Two-Dimensional Face Recognition Algorithms in the Frequency Domain

Alper Serhat Zeytunlu

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Master of Applied Science at
Concordia University
Montreal, Quebec, Canada

January 2012

CONCORDIA UNIVERSITY
SCHOOL OF GRADUATE STUDIES

This is to certify that the thesis prepared

By: Alper Serhat Zeytunlu

Entitled: "Two-Dimensional Face Recognition Algorithms in the Frequency Domain"

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science

Complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

____ Chair
Dr. R. Raut

____ Examiner, External
Dr. C.Y. Su (M.I.E.) To the Program

____ Examiner
Dr. W-P. Zhu

____ Supervisor
Dr. M.O. Ahmad

____ Supervisor
Dr. M. N. S. Swamy

Approved by: ____
Dr. W. E. Lynch, Chair
Department of Electrical and Computer Engineering

____________20_____ ____________________________________
Dr. Robin A. L. Drew
Dean, Faculty of Engineering and
Computer Science

ABSTRACT

Two-Dimensional Face Recognition Algorithms in the Frequency Domain

Alper Serhat Zeytunlu

The importance of security, law-enforcement and identity verification has necessitated the development of automated stable, fast and highly accurate algorithms for human recognition. Face recognition is one of the most popular techniques used for these purposes. Face recognition algorithms are performed on very large size of datasets obtained under various challenging conditions. Principal component analysis (PCA) is a widely used technique for face recognition. However, it has major drawbacks of (i) losing the image details due to the transformation of two-dimensional face images into one-dimensional vectors, (ii) having a large time complexity due to the use of a large size covariance matrix and (iii) suffering from the adverse effect of intra-class pose variations resulting in reduced recognition accuracy. To overcome the problem of intra-class pose variations, Fourier magnitudes have been used for feature extraction in the PCA algorithm giving rise to the so called FM-PCA algorithm. However, the time complexity of this algorithm is even higher. On the other hand, to address the other two drawbacks of the PCA algorithm, two-dimensional PCA (2DPCA) algorithms have been proposed.

This thesis is concerned with developing 2DPCA algorithms that incorporate the advantages of FM-PCA in improving the accuracy and that of 2DPCA algorithms in improving the accuracy as well as the time complexity. Towards this goal, 2DPCA algorithms, referred to as the FM-r2DPCA and FM-$(2D)^2$PCA algorithms, that use Fourier-magnitudes rather than the raw pixel values, are first developed. Extensive simulations are conducted to demonstrate the effectiveness of using the Fourier-magnitudes in providing higher recognition accuracy over their spatial domain counterparts. Next, by taking advantage of the energy compaction property of the Fourier-magnitudes, the proposed algorithms are further developed to significantly reduce their computational complexities with little loss in the recognition accuracy. Simulation results are provided to validate this claim.

ACKNOWLEDGEMENTS

First and foremost, I would like o express my deep gratitude to my thesis supervisors, Dr. M.N.S. Swamy and Dr. M. Omair Ahmad, for their guidance and mentorship throughout the years of my graduate studies at Concordia University. Without their insight, mentorship and inspiration, it would have been impossible for me to complete my research.

I wish to thank Ahmet Zeytunlu, Omer Cekelez, Cengiz Kabakci, Ozkan Agis, Afsin Kocalan, Haktan Vardaryildiz, Dr. B. Siddik Yarman, Dr. Aydin Akan, Dr. O. Nuri Ucan, Dr. Ilhan Kocaarslan, Dr. Ertugrul Cam, Dr. Ata Sevinc, Dr. Ediz Polat, Dr. Saad Bouguezel for their inspiring advice, insight and care.

I am indebted to my colleagues and friends for their motivating support and encouragement during my research at Concordia University.

I would like to express my deepest thanks to Aynur Kelci for her special and invaluable support.

Lastly, and most importantly, I owe special thanks to my bellowed family and my grandparents for their unconditional love, encouragement, patience and sacrifices. To them I dedicate this thesis.

To

my dearest parents,

grandparents

and

brother

Table of Contents

List of Figures

List of Tables

List of Symbols

C	*Covariance matrix in the PCA algorithm*
C_{c2DPCA}	*Covariance matrix in the c2DPCA algorithm*
C_{FM}	*Covariance matrix in the FM-PCA algorithm*
$C_{FM-c2DPCA}$	*Covariance matrix in the FM-c2DPCA algorithm*
$C_{FM-r2DPCA}$	*Covariance matrix in the FM-r2DPCA algorithm*
$C_{LPFM-c2DPCA}$	*Covariance matrix in the LPFM-c2DPCA algorithm*
$C_{LPFM-r2DPCA}$	*Covariance matrix in the LPFM-r2DPCA algorithm*
C_{r2DPCA}	*Covariance matrix in the r2DPCA algorithm*
d_E	*Euclidean distance*
d_F	*Frobenius distance*
d_Y	*Yang distance*
I	*Original image in the spatial domain*
$\bar{I}$	*Average of the training images in the spatial domain*
I_{FM}	*Fourier-magnitude version of an image*
$\bar{I}_{FM}$	*Average of the training images in the Fourier domain*
I_{LPFM}	*Lowpass Fourier-magnitude version of an image*
$\bar{I}_{LPFM}$	*Average of the training images' lowpass Fourier-magnitude versions*
Q	*Transformation matrix in the PCA algorithm*

Q_{FM}	*Transformation matrix in the FM-PCA algorithm*
X	*Row-directional transformation matrix for two-dimensional PCA algorithms spatial-domain*
X_{FM}	*Row-directional transformation matrix for two-dimensional PCA algorithms Fourier domain*
X_{LPFM}	*Row-directional transformation matrix for lowpass Forirer-magnitude versions of two-dimensional PCA algorithms*
Y	*Column-directional transformation matrix for two-dimensional PCA algorithms spatial domain*
Y_{FM}	*Column-directional transformation matrix for two-dimensional PCA algorithms Fourier domain*
Y_{LPFM}	*Column -directional transformation matrix for lowpass Fourier-magnitude versions of two-dimensional PCA algorithms*
Z	*Feature matrix in two-dimensional PCA algorithms*
Z_{FM}	*Feature matrix in Fourier-magnitude two-dimensional PCA algorithms*
Z_{LPFM}	*Feature matrix in lowpass Fourier-magnitude two-dimensional PCA algorithms*
$\mathit{\Gamma}$	*Concatenated image in the spatial-domain*
$\mathit{\Gamma}_{FM}$	*Concatenated image in the Fourier-domain*
$\mathit{\Phi}$	*Mean-subtracted image in the spatial-domain*

Φ_{FM}	*Mean-subtracted image in the Fourier-domain*
Ψ	*Average of the concatenated training images in the spatial domain*
Ψ_{FM}	*Average of the concatenated training images in the Fourier domain*
Ω	*Feature vector of an image in the spatial-domain PCA algorithm*
Ω_{FM}	*Feature vector of an image in the Fourier-domain PCA algorithm*

List of Acronyms

$(2D)^2$PCA	Two-Directional Two-Dimensional Principal Component Analysis
c2DPCA	Column-Directional Two-Dimensional Principal Component Analysis
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DiaPCA	Diagonal Principal Component Analysis
DWT	Discrete Wavelet Transform
FM	Fourier-Magnitude
FM-$(2D)^2$PCA	Fourier-Magnitude Two-Directional Two-Dimensional Principal Component Analysis
FM-PCA	Fourier-Magnitude Principal Component Analysis
FM-r2DPCA	Fourier-Magnitude Row-Directional Two-Dimensional Principal Component Analysis
PCA	Principal Component Analysis
r2DPCA	Row-Directional Two-Dimensional Principal Component Analysis

CHAPTER 1

Introduction

1.1 Background

Development of more sophisticated and effective techniques for human identification is becoming increasingly important in view of their applications in security, law-enforcement and verification [1]-[6]. For human identification, there are several biometric techniques, which use specific information to classify the subjects. Such specific information could be fingerprint, palmprint, signature, voice, iris, deoxyribonucleic acid (DNA), face, etc. All these types of information are unique and vary from one person to another. Many security services, airports and government agencies use biometrics for identification and security purposes.

Human face recognition is one of the most popular techniques used in biometrics. Face recognition systems are widely used for identification, security, law enforcement, etc [7]-[9]. Also, face recognition systems are indispensable in cases such as when the police officers have only the face information of a criminal. The importance of security and identification necessitates stable, fast and highly accurate algorithms for face recognition. Hence, new approaches have been proposed that are more advanced than the preceding ones with higher recognition accuracy and lower complexity under specific real-life conditions.

1.2 Face Recognition Systems

The goal of face recognition systems is to determine as to which face image belongs to which person (subject) [7]-[11]. Raw images for a face recognition system are grouped as training images and test images. The subject of each training image is already known to the system, but for the test images, subjects are assigned through a classification process. Typically, a face recognition system consists of data acquisition, feature extraction and classification subsystems [7]-[11].

In a data acquisition subsystem, raw face images are first acquired and some pre-processing is carried out to improve the recognition accuracy or to decrease the time complexity of the recognition process. The resulting data are then employed by the feature extraction subsystem for extraction of features. In the feature extraction subsystem, each image is represented as a set of features, which are signatures to characterize the data. The main advantage of this representation lies in reducing the data size in comparison to size of the original data. Also, features are useful to discriminate the details between different subjects and to determine the common characteristics of different images of the same subject. In the literature of face recognition systems, different techniques for feature extraction, such as principal component analysis (PCA) [10] and its variants, row-directional two-dimensional PCA (r2DPCA) [12], column-directional 2DPCA (c2DPCA) [13], two-directional two-dimensional PCA ($(2D)^2$PCA) [13], exist. PCA and its variants are well-known for their robustness and stability against illumination variation, noise and changes in intra-class pose details within tolerable limits.

Classification subsystem is the final module of a face recognition system. In this part, the subject of each test image is determined. There are different approaches, such as the nearest neighbour classifier, neural networks and hidden Markov model (HMM), for classification. The nearest neighbour classifier is one of the most popular classification approaches in the literature [10]-[13], [16], [18], [19], [26], [27]. In this classifier, a distance metric is used to measure the distance between the feature set of a test image and that of each of the training images. Then, the subject of the training image with the shortest distance from the test image is assigned as the subject of the test image. The Euclidean, Frobenius and Yang distances are commonly-used metrics in the nearest neighbour classifiers to compute the distance between feature-sets [10]-[13].

1.3 A Brief Review of PCA-Based Face Recognition Systems

Principal component analysis, also known as the discrete Karhunen-Loève transform, is a linear transformation with a strong energy compaction property that allows high-dimensional data to be represented compactly with a much smaller number of coefficients [10], [14], [15]. In the classical PCA technique, each face image is represented as a single point in a very high-dimensional space by concatenating a two-dimensional face image into a one-dimensional vector. In the feature extraction process, a transformation matrix is first computed using all the training image vectors, then the feature set of a test or training image is obtained using this transformation matrix. Finally, using these feature sets, a test image is classified by employing a suitable classification technique. Although the PCA technique has a strong energy compaction property to represent face images efficiently, it has significant drawbacks of (a) losing image details due to concatenation that affects the accuracy and (b) having a large time complexity due to the use of a large-size covariance matrix employed in the feature extraction module.

In 2004, Yang et al. proposed the r2DPCA technique [12] with an objective to overcome the drawbacks of PCA. The r2DPCA technique works without concatenation of the two-dimensional data. It directly employs the two-dimensional data for feature extraction. Thus, the loss of image details resulting from the concatenation process is significantly reduced. Consequently, the recognition accuracy gets improved in comparison to the conventional PCA. The covariance matrix employed in r2DPCA is smaller than the one employed in PCA. Since the computational complexity of the feature extraction process depends mainly on the size of the covariance matrix, the r2DPCA technique has a lower computational complexity than PCA does [12]. However, the r2DPCA approach, which utilizes the original face images for feature extraction, is not sufficiently pose-tolerant against intra-class pose variations.

Although significant improvements have been achieved using the r2DPCA technique, challenging real-life conditions require for more robust algorithms. One of the most common real-life problems for face recognition is intra-class pose-variation that has a negative effect on the recognition accuracy. To solve the intra-class pose-variation problem, Bhagavatula and Savvides [16] have proposed a Fourier magnitude PCA (FM-PCA) approach in 2005. In the FM-PCA approach, the Fourier magnitudes are employed for the feature extraction step of the PCA algorithm. Use of the Fourier magnitudes for feature extraction makes the algorithm more pose tolerant and the recognition accuracy is also significantly increased for face databases having pose variations. However, computing the Fourier magnitudes increases the time complexity.

During the same year that FM-PCA was proposed [16], Zhang and Zhou presented a modified version of r2DPCA addressing one of its problems, namely, that it takes into consideration only the variation in the information between the rows and not between the columns. They developed a two-directional two-dimensional PCA ($(2D)^2$PCA) technique [13] by combining r2DPCA [12] and c2DPCA [13], which is a column version of r2DPCA. The $(2D)^2$PCA algorithm takes into consideration not only the variations in the information between rows, but also that between columns. As a result, there is a very slight improvement in the recognition accuracy of $(2D)^2$PCA over that of r2DPCA, when the algorithm is run using the ORL database [17]. However, the $(2D)^2$PCA algorithm has a significant advantage over r2DPCA in terms of the time complexity in view of the smaller size of the feature matrix of the former.

In addition to the r2DPCA and $(2D)^2$PCA algorithms in the 2-D spatial domain, there are two other algorithms, namely, diagonal PCA and Diagonal PCA + r2DPCA [18] that have been reported in the literature. However, the recognition accuracies of these two algorithms, using ORL database, have been reported to be generally lower than that of the r2DPCA algorithm, in addition to the time complexities being higher than that of the r2DPCA algorithm [19]. In view of these results, these two algorithms will not be further pursued in this thesis.

1.4 Scope and Objective of the Thesis

From the brief review in Section 1.3 of the classical PCA algorithm and its derivatives, it is seen that the $(2D)^2$PCA algorithm developed by Zhang and Zhou [13] by modifying r2DPCA reduces the complexity, with the recognition accuracy remaining about the same. On the other hand, the FM-PCA algorithm developed by Bhagavatula and Savvides [16] provides a significantly higher accuracy than that of PCA, but with a computational complexity that is higher than that of PCA due to the cost of computing the Fourier magnitudes of the pixels of the image.

This thesis is aimed at developing two-dimensional PCA-based algorithms that incorporate the advantages of the FM-PCA and two-dimensional PCA algorithms. Specifically, a study is undertaken to first develop the Fourier magnitude (FM) version of the r2DPCA and $(2D)^2$PCA algorithms. Then, by taking advantage of the energy compaction property of the Fourier transform, the FM versions of both the r2DPCA and $(2D)^2$PCA algorithms are further modified to significantly reduce their computational complexities.

1.5 Organization of the Thesis

This thesis is organized as follows.

In Chapter 2, a brief review of the PCA algorithm and its subsequent modifications in the spatial domain, r2DPCA [12] and $(2D)^2$PCA [13], is presented. This chapter brings out the shortcomings of these algorithms and provides the background material necessary for the development of the work undertaken in the thesis.

In Chapter 3, FM-PCA [16] is first examined along with extensive simulations of PCA and FM-PCA using the ORL face database in which the intra-class pose variations are substantial. It is verified that the considerable improvement in the recognition accuracy is obtained at the expense of the overhead for computing the Fourier magnitudes. It is shown that the FM-PCA algorithm inherits the drawbacks of the PCA algorithm, namely, the high computational complexity in the feature extraction due to the use of very large size covariance matrix and the loss of image details resulting from the conversion of two-dimensional data into one-dimensional vectors. In order to explore the possibility of improving the accuracy as well as reducing the time complexity as compared to that of FM-PCA, two new algorithms are proposed by applying the Fourier magnitude concept to r2DPCA and its modified version $(2D)^2$PCA, which have been reported in the literature to have higher recognition accuracy and lower time complexity over PCA algorithm. Finally, extensive simulations for the r2DPCA and $(2D)^2$PCA algorithms as well as for the proposed FM-r2DPCA and FM-$(2D)^2$PCA algorithms are carried out using the ORL face database to compare the proposed algorithms with their spatial domain versions and with the FM-PCA algorithm.

Unlike Chapter 3, in which the feature extraction and classification processes are carried out based on the entire set of Fourier magnitude coefficients, in Chapter 4, using the energy compaction property of Fourier magnitudes, only a small subset of the entire Fourier magnitude coefficients, namely, the low-pass Fourier magnitude (LPFM) coefficients, are used for this purpose leading to the development of new algorithms with reduced computational complexity. Specifically, LPFM versions of the r2DPCA, and $(2D)^2$PCA algorithms are developed. Simulations are carried out to demonstrate the effectiveness of the LPFM approach over their FM counterparts in significantly decreasing the computational complexity with almost the same recognition accuracy.

In Chapter 5, the thesis is concluded by summarizing and highlighting the main contributions of the study undertaken therein. Some suggestions for further work to solve the real-life problems in face recognition are also provided.

CHAPTER 2

Review of 1-D and 2-D PCA Algorithms in the Spatial Domain

2.1 Introduction

This chapter gives a brief review of the classical PCA algorithm and its derivatives in the spatial domain in order to provide the basic ideas used in their development and to provide the background material essential for the development and solutions of the problem undertaken in this thesis. The derivatives of the PCA algorithm considered are the r2DPCA and $(2D)^2$PCA algorithms. A brief discussion of the Euclidean, Frobenius and Yang distance metrics that are generally used in classifiers of PCA-based face recognition algorithms is also included.

2.2 Principal Component Analysis

The use of PCA as a feature extraction technique is popular in pattern recognition because of its strong energy compaction property and its robustness against the data acquired under reasonably different conditions. In 1987, Sirovich and Kirby [14] proposed the idea of utilizing the principal components of the distribution of faces to characterize the variations between them. In 1991, Turk and Pentland [10] utilized the PCA technique for face recognition because of its efficiency and capability to distinguish the patterns of the subjects using a small number of coefficients. A description of the PCA algorithm follows.

A face image is basically a two-dimensional array ***I*** of size $M \times N$ given by

$$\mathbf{I} = \begin{bmatrix} \mathrm{I}(0,0) & \cdots & \mathrm{I}(0, \mathrm{N}-1) \\ \vdots & \vdots & \vdots \\ \mathrm{I}(\mathrm{M}-1,0) & \cdots & \mathrm{I}(\mathrm{M}-1, \mathrm{N}-1) \end{bmatrix} \tag{2.1}$$

In the PCA method, each face image is first concatenated by arranging the elements of ***I*** taken row-by-row (or alternatively column-by-column) into a column vector given by

$$\mathbf{\Gamma} = \begin{bmatrix} \mathrm{I}(0,0) \\ \vdots \\ \mathrm{I}(\mathrm{M}-1,0) \\ \mathrm{I}(1,0) \\ \vdots \\ \mathrm{I}(1, \mathrm{N}-1) \\ \vdots \\ \mathrm{I}(\mathrm{M}-1,1) \\ \vdots \\ \mathrm{I}(\mathrm{M}-1, \mathrm{N}-1) \end{bmatrix}_{\mathrm{MN}\times 1} \tag{2.2}$$

The average of K concatenated training samples computed as

$$\mathbf{\Psi} = \frac{1}{\mathrm{K}} \sum_{\mathrm{i}=1}^{\mathrm{K}} \mathbf{\Gamma}^{(\mathrm{i})} \tag{2.3}$$

is subtracted from each concatenated training sample giving

$$\mathbf{\Phi}^{(\mathrm{i})} = \mathbf{\Gamma}^{(\mathrm{i})} - \mathbf{\Psi} \tag{2.4}$$

Then, the mean-subtracted training samples are used to construct a matrix given by

$$\mathbf{T} = [\mathbf{\Phi}^{(1)} \quad \mathbf{\Phi}^{(2)} \quad \ldots \quad \mathbf{\Phi}^{(\mathrm{K})}] = \begin{bmatrix} \Phi^{(1)}(0) & \cdots & \Phi^{(\mathrm{K})}(0) \\ \vdots & \vdots & \vdots \\ \Phi^{(1)}(\mathrm{MN}-1) & \cdots & \Phi^{(\mathrm{K})}(\mathrm{MN}-1) \end{bmatrix} \tag{2.5}$$

where K is the number of samples in the training set. Using ***T***, the covariance matrix is obtained as

$$\mathbf{C} = \mathbf{T}\mathbf{T}^{\mathrm{T}} \tag{2.6}$$

The covariance matrix, which actually gives the relationship between the samples, is a square matrix of size $MN \times MN$. In order to obtain the principal components of the samples in the training set, the eigenvectors of the covariance matrix are computed. Because of the large size of $\boldsymbol{C}$, a direct computation of its eigenvalues and eigenvectors is practically not feasible [10]. For example, the ORL face database has 40 subjects each having 10 samples and the size of each sample is $M \times N = 112 \times 92$ [17]. If the training set consists of $K = 200$ images, the size of $\boldsymbol{C}$ becomes $MN \times MN = 10304 \times 10304$ with $MN = 10304$ eigenvalues and eigenvectors. Thus, if these eigenvalues and eigenvectors given by (2.6) are computed directly, it would result in a huge computational burden.

