors) is called the margin. x_1 and x_2 are examples of support vectors of opposite sign.

Let us start from the two classes and use labels -1 / +1 for the two classes. The sample $X = \{x^t, r^t\}$ where $r^t = +1$ if $x^t \in C_1$ and $r^t = -1$ if $x^t \in C_2$. Here, we would like to find *w* and w_0 such that

$$w^T.x^T + w_0 \geq +1 \quad \text{for} \quad r^t = +1$$

$$w^T.x^T + w_0 \leq -1 \quad \text{for} \quad r^t = -1$$

which can be rewritten as

$$r^t(w^T.x^t + w_0) \geq +1 \qquad \textbf{(2.1)}$$

Here, the important point is that we do not simply require

$$r^t (w^T .x^t + w_0) \geq 0$$

, which means we do not only want the instances to be on the right-hand side of the hyperplane but also for better generalization we want them some distance away. The distance from the hyperplane to the instances closest to it on either side is called *margin* (Figure 2.1), which we want to maximize for the best generalization.

The distance of x^t to the discriminant can be defined as

$$\frac{\left|w^T x^t + w_0\right|}{\|w\|}$$

and when $r^t \in \{+1,-1\}$, it can be rewritten as

$$\frac{r^t (w^T x^t + w_0)}{\|w\|}$$

for better separation this equation must be greater than some value ρ.

$$\frac{r^t (w^T x^t + w_0)}{\|w\|} \geq \rho, \forall t \qquad \textbf{(2.2)}$$

Here the aim is to maximize ρ. But there is infinite number of solutions that can be obtained by scaling *w*, thus $\rho\|w\|$ is fixed to 1 for a unique solution. Therefore to maximize the margin, $\|w\|$ must be minimized [3]. From all of these explanations, now the task can be written as

$$\min \quad g(w) = \frac{1}{2}\|w\|^2 \qquad \textbf{(2.3)}$$

subject to

$$r^t (w^T x^t + w_0) \geq +1, \forall t \qquad \textbf{(2.4)}$$

and the learning task can be reduced to minimization of the primal lagrange

$$L_p = \frac{1}{2}\|w\|^2 - \sum_{t=1}^{M} \alpha^t \left[r^t \left(w^T x^t + w_0\right) - 1\right] \qquad \textbf{(2.5)}$$

where α^t are defined as Lagrange multipliers and $\alpha^t \geq 0$.

From the Wolfe's theorem [4], we can take derivatives with respect to w_0 and w, and then substitute back in the primal to give Wolfe dual lagrange

$$\frac{\partial L_p}{\partial w} = 0 \Rightarrow \quad w = \sum_t \alpha^t r^t x^t \tag{2.6}$$

$$\frac{\partial L_p}{\partial w_0} = 0 \Rightarrow \quad \sum_t \alpha^t r^t = 0 \tag{2.7}$$

$$W(\alpha) = \sum_{t=1}^{M} \alpha^t - \frac{1}{2} \sum_{t=1}^{M} \sum_{s=1}^{M} \alpha^t \alpha^s r^t r^s (x^t . x^s) \tag{2.8}$$

which is maximized with respect to α^t only, subject to the constraints

$$\sum_t \alpha^t r^t = 0, \quad \text{and} \quad \alpha^t \geq 0, \forall \text{t} \ . \tag{2.9}$$

This can be solved using quadratic optimization methods. The size of the dual depends on *M*, sample size. The upper bound for time complexity is $\mathrm{O}(M^3)$, and the upper bound for space complexity is $\mathrm{O}(M^2)$.

When the problem (2.8-2.9) is solved for α^t, it is seen that only a small percentage of $\alpha^t > 0$. The set of x^t whose $\alpha^t > 0$ are called support vectors. *w* can be written as the weighted sum of the training instances that are selected as support vectors (2.6). These are the x^t, which satisfy

$$r^t (w^T x^t + w_0) = 1 \tag{2.10}$$

and lie on the margin. And w_0 is calculated as

$$w_0 = r^t - w^T x^t \tag{2.11}$$

For numerical stability, it is advised that this is done for all support vectors and an average is taken. This discriminant found by this way is named as *support vector machine (SVM).*

2.2 Soft Margins and Errors

The data points we have are not always linearly separable. In this case, the algorithm mentioned above does not work. In this situation, what we have to do is to look for a hyperplane which separates the classes with a minimum error.

The slack variables $\xi^t \geq 0$ store the deviation from the margin. We can categorize the deviation types in two groups: 1- An instance can be on the wrong side of the hyperplane so it is misclassified. 2- An instance is on the right side but lie in the margin which is not remote enough from the hyperplane. With the slack variable, the equation (2.4) becomes

$$r^t(w^T.x^t + w_0) \geq 1 - \xi^t \qquad \textbf{(2.12)}$$

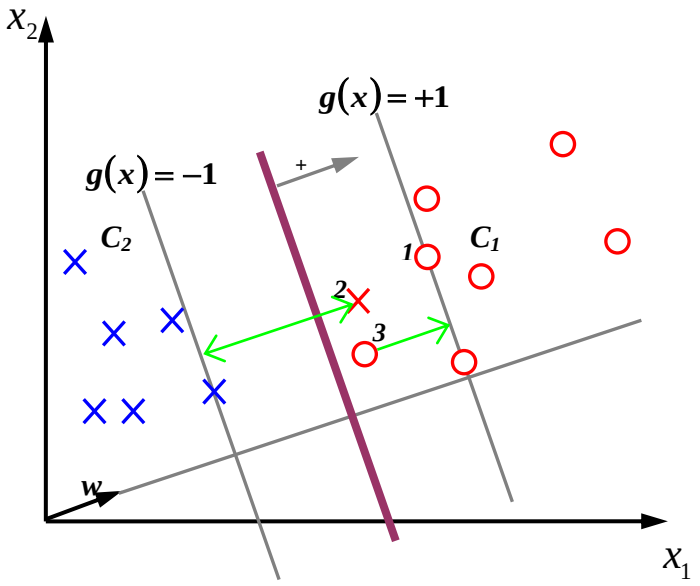


Figure 2.2: In classifying an instance, there are three possible cases: In (1), $\xi = 0$; it is on the right side and sufficiently away. In (2), $\xi = 1 + g(x) > 1;$ it is on the wrong side. In (3), $\xi = 1 - g(x), 0 < \xi < 1;$ it is on the right side but is in the margin not sufficiently away.

If $\xi^t = 0$ then there is no problem. If $0 < \xi^t < 1, x^t$ is correctly classified but is in the margin. If $\xi^t > 1, x^t$ is misclassified (Figure 2.2). The *soft error* can be defined as $\sum_t \xi^t$ and primal objective function can be written as

$$L_p = \frac{1}{2}\|w\|^2 + C\sum_t \xi^t - \sum_{t=1}^{N} \alpha^t \left[r^t\left(w^T x^t + w_0\right) - 1 + \xi^t\right] - \sum_t \mu^t \xi^t \qquad \textbf{(2.13)}$$

where μ^t are the Lagrange parameters which guarantee the positivity of ξ^t and C is the penalty factor.

The dual problem corresponding to primal (2.13) becomes:

$$W(\alpha) = \sum_{t=1}^{M} \alpha^t - \frac{1}{2} \sum_{t=1}^{M} \sum_{s=1}^{M} \alpha^t \alpha^s r^t r^s (x^t . x^s) \qquad \textbf{(2.14)}$$

subject to

$$\sum_t \alpha^t r^t = 0, \text{ and } 0 \le \alpha^t \le \mathrm{C}, \forall \mathrm{t} \ .$$

2.3 Kernel Functions

In the last section, we talk about the support vector machines by which we can control the complexity of our learning machine. By the way, for real world data linear functions are very simple. For example, by using linear functions you can not even solve the easy XOR problem (Figure 2.3).

On the other hand, we have to avoid using too complex functions. The way which uses the linear models with controllable complexity and a set of nonlinear decision functions will be an appropriate solution for real world data. Kernel functions which will be discussed in this section are this type of functions.

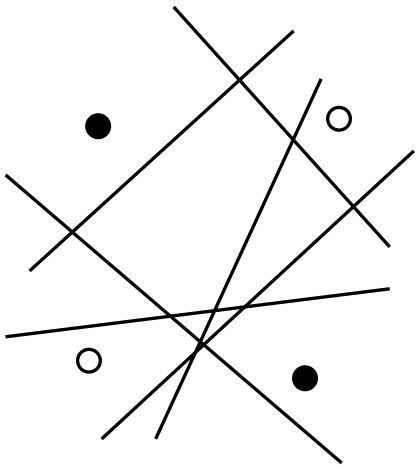


Figure 2.3: The XOR-problem: a classical example of a simple problem that can not be solved using linear functions.

Kernel Methods are a new class of pattern analysis algorithms which can operate on very general types of data and can detect very general types of relations. The basic

idea of the kernel methods is instead of applying a linear algorithm directly to the input space, firstly use a non-linear mapping ϕ on the input space and then apply a linear algorithm on this new space. If the mapping function is chosen suitably then the complex relations can be simplified and easily detected (Figure 2.4).

