11-5-2010

An Incremental Multilinear System for Human Face Learning and Recognition

Jin Wang
Florida International University, jwang006@fiu.edu

DOI: 10.25148/etd.FI10112008

FLORIDA INTERNATIONAL UNIVERSITY

Miami, Florida

AN INCREMENTAL MULTILINEAR SYSTEM FOR HUMAN FACE LEARNING AND RECOGNITION

A dissertation submitted in partial fulfillment of the

requirements for the degree of

DOCTOR OF PHILOSOPHY

in

ELECTRICAL ENGINEERING

by

Jin Wang

2010

To: Dean Amir Mirmiran
College of Engineering and Computing

This dissertation, written by Jin Wang, and entitled An Incremental Multilinear System for Human Face Learning and Recognition, having been approved in respect to style and intellectual content, is referred to you for judgment.

We have read this dissertation and recommend that it be approved.

Armando Barreto

Jean Andrian

Naphtali Rishe

Malek Adjouadi, Major Professor

Date of Defense: November 5, 2010

The dissertation of Jin Wang is approved.

Dean Amir Mirmiran
College of Engineering and Computing

Interim Dean Kevin O'Shea
University Graduate School

Florida International University, 2010

DEDICATION

I dedicate this dissertation to my parents. Without their support, understanding and love, such a milestone in my life can not be achieved.

ACKNOWLEDGMENTS

I’m appreciative to have such an excellent advisor, Dr. Malek Adjouadi. He broadened my research interests and advised me since I became a member of the CATE center. As a friend, he taught me the meaning of being a Ph.D. student and encouraged me in having my research work published in major venues. I really appreciate his unconditional support, guidance and patience. I also would like to express my gratitude and thanks to my committee members, Dr. Armando Barreto, Dr. Jean Andrian, and Dr. Naphtali Rishe, for all the help they provided and all the recommendations they have made over the years. I thank Ms. Xiaozhen You for being such a dear colleague and a friend, and for sharing novel ideas that inspired me a lot and improved the progress of my work. I am grateful to Miao, Fang, Thao, Yuling and many other friends for their encouragements to achieve such a milestone.

Most importantly, I appreciate the support provided by the National Science Fondation under grants CNS-0540592, CNC-0837556, HRD-0833093, CNS-0426125, and the Disseration Year Fellowship provided by the University Graduate School at Florida International University.

ABSTRACT OF THE DISSERTATION

AN INCREMENTAL MULTILINEAR SYSTEM FOR HUMAN FACE LEARNING AND RECOGNITION

by

Jin Wang

Florida International University, 2010

Miami, Florida

Professor Malek Adjouadi, Major Professor

This dissertation establishes a novel system for human face learning and recognition based on incremental multilinear Principal Component Analysis (PCA). Most of the existing face recognition systems need training data during the learning process. The system as proposed in this dissertation utilizes an unsupervised or weakly supervised learning approach, in which the learning phase requires a minimal amount of training data. It also overcomes the inability of traditional systems to adapt to the testing phase as the decision process for the newly acquired images continues to rely on that same old training data set. Consequently when a new training set is to be used, the traditional approach will require that the entire eigensystem will have to be generated again. However, as a means to speed up this computational process, the proposed method uses the eigensystem generated from the old training set together with the new images to generate more effectively the new eigensystem in a so-called incremental learning process.

In the empirical evaluation phase, there are two key factors that are essential in evaluating the performance of the proposed method: (1) recognition accuracy and (2) computational complexity. In order to establish the most suitable algorithm for this research, a comparative analysis of the best performing methods has been carried out first. The results of the comparative analysis advocated for the initial utilization of the multilinear PCA in our research. As for the consideration of the issue of computational complexity for the subspace update procedure, a novel incremental algorithm, which combines the traditional sequential Karhunen-Loeve (SKL) algorithm with the newly developed incremental modified fast PCA algorithm, was established. In order to utilize the multilinear PCA in the incremental process, a new unfolding method was developed to affix the newly added data at the end of the previous data. The results of the incremental process based on these two methods were obtained to bear out these new theoretical improvements. Some object tracking results using video images are also provided as another challenging task to prove the soundness of this incremental multilinear learning method.

TABLE OF CONTENTS

LIST OF TABLES

TABLE PAGE

LIST OF FIGURES

FIGURE PAGE

CHAPTER I

Introduction

1.1 General Statement of the Problem Area

This research establishes an optimization system for human face learning and recognition. Currently, the learning and recognition processes are usually applied as two separate modules in most applications. How to combine learning and recognition processes for human faces effectively is still an emergent application.