Fortunately, there is a remedy to this problem. Each eigenvector corresponds to a different amount of variation among the face images, in that an eigenvector of $\boldsymbol{C}$ corresponding to a large eigenvalue is associated with large variations in the face images. Thus, since the training set has only K samples ($K \ll MN$), no more than $K - 1$ eigenvectors are meaningful [10]. These meaningful eigenvectors (also called standard images, eigenpictures, eigenfaces, or eigenimages) are obtained by eigen-decomposing the matrix

$$\mathbf{D} = \mathbf{T}^{\mathrm{T}}\mathbf{T} \tag{2.7}$$

as

$$\mathbf{T}^{\mathrm{T}}\mathbf{T}\mathrm{d}_{\mathrm{j}} = \mathrm{e}_{\mathrm{j}}\mathrm{d}_{\mathrm{j}} \tag{2.8}$$

where e_j is an eigenvalue of $\boldsymbol{D}$ and d_j is the corresponding eigenvector. Pre-multiplying both sides of (2.8) by $\boldsymbol{T}$ yields

$$\mathbf{T}\mathbf{T}^{\mathrm{T}}\mathbf{T}\mathrm{d_j} = \mathbf{T}\mathrm{e_j d_j} = \mathrm{e_j}(\mathbf{T}\mathrm{e_j d_j}) \tag{2.9}$$

$$\mathbf{T}\mathbf{T}^{\mathrm{T}}(\mathbf{T}\mathrm{d_j}) = \mathrm{e_j}(\mathbf{T}\mathrm{d_j}) \tag{2.10}$$

Thus, it is clear from above equation that the eigenvalue e_j of $\boldsymbol{D}$ is also an eigenvalue of $\boldsymbol{C}$ and the corresponding eigenvector of $\boldsymbol{C}$ is given as $\hat{u}_j = \boldsymbol{T}d_j$. The matrix $\boldsymbol{D}$ being a square matrix of size $K \times K$ has K eigenvalues and K eigenvectors. Thus, through this process all the meaningful eigenvectors of $\boldsymbol{C}$ are obtained with much less computational burden. Next, these eigenvectors, are normalized as $u_j = \frac{\hat{u}_j}{\|\hat{u}_j\|}$ and arranged in a sequence corresponding to the decreasing eigenvalues, and denoted by $u_j,\ j = 1,2,\dots,K$, respectively.

Since the objective of the PCA technique is to significantly reduce the size of the data, each image is represented by a feature vector $\mathbf{\Omega}$ of size that is much less than MN, the dimension of a training or test sample. A training image feature vector $\mathbf{\Omega}^{(\mathbf{i})}$ is obtained by projecting the mean-subtracted training sample $\mathbf{\Phi}^{(\mathbf{i})}$ onto the normalized and ordered eigenvectors $u_j,\ j = 1,2,\dots,L$, where L is the number of chosen normalized eigenvectors that sufficiently expose the variations in the face images. The feature vector $\boldsymbol{\Omega}^{(i)}$ corresponding to the i^{th} training image sample is computed as

$$\mathbf{\Omega}^{(\mathbf{i})} = \mathbf{Q}\mathbf{\Phi}^{(\mathbf{i})} \tag{2.11}$$

where

$$\mathbf{Q} = [\mathbf{u_1} \quad \mathbf{u_2} \quad \dots \quad \mathbf{u_L}]^{\mathbf{T}} \tag{2.12}$$

is called the transformation matrix. The size of each feature vector is $L \times 1$ and it can be expressed in terms of its coefficients as

$$\mathbf{\Omega}^{(\mathbf{i})} = \begin{bmatrix} \omega_1^{(i)} \\ \omega_2^{(i)} \\ \dots \\ \omega_L^{(i)} \end{bmatrix}_{L\times 1} \tag{2.13}$$

where the j^{th} element of the of the i^{th} training image's feature vector is given by $\omega_1^{(i)} = u_1^T \mathbf{\Phi}^{(\mathbf{i})}$.

Similarly, for a test image $\boldsymbol{I}^{(test)}$, the corresponding feature vector is obtained as

$$\mathbf{\Omega}^{(\mathbf{test})} = \mathbf{Q}(\mathbf{\Gamma}^{(\mathbf{test})} - \mathbf{\Psi}) \tag{2.14}$$

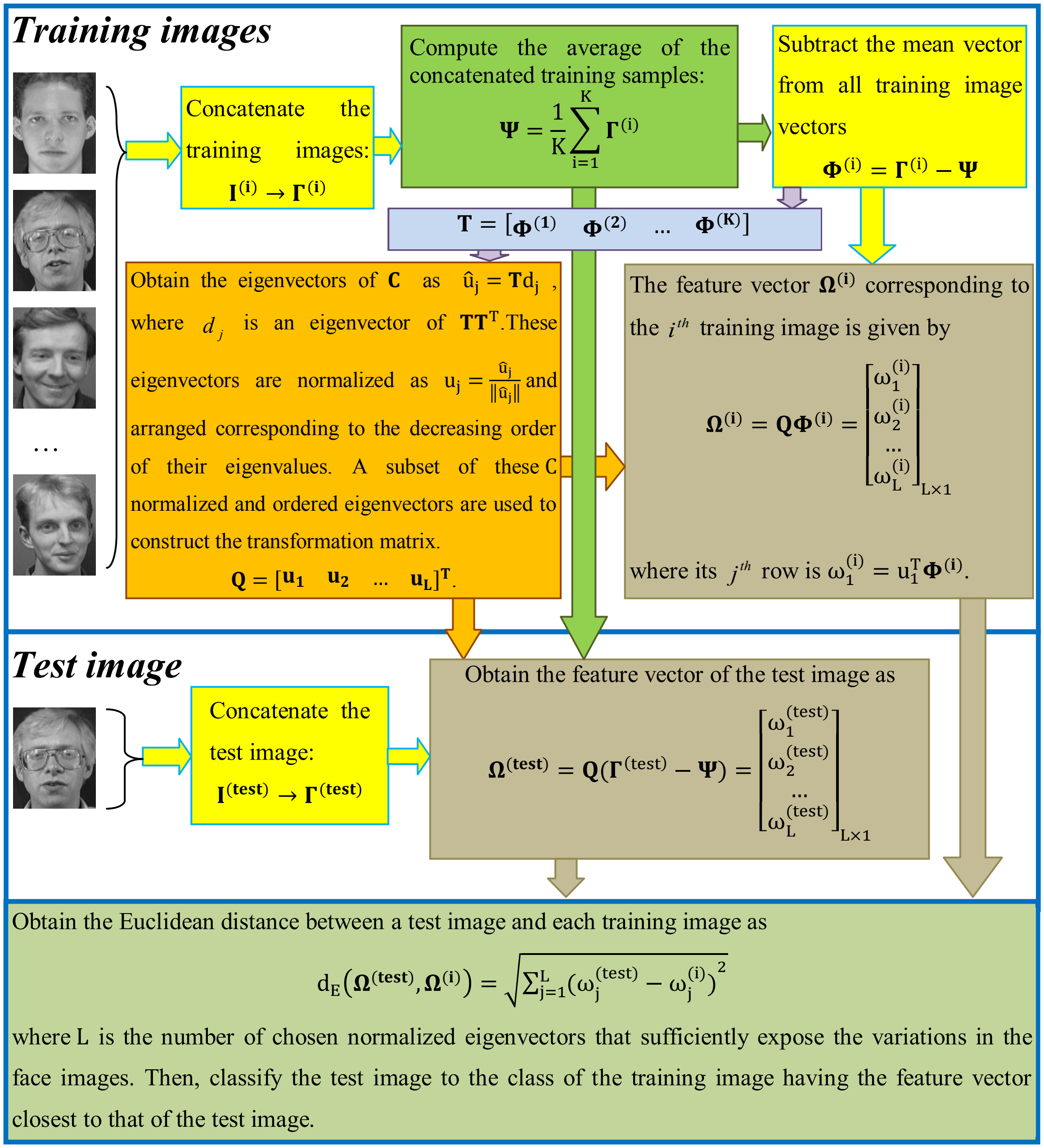
where $\mathbf{\Gamma}^{(\mathbf{test})}$ is the concatenated vector corresponding to the test image $\boldsymbol{I}^{(test)}$, $\boldsymbol{Q}$ is the transformation matrix given by (2.12) and $\boldsymbol{\Psi}$ is the mean vector given by (2.3). The feature vector of the test image is given by

$$\mathbf{\Omega}^{(\mathbf{test})} = \begin{bmatrix} \omega_1^{(test)} \\ \omega_2^{(test)} \\ \dots \\ \omega_L^{(test)} \end{bmatrix}_{L\times 1} \tag{2.15}$$

In the classification module of the PCA algorithm [10], Euclidean distance is commonly used to measure the distance between the feature vector of a test image and that of a training image. The Euclidean distance between the feature vector of a test and that of each training image is computed as

$$d_E\left(\mathbf{\Omega}^{(\mathbf{test})}, \mathbf{\Omega}^{(\mathbf{i})}\right) = \sqrt{\sum_{j=1}^{L} (\omega_j^{(test)} - \omega_j^{(i)})^2} \tag{2.16}$$

where $\omega_j^{(test)}$ and $\omega_j^{(i)}$ are, respectively, the j^{th}elements of the test image feature vector $\boldsymbol{\Omega}^{(test)}$ and the i^{th} training image feature vector $\boldsymbol{\Omega}^{(i)}$. Then, the subject of the training image corresponding to the feature vector $\boldsymbol{\Omega}^{(i)}$ that has the shortest distance from the test image feature vector $\boldsymbol{\Omega}^{(test)}$ is assigned as the subject of the test image. The block diagram of Fig.2.1 summarizes the PCA algorithm.



Training images

Concatenate the training images: $\mathbf{I}^{(\mathrm{i})} \rightarrow \mathbf{\Gamma}^{(\mathrm{i})}$

Compute the average of the concatenated training samples:

$$\mathbf{\Psi} = \frac{1}{\mathrm{K}}\sum_{\mathrm{i}=1}^{\mathrm{K}} \mathbf{\Gamma}^{(\mathrm{i})}$$

Subtract the mean vector from all training image vectors

$$\mathbf{\Phi}^{(\mathrm{i})} = \mathbf{\Gamma}^{(\mathrm{i})} - \mathbf{\Psi}$$

$$\mathbf{T} = [\mathbf{\Phi}^{(1)} \quad \mathbf{\Phi}^{(2)} \quad \dots \quad \mathbf{\Phi}^{(\mathrm{K})}]$$

Obtain the eigenvectors of $\mathbf{C}$ as $\hat{\mathrm{u}}_{\mathrm{j}} = \mathbf{T}\mathrm{d}_{\mathrm{j}}$, where d_j is an eigenvector of $\mathbf{TT}^{\mathrm{T}}$.These eigenvectors are normalized as $\mathrm{u}_{\mathrm{j}} = \frac{\hat{\mathrm{u}}_{\mathrm{j}}}{\|\hat{\mathrm{u}}_{\mathrm{j}}\|}$ and arranged corresponding to the decreasing order of their eigenvalues. A subset of these C normalized and ordered eigenvectors are used to construct the transformation matrix.

$$\mathbf{Q} = [\mathbf{u_1} \quad \mathbf{u_2} \quad \dots \quad \mathbf{u}_{\mathrm{L}}]^{\mathbf{T}}.$$

The feature vector $\mathbf{\Omega}^{(\mathrm{i})}$ corresponding to the i^{th} training image is given by

$$\mathbf{\Omega}^{(\mathrm{i})} = \mathbf{Q\Phi}^{(\mathrm{i})} = \begin{bmatrix} \omega_1^{(\mathrm{i})} \\ \omega_2^{(\mathrm{i})} \\ \dots \\ \omega_{\mathrm{L}}^{(\mathrm{i})} \end{bmatrix}_{\mathrm{L}\times1}$$

where its j^{th} row is $\omega_1^{(\mathrm{i})} = \mathrm{u}_1^{\mathrm{T}}\mathbf{\Phi}^{(\mathrm{i})}$.

Test image

Concatenate the test image: $\mathbf{I}^{(\mathbf{test})} \rightarrow \mathbf{\Gamma}^{(\mathbf{test})}$

Obtain the feature vector of the test image as

$$\mathbf{\Omega}^{(\mathbf{test})} = \mathbf{Q}(\mathbf{\Gamma}^{(\mathrm{test})} - \mathbf{\Psi}) = \begin{bmatrix} \omega_1^{(\mathrm{test})} \\ \omega_2^{(\mathrm{test})} \\ \dots \\ \omega_{\mathrm{L}}^{(\mathrm{test})} \end{bmatrix}_{\mathrm{L}\times1}$$

Obtain the Euclidean distance between a test image and each training image as

$$\mathrm{d}_{\mathrm{E}}\left(\mathbf{\Omega}^{(\mathbf{test})}, \mathbf{\Omega}^{(\mathrm{i})}\right) = \sqrt{\sum_{\mathrm{j}=1}^{\mathrm{L}}(\omega_{\mathrm{j}}^{(\mathrm{test})} - \omega_{\mathrm{j}}^{(\mathrm{i})})^2}$$

where L is the number of chosen normalized eigenvectors that sufficiently expose the variations in the face images. Then, classify the test image to the class of the training image having the feature vector closest to that of the test image.

Fig.2.1: Summary of the PCA algorithm.

2.3 Row-Directional Two-Dimensional PCA

In order to decrease the computational complexity of the PCA technique, methods have been developed in which the DCT or DWT coefficients, instead of the raw pixel coefficients of the image, are used for the feature extraction [20]-[24]. In these methods, the recognition accuracy of the PCA method is also somewhat improved by decreasing the effects of noise and illumination variations because of the use of transformed coefficients. However, the high computational complexity due to the large size covariance matrix and low recognition accuracy resulting from the concatenation of the images still remain a matter of concern of the PCA techniques using transformed data.

In 2004, Yang et al. proposed a two-dimensional PCA algorithm [12], which directly employs the two-dimensional data for feature extraction without concatenation. Consequently, the face images are less distorted, and the image details are better preserved in comparison to the PCA algorithm [12], [13]. They also modified the covariance matrix with an objective of decreasing the time complexity and increasing the recognition accuracy. The normalized eigenvectors of this covariance matrix are used to construct a linear transformation matrix. A face image matrix is then projected onto this transformation matrix row-by-row in order to obtain the feature matrix, which represents the face image better than the feature vector obtained in the PCA algorithm. A description of this 2DPCA algorithm, which henceforth will be referred to as the row-directional 2DPCA (r2DPCA) algorithm, follows.

The modified covariance matrix is computed as

$$\mathbf{C}_{\mathbf{r2DPCA}} = \frac{1}{\mathrm{K}}\sum_{\mathrm{i}=1}^{\mathrm{K}}(\mathbf{I}^{(\mathbf{i})} - \bar{\mathbf{I}})^{\mathrm{T}}(\mathbf{I}^{(\mathbf{i})} - \bar{\mathbf{I}}) \qquad (2.17)$$

where $\boldsymbol{I}^{(i)}$ is the i^{th} training image of size $M \times N$, K is the number of training images and $\bar{\boldsymbol{I}}$ is the average of the training samples computed as

$$\bar{\mathbf{I}} = \frac{1}{\mathrm{K}}\sum_{\mathrm{i}=1}^{\mathrm{K}} \mathbf{I}^{(\mathbf{i})} \tag{2.18}$$

The size of the resulting covariance matrix is $N \times N$. Next, N eigenvalues and the corresponding eigenvectors of $\boldsymbol{C}_{r2DPCA}$ are computed. The eigenvectors are normalized and arranged in a sequence $\boldsymbol{X}_j$, $j = 1,2, \dots, N$, corresponding to the decreasing eigenvalues. Similar to the PCA algorithm, the data is compressed by using a transformation matrix constructed using a subset of the normalized eigenvectors of $\boldsymbol{C}_{r2DPCA}$ as

$$\mathbf{X} = \left[\mathbf{X_1 X_2} \dots \mathbf{X_\beta}\right] \tag{2.19}$$

where $\beta << N$ is the number of chosen normalized eigenvectors, which sufficiently expose the variations among the face images. In order to obtain the feature matrix of the i^{th} training image $\boldsymbol{I}^{(i)}$, it is projected onto the transformation matrix as

$$\mathbf{Z}^{(\mathbf{i})} = \left[\mathbf{Z}_{\mathbf{1}}^{(\mathbf{i})}\mathbf{Z}_{\mathbf{2}}^{(\mathbf{i})} \dots \mathbf{Z}_{\boldsymbol{\beta}}^{(\mathbf{i})}\right] = \mathbf{I}^{(\mathbf{i})}\mathbf{X} \tag{2.20}$$

In the above equation, the elements of the j^{th} column $\boldsymbol{Z}_{\boldsymbol{j}}^{(i)}$ of the i^{th} feature matrix is obtained by projecting the i^{th} training image row-by-row onto the j^{th} projective vector of the transformation matrix, i.e., the j^{th} column vector of the i^{th} feature matrix is obtained as $\boldsymbol{Z}_{\boldsymbol{j}}^{(i)} = \boldsymbol{I}^{(i)}\boldsymbol{X}_j$, where $j = 1,2, \dots, \beta$ and $i = 1,2, \dots, K$. Similarly, the feature matrix of a test image $\boldsymbol{I}^{(test)}$ is computed by projecting it onto the transformation matrix as

$$\mathbf{Z}^{(\mathbf{test})} = \mathbf{I}^{(\mathbf{test})}\mathbf{X} \tag{2.21}$$

where $\boldsymbol{X}$ is the transformation matrix as obtained in (2.19). Each column vector of $\boldsymbol{Z}^{(i)}$ or $\boldsymbol{Z}^{(test)}$ obtained in (2.20) or (2.21) is called the principal component vector of the sample image $\boldsymbol{I}^{(i)}$ or $\boldsymbol{I}^{(test)}$ [12]. It is noted that each principal component of a two-dimensional PCA is a vector, while each principal component of PCA is a scalar.

In the classification module of the r2DPCA algorithm [12], and other two-dimensional face recognition schemes [13], [24], matrix similarity measures, instead of the Euclidean distance, are used. After obtaining the distance between the feature matrix of a test image and that of each training image, the subject of the training image whose feature matrix has the shortest distance from the test image feature matrix $\boldsymbol{Z}^{(test)}$, is assigned the subject of the test image. The Frobenius and Yang distances are commonly used matrix similarity measures in the literature for two-dimensional face recognition schemes [12], [13], [18], [19], [26]. The classifiers using these similarity measures are called as nearest neighbour classifiers. Frobenius distance metric is an adapted form of the Euclidean distance metric for the classification of the samples, which are represented as two-dimensional feature sets; it is given by

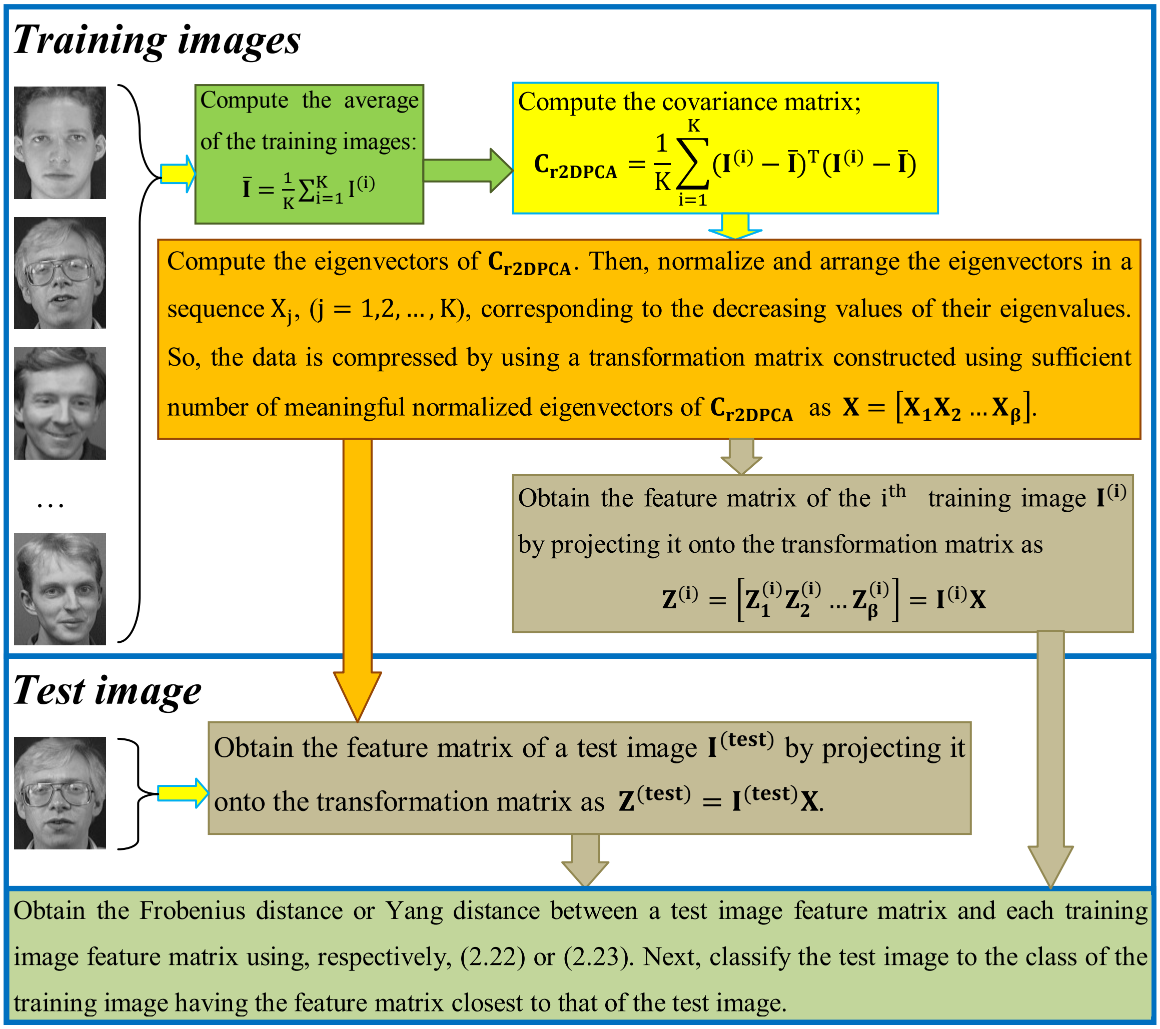
$$\mathrm{d_F}(\mathbf{Z}^{(\mathbf{test})}, \mathbf{Z}^{(\mathbf{i})}) = \sqrt{\sum_{\mathrm{j}=1}^{\gamma} \sum_{\mu=1}^{\rho} (\mathrm{Z}^{(\mathrm{test})}(\mu, \mathrm{j}) - \mathrm{Z}^{(\mathrm{i})}(\mu, \mathrm{j}))^2} \tag{2.22}$$

where $\mathbf{Z} = [\mathrm{Z}(\mu, \mathrm{j})]_{\rho\times\gamma}$. Yang distance [12], which provides a higher recognition accuracy than the Frobenius distance does, is obtained by summing the Euclidean distance between the respective columns of the test and training image feature matrices and is given by

$$\mathrm{d_Y}\left(\mathbf{Z}^{(\mathbf{test})}, \mathbf{Z}^{(\mathbf{i})}\right) = \sum_{\mathrm{j}=1}^{\gamma} \sqrt{\sum_{\mu=1}^{\rho} (\mathrm{Z}^{(\mathrm{test})}(\mu, \mathrm{j}) - \mathrm{Z}^{(\mathrm{i})}(\mu, \mathrm{j}))^2} \tag{2.23}$$

The r2DPCA technique increases the recognition accuracy and decreases the computational complexity. However, it requires many more coefficients to represent the face images than the PCA technique does. In the r2DPCA, the size of the feature matrix for an $M \times N$ face image is $M \times \beta$, and thus has many more coefficients than the feature vector of size $L \times 1$ obtained in the PCA technique.