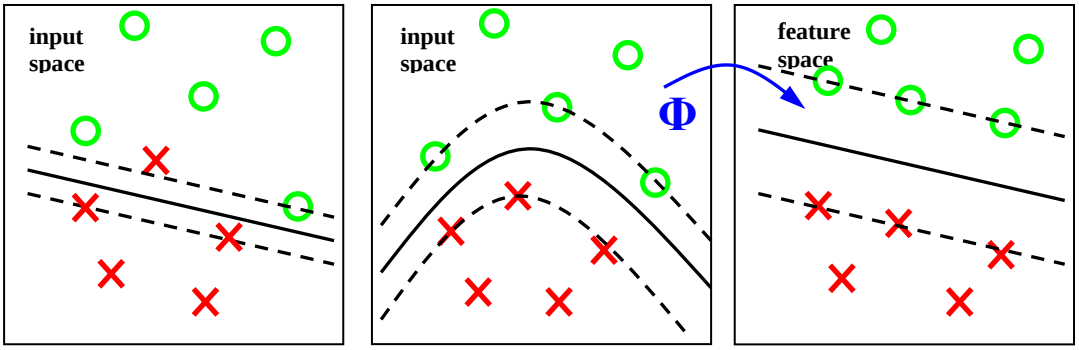


Figure 2.4: Three different views of the same data points. (a) In this example, we can not make a good separation of the input space. (b) A better separation can be made by nonlinear functions in the input space. (c) After making a mapping to a feature space using the mapping function ϕ, a better separation is obtained.

To explain what we do in figure 2.4 more formally, we apply the mapping ϕ,

$$z = \phi(x) \quad \text{where} \quad z_j = \phi_j(x),\ j = 1,.....,k$$

from the d-dimensional *x* space to the k-dimensional *z* space.

Designing an appropriate mapping function ϕ is very important. If our mapping function is not too complex to compute then, the dimension of the feature space will not be very high so only applying the mapping can be enough to classify our input data.

In Figure 2.5, there is a mapping example where the feature map $\phi(x_1, x_2) = (x_1^2, \sqrt{2}x_1x_2, x_2^2)$ maps data from R^2 to R^3.

By selecting $x_1^2, \sqrt{2}x_1x_2$ and x_2^2 as features, the new feature space can be constructed and for separation, all we need is a linear hyperplane. But, what about the case if we need to map an input space which has 70 x 70 pixels as patterns to a feature space. In this example, if we choose as nonlinearity all 5^{th} order monomials and since our dimension vectors have 4900 rows the dimension of the feature space can be computed as below.

$$\binom{5+4900-1}{5} \approx 2.3x10^{17}$$

To make such a mapping is almost impossible.

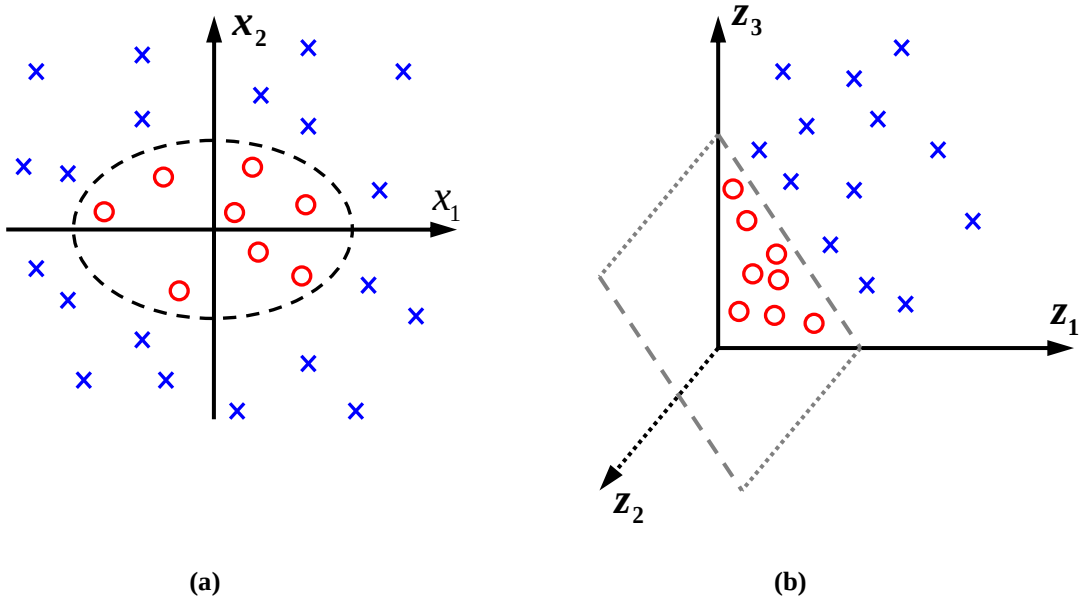


Figure 2.5: Two dimensional classification example [6].

The problem can be solved by calculating the dot products $(\phi(x).\phi(y))$. Here, we do not need patterns $\phi(x)$ to be mapped in explicit form. Now, the solution becomes simple because the dot products of the space vectors can be evaluated by a kernel

$$k(x, y) = (\phi(x).\phi(y)). \quad \textbf{(2.15)}$$

Taking the polynomial kernel as an example, the kernel function is as follows

$$k(x, y) = (x.y)^d \quad \textbf{(2.16)}$$

For d=2 the solution becomes

$$(x.y)^2 = (x_1^2, x_2^2, \sqrt{2}x_1x_2)(y_1^2, y_2^2\sqrt{2}y_1y_2)^T = (\phi(x).\phi(y)) \quad \textbf{(2.17)}$$

defining

$$\phi(x) = (x_1^2, x_2^2, \sqrt{2}x_1x_2).$$

Here, the *kernel trick* gives us the ability to apply the original algorithm as the scalar products of $\phi(x)$. If we achieve to calculate the scalar products, there is no need to

make the mapping ϕ explicitly and we can still solve the problem in huge feature space *F*. Also, the important thing is that, we do not need to know the mapping ϕ. The only thing we should need to know is the kernel function.

2.3.1 The Feature Space and Feature Map

Given any feature map ϕ from X into a vector space (possibly the higher space) called the feature space where the process can be shown as $\phi: X -> R^N$. Here X is our input vector space.

Because we use kernels for such a kind of mapping, our mapping function ϕ can be defined as

$$\phi(x) = k(x,.). \quad \textbf{(2.18)}$$

We define the vector space by taking the linear combinations of functions

$$f(.) = \sum_{i=1}^{l} \alpha_i k(x_i,.) \quad \textbf{(2.19)}$$

for arbitrary $l \in N, \alpha_i \in R$ and $x_1,.....,x_l \in X$. For all functions of the vector space (2.19) one gets

$$\langle k(x,.), f \rangle_H = f(x) \quad \textbf{(2.20)}$$

where $\langle .,. \rangle_H$ denotes the inner product in some Hilbert space that will become clearer below. In particular, we have

$$\begin{aligned} \langle k(.,x), k(.,z) \rangle_H &= \langle \phi(x), \phi(z) \rangle_\varepsilon \\ &= k(x,z). \end{aligned} \quad \textbf{(2.21)}$$

2.3.1.1 Hilbert Space and Reproducing Kernel Hilbert Spaces

A Hilbert space is a complex vector space H with an inner product $\langle .,. \rangle$ which satisfies the following requirements.

1. $\langle \alpha f + \beta g, h \rangle = \langle \alpha f, h \rangle + \langle \beta g, h \rangle$ for all real α, β all f, g, h in H
2. $\langle f, g \rangle = \langle g, f \rangle$ for all f, g in H

3. $\langle f, f \rangle \geq 0$ with equality if and only if $f = 0$

The inner product between f and $g = \sum_{j=1}^{l^1} \beta_j k(x_j^t, .)$ is defined as

$$\left\langle f, g = \sum_{i=1}^{l} \sum_{j=1}^{l'} \alpha_i \beta_j k(x_i, x_j^t) \right\rangle \quad \textbf{(2.22)}$$

With the requirements and definition (2.22), we have

$$\|f\|^2 = \langle f, f \rangle = \sum_{i,j=1}^{l} \alpha_i \alpha_j k(x_i, x_j) \geq 0 \quad \textbf{(2.23)}$$

In definition (2.20) *k* is a reproducing kernel. With the inner product on the vector space, we have obtained a pre-Hilbert space. We complete the space by adding the limit points of convergent sequences to form a Hilbert space is called *reproducing kernel Hilbert space (RKHS).*