This research uses incremental multilinear Principal Component Analysis (IMPCA) and distance-based classification method for face recognition. For existing face recognition systems, most of them need a training set in the learning process. However, the system as proposed utilizes an unsupervised or weakly supervised learning process, in other words, requiring no training set or at best a small training set initially. Moreover, the traditional systems can only recognize the testing image but can not learn from the testing image in subsequent recognition task. If new data is required to be included into the system, the new training set has to be generated again. Learning from images is another added feature of the proposed system.

1.2 Research Problem

The objective of this study is to seek an effective and integrated system that will realize both the learning and recognition processes in one setting. Moreover, the algorithms and overall approach utilized for the two processes need to be validated across large data sets containing varied faces under different circumstances. These algorithms if they are to be effective would have to yield higher recognition accuracy, faster subspace update speed and a quicker learning phase. Due to all these research aspects, the subspace update method is based on the incremental multilinear PCA and the distance-based classification method is determined to be the method to be used for the classification process.

1.3 Significance of the Study

In recent years, remarkable efforts have been extended into the face recognition problem, especially with the considerable accessibility to new technologies and the wide range of commercial applications that have become available.

Common sense dictates that all automatic face recognition systems should include two key steps. The first step is face detection and feature extraction, which is necessary to locate the face position and obtain the face features in the image for further processing. The features obtained will then be fed into the second and more challenging step that of face recognition. The recognition process remains a challenging endeavor for researchers due to the myriad of faces that can be considered and the variability in the circumstances and ways under which the images of these faces are taken.

Therefore, face recognition is considered the focal point of this research. The system as built defines how the learning and recognition phases are integrated into one system as higher recognition accuracy and faster processing time are sought.

1.4 Structure of the Research

In structuring this dissertation, a comparison of the conventional methodologies used for face recognition is introduced in Chapter 2. An overview of the software and how it is used to process and analyze the face data sets is provided. The dataset utilized in this chapter is introduced briefly. This chapter also discusses the general application steps of different methods and their ensuing computational complexities.

Chapter 3 presents the so-called modified fast principal component analysis, method with the purpose of applying it into the incremental subspace update algorithm. By comparing the similarity among the eigenvalue decomposition method, the fast principal component analysis and the modified fast principal component analysis, the advantages of the modified method is discussed and the reason why the modification is necessary to improve the performance of the algorithm is given.

Chapter 4 looks into the incremental subspace algorithm based on the performance issues raised in chapters 2 and 3. The multilinear principal component analysis has been chosen due to its better recognition accuracy and fast processing time. In order to improve the processing speed, the modified fast principal component analysis and sequencial Karhunen-Loeve are then combined to complete the mathematical foundation of the incremental multilinear principal component analysis. This chapter discusses the results of incremental subspace update between fast principal component analysis and modified fast component analysis, and compares the processing time for incremental subspace update using different methods for different modes.

Chapter 5 describes the human face learning and recognition system, where the proposed method, IMFPCA combined with SKL, is used as the incremental subspace update. Moreover, an arbitrary threshold, as determined by the bimodal histogram concept, is utilized to judge wether the face in the test image has been encountered in the previous dataset or not. The results between the proposed method and the traditional subspace update method are discussed.

Chapter 6 focuses on an object tracking application to prove the practical merits of combining IMFPCA with SKL algorithm. The tracking model is described in detail, and the tracking algorithm is tested on various image sequences with different characteristics.

Finally, Chapter 7 provides a retrospective assessment on the merits of the proposed method as well as the human face learning and recognition system. Moreover, future research directions are provided are provided as potnetial means to augment the real-world applications of the overall concept of incremental learning.

CHAPTER II

Comparative Assessment on Conventional Methodologies

2.1 Introduction

This chapter provides an overall analysis on conventional methods that have been used for face recognition. It introduces the data subjects, and software tools used in this dissertation. A thorough analysis of conventional methods helps in determining which method is best suited for the incremental learning process in seeking a low computational burden and a high recognition rate. In the experimental phase, different data sets are used to validate the reliability in the results obtained.

2.2 Related works

The Principal Component Analysis (PCA) is a well-known technique for approximating a matrix through a lower dimensional subspace. This lower dimensional subspace is constructed by eigenvectors that correspond to the most significant eigenvalues. The Eigenface system as used for face recognition was initially developed by [Turk and Pentland, 1991]. Later, other PCA-based face recognition methods were introduced with the use of the independent component analysis (ICA) [Bartlett et al., 2002; Draper et al., 2003; Yuen and Lai, 2002] and the kernel principal component analysis (KPCA) [Kim et al., 2002; Schölkopf et al., 1998] applied in kernel Hilbert space. For these aforementioned methods, there is a need to reshape a series of q $I_1 \times I_2$ input images into a matrix with a higher dimensional matrix of size $I_1 I_2 \times q$; this type of matrices may overburden the computational requirements. To decrease the computational cost due to the high dimensionality, [Yang et al., 2004] proposed the 2D PCA approach which reduced the computational complexity significantly. Moreover, [Yu and Bennamoun, 2006] extended the 2D PCA to the nD PCA for high dimensional applications. Consequently, the method proposed in this study involves the use of three-dimensional tensors ($I_1 \times I_2 \times q$) which are applied to two dimensional matrices whose structure is guided by the type of unfolding used yielding the following different 2D matrices: $I_1 \times I_2 q$, $I_2 \times I_1 q$ and $q \times I_1 I_2$, integrating at the same time the concept of multi-linear singular value (SVD) decomposition [Luthauwer et al., 2000].