Also, in the r2DPCA technique, since the rows of the i^{th} face image are projected onto the vectors of the transformation matrix as $\boldsymbol{Z}^{(i)} = \boldsymbol{I}^{(i)}\boldsymbol{X}$, only the variations between the rows of a face image are projected while ignoring the variations between the columns. The block diagram of Fig.2.2 summarizes the r2DPCA algorithm.



Compute the average of the training images: $\bar{\mathbf{I}} = \frac{1}{K}\sum_{i=1}^{K} \mathrm{I}^{(i)}$

Compute the covariance matrix; $\mathbf{C}_{\mathbf{r2DPCA}} = \frac{1}{K}\sum_{i=1}^{K}(\mathbf{I}^{(i)} - \bar{\mathbf{I}})^{T}(\mathbf{I}^{(i)} - \bar{\mathbf{I}})$

Compute the eigenvectors of $\mathbf{C}_{\mathbf{r2DPCA}}$. Then, normalize and arrange the eigenvectors in a sequence X_j, $(j = 1,2, \dots, K)$, corresponding to the decreasing values of their eigenvalues. So, the data is compressed by using a transformation matrix constructed using sufficient number of meaningful normalized eigenvectors of $\mathbf{C}_{\mathbf{r2DPCA}}$ as $\mathbf{X} = [\mathbf{X}_1 \mathbf{X}_2 \dots \mathbf{X}_\beta]$.

Obtain the feature matrix of the i^{th} training image $\mathbf{I}^{(i)}$ by projecting it onto the transformation matrix as $\mathbf{Z}^{(i)} = \left[\mathbf{Z}_1^{(i)} \mathbf{Z}_2^{(i)} \dots \mathbf{Z}_\beta^{(i)}\right] = \mathbf{I}^{(i)}\mathbf{X}$

Obtain the feature matrix of a test image $\mathbf{I}^{(\mathbf{test})}$ by projecting it onto the transformation matrix as $\mathbf{Z}^{(\mathbf{test})} = \mathbf{I}^{(\mathbf{test})}\mathbf{X}$.

Obtain the Frobenius distance or Yang distance between a test image feature matrix and each training image feature matrix using, respectively, (2.22) or (2.23). Next, classify the test image to the class of the training image having the feature matrix closest to that of the test image.

Fig.2.2: Summary of the r2DPCA algorithm.

2.4 Two-Directional Approach for 2DPCA

The row-directional 2DPCA technique [12] discussed in the previous section improves the recognition accuracy and reduces the computational complexity in comparison to the PCA technique, but it projects the variations only between the pixels of the rows of a face image and ignores the variations of those between the columns. Zhang and Zhou [13] in 2005 proposed a two-directional approach for 2DPCA in which the variations of the image details between the rows as well as the columns are taken into consideration. More specifically, they developed a technique that combines r2DPCA with a PCA technique that emphasizes the variations of the pixels between the columns. The latter is essentially a column alternative of the former. In the following, a brief description of the column alternative of the r2DPCA algorithm followed by a description of the two-directional 2DPCA is provided.

2.4.1 Column-Directional 2DPCA

In the column-directional 2DPCA technique (c2DPCA), column-directional variations of a face image are emphasized [13].

The covariance matrix is computed as

$$\mathbf{C_{c2DPCA}} = \frac{1}{\mathrm{K}}\sum_{\mathrm{i}=1}^{\mathrm{K}}(\mathbf{I}^{(\mathbf{i})} - \bar{\mathbf{I}})(\mathbf{I}^{(\mathbf{i})} - \bar{\mathbf{I}})^{\mathrm{T}} \quad (2.24)$$

where K is the number of the training images and $\bar{\boldsymbol{I}} = \frac{1}{K}\sum_{i=1}^{K}\boldsymbol{I}^{(i)}$ is the average of the training images. The size of this covariance matrix is $M \times M$. Next, M eigenvalues and the corresponding eigenvectors of $\boldsymbol{C_{c2DPCA}}$ are computed. The eigenvectors are arranged and normalized as described in Section 2.3. The transformation matrix is constructed using a sufficient number of normalized eigenvectors of $\boldsymbol{C_{c2DPCA}}$ as

$$\mathbf{Y} = [\mathbf{Y_1 Y_2} \dots \mathbf{Y_\alpha}] \tag{2.25}$$

where $\alpha \ll M$.

The feature matrix for each training image is obtained as

$$\mathbf{Z^{(i)}} = \left[\mathbf{Z_1^{(i)} Z_2^{(i)}} \dots \mathbf{Z_\alpha^{(i)}}\right] = \mathbf{Y^T I^{(i)}} \tag{2.26}$$

In (2.26), the elements of the j^{th} column $\boldsymbol{Z_j^{(i)}}$ of the i^{th} feature matrix is obtained by projecting the i^{th} training image column-by-column onto the j^{th} projective vector of the transformation matrix $\boldsymbol{Y}$, that is the j^{th} column vector of the i^{th} feature matrix is obtained as $\boldsymbol{Z_j^{(i)}} = \boldsymbol{I^{(i)}}^{\boldsymbol{T}} \boldsymbol{Y_j}$, where $j = 1,2,\dots,\alpha$ and $i = 1,2,\dots,K$. The feature matrix of a test image $\boldsymbol{I^{(test)}}$ is computed by projecting it onto the transformation matrix column-by-column as

$$\mathbf{Z^{(test)}} = \mathbf{Y^T I^{(test)}} \tag{2.27}$$

Similar to the r2DPCA technique, a nearest neighbour classifier can be used to classify the test images as described in Section 2.3. As in the case of r2DPCA, the c2DPCA technique also has a higher accuracy and lower complexity in comparison to the PCA technique.

2.4.2 Two-Directional Two-Dimensional PCA

As pointed out earlier, the two 2DPCA algorithms discussed so far project the variations of the information only between the rows or only between the columns of a face image. The size of the resulting feature matrix is very large. In order to obtain a more efficient feature representation, Zhang and Zhou [13] have proposed to take the variations of the image details between the rows as well as that between the columns into consideration by combining r2DPCA with c2DPCA. A brief description of the two-directional 2DPCA ($(2D)^2$PCA) is provided below.

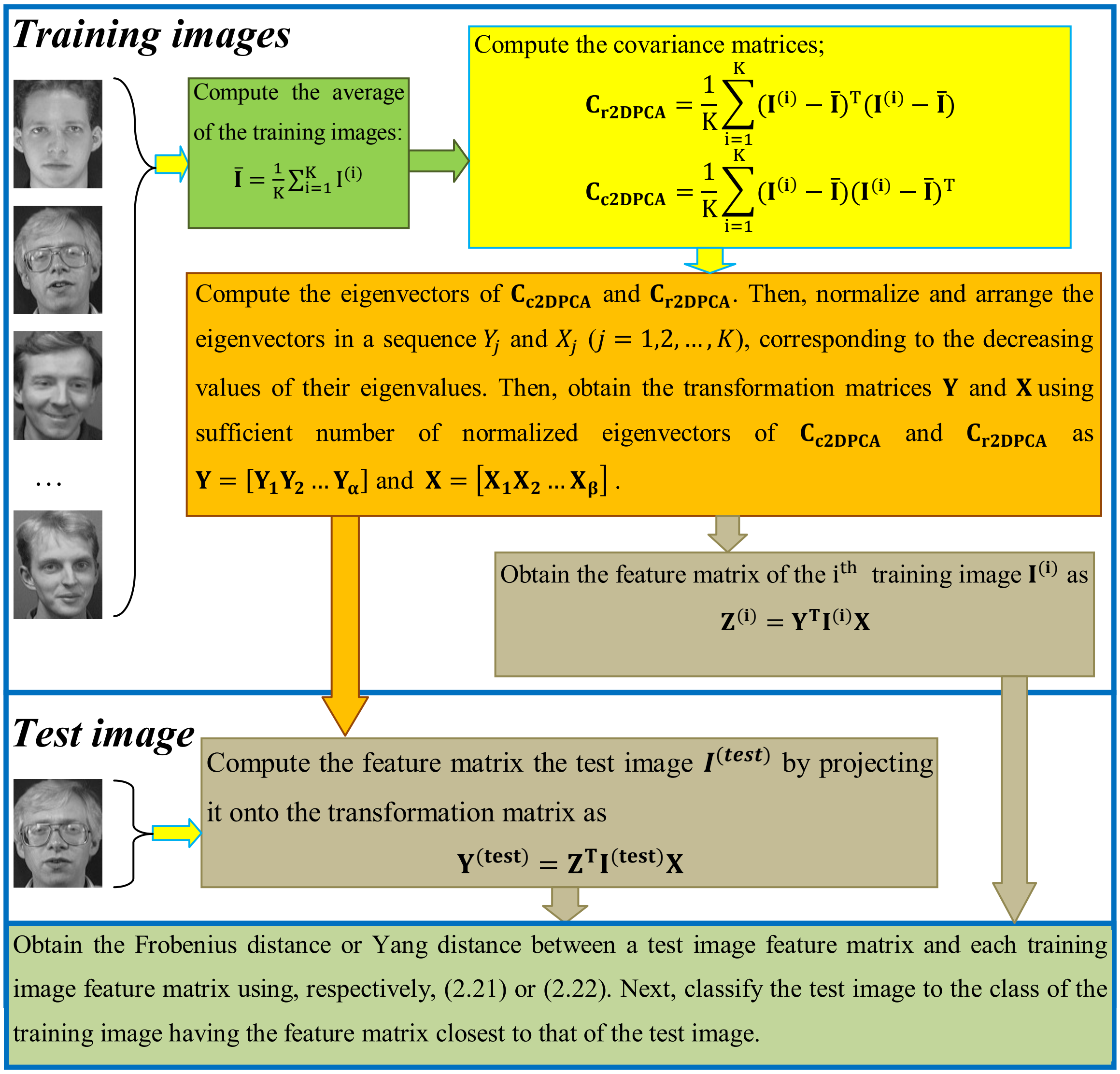
In the $(2D)^2$PCA technique, the transformation matrices $\boldsymbol{X}$ and $\boldsymbol{Y}$ given by (2.19) and (2.25) are first obtained, as described in Sections 2.3 and 2.4.1. Next, the feature matrix for each training image is computed as

$$\mathbf{Z}^{(\mathbf{i})} = \mathbf{Y}^{\mathbf{T}}\mathbf{I}^{(\mathbf{i})}\mathbf{X} \tag{2.28}$$

The size of the feature matrix $\alpha \times \beta$, where $\alpha \ll M$ and $\beta \ll N$, is chosen to sufficiently expose the variations among the face images. Similarly, the feature matrix of a test image is computed as

$$\mathbf{Z}^{(\mathbf{test})} = \mathbf{Y}^{\mathbf{T}}\mathbf{I}^{(\mathbf{test})}\mathbf{X} \tag{2.29}$$

Similar to the r2DPCA or c2DPCA technique, a nearest neighbour classifier can be used to classify the test images, as described earlier. In the $(2D)^2$PCA technique, the number of coefficients in the feature matrix for image representation is significantly reduced in comparison to the r2DPCA or c2DPCA technique. Although in the $(2D)^2$PCA technique a more efficient face representation is obtained, still it is sensitive to intra-class pose variations. A summary of the $(2D)^2$PCA algorithm is given in Fig.2.3.



Compute the average of the training images: $\bar{\mathbf{I}} = \frac{1}{K}\sum_{i=1}^{K}\mathbf{I}^{(i)}$

Compute the covariance matrices;

$$\mathbf{C}_{r2DPCA} = \frac{1}{K}\sum_{i=1}^{K}(\mathbf{I}^{(i)} - \bar{\mathbf{I}})^{T}(\mathbf{I}^{(i)} - \bar{\mathbf{I}})$$

$$\mathbf{C}_{c2DPCA} = \frac{1}{K}\sum_{i=1}^{K}(\mathbf{I}^{(i)} - \bar{\mathbf{I}})(\mathbf{I}^{(i)} - \bar{\mathbf{I}})^{T}$$

Compute the eigenvectors of $\mathbf{C}_{c2DPCA}$ and $\mathbf{C}_{r2DPCA}$. Then, normalize and arrange the eigenvectors in a sequence Y_j and X_j $(j = 1,2,\ldots,K)$, corresponding to the decreasing values of their eigenvalues. Then, obtain the transformation matrices $\mathbf{Y}$ and $\mathbf{X}$ using sufficient number of normalized eigenvectors of $\mathbf{C}_{c2DPCA}$ and $\mathbf{C}_{r2DPCA}$ as $\mathbf{Y} = [\mathbf{Y}_1\mathbf{Y}_2 \ldots \mathbf{Y}_\alpha]$ and $\mathbf{X} = [\mathbf{X}_1\mathbf{X}_2 \ldots \mathbf{X}_\beta]$.

Obtain the feature matrix of the i^{th} training image $\mathbf{I}^{(i)}$ as

$$\mathbf{Z}^{(i)} = \mathbf{Y}^{T}\mathbf{I}^{(i)}\mathbf{X}$$

Compute the feature matrix the test image $\boldsymbol{I}^{(test)}$ by projecting it onto the transformation matrix as

$$\mathbf{Y}^{(test)} = \mathbf{Z}^{T}\mathbf{I}^{(test)}\mathbf{X}$$

Obtain the Frobenius distance or Yang distance between a test image feature matrix and each training image feature matrix using, respectively, (2.21) or (2.22). Next, classify the test image to the class of the training image having the feature matrix closest to that of the test image.

Fig.2.3: Summary of the $(2D)^2$PCA algorithm.

2.5 Summary

In this chapter, a brief review of the classical PCA algorithm and its two-dimensional derivatives in the spatial domain, r2DPCA, c2DPCA and $(2D)^2$PCA, has been presented in order to provide the background material essential for the work undertaken in this thesis. Advantages and shortcomings of these techniques have been discussed in terms of the recognition accuracy and computational complexity. The PCA algorithm, which is the foundation for the development of many other algorithms, has been first introduced. A two dimensional version of the PCA technique, namely, r2DPCA that exposes the image details among the rows and has a lower computational complexity compared to that of PCA, has been presented. Next, the two-directional 2DPCA algorithm ($(2D)^2$PCA), which combines r2DPCA and its column-directional counterpart, c2DPCA, is briefly described. This algorithm, while exposing the image details between the rows as well as between the columns has a lower computational complexity in comparison to its two constituent algorithms.

CHAPTER 3

Two-Dimensional PCA Algorithms in the Fourier Domain

3.1 Introduction

As mentioned in the previous chapter, the classical PCA algorithm [10] is affected by intra-class pose variations. In an effort to overcome this problem, Bhagavatula and Savvides have proposed in [16] to apply the Fourier magnitudes of the pixels of the images for feature extraction in the PCA algorithm instead of the pixel values themselves. This resulted in considerable improvement in the recognition accuracy, but at the expense of the overhead for computing the Fourier magnitudes. As stated in the previous chapter, the two-dimensional extensions of the PCA algorithm, r2DPCA and $(2D)^2$PCA, provide higher accuracies with lower complexities as compared to PCA. Hence, we propose to employ the Fourier magnitudes for feature extraction in these two 2DPCA algorithms in order to explore the possibility of improving the accuracy as well as reducing the time complexity over that of FM-PCA. In this chapter, a brief description of the FM-PCA algorithm is given, followed by extensive simulation results, using the ORL database, to study the improvement in the recognition accuracy and the computational cost of the FM-PCA over the classical PCA. We then propose FM-versions of the two-dimensional r2DPCA and $(2D)^2$PCA algorithms. Finally, extensive simulations are carried out to study the effectiveness of applying the FM approach to the two-dimensional PCA algorithms.

3.2 Fourier-Magnitude PCA

3.2.1 A Brief Review

Although the PCA technique, discussed in Section 2.2, has been commonly used for feature extraction and data representation in the literature, it is sensitive to intra-class pose variations. In order to decrease the adverse effect of intra-class pose-variations in the PCA algorithm, Bhagavatula and Savvides [16] have employed the Fourier magnitudes (FM) of face images for the feature extraction. Using the ORL face database, in which there are substantial intra-class pose variations, they have shown that employing the Fourier magnitudes of the images for feature extraction improves the recognition accuracy of the PCA algorithm [16]. A brief explanation of the Fourier magnitude PCA (FM-PCA) follows.

In this method, the Fourier transform of a digital image of size $M \times N$ is first computed as [27], [28]

$$\mathrm{I_F(u,v)} = \sum_{\mathrm{m=0}}^{\mathrm{M-1}} \sum_{\mathrm{n=0}}^{\mathrm{N-1}} \mathrm{I(m,n)e^{-j2\pi(\frac{um}{M}+\frac{vn}{N})}} \; ; \; u = 1,2,\dots,M-1 \; ; \; v = 1,2,\dots,N-1 \quad (3.1)$$

Each Fourier coefficient given by (3.1) is a complex number expressed as $I_F(u,v) = Re\big(I_F(u,v)\big) + jIm(I_F(u,v))$, where $Re\big(I_F(u,v)\big)$ and $Im\big(I_F(u,v)\big)$ are, respectively, the real and imaginary parts of the coefficient $\mathrm{I_F(u,v)}$. The magnitude of the coefficient is given by

$$\mathrm{I_{FM}(u,v)} = |\mathrm{I_F(u,v)}| = \sqrt{\mathrm{Re}^2\big(\mathrm{I_F(u,v)}\big) + \mathrm{Im}^2(\mathrm{I_F(u,v)})} \tag{3.2}$$

The magnitudes of the Fourier coefficients for the i^{th} training image can be represented as an $M \times N$ matrix given by

$$\mathbf{I_{FM}^{(i)}} = \left[\mathrm{I_{FM}^{(i)}(u,v)}\right]_{\mathrm{M\times N}} \tag{3.3}$$

Similarly, the magnitudes of the Fourier coefficients of a test image is given by

$$\mathbf{I_{FM}^{(test)}} = \left[\mathrm{I_{FM}^{(test)}(u,v)}\right]_{\mathrm{M\times N}} \tag{3.4}$$

Any matrix given by (3.3) or (3.4) can be expressed as a column vector by rearranging the elements of $\boldsymbol{I_{FM}}$ taken row-by-row (or alternatively, column-by-column):

$$\mathbf{\Gamma_{FM}} = \begin{bmatrix} \mathrm{I_{FM}(0,0)} \\ \vdots \\ \mathrm{I_{FM}(M-1,0)} \\ \mathrm{I_{FM}(1,0)} \\ \vdots \\ \mathrm{I_{FM}(1,N-1)} \\ \vdots \\ \mathrm{I_{FM}(M-1,1)} \\ \vdots \\ \mathrm{I_{FM}(M-1,N-1)} \end{bmatrix}_{\mathrm{MN\times 1}} \tag{3.5}$$

The average of K such concatenated Fourier magnitude versions of the training samples computed as

$$\mathbf{\Psi_{FM}} = \frac{1}{\mathrm{K}}\sum_{\mathrm{i=1}}^{\mathrm{K}} \mathbf{\Gamma_{FM}^{(i)}} \tag{3.6}$$

is subtracted from each of the concatenated FM versions of the training samples giving

$$\mathbf{\Phi}_{\mathbf{FM}}^{(\mathrm{i})} = \mathbf{\Gamma}_{\mathbf{FM}}^{(\mathrm{i})} - \mathbf{\Psi}_{\mathbf{FM}} \tag{3.7}$$

Then, the above mean-subtracted training samples are used to construct a matrix given by

$$\mathbf{T}_{\mathbf{FM}} = \left[\mathbf{\Phi}_{\mathbf{FM}}^{(1)} \quad \mathbf{\Phi}_{\mathbf{FM}}^{(2)} \quad \dots \quad \mathbf{\Phi}_{\mathbf{FM}}^{(\mathrm{k})}\right] = \begin{bmatrix} \Phi_{\mathrm{FM}}^{(1)}(0) & \cdots & \Phi_{\mathrm{FM}}^{(\mathrm{K})}(0) \\ \vdots & \vdots & \vdots \\ \Phi_{\mathrm{FM}}^{(1)}(\mathrm{MN}-1) & \cdots & \Phi_{\mathrm{FM}}^{(\mathrm{K})}(\mathrm{MN}-1) \end{bmatrix} \tag{3.8}$$

Using $\boldsymbol{T_{FM}}$, we define the matrix $\boldsymbol{D_{FM}}$ as

$$\mathbf{D}_{\mathbf{FM}} = \mathbf{T}_{\mathbf{FM}}^{\mathrm{T}}\mathbf{T}_{\mathbf{FM}} \tag{3. 9}$$

Next, its eigenvalues and the corresponding eigenvectors are computed by

$$\mathbf{T}_{\mathbf{FM}}^{\mathrm{T}}\mathbf{T}_{\mathbf{FM}}\mathrm{d}_{\mathrm{j}} = \mathrm{e}_{\mathrm{j}}\mathrm{d}_{\mathrm{j}} \tag{3.10}$$

where e_j is an eigenvalue of $\boldsymbol{D_{FM}}$and d_j is the corresponding eigenvector. Pre-multiplying both sides of (3.10) by $\boldsymbol{T_{FM}}$ yields

$$\mathbf{T}_{\mathbf{FM}}\mathbf{T}_{\mathbf{FM}}^{\mathrm{T}}\mathbf{T}_{\mathbf{FM}}\mathrm{d}_{\mathrm{j}}{}^{\mathrm{T}} = \mathbf{T}_{\mathbf{FM}}\mathrm{e}_{\mathrm{j}}\mathrm{d}_{\mathrm{j}} = \mathrm{e}_{\mathrm{j}}(\mathbf{T}_{\mathbf{FM}}\mathrm{e}_{\mathrm{j}}\mathrm{d}_{\mathrm{j}}) \tag{3.11}$$

$$\mathbf{T}_{\mathbf{FM}}\mathbf{T}_{\mathbf{FM}}^{\mathrm{T}}(\mathbf{T}_{\mathbf{FM}}\mathrm{d}_{\mathrm{j}}) = \mathrm{e}_{\mathrm{j}}(\mathbf{T}_{\mathbf{FM}}\mathrm{d}_{\mathrm{j}}) \tag{3.12}$$

Since the covariance matrix $\mathbf{C}_{\mathbf{FM}} = \mathbf{T}_{\mathbf{FM}}\mathbf{T}_{\mathbf{FM}}^{\mathrm{T}}$, **t**he eigenvalue e_j of $\boldsymbol{D_{FM}}$ is also an eigenvalue of $\boldsymbol{C_{FM}}$ and the corresponding eigenvector of $\boldsymbol{C_{FM}}$ is given as $\widehat{u}_j = \boldsymbol{T_{FM}}d_j$. Next, these eigenvectors, are normalized as $u_j = \frac{\widehat{u}_j}{\|\widehat{u}_j\|}$ and arranged in a sequence corresponding to the decreasing eigenvalues, and denoted by u_j, $j = 1,2,\dots,K$, respectively.