2.3.1.2 Mercer Kernels

Mercer Kernel functions can be viewed as a measure of the similarity between two data points that are embedded in a high, possibly infinite dimensional feature space. For a finite sample of data *X*, the kernel function yields a symmetric *N x N* positive definite matrix, where the (*i; j*) entry corresponds to the similarity between (x_i; x_j) as measured by the kernel function. Because of the positive definite property, such a Mercer Kernel can be written as the outer product of the data in the feature space. Thus, if $\phi(x_i) : R^d \rightarrow F$ is the (perhaps implicitly) defined embedding function, we have

$$K(x_i; x_j) = \phi(x_i).\phi^T(x_j) \quad \textbf{(2.24)}$$

Typical kernel functions include the Gaussian kernel for which

$$K(x_i; x_j) = \phi(x_i).\phi^T(x_j) = \exp\left(-\frac{1}{2\sigma^2}\|x_i - x_j\|^2\right) \quad \textbf{(2.25)}$$

and the polynomial kernel

$$K(x_i; x_j) = \phi(x_i).\phi^T(x_j) = \langle x_i; x_j \rangle^p \quad \textbf{(2.26)}$$

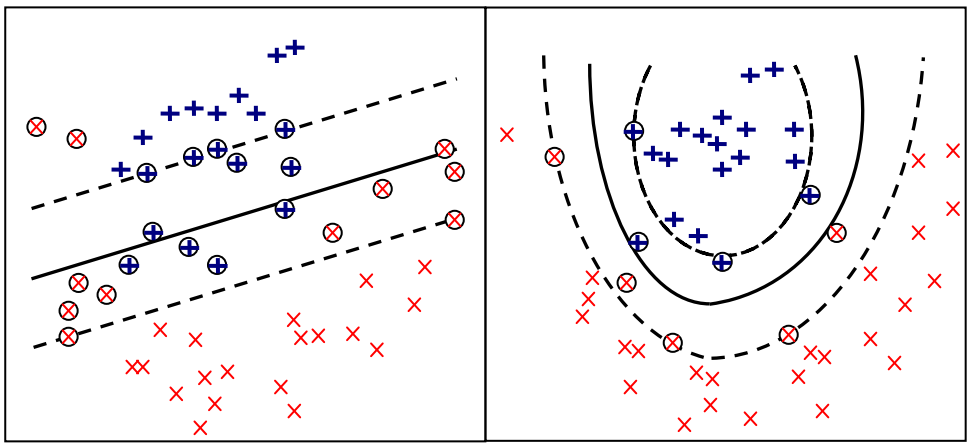
2.3.2 Examples of Kernels and Illustration

There are many types of kernel which is used in an SVM. In Table 2.1 some commonly used of kernels are listed. A kernel is said as an acceptable kernel if and only if it can be shown as an inner product in a feature space that is what we named as Mercer's condition [2].

Table 2.1: Common kernels

Polynomial	$k(x, y) = \left(\left(x^T y\right) + \theta\right)^d$	**(2.27)**
Gaussian	$k(x, y) = \exp\left(-\left\|x - y\right\|^2 / \sigma^2\right)$	**(2.28)**
Sigmoidal	$k(x, y) = \tanh(k(x^T . y) - \theta)$	**(2.29)**

Figure 2.6 shows a toy example using a Gaussian kernel of a non-linear SVM with a RBF kernel. It is seen that using non–linearity by the kernel function, the dataset becomes linearly separable in the feature space F [8].



(a) **(b)**

Figure 2.6: Illustration of a linear (left panel) and a nonlinear SVM with RBF kernel (right panel) on a toy dataset. Training examples from the two classes are represented by 'x' and '+', respectively. The solid curve shows the decision surface, i.e. $(w.\phi(x)) + b = 0$, the dotted curves show the margin area, i.e. $(w.\phi(x)) + b = \pm 1$. Support vectors are marked by small circles. It can be observed that only very few training patterns become support vectors, namely those that are inside the margin area or misclassified.

2.4 Summary

In the previous chapter, some basic notations and concepts about SVMs and kernel functions are reviewed. Kernel functions also give us the opportunity to work on very high dimensional spaces. By using the kernel trick we are able to express an algorithm with the dot products of the input data. Without kernel functions the algorithms are only capable of finding linear hyperplanes. By using kernel functions, we are able to apply linear methods to vectorial as well as non-vectorial data. Because of these many advantages, researchers have been worked rigorously on developing new kernel algorithms [7].

3. PCA AND FISHER DISCRIMINANT ANALYSIS METHODS

In this section, we give a brief review of PCA as a dimensionality reduction method, we explain the original Fisherface algorithm and we mention about our main algorithm which is KFDA. We also give a brief review of the distance measure that we used in our algorithms.

3.1 Principal Component Analysis

Principle Component Analysis (PCA) is a linear transformation algorithm that uses the principle components of the input data set for calculating the variance between them. PCA adjusts the coordinate system such that, the greatest variance of the projected data set lies on the first axis, the second largest variance lies on the second axis, etc. PCA is very effective on dimension reduction. Since it preserves most of the data variation in dimension reduction, PCA is used commonly in face recognition programs for this purpose.

Assume that we have a data set

$$X = [x_1, x_2, \dots, x_n] \quad \textbf{(3.1)}$$

where n is the number of samples in class X. We first calculate the mean of class X and then subtract from our class elements.

$$\bar{x}_i = x_i - m \quad , \quad m = \frac{1}{n}\sum_{i=1}^{n} x_i \quad \textbf{(3.2)}$$

$$\bar{X} = \left[\bar{x}_1, \bar{x}_2, ..., \bar{x}_n\right] \quad \textbf{(3.3)}$$

The covariance matrix Ω is calculated by

$$\Omega = \bar{X}\bar{X}^T \quad \textbf{(3.4)}$$

After calculating the covariance matrix we can find the eigenvalues and corresponding eigenvectors of our covariance matrix by

$$\Omega V = \Lambda V \tag{3.5}$$

where

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & ... & 0 \\ 0 & \lambda_2 & ... & . \\ . & . & . & 0 \\ 0 & ... & 0 & \lambda_p \end{bmatrix} \tag{3.6}$$

is the diagonal matrix of eigenvalues corresponding to the eigenvectors of

$$V = [v_1 \mid v_2 \mid \ \ldots \ v_p] \quad . \tag{3.7}$$

The eigenvector corresponding to the largest eigenvalue represents the basis vector which contains most variance.

The i[th] sample, x_i , can be projected into the PCA space by

$$\hat{x}_i = \overline{V}^T x_i \ . \tag{3.8}$$

3.2 Fisher Linear Discriminant Analysis and Fisherfaces

The approach that was adopted by Fisher is named as Fisher Linear Discriminant Analysis (FLDA) [9]. The main idea in this approach is to find a linear combination of variables which separates the classes in a right way. This method provides this correct separation by controlling the scatters of classes. By maximizing the ratio of the between-class scatter and within-class scatter, a good separation can be obtained. The mathematical notation can be written as

$$J(w) = \frac{w^T S_B w}{w^T S_W w} \tag{3.9}$$

where the between scatter S_B and the within scatter S_W can be defined as

$$S_B = \sum_{i=1}^{C} N_i (m_i - m)(m_i - m)^T \tag{3.10}$$

and

$$S_W = \sum_{i=1}^{c} \sum_{x_k \in X_i} (x_k - m_i)(x_k - m_i)^T \quad \textbf{(3.11)}$$

where m_i is the mean matrix of class X_i and N_i is the number of samples in class X_i. Here, what we are looking for is a set of feature vectors w_i that maximizes the equation (3.9). This leads to the generalized symmetric eigenvector equation

$$S_B W = S_W W \Lambda \quad \textbf{(3.12)}$$

where W is the matrix whose columns consist of w_i and Λ is the diagonal matrix of eigenvalues. If S_W is full-rank, then the equation (3.13) can be written as

$$S_W^{-1} S_B W = W \Lambda \quad \textbf{(3.13)}$$

Here, the eigenvectors corresponding to the largest eigenvalues are used for feature extraction. Since there can be maximum c-1 nonzero generalized eigenvalues, the upper bound for the number of eigenvalues can be maximum c-1 where c is the number of classes.

Because FLDA makes a class oriented linear projection, it has advantages to the other kind of linear algorithms. In Figure 3.1 there is a comparison of FLDA and PCA [9].

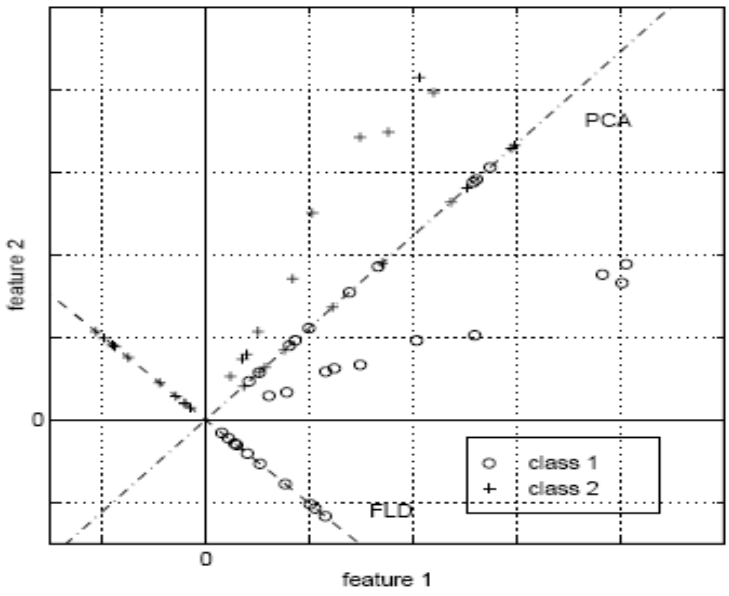


Figure 3.1: A comparison of FLDA and PCA

In this example, the number of classes is 2 and the number of samples in each class is 20. Here, it is seen that Fisher's class specific algorithm gives better results than PCA in image classification. Despite the large value of the PCA's total scatter, Fisher's between-class scatter makes a very good separation between classes; therefore FLDA makes better classification than PCA.