2.3 Data and subjects

The experiments were conducted using [AT&T, the Database of Faces] (ORL Database of Faces formerly) given its widespread usage in the literature. This database of faces, which is composed of face images taken in a laboratory setting between April 1992 and April 1994, was first used in a face recognition project with the Cambridge University Engineering Department.

There are 40 subjects with 10 images per subject. For some subjects, the images were taken at different times, with different lighting conditions, different facial expressions (open/closed eyes, smiling/not smiling, etc.) and with different facial details and expressions. These images are grayscale images with 112x92 in resolution, and they were taken against a dark homogeneous background with the subjects in an upright, frontal position (with tolerance for some side movement). Figure 2.1 shows some sample images from the database as illustrative examples.

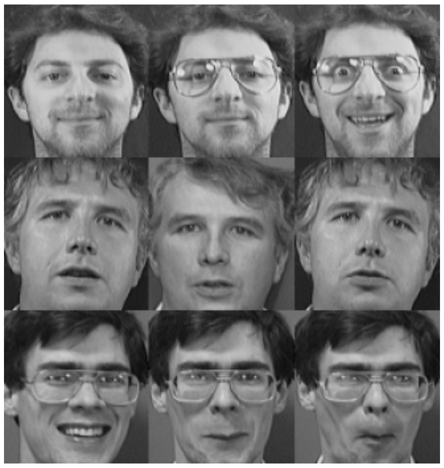


Figure 2.1 Sample images from the AT&T database of faces

2.4 Processing environment

All the programs were executed in Matlab on Windows Vista based PC, with the configuration of Intel Core 2 CPU T5200 1.60GHz and 2G RAM.

MATLAB (http://www.mathworks.com/) has already been used to develop different tools for face recognition and some of them can be utilized directly. For example, the Matlab toolbox for pattern recognition [Duin et al., 2004] was developed by the pattern recognition group in Delft University of Technology. Another application toolbox called the INface toolbox for illumination invariant face recognition [Struc, 2010] was provided with the purpose of maintaining consistency in the way facial characteristics are perceived or recorded.

2.5 Comparisons of different methods

To value the established groundwork that has guided the progress of incremental learning leading to face recognition, four most useful techniques together with the newly developed multilinear PCA are compared in terms of recognition accuracy and computational complexity.

2.5.1 Conventional methods

There are four conventional methods that have been quite used in this field of research with varying degrees of success. These include the PCA, ICA, kernel PCA and 2D PCA. Their detailed steps (once obtaining the projections of the training images) are as follows.

2.5.1.1 Principal Component Analysis

The principal component analysis (PCA) is a statistic technique that is widely used in the fields of pattern recognition, image compression, and decision making processes. It can express the data with the purpose of emphasizing either their similarities or their differences.

The following constitute the main steps needed to perform the PCA.

- Collect the data as q images of size $I_1 \times I_2$.
- Establish the data matrix and resize it by setting each image as a $I_1 I_2$ vector, then with the q images set column-wise, generate a new matrix of size $I_1 I_2 \times q$.
- Determine the centered matrix by computing the mean vector and subtracting it from each column vector. This process produces a data set with zero mean.
- Obtain the scatter matrix of the centered matrix producing a matrix of size $I_1 I_2 \times I_1 I_2$.
- Calculate the eigenvectors and eigenvalues of the scatter matrix.
- Select appropriate components to form a feature basis on the basis of the first k eigenvectors that are chosen.
- Obtain the projection of the training samples.

2.5.1.2 Independent Component Analysis

ICA and PCA are closely related in that the PCA can be considered as a special case of ICA. The purpose of ICA is to minimize the statistical dependence between the basis vectors. In the procedure for implementing ICA, additional steps are included beyond the required steps of the PCA which are initially used to reduce dimensionality prior to performing the ICA.

The main steps of the ICA are as follows:

- Follow the steps of the PCA to obtain the feature basis.
- Compute the projections of the images into the feature basis.
- Find the Whitening matrix to minimize the statistical dependence.
- Obtain the ICA representation of the image.