Each sample is represented by a feature vector $\boldsymbol{\Omega}_{\boldsymbol{FM}}$. A training feature vector $\boldsymbol{\Omega}_{\boldsymbol{FM}}^{(\boldsymbol{i})}$ is obtained by projecting $\boldsymbol{\Phi}_{\boldsymbol{FM}}^{(i)}$ onto the normalized and ordered eigenvectors u_j, $j = 1,2,\dots,L$, where L is the number of chosen normalized eigenvectors that sufficiently expose the variations in the face images. The feature vector $\boldsymbol{\Omega}^{(i)}$ corresponding to the i^{th} training image sample is computed as

$$\mathbf{\Omega}_{\mathbf{FM}}^{(\mathbf{i})} = \mathbf{Q}_{\mathbf{FM}}\mathbf{\Phi}_{\mathbf{FM}}^{(\mathrm{i})} \tag{3.13}$$

where

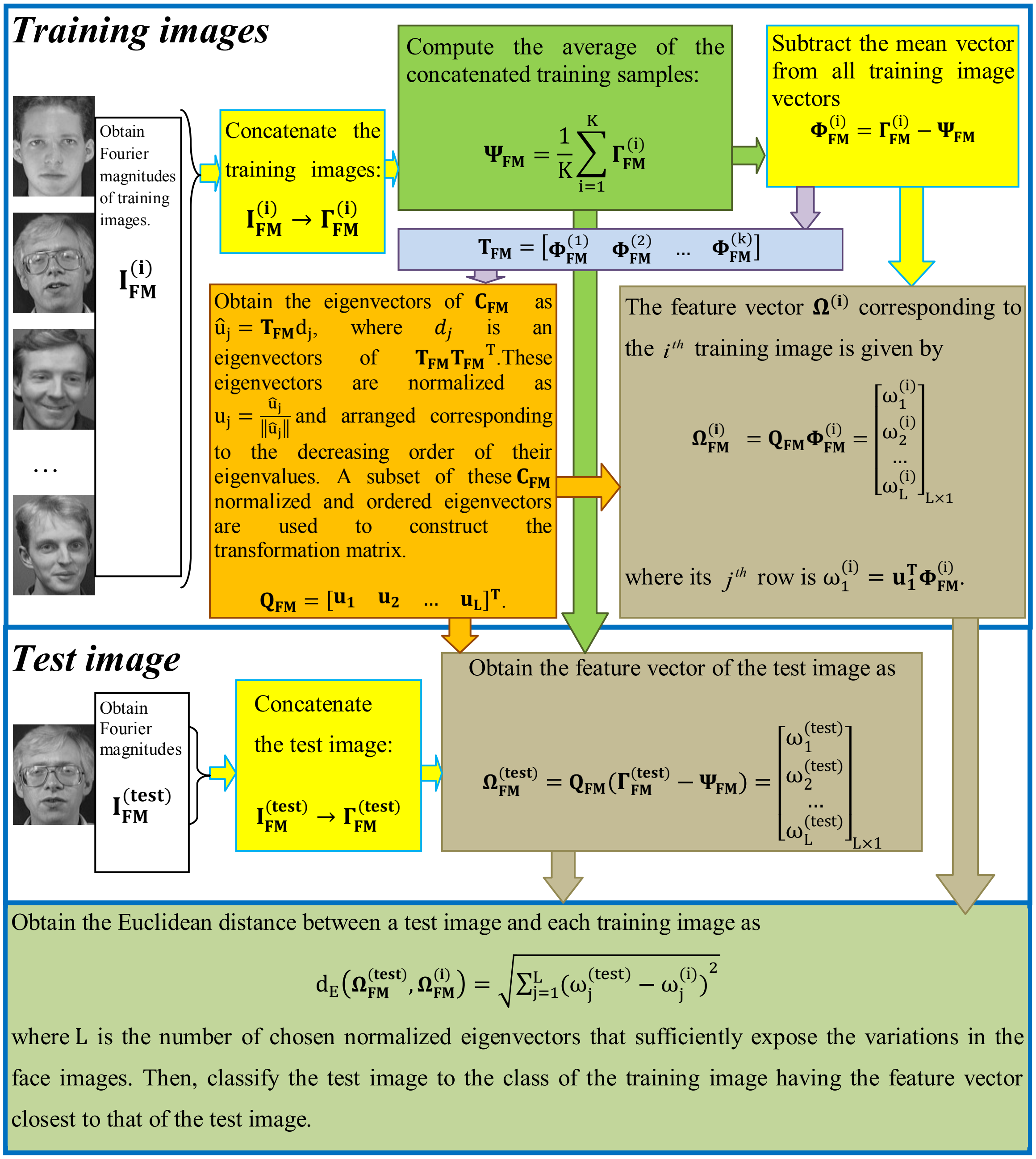
$$\mathbf{Q}_{\mathbf{FM}} = [\mathbf{u_1} \quad \mathbf{u_2} \quad \dots \quad \mathbf{u_L}]^{\mathbf{T}} \tag{3.14}$$

is called the transformation matrix. Similarly, for a test image $\boldsymbol{I}_{\boldsymbol{FM}}^{(\boldsymbol{test})}$, the corresponding feature vector is obtained as

$$\mathbf{\Omega}_{\mathbf{FM}}^{(\mathbf{test})} = \mathbf{Q}_{\mathbf{FM}}(\mathbf{\Gamma}_{\mathbf{FM}}^{(\mathrm{test})} - \mathbf{\Psi}_{\mathbf{FM}}) \tag{3.15}$$

where $\boldsymbol{\Gamma}_{\boldsymbol{FM}}^{(test)}$is the concatenated vector corresponding to the test image's Fourier magnitude version $\boldsymbol{I}_{\boldsymbol{FM}}^{(test)}$, $\boldsymbol{Q}_{\boldsymbol{FM}}$ is the transformation matrix given by (3.14) and $\boldsymbol{\Psi}_{\boldsymbol{FM}}$ is the mean vector given by (3.6).

As in the case of the PCA algorithm, the Euclidean distance is used to measure the distance between the feature vector of a test image and that of a training image. Then, the subject of the training image corresponding to the feature vector $\boldsymbol{\Omega}^{(i)}$ that has the shortest distance from the test image feature vector $\boldsymbol{\Omega}^{(test)}$ is assigned as the subject of the test image. A block diagram of Fig.3.1 summarizes the FM-PCA algorithm.



Training images

Obtain Fourier magnitudes of training images.

$\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})}$

Concatenate the training images: $\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})} \rightarrow \mathbf{\Gamma}_{\mathbf{FM}}^{(\mathbf{i})}$

Compute the average of the concatenated training samples:

$$\mathbf{\Psi}_{\mathbf{FM}} = \frac{1}{K}\sum_{i=1}^{K}\mathbf{\Gamma}_{\mathbf{FM}}^{(i)}$$

Subtract the mean vector from all training image vectors

$$\mathbf{\Phi}_{\mathbf{FM}}^{(i)} = \mathbf{\Gamma}_{\mathbf{FM}}^{(i)} - \mathbf{\Psi}_{\mathbf{FM}}$$

$$\mathbf{T}_{\mathbf{FM}} = [\mathbf{\Phi}_{\mathbf{FM}}^{(1)} \quad \mathbf{\Phi}_{\mathbf{FM}}^{(2)} \quad \dots \quad \mathbf{\Phi}_{\mathbf{FM}}^{(k)}]$$

Obtain the eigenvectors of $\mathbf{C}_{\mathbf{FM}}$ as $\hat{u}_j = \mathbf{T}_{\mathbf{FM}} d_j$, where d_j is an eigenvectors of $\mathbf{T}_{\mathbf{FM}}\mathbf{T}_{\mathbf{FM}}{}^{T}$.These eigenvectors are normalized as $u_j = \frac{\hat{u}_j}{\|\hat{u}_j\|}$ and arranged corresponding to the decreasing order of their eigenvalues. A subset of these $\mathbf{C}_{\mathbf{FM}}$ normalized and ordered eigenvectors are used to construct the transformation matrix.

$$\mathbf{Q}_{\mathbf{FM}} = [\mathbf{u_1} \quad \mathbf{u_2} \quad \dots \quad \mathbf{u}_L]^{\mathbf{T}}.$$

The feature vector $\mathbf{\Omega}^{(\mathbf{i})}$ corresponding to the i^{th} training image is given by

$$\mathbf{\Omega}_{\mathbf{FM}}^{(i)} = \mathbf{Q}_{\mathbf{FM}}\mathbf{\Phi}_{\mathbf{FM}}^{(i)} = \begin{bmatrix} \omega_1^{(i)} \\ \omega_2^{(i)} \\ \dots \\ \omega_L^{(i)} \end{bmatrix}_{L\times 1}$$

where its j^{th} row is $\omega_1^{(i)} = \mathbf{u}_1^{\mathbf{T}}\mathbf{\Phi}_{\mathbf{FM}}^{(i)}$.

Test image

Obtain Fourier magnitudes

$\mathbf{I}_{\mathbf{FM}}^{(\mathbf{test})}$

Concatenate the test image: $\mathbf{I}_{\mathbf{FM}}^{(\mathbf{test})} \rightarrow \mathbf{\Gamma}_{\mathbf{FM}}^{(\mathbf{test})}$

Obtain the feature vector of the test image as

$$\mathbf{\Omega}_{\mathbf{FM}}^{(\mathbf{test})} = \mathbf{Q}_{\mathbf{FM}}(\mathbf{\Gamma}_{\mathbf{FM}}^{(\mathbf{test})} - \mathbf{\Psi}_{\mathbf{FM}}) = \begin{bmatrix} \omega_1^{(test)} \\ \omega_2^{(test)} \\ \dots \\ \omega_L^{(test)} \end{bmatrix}_{L\times 1}$$

Obtain the Euclidean distance between a test image and each training image as

$$d_E(\mathbf{\Omega}_{\mathbf{FM}}^{(\mathbf{test})}, \mathbf{\Omega}_{\mathbf{FM}}^{(\mathbf{i})}) = \sqrt{\sum_{j=1}^{L}(\omega_j^{(test)} - \omega_j^{(i)})^2}$$

where L is the number of chosen normalized eigenvectors that sufficiently expose the variations in the face images. Then, classify the test image to the class of the training image having the feature vector closest to that of the test image.

Fig.3.1: Summary of the FM-PCA algorithm.

3.2.2 Simulation Results

In this subsection, detailed simulation results for the conventional PCA algorithm as well as for the FM-PCA algorithm are provided in terms of the recognition accuracy and the computational complexity. Simulations are performed on a 2.8 GHz Intel Core i7 CPU with 4GB RAM and Windows 7 operating system. Simulations are carried out using MATLAB [29]. The algorithms are tested on the Olivetti Research Laboratory (ORL) face database [17], a benchmark database in the literature [12], [13], [18], [19], [22], [23], [26]. The database consists of face images taken against a dark homogeneous background and represented with an 8 bit-greyscale. There are 40 subjects, each having 10 different images of size 92×112. The images of each subject vary from one another in terms of pose, expression and zooming. In addition, the intra-class pose variations are substantial in this database. The complete set of images in the database are shown in Fig. 3.2.

For the sake of clarity in explaining how the simulations are conducted, we assign indices 1-to-10, starting with the left most image to the right most image, for each subject. A set of 5 indices are chosen randomly and the five images in each subject corresponding to these indices are chosen as the training images (thus, constituting a total of 200 training images) to train a given algorithm. For each of the training image, the corresponding feature matrix is obtained. For a given test image, the Euclidean distance between the feature vector of the test image and that of each of the 200 vectors of the training images is obtained, and the one with the closest distance chosen to determine the

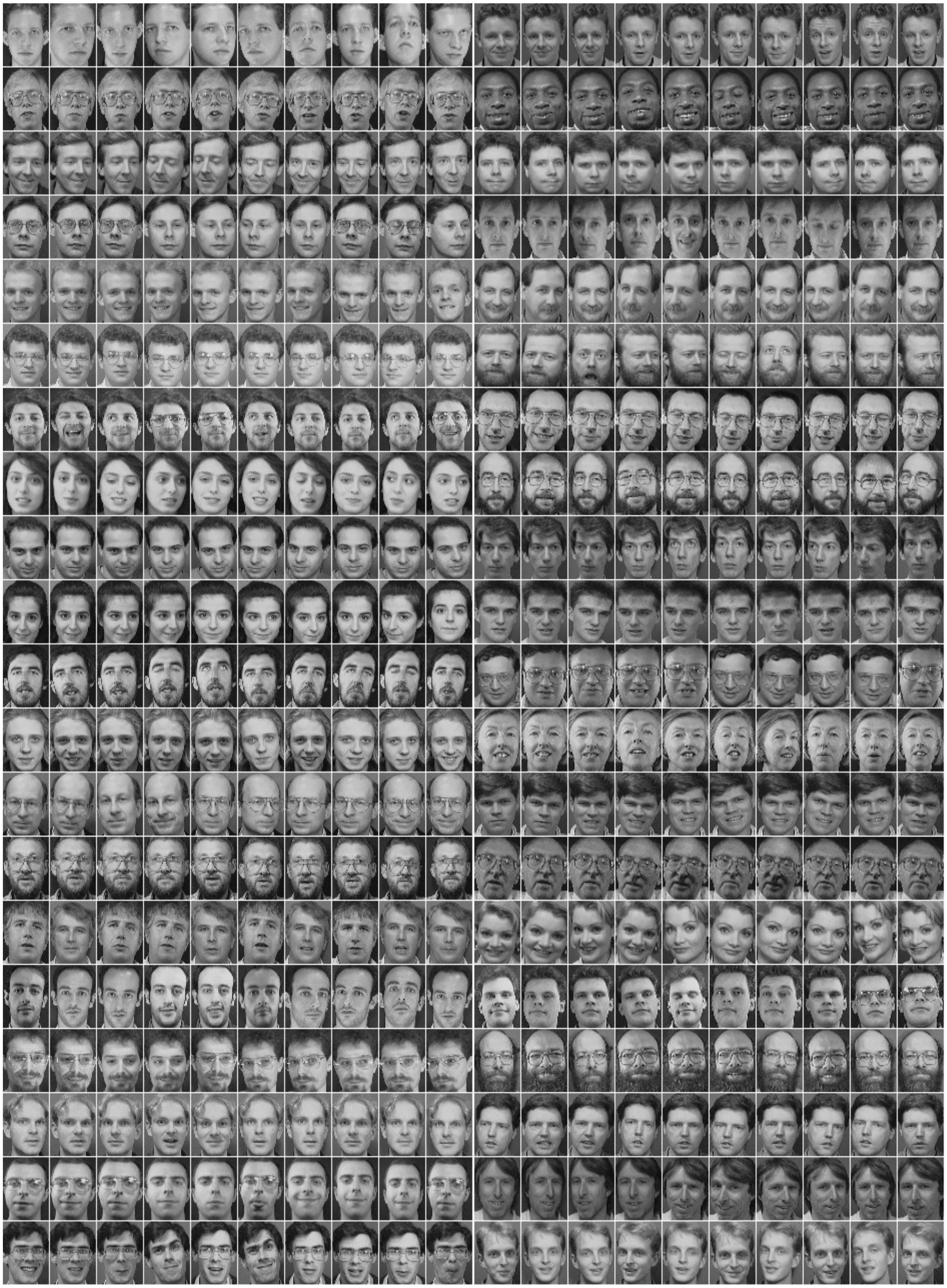


Fig.3.2: Face images in the ORL face database.

class of the test image under consideration. This is repeated for each of the 200 test images and the recognition accuracy found.

The above procedure is repeated 49 times, each time choosing randomly a set of 5 indices that is distinct from the sets of the previous runs of the algorithm. Finally, the recognition accuracy of the algorithm is obtained as the average of the recognition accuracies derived from the 50 runs.

Table 3.1 gives the performance of the PCA and FM-PCA algorithms in terms of the recognition accuracy for different sizes of the feature vector. The recognition accuracy as a function of the feature vector size is also illustrated in Fig. 3.3. It is clearly seen from the table and this figure that the use of the Fourier magnitudes in feature extraction for PCA improves its recognition accuracy substantially.

We now consider the time complexities for the PCA and FM-PCA algorithms. For this purpose, each algorithm is run 10 times and the average execution time is computed; ten runs are considered sufficient to compute the time complexity, since the algorithms are not data dependent for a fixed image size. The average total time taken for each of the algorithms is given in Table 3.2. As expected, it is seen from this table that FM-PCA has a higher time complexity than that of FM-PCA due to the overhead involved in the computation of the Fourier magnitudes.

In conclusion, it is to be noted that there is about 3% improvement in the recognition accuracy, but at the cost of a 23% increase in the computational time.

Table 3.1 : Recognition accuracy in percentage of the PCA and FM-PCA algorithms.

Number of principal components	Recognition accuracy (%)	
	PCA	FM-PCA
10	89.9	92.6
11	90.3	93.1
12	90.7	93.5
13	90.8	93.7
14	90.9	93.9
15	91.1	94.1
16	91.1	94.2
17	91.2	94.3
18	91.2	94.3
19	91.3	94.4
20	91.4	94.4
21	91.5	94.5
22	91.5	94.6
23	91.6	94.7
24	91.7	94.7
25	91.7	94.7
26	91.8	94.8
27	91.8	94.8
28	91.9	94.8
29	91.9	94.8
30	91.9	94.8
31	91.9	94.8
32	91.9	94.8
33	91.9	94.8
34	91.9	94.9
35	91.9	94.9
36	91.9	94.8
37	91.9	94.9
38	92	94.8
39	92	94.9
40	92	94.8

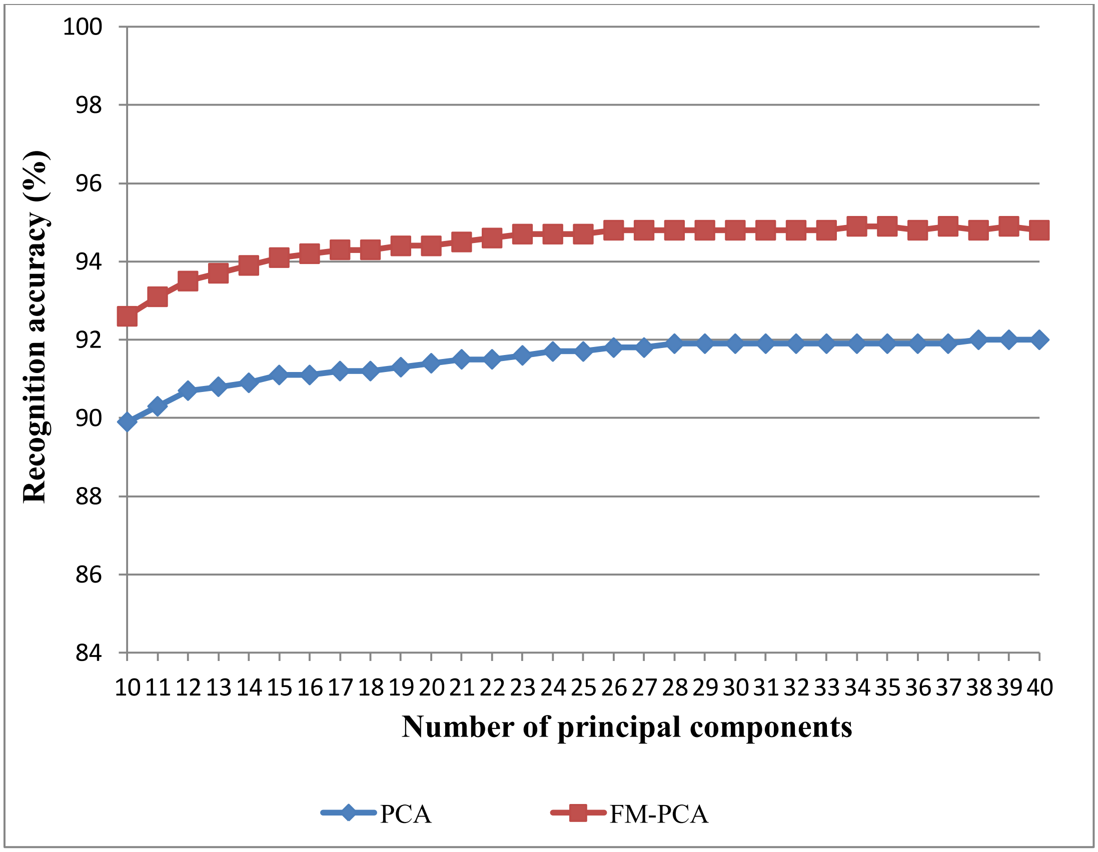


Fig.3.3: Recognition accuracies for the PCA and FM-PCA algorithms using the Euclidean distance for classification.