In FLDA the singularity of the within class scatter matrix can be a problem for face recognition. In face recognition programs, mostly the number of training images is less than the number of pixels in each image. The rank of the within class scatter matrix can be at most *N-c* where *N* is the number of images in each class and *c* is the class count, and in most cases *N* is smaller than the number of pixels in each image. Thus, choosing a within class scatter matrix that makes the within class scatter of the projected samples zero is possible. This is the dilemma of the FLDA.

To solve this singularity of S_W a method which is called Fisherfaces is implemented. Instead of direct usage of FLDA, this method first projects the image to a lower dimensional space and overcomes the singularity of the within class scatter matrix S_W. To make the matrix S_W nonsingular, these Fisherfaces reduce the dimension of the feature space to *N-c* by using PCA and then apply the original FLDA to reduce the dimension to *c-1*. The new equation of $J(w)$ can be written as

$$J(w)^T = J(w)_{fld}^T J(w)_{pca}^T \tag{3.14}$$

where

$$J(w)_{pca} = \arg\max_W \left| W^T S_T W \right| \tag{3.15}$$

$$J(w)_{fld} = \arg\max_W \frac{\left| W^T J(w)_{pca}^T S_B J(w)_{pca} W \right|}{\left| W^T J(w)_{pca}^T S_W J(w)_{pca} W \right|}. \tag{3.16}$$

By using this new notation (3.16), we solve the problem of FLDA. And because the usage of W_{pca} throws only the smallest principal components of S_B and S_W, most of the information of the between class scatter matrix and within class scatter matrix are still preserved.

3.2.1 Why Fisher Can Be Really Bad?

As mentioned in above sections, FLDA produces at most *c-1* projections. If the classification error estimates establish that more features are needed, FLDA is not enough alone so some other method must be employed to provide those additional features. Because FLDA is a parametric method, it assumes unimodal Gaussian likehoods. If the distributions are significantly non-Gaussian, the FLDA projections will not be able to preserve any complex structure of the data, which may be needed for classification as seen below.

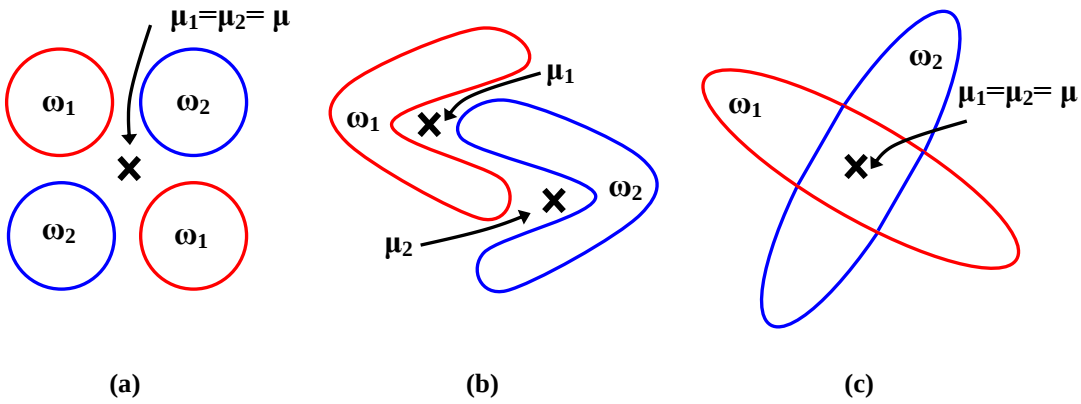


Figure 3.2: The weak points of FLDA where it can not separate the classes correctly.

3.3 Kernel Fisher Discriminant Analysis

In real life, linear discriminant information may not be enough especially for analyzing face variations under different illumination and pose. In this case, the non-linear information of the input data should be analyzed. KFDA gives us the opportunity to look for non-linear directions of the input data. The basic idea of KFDA is to apply a non-linear mapping to the data in input space, and then computing FLDA on this new mapped feature space *F*. Through this way, we achieve to obtain non-linear information of the input data.

Let Φ be a non-linear mapping in feature space *F*. By Fisher's theorem [9], to find the linear discriminant in this feature space we need to maximize

$$J(w) = \frac{w^T S_B^{\phi} w}{w^T S_W^{\phi} w} \qquad \textbf{(3.17)}$$

where $w \in F$ and S_B^{ϕ} is the between class scatter and S_W^{ϕ} is the within class scatter matrices in this new feature space *F*

$$S_B^{\Phi} = \frac{1}{C(C-1)} \sum_{i=1}^{C} \sum_{j=1}^{C} (m_i^{\Phi} - m_j^{\Phi})(m_i^{\Phi} - m_j^{\Phi})^T$$

$$S_W^{\Phi} = \frac{1}{C} \sum_{i=1}^{C} \frac{1}{n_i} \sum_{j=1}^{n_i} (\Phi(x_j) - m_i^{\Phi})(\Phi(x_j) - m_i^{\Phi})^T \quad \textbf{(3.18)}$$

with

$$m_i^{\Phi} = (1/n_i) \sum_{j=1}^{n_i} \Phi(x_j) \ . \quad \textbf{(3.19)}$$

Since we map input data from a low dimensional space to a higher dimensional space *F*, mapping the data directly to the feature space *F* is almost impossible. The solution of the problem arises with the kernel trick. By computing the inner products of the input data $(\Phi(x).\Phi(y))$, we can achieve to map input data to a higher dimensional space. Mercer kernels [5] give us the opportunity for such a kind of mapping. Polynomial kernel k(x,y) = $(x.y)^d$ or Gaussian kernel $k(x, y) = \exp(-\|x - y\|^2 / \sigma)$ are the most useful kernels for this type of problem. The polynomial kernel function that we use in our experiments is

$$k(x, y) = (\Phi(x).\Phi(y)) = (a(x.y) + b)^d \ . \quad \textbf{(3.20)}$$

Here a, b, σ and d are positive constants.

From the equation (3.17), because any solution $w \in F$ must lie in the span of all the samples in *F*, *w* can be written as

$$w = \sum_{i=1}^{n_i} \alpha_i \Phi(x_i) \quad \textbf{(3.21)}$$

From the definition of S_B^{Φ} and equation (3.21), we can write

$$w^T S_B^{\phi} w = \alpha^T K_B \alpha \quad \textbf{(3.22)}$$

where the definition of K_B is as follows

$$K_B = \frac{1}{C(C-1)} \sum_{i=1}^{C} \sum_{j=1}^{C} (\mu_i^{\Phi} - \mu_j^{\Phi})(\mu_i^{\Phi} - \mu_j^{\Phi})^T \quad \textbf{(3.23)}$$

and

$$\mu_i^{\Phi} = \left(\frac{1}{n_i} \sum_{j=1}^{n_i} k(x_1, x_j), \frac{1}{n_i} \sum_{j=1}^{n_i} k(x_2, x_j), ..., \frac{1}{n_i} \sum_{j=1}^{n_i} k(x_n, x_j) \right)^T \quad \textbf{(3.24)}$$

, and through a similar transformation as in (3.22), it can be found that

$$w^T S_W^{\phi} w = \alpha^T K_W \alpha \quad \textbf{(3.25)}$$

where

$$K_W = \frac{1}{C} \sum_{i=1}^{C} \frac{1}{n_i} \sum_{j=1}^{n_i} (\zeta_j - \mu_i^{\Phi})(\zeta_j - \mu_i^{\Phi})^T \quad \textbf{(3.26)}$$

and

$$\zeta_j = \left(k(x_1, x_j), k(x_2, x_j), ..., k(x_n, x_j) \right)^T . \quad \textbf{(3.27)}$$

Using the equation (3.22) and (3.25), we can find the FLDA in *F* by maximizing

$$J(\alpha) = \frac{\alpha^T K_B \alpha}{\alpha^T K_W \alpha} \quad . \quad \textbf{(3.28)}$$

This problem can be solved by finding the leading eigenvectors of $K_W^{-1} K_B$ as in FLDA. This approach is called as Kernel Fisher Discriminant Analysis (KFDA) [10]. We can calculate the projection of a pattern *x* onto *w* from the equation

$$\left(w.\Phi(x) \right) = \sum_{i=1}^{l} \alpha_i k(x_i, x). \quad \textbf{(3.29)}$$

3.3.1 Cosine Kernel Function

Selecting a suitable kernel function is very important in face recognition algorithms. Different kernel functions will form different constructions of implicit feature space. In our experiments, we use the cosine kernel function which is based on the original polynomial kernel in order to improve the performance of KFDA. It is as follows

$$\tilde{k}(x,y) = \frac{k(x,y)}{\sqrt{k(x,x)k(y,y)}} \quad \textbf{(3.30)}$$

where $k(x,y)$ is the original polynomial kernel. Cosine measurement can be used as the similarity measure so instead of using only the inner product measurement [11], we used the cosine measurement in KFDA.