It should be noted that the vectors in feature basis of the ICA are neither orthogonal nor ranked in order.

2.5.1.3 Kernel Principal Component Analysis

Kernel PCA is the PCA applied to the data that is nonlinearly mapped into a higher dimensional feature space. The major steps in this case are:

- Find the covariance matrix in kernel space.
- Decompose the matrix.
- Obtain the projection of the training samples in the kernel space.

2.5.1.4 2D Principal Component Analysis

The 2D PCA changes the PCA by keeping the shape of original images and performing the decomposition directly on the mean covariance matrix of the images in order to find the feature basis. The steps of the 2D PCA are as follows:

- Collect a set of data as q images of size $I_1 \times I_2$ each.
- Get the centered images and compute the mean matrix of the dataset, and subtract the mean matrix from each image in the dataset.
- Calculate the mean scatter matrix.
- Perform the eigen decomposition on the mean scatter matrix.
- Select the components and generate the feature basis.
- Obtain the projections of the training samples.

The above are just the general description of the steps to find the projections of the training samples. The detailed steps with mathematical explanations will be discussed in due course as their computational complexity is considered.

2.5.2 Multilinear Principal Component Analysis

In this section, some basic multilinear algebra will be introduced first. Then, the steps of Multilinear PCA are discussed.

2.5.2.1 Basic Multilinear Algebra

A high-order tensor is denoted as $\mathcal{A} \in \Re^{I_1 \times I_2 \ldots \times I_N}$, where I_n *with* $n = 1, \ldots, N$ represents the size of the n^{th} dimension of the tensor. The mode-*n* product of a tensor $\mathcal{A}$ by a matrix $U \in \Re^{J_n \times I_n}$, denoted by $\mathcal{A} \times_n U$ is determined by the tensor entries $(\mathcal{A} \times_n U)_{i_1,\ldots,i_{n-1},j_n,i_{n+1},\ldots,i_N} = \sum_{i_n} \mathcal{A}_{i_1 \ldots i_N} u_{j_n i_n}$, where i_n denotes the mode-n of $\mathcal{A}$. The scalar product of two tensors $\mathcal{A}, \mathcal{B} \in \Re^{I_1 \times I_2 \times \ldots \times I_N}$ is defined as $\langle \mathcal{A}, \mathcal{B} \rangle = \sum_{i_1} \sum_{i_2} \cdots \sum_{i_N} \mathcal{A}_{i_1 i_2 \ldots i_N} \cdot \mathcal{B}_{i_1 i_2 \ldots i_N}$. And the Frobenius norm of a tensor

$\mathcal{A} \in \Re^{I_1 \times I_2 \ldots \times I_N}$ is defined as $\|\mathcal{A}\| = \sqrt{\langle \mathcal{A}, \mathcal{A} \rangle}$. Any tensor therefore can be expressed as the product $\mathcal{A} = \mathcal{G} \times_1 U^{A_{(1)}} \times_2 U^{A_{(2)}} \ldots \times_N U^{A_{(N)}}$ where $\mathcal{G} \in \Re^{J_1 \times J_2 \ldots \times J_N}$ is the core tensor defined as $\mathcal{G} = A \times_1 U^{A_{(1)}} \times_2 U^{A_{(2)}} \ldots \times_N U^{A_{(N)}}$, with $U^{A_{(i)}} = (u_1^{A_{(i)}}, u_2^{A_{(i)}}, \ldots, u_{k_{A_{(i)}}}^{A_{(i)}})$ being a unitary matrix. Mode-*n* unfolding yields a matrix $A_{(n)} \in \Re^{I_n \times I_1 I_2 \ldots I_{n-1} I_{n+1} \ldots I_N}$. The different unfolding mechanisms for a 3rd-order tensor are illustrated in Figure 2.2.