Table 3.2: Time complexity in seconds of the PCA and FM-PCA algorithms.

Algorithm and number of principal components	**Total time complexity (s)**
PCA 35	3.016
FM-PCA **35**	**3.719**
PCA 38	3.027
FM-PCA **38**	**3.730**

3.3 Proposed Two-Dimensional PCA Algorithms in the Fourier Domain

As seen from the previous section, the use of the Fourier magnitudes in the PCA algorithm improves the recognition accuracy, but at the expense of increased time complexity. On the other hand, we also know that the two-dimensional extensions of the PCA algorithm have lower time complexities, while providing at the same time higher accuracies compared to the PCA. Hence, in this section, we develop Fourier magnitude versions of the two-dimensional PCA algorithms, r2DPCA and (2D)^{2}PCA, and refer to them as FM-r2DPCA and FM-(2D)^{2}PCA, respectively.

3.3.1 Fourier-Magnitude r2DPCA

The Fourier magnitudes for the pixels of an image are computed first using (3.2). The magnitudes of the Fourier coefficients for the i^{th} training image can be represented as an $M \times N$ matrix given by $\boldsymbol{I}_{\boldsymbol{FM}}^{(\boldsymbol{i})} = \left[I_{FM}^{(i)}(u,v)\right]_{M\times N}$. Similarly, the FM version of a test image is represented as $\boldsymbol{I}_{\boldsymbol{FM}}^{(\boldsymbol{test})} = \left[I_{FM}^{(test)}(u,v)\right]_{M\times N}$. The Fourier magnitude matrices of the training images are then used to obtain the covariance matrix of FM-r2DPCA as

$$\mathbf{C}_{\mathbf{FM-r2DPCA}} = \frac{1}{\mathrm{K}}\sum_{\mathrm{i}=1}^{\mathrm{K}}(\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})} - \bar{\mathbf{I}}_{\mathbf{FM}})^{\mathrm{T}}(\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})} - \bar{\mathbf{I}}_{\mathbf{FM}}) \qquad (3.16)$$

where K is the number of training samples and $\boldsymbol{\bar{I}_{FM}}$ is the matrix in which each entry is the average of the corresponding entries in the matrices $\boldsymbol{I_{FM}^{(i)}}$, $i = 1,2, \dots, K$, and is given by

$$\bar{\mathbf{I}}_{\mathbf{FM}} = \frac{1}{\mathrm{K}} \sum_{\mathrm{i}=1}^{\mathrm{K}} \mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})} \tag{3.17}$$

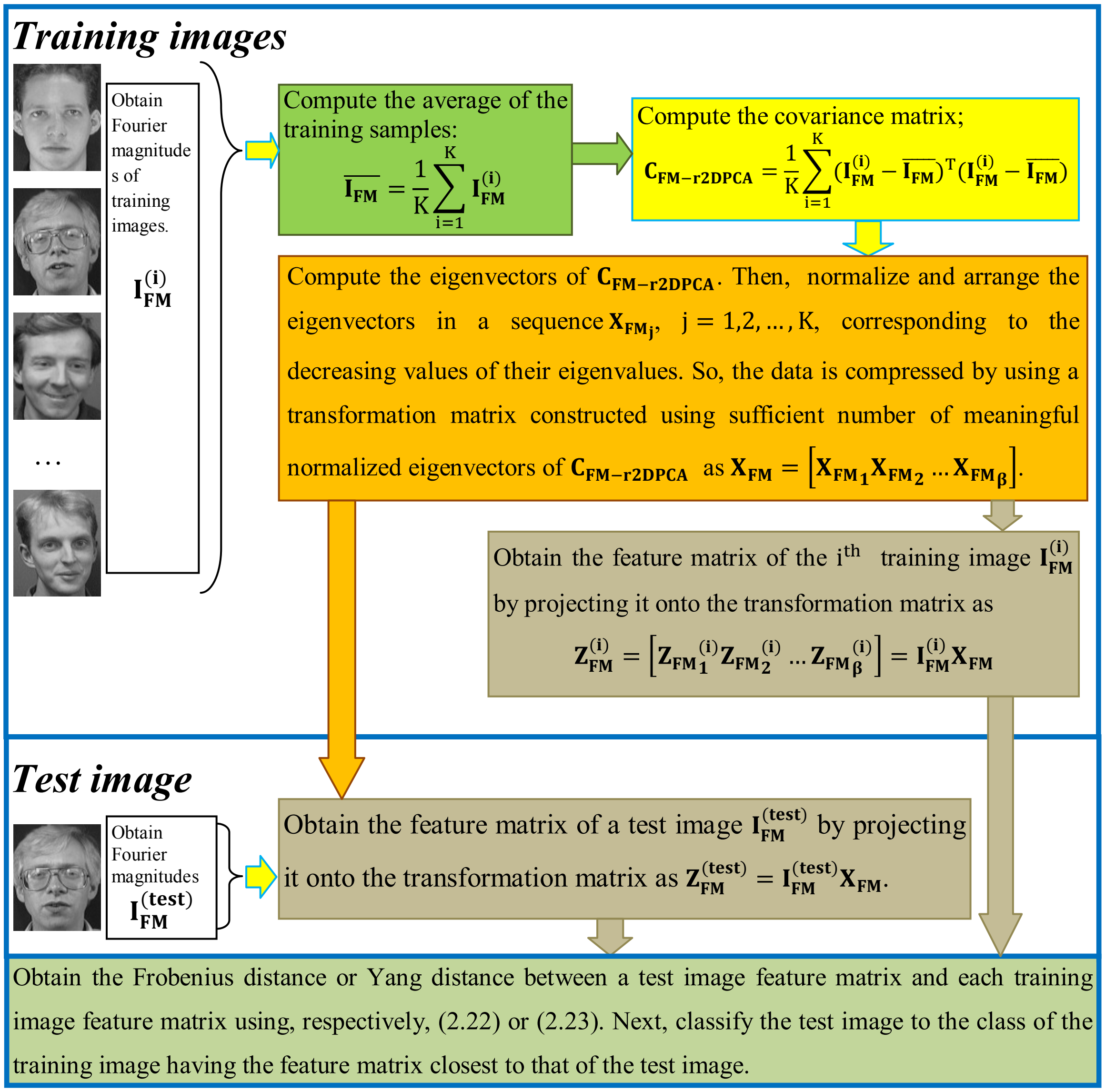
Next, the eigenvalues and eigenvectors of $\boldsymbol{C_{FM-r2DPCA}}$ are computed. The eigenvectors are normalized and rearranged in descending order of the corresponding eigenvalues. A transformation matrix $\mathbf{X_{FM}} = \left[\mathbf{X_{FM_1} X_{FM_2} \dots X_{FM_\beta}}\right]$ is obtained using sufficient number of normalized eigenvectors of $\boldsymbol{C_{FM-r2DPCA}}$. The, feature matrices for the i^{th} training sample and the test sample are, respectively, given by

$$\mathbf{Z_{FM}^{(i)}} = \mathbf{I_{FM}^{(i)} X_{FM}} \tag{2. 18}$$

and

$$\mathbf{Z_{FM}^{(test)}} = \mathbf{I_{FM}^{(test)} X_{FM}} \tag{2. 19}$$

For the classification module of the FM-r2DPCA algorithm, the Frobenius or Yang distance metric, given by (2.22) and (2.23), is used. The distance between the feature matrix of a test sample and that of each training sample is obtained. Then, the subject of the training sample, whose feature matrix has the shortest distance from the test image feature matrix $\boldsymbol{Z_{FM}^{(test)}}$, is assigned as the subject of the test image. The block diagram of Fig.3.4 summarizes the FM-r2DPCA algorithm.



Training images

Obtain Fourier magnitudes of training images. $\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})}$

Compute the average of the training samples: $\overline{\mathbf{I}_{\mathbf{FM}}} = \frac{1}{K}\sum_{i=1}^{K}\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})}$

Compute the covariance matrix; $\mathbf{C}_{\mathbf{FM-r2DPCA}} = \frac{1}{K}\sum_{i=1}^{K}(\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})} - \overline{\mathbf{I}_{\mathbf{FM}}})^{T}(\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})} - \overline{\mathbf{I}_{\mathbf{FM}}})$

Compute the eigenvectors of $\mathbf{C}_{\mathbf{FM-r2DPCA}}$. Then, normalize and arrange the eigenvectors in a sequence $\mathbf{X}_{\mathbf{FM}_j}$, $j = 1,2,\dots,K$, corresponding to the decreasing values of their eigenvalues. So, the data is compressed by using a transformation matrix constructed using sufficient number of meaningful normalized eigenvectors of $\mathbf{C}_{\mathbf{FM-r2DPCA}}$ as $\mathbf{X}_{\mathbf{FM}} = \left[\mathbf{X}_{\mathbf{FM}_1}\mathbf{X}_{\mathbf{FM}_2}\dots\mathbf{X}_{\mathbf{FM}_\beta}\right]$.

Obtain the feature matrix of the ith training image $\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})}$ by projecting it onto the transformation matrix as

$\mathbf{Z}_{\mathbf{FM}}^{(\mathbf{i})} = \left[\mathbf{Z}_{\mathbf{FM}_1}^{(\mathbf{i})}\mathbf{Z}_{\mathbf{FM}_2}^{(\mathbf{i})}\dots\mathbf{Z}_{\mathbf{FM}_\beta}^{(\mathbf{i})}\right] = \mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})}\mathbf{X}_{\mathbf{FM}}$

Test image

Obtain Fourier magnitudes $\mathbf{I}_{\mathbf{FM}}^{(\mathbf{test})}$

Obtain the feature matrix of a test image $\mathbf{I}_{\mathbf{FM}}^{(\mathbf{test})}$ by projecting it onto the transformation matrix as $\mathbf{Z}_{\mathbf{FM}}^{(\mathbf{test})} = \mathbf{I}_{\mathbf{FM}}^{(\mathbf{test})}\mathbf{X}_{\mathbf{FM}}$.

Obtain the Frobenius distance or Yang distance between a test image feature matrix and each training image feature matrix using, respectively, (2.22) or (2.23). Next, classify the test image to the class of the training image having the feature matrix closest to that of the test image.

Fig.3.4: Summary of the FM-r2DPCA algorithm.

3.3.2 Fourier Magnitude $(2D)^2$PCA

As discussed in Section 2.4.2, the $(2D)^2$PCA algorithm has a lower computational complexity than that of even the r2DPCA algorithm. Hence, we now develop an FM-version of the $(2D)^2$PCA algorithm. Henceforth, this algorithm is referred to as the FM-$(2D)^2$PCA algorithm.

The row-directional covariance matrix $\boldsymbol{C_{FM-r2DPCA}}$ is computed using (3.16) and its column-directional counterpart is computed as

$$\mathbf{C_{FM-c2DPCA}} = \frac{1}{\mathrm{K}}\sum_{\mathrm{i}=1}^{\mathrm{K}}(\mathbf{I_{FM}^{(i)}} - \bar{\mathbf{I}}_{\mathbf{FM}})(\mathbf{I_{FM}^{(i)}} - \bar{\mathbf{I}}_{\mathbf{FM}})^{\mathrm{T}} \tag{3.20}$$

The eigenvalues and eigenvectors of $\boldsymbol{C_{FM-r2DPCA}}$ and $\boldsymbol{C_{FM-c2DPCA}}$ are then computed. Next, the eigenvectors of each of the two covariance matrices are normalized and arranged in descending order of the corresponding eigenvalues. Then, the row-directional and column-directional transformation matrices $\mathbf{X_{FM}} = \left[\mathbf{X_{FM_1}X_{FM_2}} \ldots \mathbf{X_{FM_\beta}}\right]$ and $\mathbf{Y_{FM}} = \left[\mathbf{Y_{FM_1}Y_{FM_2}} \ldots \mathbf{Y_{FM_\alpha}}\right]$ are obtained using sufficient numbers of the eigenvectors of $\mathbf{C_{FM-r2DPCA}}$ and $\mathbf{C_{FM-c2DPCA}}$, respectively.

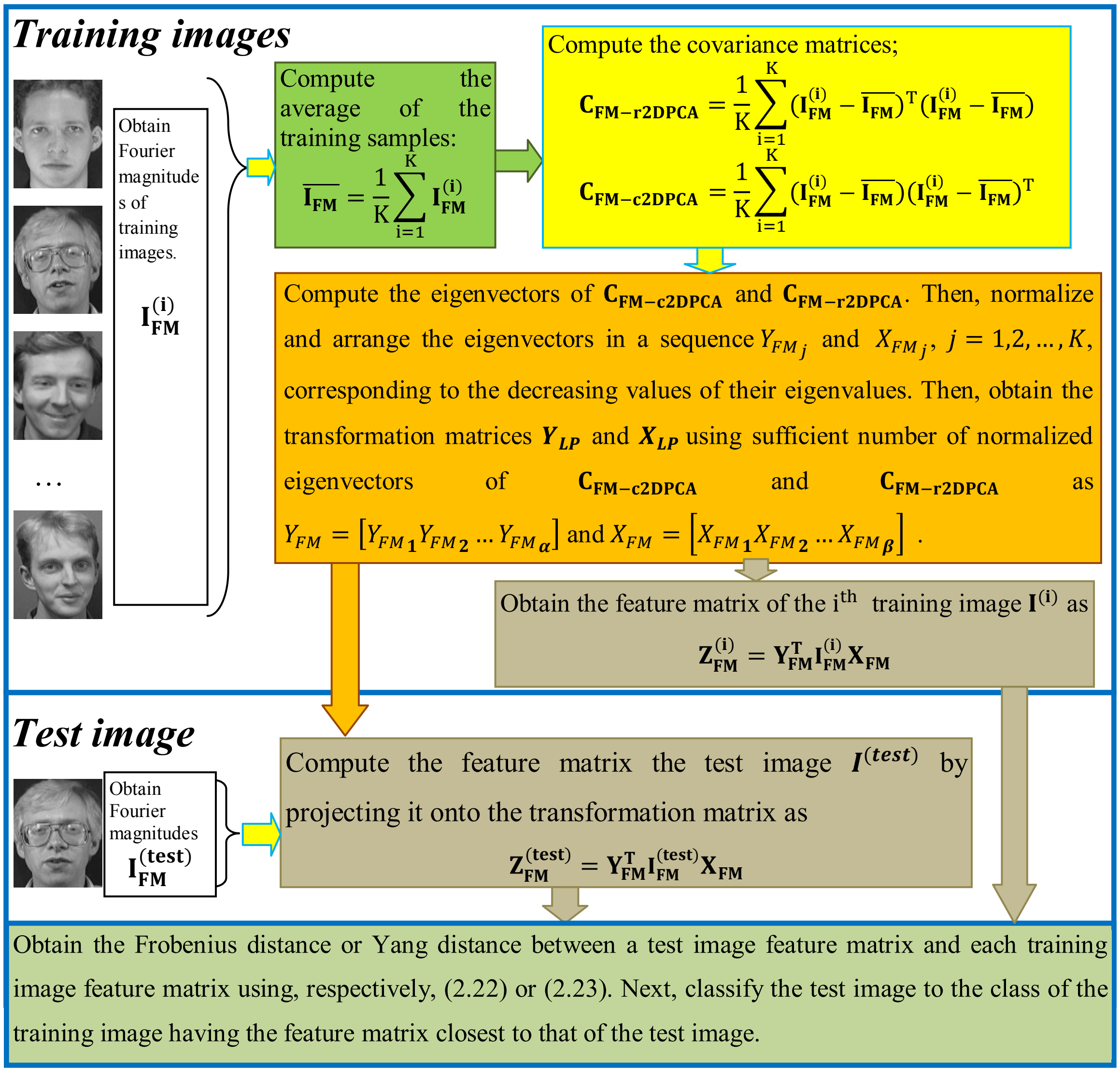
The feature matrices for the i^{th} training sample and the test sample are, respectively, given by

$$\mathbf{Z_{FM}^{(i)}} = \mathbf{Y_{FM}^{T}I_{FM}^{(i)}X_{FM}} \tag{3.21}$$

and

$$\mathbf{Z_{FM}^{(test)}} = \mathbf{Y_{FM}^{T}I_{FM}^{(test)}X_{FM}} \tag{3.22}$$

For the purpose of classification, we again employ the Frobenius as well as the Yang distance metrics given by (2.22) and (2.23), respectively. The distance between the feature matrix of a test sample and that of each of the training samples is obtained. Then, the subject of the training sample, whose feature matrix has the shortest distance from the feature matrix of the test sample, is assigned as the subject of the test image. A summary of the FM-(2D)^{2}PCA algorithm is given in Fig. 3.5.



Obtain Fourier magnitudes of training images. $\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})}$

Compute the average of the training samples: $\overline{\mathbf{I}_{\mathbf{FM}}} = \frac{1}{\mathrm{K}}\sum_{\mathrm{i}=1}^{\mathrm{K}} \mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})}$

Compute the covariance matrices;

$$\mathbf{C}_{\mathbf{FM-r2DPCA}} = \frac{1}{\mathrm{K}}\sum_{\mathrm{i}=1}^{\mathrm{K}} (\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})} - \overline{\mathbf{I}_{\mathbf{FM}}})^{\mathrm{T}}(\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})} - \overline{\mathbf{I}_{\mathbf{FM}}})$$

$$\mathbf{C}_{\mathbf{FM-c2DPCA}} = \frac{1}{\mathrm{K}}\sum_{\mathrm{i}=1}^{\mathrm{K}} (\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})} - \overline{\mathbf{I}_{\mathbf{FM}}})(\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})} - \overline{\mathbf{I}_{\mathbf{FM}}})^{\mathrm{T}}$$

Compute the eigenvectors of $\mathbf{C}_{\mathbf{FM-c2DPCA}}$ and $\mathbf{C}_{\mathbf{FM-r2DPCA}}$. Then, normalize and arrange the eigenvectors in a sequence Y_{FM_j} and X_{FM_j}, $j = 1,2,\ldots,K$, corresponding to the decreasing values of their eigenvalues. Then, obtain the transformation matrices $\boldsymbol{Y_{LP}}$ and $\boldsymbol{X_{LP}}$ using sufficient number of normalized eigenvectors of $\mathbf{C}_{\mathbf{FM-c2DPCA}}$ and $\mathbf{C}_{\mathbf{FM-r2DPCA}}$ as $Y_{FM} = [Y_{FM_1} Y_{FM_2} \ldots Y_{FM_\alpha}]$ and $X_{FM} = [X_{FM_1} X_{FM_2} \ldots X_{FM_\beta}]$.

Obtain the feature matrix of the i$^{\text{th}}$ training image $\mathbf{I}^{(\mathbf{i})}$ as

$$\mathbf{Z}_{\mathbf{FM}}^{(\mathbf{i})} = \mathbf{Y}_{\mathbf{FM}}^{\mathbf{T}}\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})}\mathbf{X}_{\mathbf{FM}}$$

Obtain Fourier magnitudes $\mathbf{I}_{\mathbf{FM}}^{(\mathbf{test})}$

Compute the feature matrix the test image $\boldsymbol{I}^{(\boldsymbol{test})}$ by projecting it onto the transformation matrix as

$$\mathbf{Z}_{\mathbf{FM}}^{(\mathbf{test})} = \mathbf{Y}_{\mathbf{FM}}^{\mathbf{T}}\mathbf{I}_{\mathbf{FM}}^{(\mathbf{test})}\mathbf{X}_{\mathbf{FM}}$$

Obtain the Frobenius distance or Yang distance between a test image feature matrix and each training image feature matrix using, respectively, (2.22) or (2.23). Next, classify the test image to the class of the training image having the feature matrix closest to that of the test image.

Fig.3.5: Summary of the FM-(2D)^{2}PCA algorithm.