3.3.2 Problems of KFDA

KFDA has numerical problems since K_W is sometimes singular. In this case, we can not find the leading eigenvectors of $K_W^{-1}K_B$. A simple way to avoid this problem is adding a multiple identity matrix to K_W so that the new equation becomes

$$K_{W_\mu} = K_W + \mu I \quad \textbf{(3.31)}$$

where μ is a very small constant. In addition to this, KFDA can not handle with the case where there is only one sample for each person for training as FLDA. When this happens, it is better to use some tricks like first generating multiple images from a single image. And also selecting a suitable kernel function for different tasks is still an open problem.

3.4 Distance Measure

After we project our images into a subspace, it has to be determined that which images are the most like one to another. In our experiments, to find the distance of the projected images, we use L_2 norm distance measure.

***L_2* Norm:** The L_2 norm is also known as the Euclidean norm or the Euclidean distance. It sums up the squared differences between pixels. The L_2 norm of an image A and an image B can be calculated as:

$$L_2(A,B) = \sum_{i=1}^{N} (A_i - B_i)^2 \quad \textbf{(3.32)}$$

4. EXPERIMENTS

The aim of this work is to compare the linear and non-linear discriminant analysis technique methods of Fisher Discriminant Analysis which are called Fisherfaces and KFDA. Therefore, in this part we give information about the experimental data, the application, and the results of these two algorithms.

4.1 Face Databases

In our experiments, Yale, FERET, ORL, and Yale Database B are selected to compare the performance of KFDA and FLDA. For face expression tests we used the Yale and FERET face databases, for pose variations ORL database is used, and for illumination changes Yale and Yale Database B are used. For general recognition tests, FERET and a mixture of Yale and ORL databases are used. The brief description of used databases is below.

1) The Yale database which consists of 165 images of 15 individuals. These images have different types of expression and illumination sources (Figure 4.1).

Figure 4.1: The images belonging to a person from Yale Face Database

There are male and female people in the database. Some of them have beard and some of them are wearing glasses. The first image is the natural pose of a person which is illuminated from everywhere. The second image is obtained under same conditions but with glasses. If a person wears glasses, the other pictures are photographed with glasses, if not the pictures are photographed without glasses. The pictures 3-5 are photographed by using Luxolamp. The last six pictures have different expressions of a person which are illuminated from everywhere.

2) The ORL database which consists of 400 images of 40 individuals, containing quite a high degree of variability in expression, pose, and facial details. 10 people are selected from the ORL face database (Figure 4.2).

Figure 4.2: The images belonging to a person from ORL Face Database

All the images are resized to 70x70.

3) The Yale Database B contains 5760 single light sources images of 10 subjects under 64 different lighting conditions for 9 poses. Since this database is only used for illumination tests, only the frontal face images are taken. As shown in Figure 4.3 the face images are divided into the four subsets according to the angle between the camera axis and light source direction as shown in Table 4.1 [12].

Table 4.1: Subsets divided according to light source direction

Subsets	**1**	**2**	**3**	**4**
Lighting angle (°)	0~12	13~25	26~50	51~77
Number of images	70	120	120	140

Subset 1

Subset 2

Subset 3

Subset 4

Figure 4.3: The images belonging to a person that is divided into four subsets.

4) In FERET database we select 50 people, four images for each person. All images are processed and images from a person are shown in Figure 4.4.

Figure 4.4: The images belonging to a person from FERET database.

5) We also use another subset of FERET database for general face recognition. In this subset there are 15 people (Figure 4.5).

Figure 4.5: The images belonging to a person from FERET database.

4.2 Application

To see the results of the two algorithms an application is developed in a machine which has 1 GB RAM and Intel Centrino 1.73 GHZ CPU under Windows XP platform. The code of the application is written in Borland C++ Builder 6.0© with C programming language.

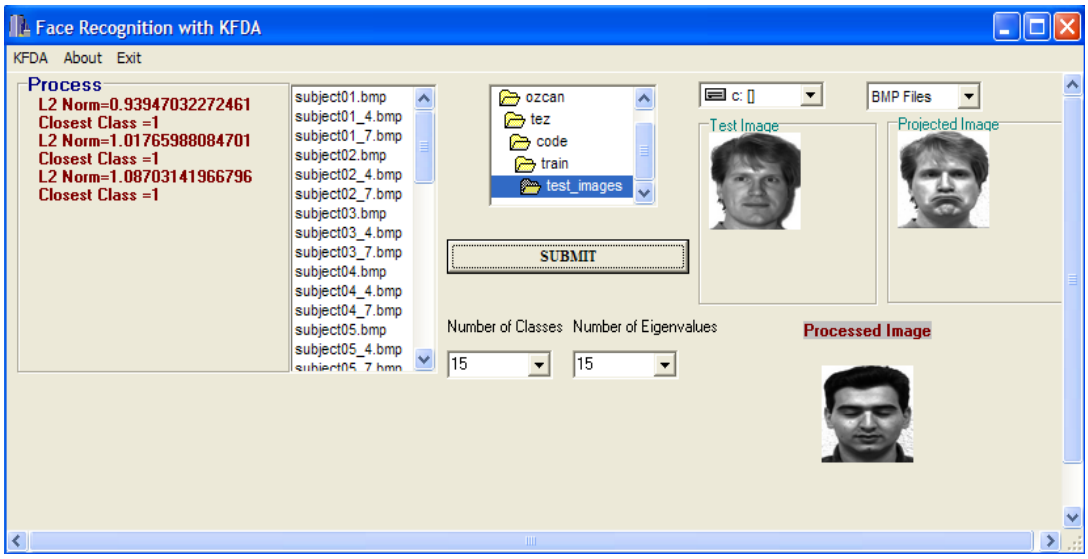


Figure 4.6: Interface of the Program

4.3 Experimental Results

We made many tests under the mixture of the databases and the databases themselves. We also made tests with facial expressions, pose variations, and illumination themselves. The size of the images that we used is 70x70 pixels. The parameters of our cosine polynomial kernel are as follows, a = (10^{-3} / size of image), b = 0, and the degree d = 2. In each test, we use three different types of images. First type is the original images of the databases. Second type is the images to which histogram equalization is applied before testing KFDA and FLDA. In performance results, second type KFDA and FLDA are shown as HEKFDA and HEFLDA sequentially. In third type, we divided each image into four and applied histogram equalization to each part separately. We call this method as divided histogram equalization. In performance results, third type KFDA and FLDA are shown as DHEKFDA and DHEFLDA separately. The results of our tests with different feature numbers can be seen below.

4.3.1 FLDA and KFDA Performance Results

Test-1: In this test, facial expressions are tested by using the Yale database. The 8 of the 11 pictures per person are selected. The three of them are used as train and the other five are used as test images. (Figure 4.7, Figure 4.8). Performance results are as follows:

Figure 4.7: The train image set belonging to a person

Figure 4.8: The test image set belonging to a person

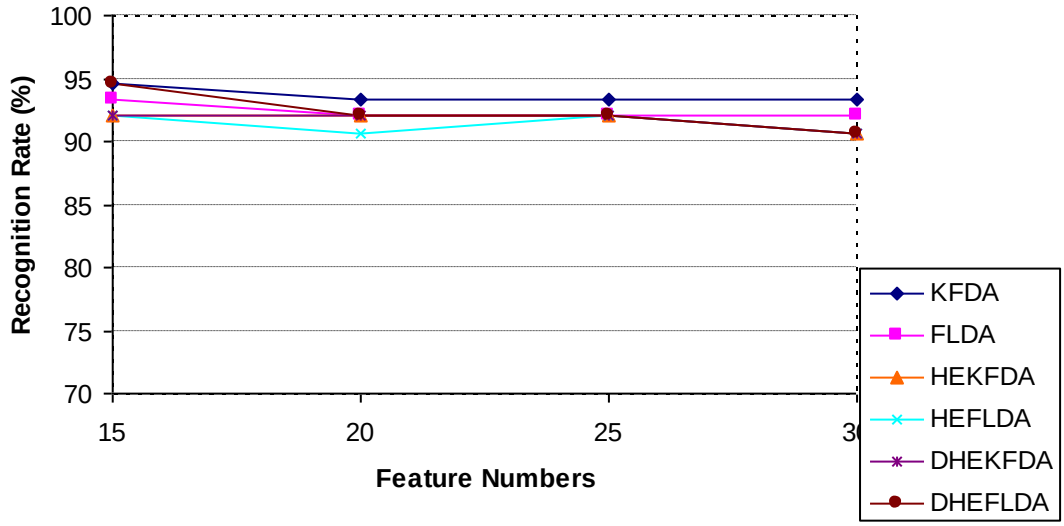


Figure 4.9: Comparison of methods under facial expressions

Table 4.2: Performance results of methods under facial expressions

	15	**20**	**25**	**30**
KFDA	%94.6	%93.3	%93.3	%93.3
FLDA	%93.3	%92	%92	%92
HEKFDA	%92	%92	%92	%90.6
HEFLDA	%92	%90.6	%92	%90.6
DHEKFDA	%92	%92	%92	%90.6
DHEFLDA	%94.6	%92	%92	%90.6