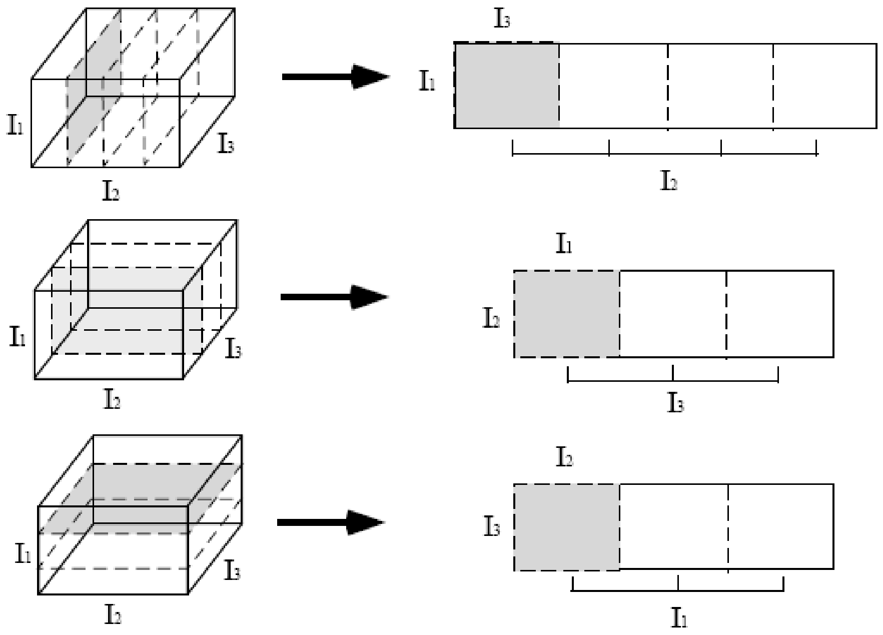


Figure 2.2 Unfolding mechanisms for a 3rd-order tensor

2.5.2.2 Multilinear Principal Component Analysis

In contrast to using the mean covariance matrix as in 2D PCA, images are retained as 2D matrices instead of 1D vector in the PCA as shown in Figure 2.3 (a), and the matrices are utilized to generate a 3D tensor as given in Figure 2.3 (b). Then the multilinear algorithm is conducted to find the feature basis for the tensor. Moreover, the dimension involved for feature basis is *N-1* without considering the dimension of time. Specifically for our case, the face image is a 2D matrix; hence

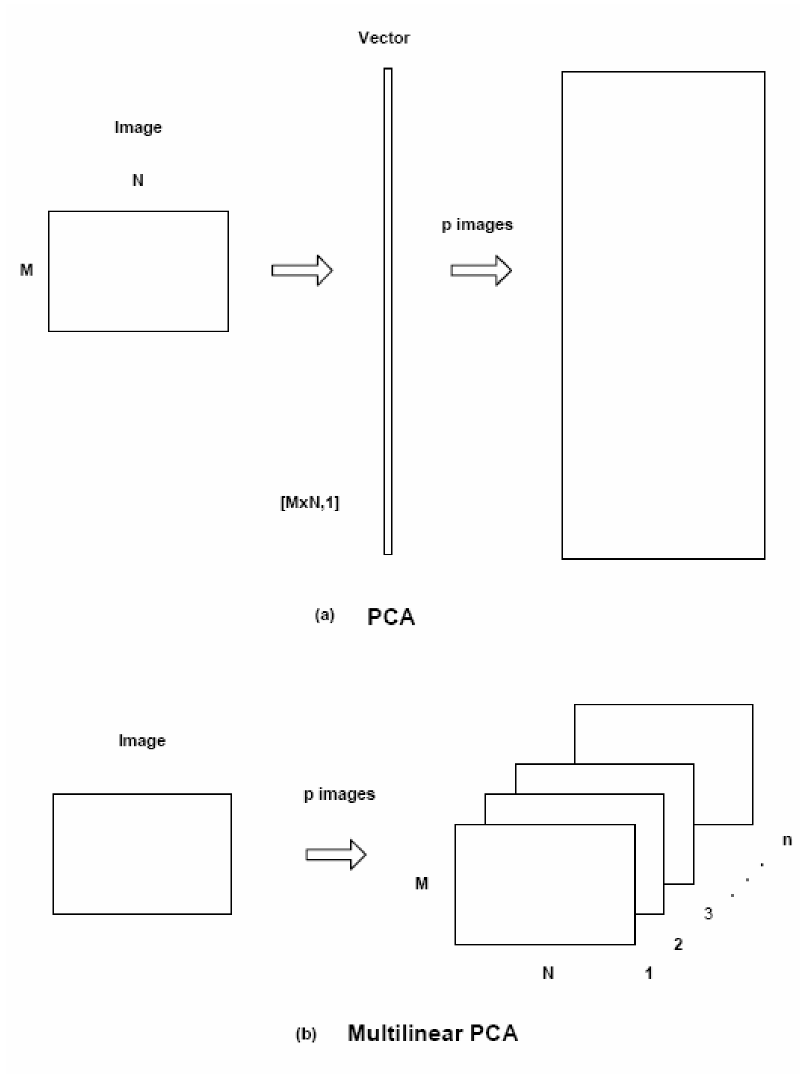


Figure 2.3 Data generation for PCA and multilinear PCA

features of two dimensions are obtained for the recognition process.

The steps considered for multilinear PCA are as follows:

- Collect a set of data as q images with size $I_1 \times I_2$ each.
- Generate the data tensor of size $I_1 \times I_2 \times q$.
- Compute the centered tensor along the time dimension q.
- Unfold the centered tensor in mode-1 and mode-2, which gives two matrices with sizes $I_1 \times I_2 q$ and $I_2 \times I_1 q$, respectively.