3.4 Experimental Results

In this section, detailed simulation results for the r2DPCA, FM-r2DPCA, (2D)^{2}PCA and FM-(2D)^{2}PCA algorithms are provided and the algorithms compared in terms of the recognition accuracy and the computational complexity. In order to obtain the simulation results, the hardware and software platforms described in Section 3.2.2 are utilized. As mentioned earlier, we assign indices 1-to-10 for each subject starting with the left most image to the right most image. A set of 5 indices are chosen randomly and the five images in each subject corresponding to these indices are chosen as the training images (thus, constituting a total of 200 training images) to train a given algorithm. For each of the training image, the corresponding feature matrix is obtained. For a given test image, the Frobenius or the Yang distance between the feature matrix of the test image and that of each of the 200 feature matrices of the training images is obtained and the one with the closest distance chosen to determine the class of the test image under consideration. This is repeated for each of the 200 test images and the recognition accuracy found. The above procedure is repeated 49 times, each time choosing randomly a set of 5 indices that is distinct from the sets of the previous runs of the algorithm. Finally, the recognition accuracy of the algorithm is obtained as the average of the recognition accuracies derived from the 50 runs.

Tables 3.3 and 3.4 give, respectively, the performance of the r2DPCA, FM-r2DPCA, $(2D)^2$PCA and FM-$(2D)^2$PCA algorithms in terms of the recognition accuracy using the Frobenius and Yang distance metrics for classification. The first column in these tables gives the name of the algorithm along with the size of the feature matrix.

Table 3.3: Recognition accuracy in percentage of the various algorithms using the Frobenius distance as the metric for classification.

Algorithm	**Recognition accuracy**							
	Number of principal component vectors (β)							
	3	**4**	**5**	**6**	**7**	**8**	**9**	**10**
r2DPCA $(112 \times \beta)$	95.1	95.3	95.0	95.0	95.0	95.0	94.8	95.0
FM-r2DPCA $\mathbf{(112 \times \beta)}$	**97.2**	**97.5**	**98.0**	**97.8**	**98.0**	**97.8**	**98.0**	**97.9**
$(2D)^2$PCA $(21 \times \beta)$	94.9	95.0	95.3	95.0	95.1	95.0	95.0	95.2
FM-$\mathbf{(2D)^2}$PCA $\mathbf{(21 \times \beta)}$	**97.9**	**98.1**	**98.2**	**98.1**	**98.2**	**98.1**	**98.1**	**98.2**
$(2D)^2$PCA $(23 \times \beta)$	95.2	95.3	95.3	95.2	95.2	95.2	95.1	95.2
FM-$\mathbf{(2D)^2}$PCA $\mathbf{(23 \times \beta)}$	**98.3**	**98.2**	**98.3**	**98.3**	**98.2**	**98.2**	**98.2**	**98.2**
$(2D)^2$PCA $(25 \times \beta)$	95.2	95.0	95.1	95.0	95.1	95.0	95.0	95.0
FM-$\mathbf{(2D)^2}$PCA $\mathbf{(25 \times \beta)}$	**98.1**	**98.2**	**98.2**	**98.0**	**98.1**	**98.0**	**98.1**	**98.0**

Table 3.4: Recognition accuracy in percentage of the various algorithms using the Yang distance as the metric for classification.

Algorithm	**Recognition accuracy**							
	Number of principal component vectors (β)							
	3	**4**	**5**	**6**	**7**	**8**	**9**	**10**
r2DPCA $(112 \times \beta)$	96.0	96.0	95.8	95.7	95.4	95.2	95.6	95.5
FM-r2DPCA $\boldsymbol{(112 \times \beta)}$	**97.3**	**98.0**	**98.5**	**98.4**	**98.4**	**98.4**	**98.3**	**98.5**
(2D)^{2}PCA $(21 \times \beta)$	95.5	95.7	95.8	95.7	95.6	95.3	95.7	95.5
FM-(2D)^{2}PCA $\boldsymbol{(21 \times \beta)}$	**97.6**	**98.8**	**98.9**	**98.8**	**98.7**	**98.7**	**98.8**	**98.7**
(2D)^{2}PCA $(23 \times \beta)$	96.0	96.5	96.4	95.7	95.5	95.3	95.6	95.6
FM-(2D)^{2}PCA $\boldsymbol{(23 \times \beta)}$	**98.1**	**98.9**	**98.8**	**98.9**	**98.9**	**98.8**	**98.8**	**98.8**
(2D)^{2}PCA $(25 \times \beta)$	96.0	96.0	96.0	95.7	95.5	95.3	95.7	95.5
FM-(2D)^{2}PCA $\boldsymbol{(25 \times \beta)}$	**97.3**	**98. 8**	**98.8**	**98.8**	**98.8**	**98.6**	**98.7**	**98.7**

It is seen from these tables that, as expected, the FM-r2DPCA and FM-(2D)^{2}PCA algorithms provide higher recognition accuracies compared to that provided by r2DPCA and (2D)^{2}PCA, respectively, regardless of whether the Frobenius or the Yang distance is used as the metric. It is to be noted that even though the recognition accuracies of r2DPCA and (2D)^{2}PCA are about the same, the application of FM on (2D)^{2}PCA results in better accuracies than when applied on r2DPCA. The results for r2DPCA and FM-r2DPCA as well as for (2D)^{2}PCA and FM-(2D)^{2}PCA, when the number of rows in the resulting feature matrix is 23, are also illustrated in Figs. 3.6 and 3.7 for the two metrics. It is to be pointed out that the FM-(2D)^{2}PCA algorithm provides the best recognition accuracy amongst all the PCA algorithms considered.

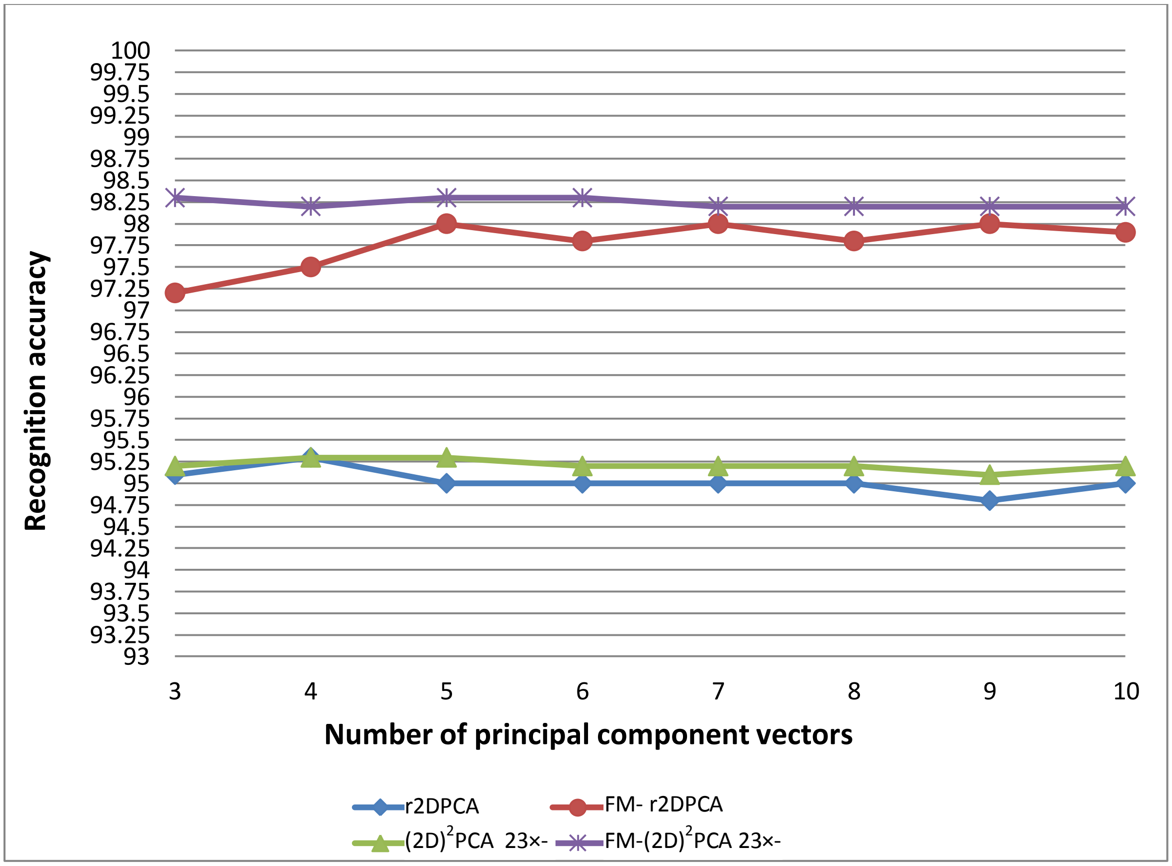


Fig.3.6: Recognition accuracy for the r2DPCA, FM-r2DPCA, (2D)^{2}PCA and FM-(2D)^{2}PCA algorithms using the Frobenius distance for classification.

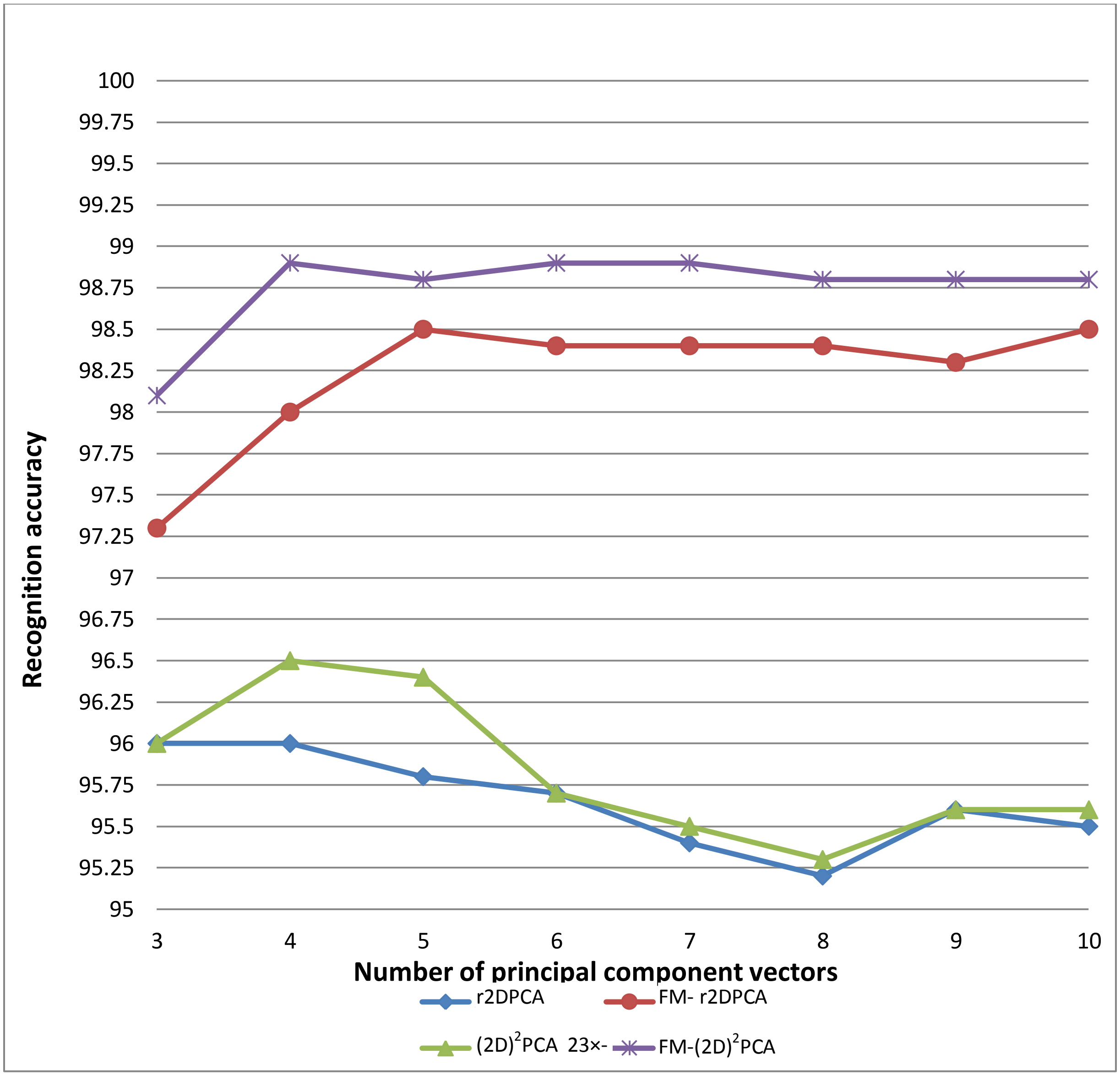


Fig.3.7: Recognition accuracy for the r2DPCA, FM-r2DPCA, $(2D)^2$PCA and FM-$(2D)^2$PCA algorithms using the Yang distance for classification.

We now consider the time complexities for the r2DPCA, FM-r2DPCA, $(2D)^2$PCA and FM-$(2D)^2$PCA algorithms using the Frobenius and Yang distance metrics for classification and the results are given in Tables 3.5 and 3.6, respectively. In these tables, the execution times for data acquisition, feature extraction (consisting of transformation matrix extraction and transformation) and classification, as well as the total time for each algorithm are given. Each algorithm is executed 10 times and this number can be considered sufficient to compute the time complexity, since the algorithms are not data dependent for a fixed image size. It is seen from these tables that, as expected, the time complexities of FM-r2DPCA and FM-$(2D)^2$PCA are higher than that of their spatial domain counterparts. However, these complexities are much lower than that of FM-PCA, as seen from Table 3.2. Further, between the two algorithms, FM-$(2D)^2$PCA has a lower time complexity than that of FM-r2DPCA in view of the much smaller size feature matrix of the former. Thus, FM-$(2D)^2$PCA not only provides the highest recognition accuracy amongst all the PCA algorithms considered, but also has the lowest time complexity amongst all the Fourier magnitude PCA algorithms.

3.5 Summary

In this chapter, a comprehensive simulation study of the classical PCA algorithm and the FM-PCA algorithm, wherein the Fourier magnitudes of the pixel values rather than the raw pixel values are used, has been first undertaken. This study has confirmed that the use of Fourier magnitudes improves substantially the recognition performance of the PCA algorithm, but at the expense of an increased time complexity. This observation coupled with the fact that the two-dimensional extensions of the PCA algorithm provide higher accuracies with lower complexities as compared to PCA, has motivated us to apply the concept of Fourier magnitudes to two 2DPCA algorithms. Using this concept, two new Fourier magnitude based 2DPCA algorithms, FM-r2DPCA and FM-$(2D)^2$PCA, have been developed. Next, extensive simulations of r2DPCA, $(2D)^2$PCA, FM-r2DPCA

Table 3.5: Time complexity in seconds of the r2DPCA, FM- r2DPCA, $(2D)^2$PCA and FM-$(2D)^2$PCA algorithms using the Frobenius distance as the metric.

Algorithm and size of feature matrix	**Time (s)**				
	Data acquisition	**Feature extraction**		**Classification**	**Total**
		Transfer matrix extraction	**Transformation**		
r2DPCA					
(112×4)	0.429	0.052	0.018	0.301	0.800
(112×5)	0.429	0.052	0.019	0.347	0.847
FM-r2DPCA					
$\mathbf{(112 \times 4)}$	**1.132**	**0.052**	**0.018**	**0.301**	**1.503**
$\mathbf{(112 \times 5)}$	**1.132**	**0.052**	**0.019**	**0.347**	**1.550**
$(2D)^2$PCA					
(21×4)	0.429	0.081	0.024	0.053	0.587
(21×5)	0.429	0.081	0.024	0.058	0.592
FM-$\mathbf{(2D)^2}$PCA					
$\mathbf{(21 \times 4)}$	**1.132**	**0.081**	**0.024**	**0.053**	**1.290**
$\mathbf{(21 \times 5)}$	**1.132**	**0.081**	**0.024**	**0.058**	**1.295**
$(2D)^2$PCA					
(23×4)	0.429	0.081	0.024	0.054	0.588
(23×5)	0.429	0.081	0.025	0.058	0.593
FM-$\mathbf{(2D)^2}$PCA					
$\mathbf{(23 \times 4)}$	**1.132**	**0.081**	**0.024**	**0.054**	**1.291**
$\mathbf{(23 \times 5)}$	**1.132**	**0.081**	**0.025**	**0.058**	**1.296**
$(2D)^2$PCA					
(25×4)	0.429	0.081	0.025	0.055	0.590
(25×5)	0.429	0.081	0.026	0.060	0.596
FM-$\mathbf{(2D)^2}$PCA					
$\mathbf{(25 \times 4)}$	**1.132**	**0.081**	**0.025**	**0.055**	**1.293**
$\mathbf{(25 \times 5)}$	**1.132**	**0.081**	**0.026**	**0.060**	**1.299**

Table 3.6: Time complexity in seconds of the r2DPCA, FM- r2DPCA, $(2D)^2$PCA and FM-$(2D)^2$PCA algorithms using the Yang distance as the metric.

Algorithm and size of feature matrix	**Time (s)**				
	Data acquisition	**Feature extraction**		**Classification**	**Total**
		Transfer matrix extraction	**Transformation**		
r2DPCA					
(112 × 4)	0.429	0.052	0.018	0.308	0.807
(112 × 5)	0.429	0.052	0.019	0.354	0.854
FM-r2DPCA					
(112 × 4)	**1.132**	**0.052**	**0.018**	**0.308**	**1.510**
(112 × 5)	**1.132**	**0.052**	**0.019**	**0.354**	**1.557**
$(2D)^2$PCA					
(21 × 4)	0.429	0.081	0.024	0.055	0.589
(21 × 5)	0.429	0.081	0.024	0.061	0.595
FM-$(2D)^2$PCA					
(21 × 4)	**1.132**	**0.081**	**0.024**	**0.055**	**1.292**
(21 × 5)	**1.132**	**0.081**	**0.024**	**0.061**	**1.298**
$(2D)^2$PCA					
(23 × 4)	0.429	0.081	0.024	0.057	0.591
(23 × 5)	0.429	0.081	0.025	0.062	0.597
FM-$(2D)^2$PCA					
(23 × 4)	**1.132**	**0.081**	**0.024**	**0.057**	**1.294**
(23 × 5)	**1.132**	**0.081**	**0.025**	**0.062**	**1.300**
$(2D)^2$PCA					
(25 × 4)	0.429	0.081	0.025	0.059	0.594
(25 × 5)	0.429	0.081	0.026	0.064	0.600
FM-$(2D)^2$PCA					
(25 × 4)	**1.132**	**0.081**	**0.025**	**0.059**	**1.297**
(25 × 5)	**1.132**	**0.081**	**0.026**	**0.064**	**1.303**

and FM-$(2D)^2$PCA have been carried out showing the effectiveness of applying the FM approach to the two-dimensional PCA algorithms in improving their recognition accuracy. Further, the simulation results have also shown that the time complexities of the two proposed algorithms are also lower than that of FM-PCA. Finally, it is pointed out that the FM-$(2D)^2$PCA provides the highest recognition accuracy amongst all the PCA algorithms considered, whether they be 1-D, 2-D, spatial domain or Fourier domain; further, it has the lowest time complexity amongst all the Fourier magnitude PCA algorithms.

CHAPTER 4

Reduced Time Complexity Fourier-Magnitude 2DPCA Algorithms

4.1 Introduction

The approach of using Fourier magnitudes rather than the raw pixel values of an image in 2DPCA algorithms, presented in the previous chapter, substantially improves the recognition accuracies, but this improvement is obtained at the cost of increased time complexity. Since the Fourier magnitudes possess energy compaction property, in this chapter we propose to reduce the time complexity of the Fourier-magnitude 2DPCA algorithms by discarding the high frequency Fourier magnitude coefficients.

4.2 Lowpass Fourier-Magnitude 2DPCA Algorithms

In order to reduce the time complexity of the FM-r2DPCA and FM-$(2D)^2$PCA algorithms, we propose to use the DFT's energy compaction property, i.e., the property that most of the energy of practical signals lies in the low frequency band. Fig. 4.1(a) shows an original image from the ORL database and Fig. 4.1(b) its Fourier magnitude. It is seen from Fig. 4.1(b) that almost all of the Fourier coefficients with high energy are located at the corners, when the coefficients are not centered. Thus, only these Fourier magnitude coefficients, which contain most of the signal energy, can be used to perform the principal component analysis without any appreciable loss in the recognition accuracy. Fig. 4.2 shows how a reduced-size data set is obtained by cropping only a certain number of Fourier magnitude coefficients located at the corners of the complete Fourier magnitude spectrum. It is seen from this figure that the dataset on which the principal component analysis has to be performed gets reduced from $M \times N$ to $\tau \times \tau$.

In the following subsections, we present PCA algorithms based on this new data set, which we will refer to as the lowpass Fourier-magnitude row-directional 2DPCA (LPFM-r2DPCA) and lowpass Fourier-magnitude two-directional 2DPCA (LPFM-(2D)^{2}PCA) algorithms.

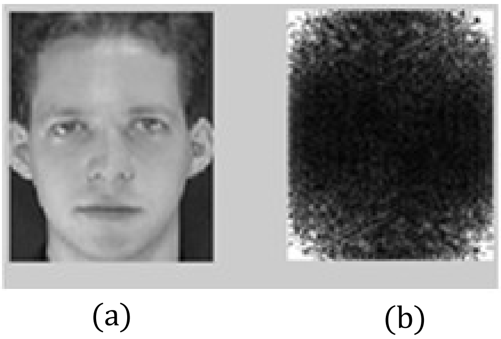


Fig. 4.1: (a) A face image from ORL database and (b) its Fourier magnitude spectrum.