Test-2: In this test, the effect of the illumination is tested by using the Yale database. Seven of the images belonging to a person are selected as train and the other three pictures are selected as test images. (Figure 4.10, Figure 4.11, Figure 4.12, Figure 4.13, Figure 4.14, Figure 4.15). Performance results are as follows:

Figure 4.10: The train image set belonging to a person

Figure 4.11: The test image set belonging to a person

Figure 4.12: The train image set belonging to a person, to which histogram equalization is applied

Figure 4.13: The test image set belonging to a person, to which histogram equalization is applied

Figure 4.14: The train image set belonging to a person, to which divided histogram equalization is applied

Figure 4.15: The test image set belonging to a person, to which divided histogram equalization is applied

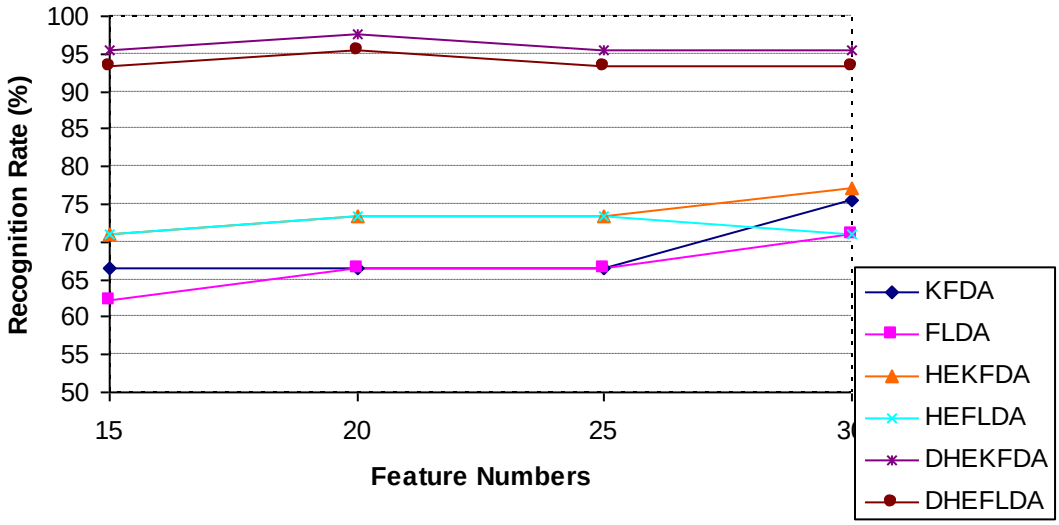


Figure 4.16: Comparison of methods under illumination. HE means that before processing the image histogram equalization is applied. DHE means that before processing the image divided histogram equalization is applied.

Table 4.3: Performance results of methods under illumination

	15	**20**	**25**	**30**
KFDA	%66.6	%66.6	%66.6	%75.5
FLDA	%62.2	%66.6	%66.6	%71.1
HEKFDA	%71.1	%73.3	%77.1	%73.3
HEFLDA	%71.1	%73.3	%73.3	%71.1
DHEKFDA	%95.5	%97.7	%95.5	%95.5
DHEFLDA	%93.3	%97.7	%93.3	%93.3

Test-3: In this test, the effect of pose variations is tested by using the ORL database. Four of the images belonging to a person are selected as train and other different pose images are selected as test images. (Figure 4.17, Figure 4.18, Figure 4.19, Figure 4.20, Figure 4.21, Figure 4.22). Performance results are as follows:

Figure 4.17: The train image set belonging to a person

Figure 4.18: The test image set belonging to a person

Figure 4.19: The train image set belonging to a person, to which histogram equalization is applied

Figure 4.20: The test image set belonging to a person, to which histogram equalization is applied

Figure 4.21: The train image set belonging to a person, to which divided histogram equalization is applied

Figure 4.22: The test image set belonging to a person, to which divided histogram equalization is applied

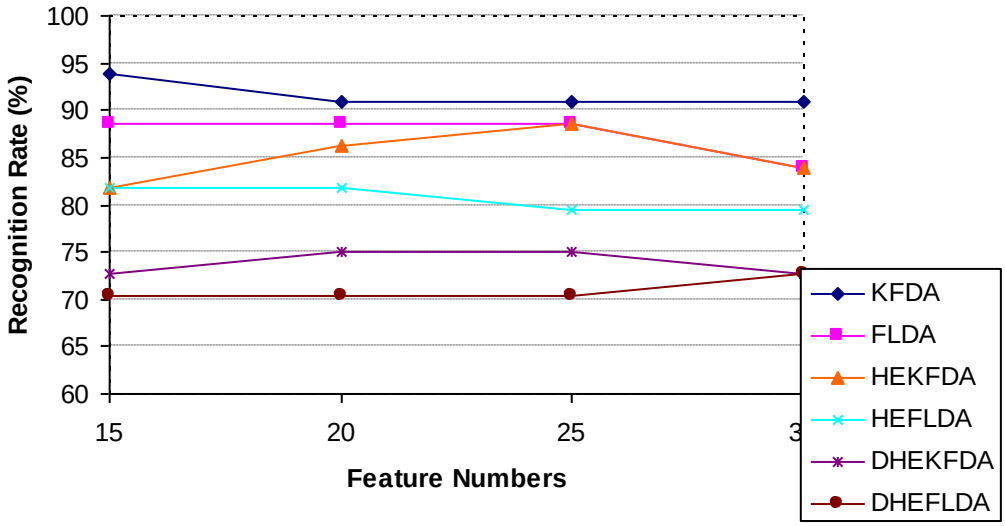


Figure 4.23: Comparison of methods under pose variations.

Table 4.4: Performance results of methods under pose variations

	15	**20**	**25**	**30**
KFDA	%93.8	%90.9	%90.9	%90.9
FLDA	%88.6	%88.6	%88.6	%84
HEKFDA	%81.8	%86.3	%88.6	%84
HEFLDA	%81.8	%81.8	%79.5	%79.5
DHEKFDA	%72.7	%75	%75	%72.7
DHEFLDA	%70.4	%70.4	%70.4	%72.7

Test-4: In this test, we use a mixture of Yale and ORL face database. There are 15 people from Yale and 10 people from ORL face databases. The four of the images belonging to a person are used as train and the others are used as test. Performance results are as follows:

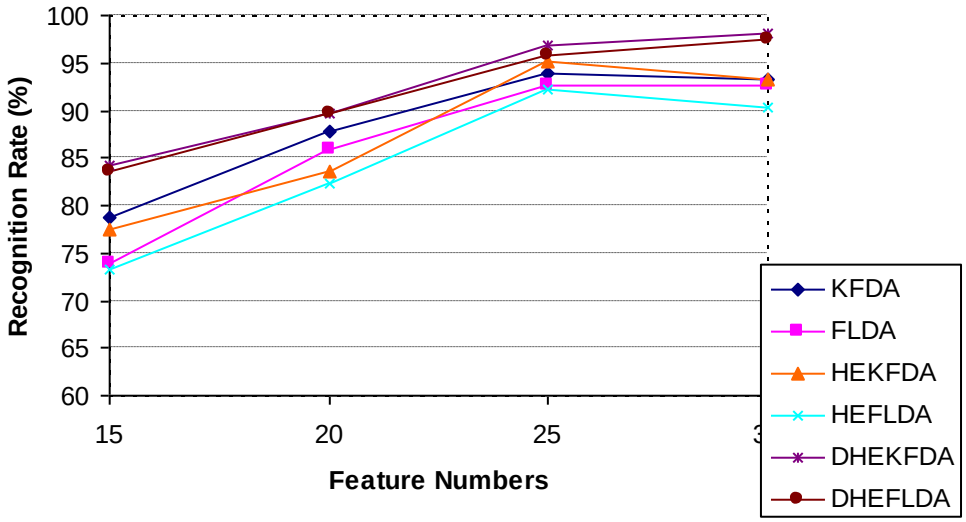


Figure 4.24: Comparison of methods in a mixture database

Table 4.5: Performance results of methods in a mixture database

	15	**20**	**25**	**30**
KFDA	%78.7	%87.8	%93.9	%93.3
FLDA	%73.6	%86	%92.7	%92.7
HEKFDA	%77.5	%83.6	%95.1	%93.3
HEFLDA	%73.3	%82.4	%92.1	%90.3
DHEKFDA	%84.2	%89.6	%96.9	%98.1
DHEFLDA	%83.6	%89.6	%95.7	%97.5