- Find the feature basis for these two modes.
- Compute the projections of the training image into these two modes.

2.5.2.3 Classification method

The so-called L-2 norm distance is used for recognition purposes. For all methods, the projection of the testing image is the same as the training image. However, due to the two dimensional and multilinear characteristics, an extra step for 2D PCA and MPCA is required for storage and computational convenience. The projections are also rearranged into one vector.

Suppose the projection of the testing image is y_t and the projection of the p^{th} training image is y_p. Then the norm between projections of the two images is determined as $d(p) = \left\| y_t - y_p \right\|$. The index of minimum distance expressed by *min(d)* gives the recognition result in the training data for the given testing image.

2.5.3 Computational complexity

The computational complexities of the different methods and their different processing steps are assessed. These do not include the time for image loading as such a task is performed for all methods. Suppose the matrix A with size $I_1 I_2 \times q$, B with image size $I_1 \times I_2$ and a tensor $\mathcal{A}$ with size $I_1 \times I_2 \times q$, the following are the processing times required for the different methods.

2.5.3.1 Principal Component Analysis

- Centered matrix $\hat{A} = A - \overline{A}$ requires $I_1 I_2$
- Finding the matrix of $\hat{A}^T \hat{A}$ requires $2q^2 I_1 I_2$

- Eigen decomposition of $\hat{A}^T\hat{A} = U\Sigma U^T$ requires $O(q^3)$
- Computing eigenvectors $\hat{A}U$ requires $2q^2 I_1 I_2$
- Projection of the training samples with the k largest eigenvectors in $\hat{A}U$ by $\hat{A}(:,i)^T(\hat{A}U(:,1:k)), \quad i = 1,2,...,q$ requires $2qkI_1I_2$

  Total : $I_1I_2 + 4q^2I_1I_2 + 2qkI_1I_2 + O(q^3)$

2.5.3.2 Independent Component Analysis

- Following the same steps as with the Principal Component analysis to get the eigenvectors $R = \hat{A}U$ would require $I_1I_2 + 4q^2I_1I_2 + O(q^3)$.
- To get PCA representation of training images $X = (R(:,k))^T\hat{A}$ which keep k eigenvectors corresponding to k largest requires $2qkI_1I_2$.
- Centered $\hat{X}$ requires I_1I_2k and get the $\hat{X}\hat{X}^T$ needs $2kI_1^2I_2^2$.
- Finding the whitening matrix $W_Z = 2(\hat{X}\hat{X}^T)^{-\frac{1}{2}}$ requires $O(k^3)$.
- Updating $\hat{X} = W_Z\hat{X}$ requires $2k^2I_1I_2$.
- Getting the generative model of data $\hat{X} = W_Z^{-1}\hat{X}$ requires $2k^2I_1I_2 + O(k^3)$.
- Training the output W related to the iteration $Iter$, here let's set the operation as F_0.
- Get the ICA represents of image $F = R(WW_Z)^{-1}$ requires $2qk + 3k^3 + O(k^3)$.

  Total: $$\begin{aligned}&I_1I_2 + 4q^2I_1I_2 + 2qkI_1I_2 + I_1I_2k + 2kI_1^2I_2^2 + \\ &4k^2I_1I_2 + 2qk^2 + 2k^3 + F_0 + O(k^3) + O(q^3)\end{aligned}$$

2.5.3.3 Kernel Principal Component Analysis

- Finding μ_{A_F} requires q^2
- Computing $A_F^TA_F$ requires q^2K_0, where K_0 represents the kernel evaluation,

related to $I_1 I_2$ and kernel type.

- Computing $\mu_{A_F}^T A_F^T A_F \mu_{A_F}$ requires $4q^3$.
- Eigen decomposition of $\mu_{A_F}^T A_F^T A_F \mu_{A_F} = Q \Delta Q^T$ requires $O(q^3)$.
- Expansion coefficients $\alpha = Q(:,1:k)\Delta(1:k,1:k)^{-\frac{1}{2}}$ (Note: Δ is a diagonal matrix, we thus ignore the operation of inverse and square root) require $2qk^2$.
- Projection of the training image $A_F(i)^T A_F \alpha - \mu_{A_F}^T A_F^T A_F \alpha, \quad i = 1,2,...q$ ($A_F^T A_F$ already obtained) requires $q(K_0 + 2qk + 2q^3 + 2q^2 k)$.