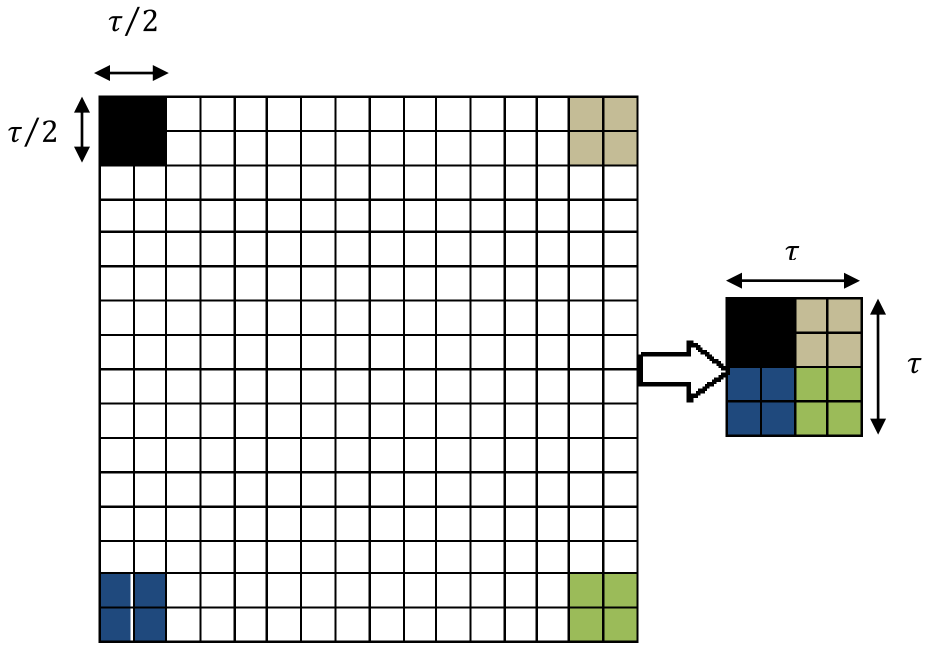


Fig.4.2: How to obtain the new form of the sample using its Fourier magnitudes.

4.2.1 Lowpass Fourier-Magnitude r2DPCA

The Fourier magnitudes for a training or a test image are computed first using (3.2). The magnitudes of the Fourier coefficients for the i^{th} training image and a test image are represented as $M \times N$ matrices given by $\boldsymbol{I}_{\boldsymbol{FM}}^{(\boldsymbol{i})} = \left[I_{FM}^{(i)}(u,v)\right]_{M\times N}$ and $\boldsymbol{I}_{\boldsymbol{FM}}^{(\boldsymbol{test})} = \left[I_{FM}^{(test)}(u,v)\right]_{M\times N}$, respectively. Let $\boldsymbol{I}_{\boldsymbol{LPFM}}^{(\boldsymbol{i})}$ and $\boldsymbol{I}_{\boldsymbol{LPFM}}^{(\boldsymbol{test})}$ represent, respectively, the cropped versions of $\boldsymbol{I}_{\boldsymbol{FM}}^{(\boldsymbol{i})}$ and $\boldsymbol{I}_{\boldsymbol{FM}}^{(\boldsymbol{test})}$, obtained by retaining only the high energy coefficients of the Fourier magnitudes, as explained previously. The new row-directional covariance matrix $\boldsymbol{C}_{\boldsymbol{LPFM-r2DPCA}}$ is computed as

$$\mathbf{C}_{\mathbf{LPFM-r2DPCA}} = \frac{1}{\mathrm{K}}\sum_{\mathrm{i}=1}^{\mathrm{K}}(\mathbf{I}_{\mathbf{LPFM}}^{(\mathbf{i})} - \bar{\mathbf{I}}_{\mathbf{LPFM}})^{\mathrm{T}}(\mathbf{I}_{\mathbf{LPFM}}^{(\mathbf{i})} - \bar{\mathbf{I}}_{\mathbf{LPFM}}) \tag{4.1}$$

where $\bar{\boldsymbol{I}}_{\boldsymbol{LPFM}}$ is the matrix in which each entry is the average of the corresponding entries in the matrices $\boldsymbol{I}_{\boldsymbol{LPFM}}^{(\boldsymbol{i})}$, $i = 1,2,\dots,K$, and is given by

$$\bar{\mathbf{I}}_{\mathbf{LPFM}} = \frac{1}{\mathrm{K}}\sum_{\mathrm{i}=1}^{\mathrm{K}}\mathbf{I}_{\mathbf{LPFM}}^{(\mathbf{i})} \tag{4.2}$$

The eigenvalues and eigenvectors of $\boldsymbol{C}_{\boldsymbol{LPFM-r2DPCA}}$ are then computed. Next, the eigenvectors are normalized and arranged in descending order of the corresponding eigenvalues. Then, the row-directional transformation matrix $\boldsymbol{X}_{\boldsymbol{LPFM}} = \left[\boldsymbol{X}_{\boldsymbol{LPFM}_{\boldsymbol{1}}}\boldsymbol{X}_{\boldsymbol{LPFM}_{\boldsymbol{2}}}\dots\boldsymbol{X}_{\boldsymbol{LPFM}_{\boldsymbol{\beta}}}\right]$ is obtained using a sufficient number of the eigenvectors of $\mathbf{C}_{\mathbf{LPFM-r2DPCA}}$.

The feature matrices for the *i*th training sample and the test sample are, respectively, given by

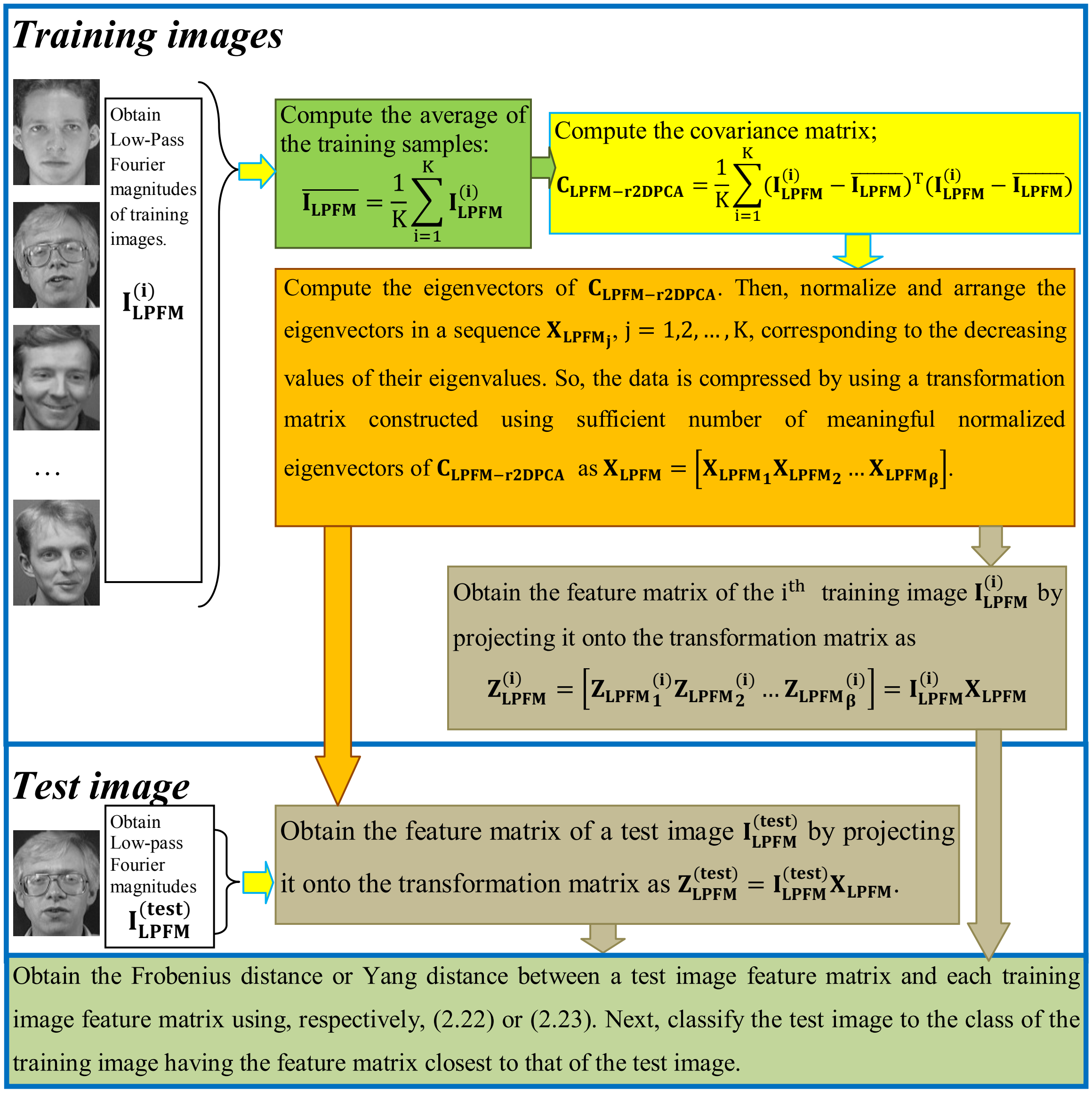
$$\mathbf{Z}_{\mathbf{LPFM}}^{(\mathbf{i})} = \mathbf{I}_{\mathbf{LPFM}}^{(\mathbf{i})}\mathbf{X}_{\mathbf{LPFM}} \tag{4.3}$$

and

$$\mathbf{Z}_{\mathbf{LPFM}}^{(\mathbf{test})} = \mathbf{I}_{\mathbf{LPFM}}^{(\mathbf{test})}\mathbf{X}_{\mathbf{LPFM}} \tag{4.4}$$

For the classification module of the LPFM-r2DPCA algorithm, the Frobenius or the Yang distance metric, given by (2.22) and (2.23), can be used. The distance between the feature matrix of a test sample and that of each of the training samples is obtained.

Then, the subject of the training sample, whose feature matrix has the shortest distance from the feature matrix of the test sample, is assigned as the subject of the test image. A summary of the LPFM-r2DPCA algorithm is given in Fig. 4.3.



Obtain Low-Pass Fourier magnitudes of training images. $\mathbf{I}^{(i)}_{\mathbf{LPFM}}$

Compute the average of the training samples: $\overline{\mathbf{I}_{\mathbf{LPFM}}} = \frac{1}{K}\sum_{i=1}^{K}\mathbf{I}^{(i)}_{\mathbf{LPFM}}$

Compute the covariance matrix; $\mathbf{C}_{\mathbf{LPFM-r2DPCA}} = \frac{1}{K}\sum_{i=1}^{K}(\mathbf{I}^{(i)}_{\mathbf{LPFM}} - \overline{\mathbf{I}_{\mathbf{LPFM}}})^{T}(\mathbf{I}^{(i)}_{\mathbf{LPFM}} - \overline{\mathbf{I}_{\mathbf{LPFM}}})$

Compute the eigenvectors of $\mathbf{C}_{\mathbf{LPFM-r2DPCA}}$. Then, normalize and arrange the eigenvectors in a sequence $\mathbf{X}_{\mathbf{LPFM}_j}$, $j = 1,2,\dots,K$, corresponding to the decreasing values of their eigenvalues. So, the data is compressed by using a transformation matrix constructed using sufficient number of meaningful normalized eigenvectors of $\mathbf{C}_{\mathbf{LPFM-r2DPCA}}$ as $\mathbf{X}_{\mathbf{LPFM}} = \left[\mathbf{X}_{\mathbf{LPFM}_1}\mathbf{X}_{\mathbf{LPFM}_2}\dots\mathbf{X}_{\mathbf{LPFM}_\beta}\right]$.

Obtain the feature matrix of the i^{th} training image $\mathbf{I}^{(i)}_{\mathbf{LPFM}}$ by projecting it onto the transformation matrix as

$$\mathbf{Z}^{(i)}_{\mathbf{LPFM}} = \left[\mathbf{Z}_{\mathbf{LPFM}_1}{}^{(i)}\mathbf{Z}_{\mathbf{LPFM}_2}{}^{(i)}\dots\mathbf{Z}_{\mathbf{LPFM}_\beta}{}^{(i)}\right] = \mathbf{I}^{(i)}_{\mathbf{LPFM}}\mathbf{X}_{\mathbf{LPFM}}$$

Obtain Low-pass Fourier magnitudes $\mathbf{I}^{(\mathbf{test})}_{\mathbf{LPFM}}$

Obtain the feature matrix of a test image $\mathbf{I}^{(\mathbf{test})}_{\mathbf{LPFM}}$ by projecting it onto the transformation matrix as $\mathbf{Z}^{(\mathbf{test})}_{\mathbf{LPFM}} = \mathbf{I}^{(\mathbf{test})}_{\mathbf{LPFM}}\mathbf{X}_{\mathbf{LPFM}}$.

Obtain the Frobenius distance or Yang distance between a test image feature matrix and each training image feature matrix using, respectively, (2.22) or (2.23). Next, classify the test image to the class of the training image having the feature matrix closest to that of the test image.

Fig.4.3: Summary of the LPFM-r2DPCA algorithm.

4.2.2 Lowpass Fourier-Magnitude (2D)^{2}PCA

As in the case of the LPFM-r2DPCA algorithm, the row-directional covariance matrix $\boldsymbol{C_{LPFM-r2DPCA}}$ is first obtained using (4.1) and its column-directional counterpart is then computed as

$$\mathbf{C_{LPFM-c2DPCA}} = \frac{1}{K}\sum_{i=1}^{K}(\mathbf{I_{LPFM}^{(i)}} - \bar{\mathbf{I}}_{\mathbf{LPFM}})(\mathbf{I_{LPFM}^{(i)}} - \bar{\mathbf{I}}_{\mathbf{LPFM}})^{T} \tag{4.5}$$

The eigenvalues and eigenvectors of $\boldsymbol{C_{LPFM-r2DPCA}}$ and $\boldsymbol{C_{LPFM-c2DPCA}}$ are then computed. The eigenvectors of each of the two covariance matrices are normalized and arranged in descending order of the corresponding eigenvalues. The row-directional and column-directional transformation matrices $\boldsymbol{X_{LPFM}} = \left[\boldsymbol{X_{LPFM_1}}\boldsymbol{X_{LPFM_2}} \dots \boldsymbol{X_{LPFM_\beta}}\right]$ and $\boldsymbol{Y_{LPFM}} = \left[\boldsymbol{Y_{LPFM_1}}\boldsymbol{Y_{LPFM_2}} \dots \boldsymbol{Y_{LPFM_\alpha}}\right]$ are obtained using sufficient numbers of the eigenvectors of $\mathbf{C_{LPFM-r2DPCA}}$ and $\mathbf{C_{LPFM-c2DPCA}}$, respectively.

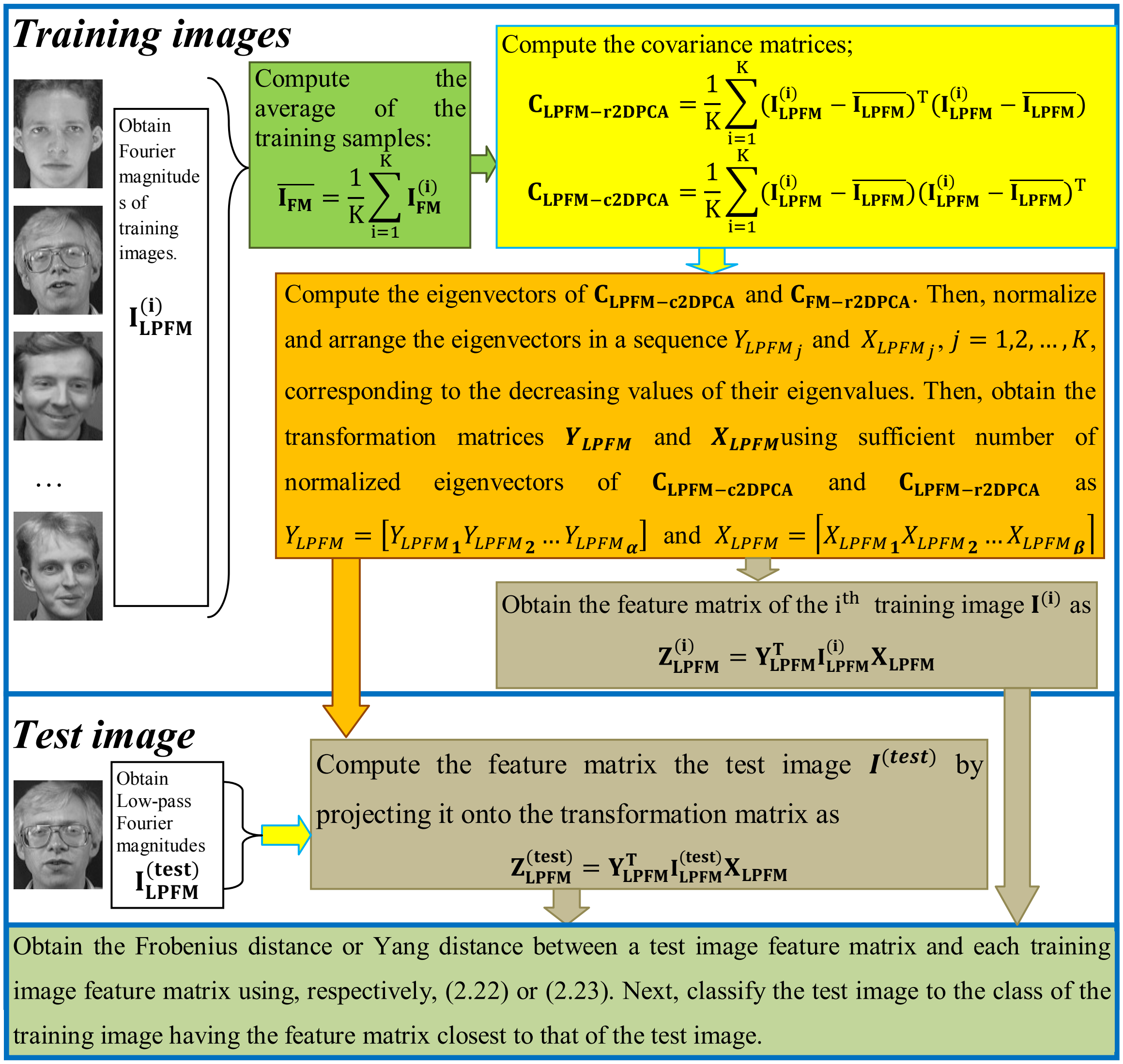
The feature matrices for the i^{th} training sample and the test sample are, respectively, given by

$$\mathbf{Z_{LPFM}^{(i)}} = \mathbf{Y_{LPFM}^{T}}\mathbf{I_{LPFM}^{(i)}}\mathbf{X_{LPFM}} \tag{4.6}$$

and

$$\mathbf{Z_{LPFM}^{(test)}} = \mathbf{Y_{LPFM}^{T}}\mathbf{I_{LPFM}^{(test)}}\mathbf{X_{LPFM}}\mathbf{X} \tag{4.7}$$

For classification, we again employ the Frobenius as well as the Yang distance metrics given by (2.22) and (2.23), respectively. The distance between the feature matrix of a test sample and that of each of the training samples is obtained. Then, the subject of the training sample, whose feature matrix has the shortest distance from the feature matrix of the test sample, is assigned as the subject of the test image. A summary of the LPFM-(2D)^{2}PCA algorithm is given in Fig.4.4.



Training images
Obtain Fourier magnitudes of training images.
$\mathbf{I}_{\mathbf{LPFM}}^{(\mathbf{i})}$
…
Compute the average of the training samples:
$\overline{\mathbf{I}_{\mathbf{FM}}} = \frac{1}{K}\sum_{i=1}^{K}\mathbf{I}_{\mathbf{FM}}^{(\mathbf{i})}$
Compute the covariance matrices;
$\mathbf{C}_{\mathbf{LPFM-r2DPCA}} = \frac{1}{K}\sum_{i=1}^{K}(\mathbf{I}_{\mathbf{LPFM}}^{(\mathbf{i})} - \overline{\mathbf{I}_{\mathbf{LPFM}}})^{T}(\mathbf{I}_{\mathbf{LPFM}}^{(\mathbf{i})} - \overline{\mathbf{I}_{\mathbf{LPFM}}})$
$\mathbf{C}_{\mathbf{LPFM-c2DPCA}} = \frac{1}{K}\sum_{i=1}^{K}(\mathbf{I}_{\mathbf{LPFM}}^{(\mathbf{i})} - \overline{\mathbf{I}_{\mathbf{LPFM}}})(\mathbf{I}_{\mathbf{LPFM}}^{(\mathbf{i})} - \overline{\mathbf{I}_{\mathbf{LPFM}}})^{T}$
Compute the eigenvectors of $\mathbf{C}_{\mathbf{LPFM-c2DPCA}}$ and $\mathbf{C}_{\mathbf{FM-r2DPCA}}$. Then, normalize and arrange the eigenvectors in a sequence Y_{LPFM_j} and X_{LPFM_j}, $j = 1,2,\dots,K$, corresponding to the decreasing values of their eigenvalues. Then, obtain the transformation matrices $\boldsymbol{Y_{LPFM}}$ and $\boldsymbol{X_{LPFM}}$using sufficient number of normalized eigenvectors of $\mathbf{C}_{\mathbf{LPFM-c2DPCA}}$ and $\mathbf{C}_{\mathbf{LPFM-r2DPCA}}$ as $Y_{LPFM} = [Y_{LPFM_1} Y_{LPFM_2} \dots Y_{LPFM_\alpha}]$ and $X_{LPFM} = [X_{LPFM_1} X_{LPFM_2} \dots X_{LPFM_\beta}]$
Obtain the feature matrix of the ith training image $\mathbf{I}^{(\mathbf{i})}$ as
$\mathbf{Z}_{\mathbf{LPFM}}^{(\mathbf{i})} = \mathbf{Y}_{\mathbf{LPFM}}^{\mathbf{T}}\mathbf{I}_{\mathbf{LPFM}}^{(\mathbf{i})}\mathbf{X}_{\mathbf{LPFM}}$
Test image
Obtain Low-pass Fourier magnitudes
$\mathbf{I}_{\mathbf{LPFM}}^{(\mathbf{test})}$
Compute the feature matrix the test image $\boldsymbol{I^{(test)}}$ by projecting it onto the transformation matrix as
$\mathbf{Z}_{\mathbf{LPFM}}^{(\mathbf{test})} = \mathbf{Y}_{\mathbf{LPFM}}^{\mathbf{T}}\mathbf{I}_{\mathbf{LPFM}}^{(\mathbf{test})}\mathbf{X}_{\mathbf{LPFM}}$
Obtain the Frobenius distance or Yang distance between a test image feature matrix and each training image feature matrix using, respectively, (2.22) or (2.23). Next, classify the test image to the class of the training image having the feature matrix closest to that of the test image.