Test-5: In this test, we use the FERET database. As a subset 15 people are selected. Four images belonging to a person is used as train images, and two images are used as test images. Performance results are as follows:

Figure 4.25: The train image set belonging to a person

Figure 4.26: The test image set belonging to a person

Figure 4.27: The train image set belonging to a person, to which histogram equalization is applied

Figure 4.28: The test image set belonging to a person, to which histogram equalization is applied

Figure 4.29: The train image set belonging to a person, to which divided histogram equalization is applied

Figure 4.30: The test image set belonging to a person, to which divided histogram equalization is applied

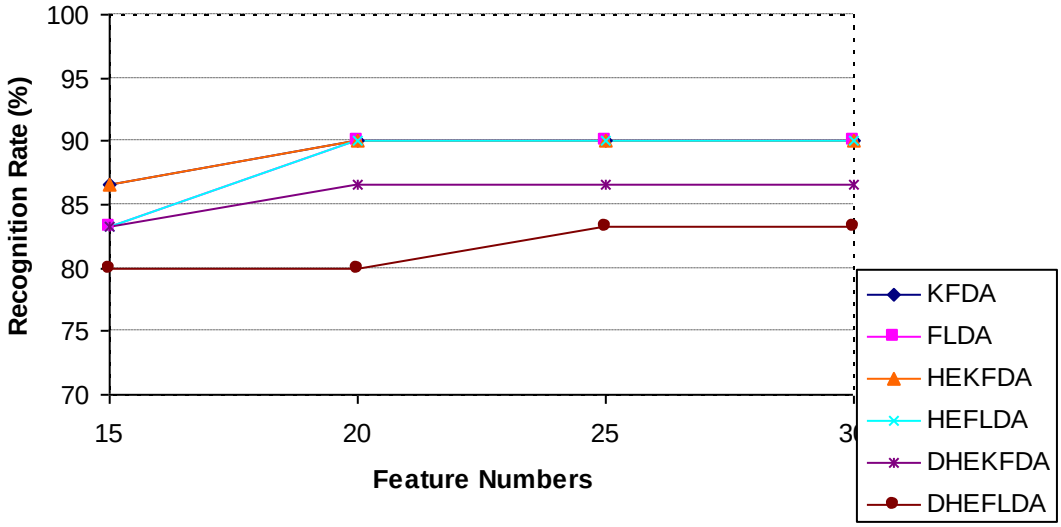


Figure 4.31: Comparison of methods in FERET database

Table 4.6: Performance results of methods in FERET database

	15	**20**	**25**	**30**
KFDA	%86.6	%90	%90	%90
FLDA	%83.3	%90	%90	%90
HEKFDA	%86.6	%90	%90	%90
HEFLDA	%83.3	%90	%90	%90
DHEKFDA	%83.3	%86.6	%86.6	%86.6
DHEFLDA	%80	%80	%83.3	%83.3

Test-6: In this test we use the FERET database. As a subset 50 people are selected. Three images belonging to a person is used as train images, and one image is used as test image. Performance results are as follows:

Figure 4.32: The train image set belonging to a person

Figure 4.33: The test image belonging to a person

Figure 4.34: The train image set belonging to a person, to which divided histogram equalization is applied

Figure 4.35: The test image belonging to a person, to which divided histogram equalization is applied

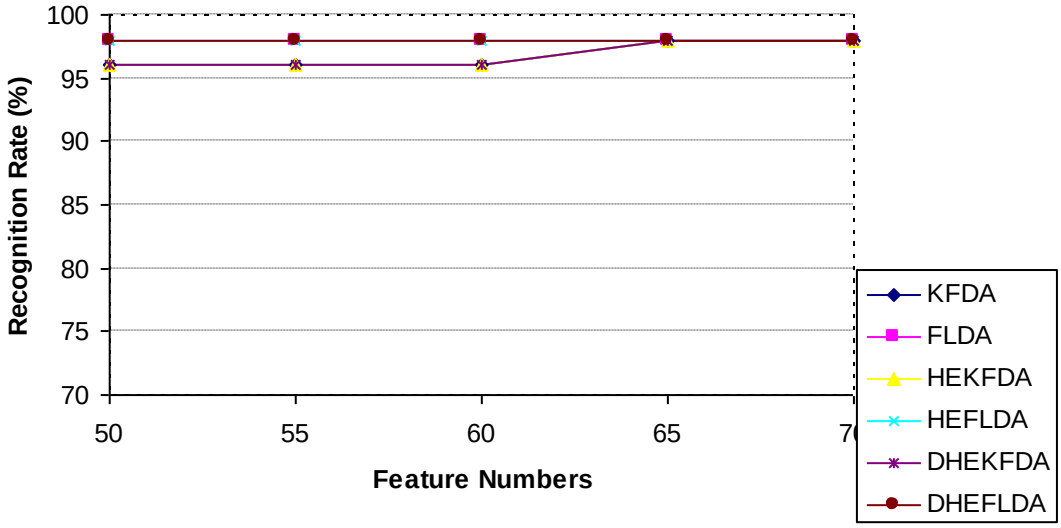


Figure 4.36: Comparison of methods in FERET database

Table 4.7: Performance results of methods in FERET database

	50	**55**	**60**	**65**	**70**
KFDA	%96	%96	%96	%98	%98
FLDA	%98	%98	%98	%98	%98
HEKFDA	%96	%96	%96	%98	%98
HEFLDA	%98	%98	%98	%98	%98
DHEKFDA	%96	%96	%96	%98	%98
DHEFLDA	%98	%98	%98	%98	%98

Test-7: In this test, the effect of the illumination is tested by using the Yale database B. Four pictures of each person from subset 1 are selected as train images and the others are used as test images (Figure 4.3). Performance results are shown below:

Table 4.8: Performance results of methods in Yale database B under different subsets

	Subset 1	**Subset 2**	**Subset 3**	**Subset 4**
KFDA	%100	%85	%49.1	%11.4
FLDA	%100	%82.5	%46.6	%10
HEKFDA	%100	%88.3	%60	%31.4
HEFLDA	%100	%87.5	%52.5	%29.2
DHEKFDA	%100	%95.8	%65	%47.6
DHEFLDA	%100	%87.5	%56.6	%38

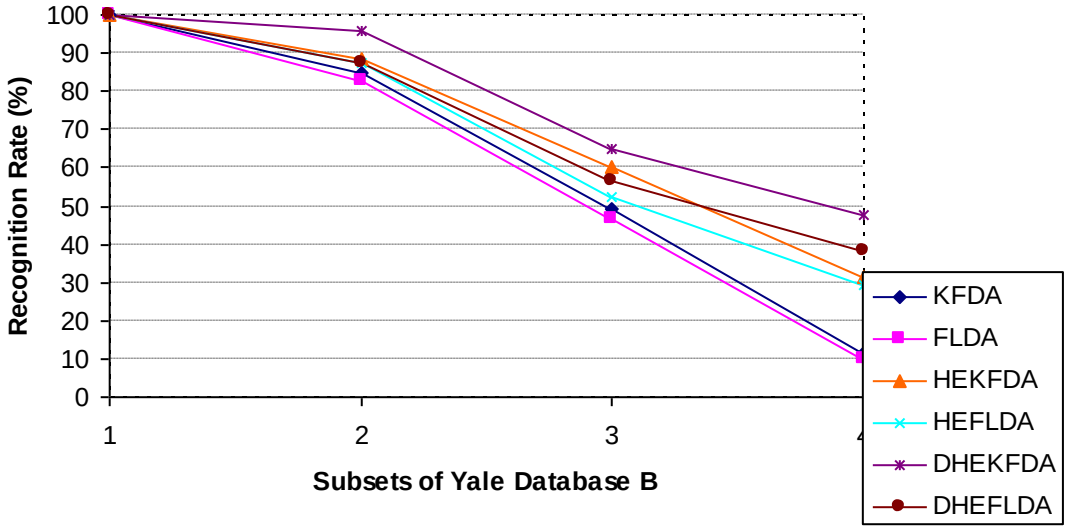


Figure 4.37: Comparison of methods in Yale database B under different subsets

As shown from seven test results, in general KFDA gives better results than FLDA. But the recognition rates of these two algorithms are very close to each other. For illumination conditions before processing the images, applying histogram equalization to images increases the recognition rates. Also, dividing the images to four parts and applying histogram equalization to each part separately gives remarkable recognition rates under illumination conditions.