Total: $q^2 + q^2 K_0 + 4q^3 + 2qk^2 + qK_0 + 2q^2 k + 2q^4 + 2q^3 k + O(q^3)$

2.5.3.4 2D Principal Component Analysis

- Centered matrix $\hat{B}_i = B_i - \overline{B}_{1\_q}, \quad i = 1,2,...,q$ requires $qI_1 I_2$.
- Finding the covariance matrix for each centered image $\hat{B}_i \hat{B}_i^T$ requires $2qI_1 I_2$.
- Obtaining mean covariance matrix $\overline{\hat{B}_i \hat{B}_i^T}$ requires qI_1^2.
- Eigen decomposition of $\overline{\hat{B}_i \hat{B}_i^T} = U \Sigma U^T$ needs $O(I_1^3)$.
- Projection of the training samples $\hat{B}_i^T U(:,1:k), \quad i = 1,2,...,q$ requires $2qkI_1$.

Total: $qI_1 I_2 + 2qI_1 I_2^2 + qI_1^2 + 2qkI_1 + O(I_1^3)$

2.5.3.5 Multilinear Principal Component Analysis

- Centered tensor $\hat{\mathcal{A}} = \mathcal{A} - \bar{\mathcal{A}}$ requires $qI_1 I_2$.
- Unfolding to $A_{(1)}$ with size $I_1 \times I_2 q$ and finding covariance matrix $A_{(1)} A_{(1)}{}^T$ requires $2kI_1^2 I_2$.
- Unfolding to $A_{(2)}$ with size $I_2 \times I_1 q$ and finding covariance matrix $A_{(2)} A_{(2)}{}^T$ requires $2kI_1 I_2{}^2$.

- Finding the Eigen decomposition of $A_{(1)}A_{(1)}{}^T$ and $A_{(1)}A_{(1)}{}^T$ requires $O(I_1^3)$ and $O(I_2^3)$ respectively.
- Projection of the training samples to two modes needs $2qk_{A_{(1)}}I_1 + 2qk_{A_{(2)}}I_2$, where $k_{A_{(1)}}$ and $k_{A_{(2)}}$ are the numbers of the eigenvectors corresponding to largest eigenvalues.

Total: $qI_1I_2 + 2qI_1{}^2I_2 + 2qI_1I_2^2 + 2qk_{A_{(1)}}I_1 + 2qk_{A_{(2)}}I_2 + O(I_1^3) + O(I_2^3)$

2.5.4 Experiment results

Since PCA based methods are essentially dimensionality reduction methods, the reconstruction of the original images from fewer dimensions should be assessed for a performance evaluation using contemporary methods which include in this case PCA, 2D PCA, and multilinear PCA. The KPCA method is not included in the reconstruction process since it was found to be computationally taxing.

2.5.4.1 Reconstruction comparison

PCA: Given an image B with size $I_1 \times I_2$, the image is reshaped into a vector B_v with size $I_1I_2 \times 1$ after centering. The reconstruction vector is defined as $recon(\hat{B}_v) = UU^T\hat{B}_v$. Then reshape $recon(\hat{B}_v) + \overline{B}_v$ back to size to constitute the reconstruction image needed, where $\overline{B}_v$ is the mean of all image vectors.

2D PCA: With the same definitions given in the previous section on the PCA, the reconstructed image in this case can be obtained using the formula $recon(B) = UU^T\hat{B} + \overline{B}$.

Multilinear PCA: The reconstruction in this case is performed using:

$$recon(B) = [(\hat{B} \times_1 U_{(1)}{}^T) \times_1 U_{(1)} + (\hat{B} \times_2 U_{(2)}{}^T) \times_2 U_{(2)}] / 2 + \overline{B} .$$

To facilitate the comparison of the different reconstruction results using the aforementioned methods, Figure 2.4 provides the reconstruction images of a given subject with different number of eigenvectors retained. In this experiment, 360 images were considered to compute the feature basis.