Fig.4.4: Summary of the LPFM-(2D)^{2}PCA algorithm.

4.3 Experimental Results

In this section, detailed simulation results for the LPFM-r2DPCA and LPFM-$(2D)^2$PCA algorithms are provided and the results compared with those of FM-r2DPCA and FM-$(2D)^2$PCA, in terms of the recognition accuracy and the computational complexity. In order to obtain the simulation results, the hardware and software platforms described in Section 3.2.2 are utilized. In order to determine a suitable value for the cropping parameter τ (see Fig. 4.2), extensive experiments were carried out to find the energy content of the cropped magnitude spectrum of the images in the ORL database for various values of τ. From these experiments it was found that, on an average, 99% of the energy of the original image is contained in the first 26×26 lowpass coefficients of the magnitude spectrum of the image. In view of this, $\tau = 26$ is used in all the simulations performed to determine the recognition accuracy and the time complexity of the two algorithms proposed in this chapter.

As mentioned earlier, we assign indices 1 to 10 for each subject in the ORL database starting with the left most image to the right most image (see Fig. 3.2). A set of 5 indices are chosen randomly and the five images in each subject corresponding to these indices are chosen as the training images (thus, constituting a total of 200 training images) to train a given algorithm. For each of the training image, the corresponding feature matrix is obtained. For a given test image, the Frobenius or the Yang distance between the feature matrix of the test image and that of each of the 200 feature matrices of the training images is obtained and the one with the closest distance chosen to determine the class of the test image under consideration. This is repeated for each of the 200 test images. The above procedure is repeated, each time choosing randomly a set of 5 indices that is distinct from the sets of the previous runs of the algorithm. Finally, the recognition accuracy of the algorithm is obtained as the average of the recognition accuracies derived from 50 runs.

Tables 4.1 and 4.2 give the recognition accuracies of the FM-r2DPCA and LPFM-r2DPCA algorithms using, respectively, the Frobenius and Yang distance metrics.

Table 4.1: Recognition accuracy in percentage of the FM-r2DPCA and LFPM-r2DPCA algorithms using the Frobenius distance as the metric for classification.

Algorithm and size of feature matrix	**Recognition accuracy**							
	Number of principal component vectors (β)							
	3	**4**	**5**	**6**	**7**	**8**	**9**	**10**
FM-r2DPCA (112 × β)	97.2	97.5	98.0	97.8	98.0	97.8	98.0	97.9
LPFM-r2DPCA (24 × β)	97.0	97.4	98.0	97.8	98.0	97.9	97.8	97.7
LPFM-r2DPCA (26 × β)	**97.1**	**97.6**	**98.0**	**98.0**	**98.0**	**97.8**	**98.0**	**97.9**
LPFM-r2DPCA (28 × β)	97.2	97.4	98.1	98.0	98.0	98.0	98.0	98.1

Table 4.2: Recognition accuracy in percentage of the FM-r2DPCA and LFPM-r2DPCA algorithms using the Yang distance as the metric for classification.

Algorithm and size of feature matrix	**Recognition accuracy**							
	Number of principal component vectors (β)							
	3	**4**	**5**	**6**	**7**	**8**	**9**	**10**
FM-r2DPCA ($112 \times \beta$)	97.3	98.0	98.5	98.4	98.4	98.4	98.3	98.5
LPFM-r2DPCA ($24 \times \beta$)	97.1	97.9	98.3	98.3	98.4	98.3	98.3	98.3
LPFM-r2DPCA ($26 \times \beta$)	**97.4**	**98.0**	**98.3**	**98.5**	**98.4**	**98.4**	**98.4**	**98.5**
LPFM-r2DPCA ($28 \times \beta$)	97.4	98.1	98.5	98.5	98.5	98.4	98.5	98.4

It is seen from these tables that, when $\tau = 26$, the LPFM-r2DPCA algorithm provides about the same recognition accuracy compared to that provided by FM-r2DPCA, regardless of whether the Frobenius or the Yang distance is used as the metric. It is also seen that the accuracy is reduced when τ is decreased to 24, whereas it is not improved if τ increased to 28, showing that the choice of $\tau = 26$ is most appropriate from the point of view of accuracy. Therefore, only 676 lowpass Fourier-magnitude coefficients are used for feature extraction rather than 10304 coefficients.

For the purpose of determining the time complexity, the two algorithms are executed 10 times and the average time taken. The time complexities for the FM-r2DPCA and LPFM-r2DPCA algorithms using the Frobenius and Yang distance metrics for classification and the results are given in Tables 4.3 and 4.4, respectively. It is seen from these tables that, as expected, the time complexity of LPFM-r2DPCA is significantly smaller than that of FM-r2DPCA, whether the Frobenius or Yang distance is used for classification and the complexity increases with increasing value of τ. This fact coupled with our earlier observation regarding accuracy, it is seen that there is a reduction of at least 36% in time complexity with little loss in the accuracy for $\tau = 26$.

Table 4.3: Time complexity in seconds of the FM-r2DPCA and LPFM-r2DPCA algorithms using the Frobenius distance as the metric.

Algorithm and size of feature matrix	**Time (s)**				
	Data acquisition	**Feature extraction**		**Classification**	**Total**
		Transfer matrix extraction	**Transformation**		
FM-r2DPCA (112 × 5)	1.132	0.052	0.019	0.347	1.550
LPFM-r2DPCA (24 × 5)	0.880	0.010	0.005	0.061	0.956
LPFM-r2DPCA **(26 × 5)**	**0.882**	**0.010**	**0.005**	**0.065**	**0.962**
LPFM-r2DPCA (28 × 5)	0.885	0.010	0.005	0.067	0.967

Table 4.4: Time complexity in seconds of the FM-r2DPCA and LPFM-r2DPCA algorithms using the Yang distance as the metric.

Algorithm and size of feature matrix	**Time (s)**				
	Data acquisition	**Feature extraction**		**Classification**	**Total**
		Transfer matrix extraction	**Transformation**		
FM-r2DPCA (112 × 5)	1.132	0.052	0.019	0.354	1.557
LPFM-r2DPCA (24 × 5)	0.880	0.010	0.005	0.069	0.964
LPFM-r2DPCA (26 × 5)	**0.882**	**0.010**	**0.005**	**0.080**	**0.977**
LPFM-r2DPCA (28 × 5)	0.885	0.010	0.005	0.084	0.984

Tables 4.5 and 4.6 give the recognition accuracies of the FM-(2D)^{2}PCA and LPFM-(2D)^{2}PCA algorithms using, respectively, the Frobenius and Yang distance metrics. Tables 4.7 and 4.8 give time complexities for the algorithms corresponding to the two metrics. It is seen from these tables that, as in the case of LPFM-r2DPCA, the recognition accuracy of LPFM-(2D)^{2}PCA remains about the same as that of FM-(2D)^{2}PCA when $\tau = 26$, with a reduction in time complexity of about 25.5%. When τ is decreased to 24, the time complexity is slightly reduced, but with a lower recognition accuracy. On the other hand, when τ is increased to 28, the time complexity is increased with little gain in recognition accuracy.

Table 4.5: Recognition accuracy in percentage of the FM-(2D)^{2}PCA and LPFM-(2D)^{2}PCA algorithms using the Frobenius distance as the metric for classification.

Algorithm, data size and size of feature matrix	**Recognition accuracy**							
	Number of principal component vectors (β)							
	3	**4**	**5**	**6**	**7**	**8**	**9**	**10**
FM-(2D)^{2}PCA (23 × β)	98.3	98.2	98.3	98.3	98.2	98.2	98.2	98.2
LPFM-(2D)^{2}PCA ($\tau = 24$) (23 × β)	97.9	98.0	98.0	98.0	98.0	98.0	98.1	98.1
LPFM-(2D)^{2}PCA ($\tau = 26$) (23 × β)	**98.3**	**98.3**	**98.2**	**98.3**	**98.3**	**98.1**	**98.2**	**98.0**
LPFM-(2D)^{2}PCA ($\tau = 28$) (23 × β)	98.3	98.3	98.3	98.3	98.3	98.2	98.2	98.3

Table 4.6: Recognition accuracy in percentage of the FM-(2D)^{2}PCA and LPFM-(2D)^{2}PCA algorithms using the Yang distance as the metric for classification.

Algorithm, data size and size of feature matrix	**Recognition accuracy**							
	Number of principal component vectors (β)							
	3	**4**	**5**	**6**	**7**	**8**	**9**	**10**
FM-(2D)^{2}PCA (23 × β)	98.1	98.9	98.8	98.9	98.9	98.8	98.8	98.8
LPFM-(2D)^{2}PCA ($\tau = 24$) (23 × β)	98.2	98.2	98.4	98.3	98.2	98.5	98.5	98.6
LPFM-(2D)^{2}PCA ($\tau = 26$) (23 × β)	**98.2**	**98.7**	**98.6**	**98.8**	**98.9**	**98.7**	**98.8**	**98.7**
LPFM-(2D)^{2}PCA ($\tau = 28$) (23 × β)	98.2	98.9	98.9	98.9	98.8	98.8	98.9	98.7

Table 4.7: Time complexity in seconds of the FM-(2D)^{2}PCA and LPFM-(2D)^{2}PCA algorithms using the Frobenius distance as the metric.

Algorithm and size of feature matrix	**Time (s)**				
	Data acquisition	**Feature extraction**		**Classification**	**Total**
		Transfer matrix extraction	**Transformation**		
FM-(2D)^{2}PCA $(23 \times \beta)$	1.132	0.081	0.025	0.058	1.296
LPFM-(2D)^{2}PCA $(\tau = 24)$ $(23 \times \beta)$	0.880	0.012	0.007	0.058	0.957
LPFM-(2D)^{2}PCA $(\tau = 26)$ $(23 \times \beta)$	**0.882**	**0.013**	**0.007**	**0.058**	**0.960**
LPFM-(2D)^{2}PCA $(\tau = 28)$ $(23 \times \beta)$	0.885	0.013	0.007	0.058	0.963

Table 4.8: Time complexity in seconds of the FM-(2D)^{2}PCA and LPFM-(2D)^{2}PCA algorithms using the Yang distance as the metric.

Algorithm and size of feature matrix	**Time (s)**				
	Data acquisition	**Feature extraction**		**Classification**	**Total**
		Transfer matrix extraction	**Transformation**		
FM-(2D)^{2}PCA $(23 \times \beta)$	1.132	0.081	0.025	0.062	1.300
LPFM-(2D)^{2}PCA $(\tau = 24)$ $(23 \times \beta)$	0.880	0.012	0.007	0.062	0.961
LPFM-(2D)^{2}PCA $(\tau = 26)$ $(23 \times \beta)$	**0.882**	**0.013**	**0.007**	**0.062**	**0.964**
LPFM-(2D)^{2}PCA $(\tau = 28)$ $(23 \times \beta)$	0.885	0.013	0.007	0.062	0.967

In conclusion, the simulation results show that the LPFM approach for both the r2DPCA and $(2D)^2$PCA algorithms can provide more or less the same recognition accuracy as compared to that provided by FM-r2DPCA and FM-$(2D)^2$PCA, respectively, but at a substantially reduced time complexity. It is further noted that the Yang distance metric gives better recognition accuracy than that using the Frobenius distance metric, with about the same time complexity. This result is consistent with our findings in Chapter 3 as well as that reported in [19].

4.4 Summary

In this chapter, using the energy compaction property of Fourier magnitudes, FM-r2DPCA and FM-$(2D)^2$PCA algorithms proposed in Chapter 3 are further developed to reduce their time complexities without an appreciable loss in the recognition accuracy. Towards this goal, a new dataset containing an appropriate number of lowpass Fourier magnitudes is first obtained. Based on extensive experiments, it was established that, on an average, 99% of the energy of the original image is contained in the first 26×26 lowpass coefficients of the images in the ORL database.

Next, extensive simulations of LPFM-r2DPCA and LPFM-$(2D)^2$PCA have been carried out showing the effectiveness of these algorithms over the FM-r2DPCA and FM-$(2D)^2$PCA algorithms in substantially reducing the time complexity with little loss in the recognition accuracy. The simulation results have also shown that LPFM-$(2D)^2$PCA has the lowest time complexity amongst all these algorithms.

Chapter 5

Conclusion and Future Work

5.1 Concluding Remarks

Human identification systems are becoming increasingly popular in view of their importance of security, law-enforcement and identity verification. For human identification, specific information, such as that contained in fingerprint, palmprint, signature, voice, iris, DNA and face, is used. Face recognition is one of the most popular techniques used for human identification and has led to the development of fast and highly accurate algorithms. In practice, face recognition systems need to work on very large size of datasets obtained under various challenging conditions.

Principal component analysis (PCA) is a widely used technique for face recognition. However, it has major drawbacks of losing image details, having a large time complexity and suffering from the adverse effect of intra-class pose variations resulting in reduced recognition accuracy. To overcome the problem of intra-class pose variations, Fourier magnitudes have been used for feature extraction in the PCA algorithm giving rise to the so-called FM-PCA algorithm. However, the time complexity of this algorithm is even higher. On the other hand, to address the other two drawbacks of the PCA algorithm, two-dimensional PCA (2DPCA) algorithms have been proposed.

In this thesis, incorporating the advantages of FM-PCA in improving the recognition accuracy and that of 2DPCA algorithms in improving the accuracy as well as the time complexity, new algorithms have been proposed. Then, the proposed algorithms have been further investigated in view of the energy compaction property of the Fourier magnitudes of the image pixels.

In Chapter 3, two-dimensional PCA algorithms, that utilize the Fourier magnitudes for feature extraction and incorporate the advantages of the FM-PCA and two-dimensional PCA algorithms, have been developed [30]. Extensive simulations have been conducted using the Olivetti Research Laboratory (ORL) face database, a benchmark database in the literature, in which the intra-class pose variations are substantial. Simulation results have shown that applying the Fourier magnitude concept on the two-dimensional PCA algorithms results in improving their recognition accuracy. It has also been shown that the new algorithms have higher recognition accuracy and lower time complexity as compared to FM-PCA.

In Chapter 4, by taking advantage of the energy compaction property of the Fourier transform, the algorithms proposed in Chapter 3 have been further investigated to develop new algorithms that significantly reduce the computational complexities with little loss in the recognition accuracy [31]. This has been achieved by utilizing only a small fraction of the lowpass Fourier-magnitude coefficients of image data rather than all the coefficients for feature extraction. Extensive experiments have been conducted to demonstrate the effectiveness of these algorithms in terms of their recognition accuracy and time complexity.

In conclusion, this study has demonstrated the effectiveness of using Fourier magnitudes rather than the raw pixel values in two-dimensional PCA algorithms in substantially improving their recognition accuracy. Further, it has also been demonstrated that the use of the energy compaction property of practical signals can significantly reduce the time complexity of the Fourier magnitude versions of two-dimensional algorithms with little loss in their recognition accuracy.

5.2 Future Work

The concept of using the Fourier magnitudes rather than the raw pixel values in conjunction with the utilization of the energy compaction property can also be applied to two-dimensional linear discriminant analysis algorithms. It can also be applied in developing algorithms for palmprint and fingerprint recognition problems, as well as for recognition problems using fused biometric data.

References

[1] John D. Woodward Jr., Nicholas M. Orlans and Peter T. Higgins, *Biometrics: Identity Assurance in the Information Age*, McGraw-Hill Osborne Media, 2002.

[2] James Wayman, Anil Jain, Davide Maltoni and Dario Maio, *Biometric Systems: Technology, Design and Performance Evaluation*, Springer, 2004.

[3] Anil K. Jain, Patrick Flynn and Arun A. Ross, *Handbook of Biometrics*, Springer, 2008.

[4] N. K. Ratha and Venu Govindaraju, *Advances in Biometrics Sensors, Algorithms and Systems*, Springer, 2008.

[5] Massimo Tistarelli, Stan Z. Li and Rama Chellappa, *Handbook of Remote Biometrics for Surveillance and Security*, Springer, 2009.

[6] Anil K. Jain, Arun A. Ross and Karthik Nandakumar, *Introduction to Biometrics*, Springer, 2011.

[7] Harry Wechsler, *Reliable Face Recognition Methods: System Design, Implementation and Evaluation*, Springer, 2007.

[8] Riad Hammoud, Besma Abidi, Mongi A. Abidi, *Face Biometrics for Personal Identification: Multi-Sensory Multi-Modal Systems*, Springer-Verlag Berlin Heidelberg, 2007.

[9] Daijin Kim, Jaewon Sung, *Automated Face Analysis: Emerging Technologies and Research*, IGI Publishing Hershey, 2009.

[10] M. Turk and A. Pentland, "Eigenfaces for Recognition," *Journal of Cognitive Neuroscience,* vol. 3, no. 1, pp. 71-86, 1991.

[11] Yang, J. and Yang, J.Y. "From image vector to matrix: a straightforward image projection technique - IMPCA vs. PCA," *Pattern Recognition*, vol. 35, pp.1997-1999, 2002.

[12] Jian Yang, David Zhang, Alejandro F. Frangi, Jing-yu Yang, "Two-Dimensional PCA: A New Approach to Appearance-Based Face Representation and Recognition", *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 26 no. 1, p.131-137, 2004.

[13] D. Q. Zhang and Z. H. Zhou, "(2D) PCA: Two-directional two-dimensional PCA for efficient face representation and recognition," *Neurocomputing*, vol. 69, pp. 224–231, 2005.

[14] L. Sirovich and M. Kirby, "Low-Dimensional Procedure for Characterization of Human Faces," *J. Optical Soc. Am. A*, vol. 4, pp. 519-524, 1987.

[15] M. Kirby and L. Sirovich, “Application of the KL Procedure for the Characterization of Human Faces,” *IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 12, no. 1*, pp. 103-108, Jan. 1990

[16] Ramamurthy Bhagavatula and Marios Savvides, "Eigen and Fisher-Fourier Spectra for Shift Invariant Pose-Tolerant Face Recognition,” *Pattern Recognition and Image Analysis*, Springers, vol. 3687/2005*,* pp. 351-359, 2005.

[17] ORL, 1992. The ORL face database at the AT&T (Olivetti) Research Laboratory. Available from: <http://www.uk.research.att.com/facedatabase.html>.

[18] D. Zhang, Z. H. Zhou and S. Chen, "Diagonal Principal Component Analysis for Face Recognition", *Pattern Recognition*, 39(1): pp. 140-142, 2006.

[19] Bengherabi M., Mezai L., Harizi F., Cheriet M., Guessoum A, "Face recognition based on 2DPCA, DIAPCA and DIA2DPCA in DCT domain," *The 5th International Multi Conference on Systems, Signals and Devices, IEEE SSD*, Amman, Jordan, July 20-23, 2008.

[20] G. C. Feng, P. C. Yuen, and D. Q. Dai, "Human face recognition using PCA on wavelet subband," *Journal of Electronic Imaging*, vol. 9, no. 2, pp.226 - 233, 2000.

[21] W. Chen, M. J. Er and S. Wu, "PCA and LDA in DCT Domain," *Pattern Recognition Letters*, Vol. 26, No. 15, pp. 2474-2482, 2005.

[22] Hong Wang, Su Yang and Wei Liao ,"An Improved PCA Face Recognition Algorithm Based on the Discrete Wavelet Transform and the Support Vector Machines," *International Conference on Computational Intelligence and Security Workshops*, pp. 308 - 311, 2007.

[23] P. Nicholl and A. Amira, "DWT/PCA face recognition using automatic coefficient selection," *The 4th IEEE International Workshop on Electronic Design, Test and Applications (DELTA '08)*, pp. 390-393, Hong Kong, January 2008.

[24] Mohammed Alwakeel and Zyad Shaaban, "Face Recognition Based on Haar Wavelet Transform and Principal Component Analysis via Levenberg-Marquardt Backpropagation Neural Network," *European Journal of Scientific Research ISSN 1450-216X*, Vol.42 No.1, pp.25-31, 2010.

[25] K.-L. Du and M.N.S. Swamy, *Neural networks in a softcomputing framework*, Springer, 2006.

[26] Moataz M. Abdelwahab, and Wasfy B. Mikhael, "Recognition of Noisy Facial Images Employing Transform-Domain Two-Dimensional Principal Component Analysis," *The 49th IEEE International MWSCAS*, San Juan, Puerto Rico, August 6-9, 2006.

[27] Sergios Theodoridis, Konstantinos Koutroumbas, *Pattern recognition*, Academic Press, 2007.

[28] Rafael C. González and Richard Eugene Woods, *Digital Image Processing*, 3rd ed., Prentice Hall, 2008.

[29] Matlab, *The Language of Technical Computing*, Version 7.9, 2009, [Online] http://www.mathworks.com.

[30] Alper S. Zeytunlu, M. Omair Ahmad and M.N.S. Swamy, "Two-Dimensional Face Recognition Algorithms in the Frequency Domain," *in Proceedings of the 25th IEEE Canadian Conference on Electrical and Computer Engineering,* Montreal, Quebec, April 29-May 2, 2012.

[31] Alper S. Zeytunlu, M. Omair Ahmad and M.N.S. Swamy, "A Frequency-Domain Face Recognition Algorithm with Reduced Time Complexity," submitted to *the 10th IEEE International NEWCAS Conference*, Montreal, Quebec, June 17-20, 2012.