Test-8: In this test, the general recognition rate of Yale database is tested. Eight pictures of each person (subject1, 2, 3, 4, 5, 6, 7, 8) are selected as train images and

the rest of them (subject 9, 10, 11) are selected as test images. Performance results are shown below:

Table 4.9: Performance results of methods in YALE database

	15	**20**	**25**	**30**
KFDA	%95.5	%97.7	%97.7	%97.7
FLDA	%95.5	%95.5	%97.7	%97.7
HEKFDA	%93.3	%93.3	%95.5	%95.5
HEFLDA	%93.3	%95.5	%95.5	%95.5
DHEKFDA	%93.3	%95.5	%95.5	%95.5
DHEFLDA	%93.3	%93.3	%93.3	%95.5

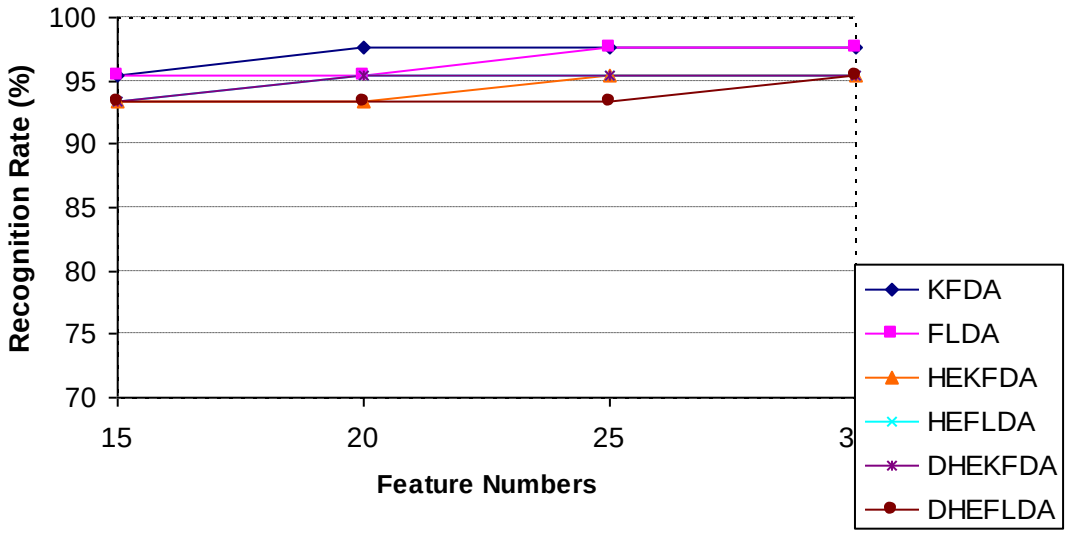


Figure 4.38: Comparison of methods in YALE database

Test-9: In this test, one of the facial expressions is tested separately. The number of eigenvalues used in this test is 25. Performance results are shown below:

1- center-light , 2- with glasses, 3- happy, 4- left-light, 5- no glasses,

6 – normal, 7- right-light, 8- sad, 9- sleepy, 10- surprised, 11- wink

Table 4.10: Performance results of methods in YALE database for separate facial expressions

	1	2	3	4	5-6	7	8-9	10	11
KFDA	%80	%93.3	%100	%33.3	%100	%40	%100	%100	%93.3
FLDA	%80	%93.3	%100	%26.6	%100	%33.3	%100	%93.3	%93.3
HEKFDA	%80	%93.3	%100	%73.3	%100	%80	%100	%100	%93.3
HEFLDA	%80	%86.6	%100	%66.6	%100	%73.3	%100	%93.3	%93.3
DHEKFDA	%80	%93.3	%100	%93.3	%100	%93.3	%100	%100	%93.3
DHEFLDA	%80	%86.6	%100	%86.6	%100	%86.6	%100	%93.3	%93.3

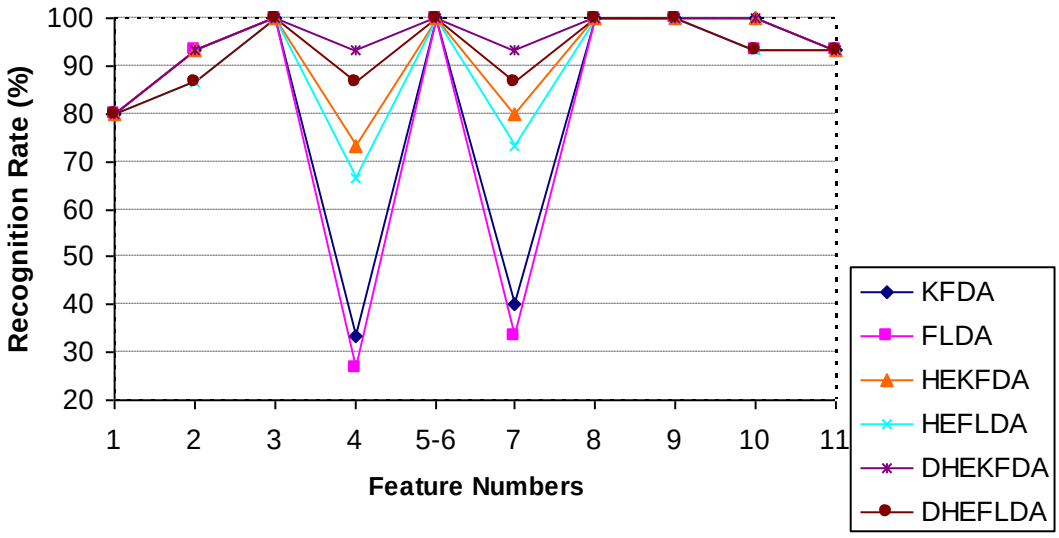


Figure 4.39: Comparison of methods in YALE database for separate facial expressions

5. CONCLUSION

In this thesis, KFDA is studied. As a case study, the performance of KFDA is analyzed in many face databases like Yale, FERET, ORL, and etc. and is compared with FLDA. In previous studies, it has been seen that FLDA has an average success rate of 90 percent under facial expressions and is affected less from lighting and illumination conditions than the other methods like PCA, local feature analysis, and etc. But, because FLDA only uses the linear information, it can not benefit from the non-linear information. Here, the kernel trick is used to first project the input data into a new space and then FLDA is applied to this new space. By using not only the linear information but also the non-linear information, KFDA gives us more success rate than FLDA. The expected success rate of KFDA is over 90 percent. In future work, we will focus on reducing the effects of illumination on images before applying the KFDA algorithm.

As a final conclusion, the methods applied indicate that, kernel based version of the Fisher Discriminant Analysis is more convenient for face recognition than the original Fisher Discriminant Analysis.

REFERENCES

[1] **ICAO.**, 2004. Biometrics Deployment of Machine Readable Travel Documents: Technical Report, ICAO TAG MRTD/NTWG.

[2] **Vapnik, V.**, 1998. *Statistical Learning Theory*, Wiley, New York.

[3] **Cortes, C. and Vapnik, V.**, 1995. Support-vector networks. *Machine Learning*, 20:237-297.

[4] **Wolfe, P.**, 1961. A duality theorem for non-linear programming. *Quart. Appl. Math*, 19:239–244.

[5] **Mercer, J.**, 1909. Functions of positive and negative type and their connection with the theory of integral equations. *Proc. Roy. Soc. London Ser. A*, 209:415–446.

[6] **Schölkopf, B. and Smola, A.J.**, 2002. *Learning with Kernels*. MIT Press, Cambridge, MA.

[7] **Schölkopf, B., Smola, A.J., Williamson, R.C. and Bartlett, P.L.**, 2000. New support vector algorithms. *Neural Computation*, 12:1207 – 1245.

[8] **Mika, S., Rätsch, G., Weston, J., Schölkopf, B. and Müller, K.R.**, 1999. Fisher Discriminant Analysis with kernels. in *Proc. IEEE Workshop on Neural Networks for Signal Processing IX*, 41-48.

[9] **Belheumer, P., Hespanha, J. and Kreigman, D.**, 1997. Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(7):711-720

[10] **Mika, S., Rätsch, G. and Müller, K.R.**, 2001. A mathematical programming Approach to the kernel Fisher algorithm. In T.K. Leen, T.G. Dietterich, and V Tresp, editors, *Advances in Neural Information Processing Systems*, 13:591–597. MIT Press.

[11] **Cristianini, N., Taylor, J.S. and Kandola, J.**, 2001. Spectral kernel methods for Clustering. in *Proc. Int. Conf. Neural Information Processing Systems*, 649-655.

[12] **Georghiades, A. S., Belhumeur, P. N. and Jacobs, D.W.**, 2001. From few to many: illumination cone models for face recognition under variable lighting and pose. *IEEE Trans. Pattern Anal. Mach. Intell*,23(6):630-660.

$$J(w) = \frac{w^T S_B w}{w^T S_W w}$$

RESUME

Özcan Çavuş was born in 1979, in Kocaeli. He was graduated from Anatolian Elementary School, Oruç Reis Anatolian High School, and Kocaeli Science High School respectively. In 2002, he was gratuated from Istanbul Technical University Control&Computer Engineering Department and he started the master program in Computer Engineering Department in the same university. He is now working at Oyak Technology as a computer engineer.