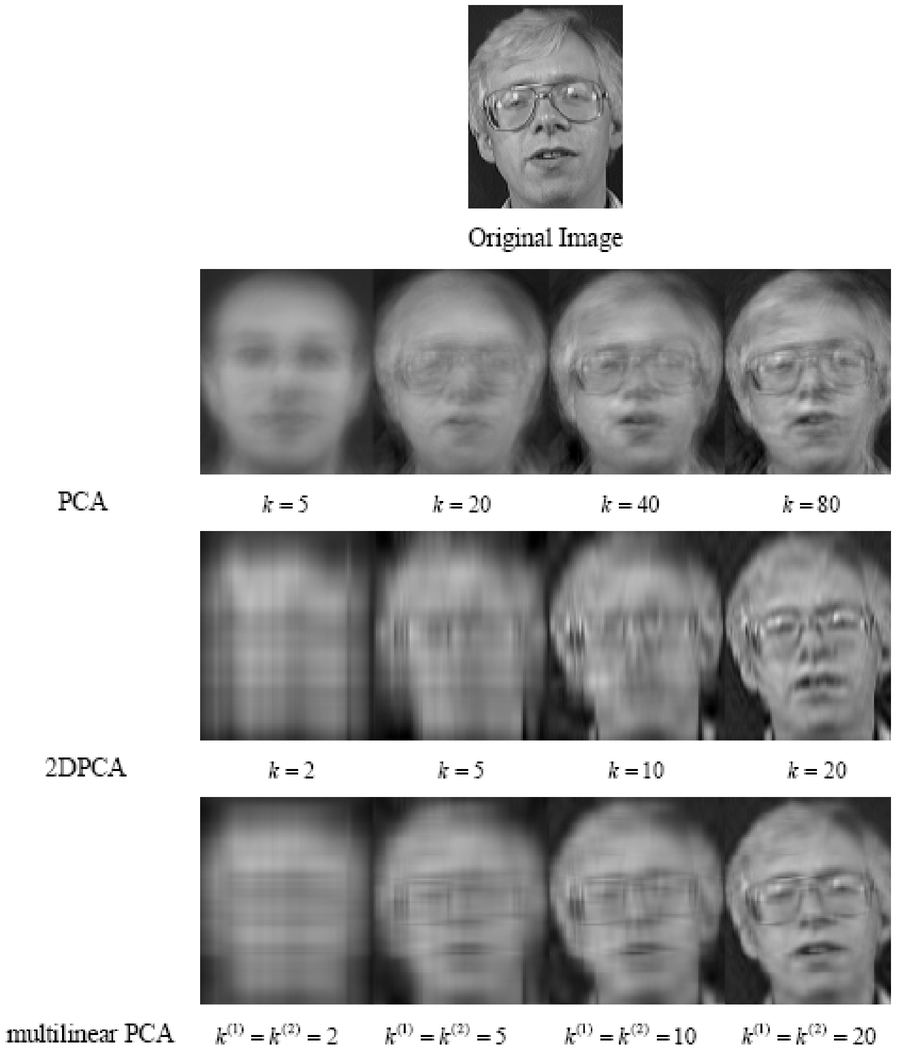


Figure 2.4 Original image and reconstructed images based on different methods and parameters.

From these results, it is apparent that the 2D PCA and the multilinear PCA are superior to the PCA. The PCA would thus need to retain more eigenvectors for the reconstruction of images in order to minimize the error. The 2D PCA and the multilinear PCA performed equally well, and needed a much smaller subset of the eigenvectors in contrast to the PCA for similar reconstruction results. However, it is emphasized that the reconstruction error of the multilinear PCA (0.0080) was less than that of the 2D PCA (0.0101).

2.5.4.2 Face recognition

The top recognition rates for different face recognition algorithms including ICA, PCA, KPCA (Gaussian and Polynomial kernel), 2D PCA and multilinear PCA for two sets of experimental data are given in Table 2.1. For the five-to-five dataset, five images were chosen randomly of one subject for feature bases and the remaining five images were used for testing. Recall that ten images were considered for each subject. For the other leave-one-out dataset, the first nine images were kept out of ten for feature bases extraction and the one left is used as the testing image.

For the five-to-five dataset, both multilinear PCA and KPCA with Gaussian kernel achieve the highest recognition rate of 93.5%. In the other test, multilinear PCA is superior to all the other algorithms and its recognition rate reached 97.5%. In other words, only one image in the testing set was recognized incorrectly.

Table 2.1 Comparison of the top recognition rates (%) for different methods

Method	Five-to-Five	Leave-one-out
ICA	85.5%	92.5%
PCA	91.0%	95%
KPCA (Gaussian, sigma=256x16)	93.5%	92.5%
KPCA (Polynomial, d=3)	90.5%	95%
2D PCA	92.5%	95%
Multilinear PCA	93.5%	97.5%

To complement this evaluation process, and as a complete retrospective, Table 2.2 provides the different memory requirements for each method and indicates the operational complexity for each in terms of both the generalized formulas derived earlier and the associated run time.

In Table 2.2, the average running time is obtained by averaging 100 trials of the training and projecting procedures (excluding data matrix generation and centering steps). For this test, the parameters are set to be the same as the aforementioned leave-one-out experiment.

Table 2.2 Memory requirement and operation complexity among methods

Method	Operation (Feature extraction & training samples projection)	Memory unit	Average Running time in Matlab
ICA	$I_1I_2 + 4q^2I_1I_2 + 2qkI_1I_2 + I_1I_2k + 2kI_1^2I_2^2 + 4k^2I_1I_2 + 2qk^2 + 2k^3 + F_0 + O(k^3) + O(q^3)$	qk $k <= q$	209.5469s
PCA	$I_1I_2 + 4q^2I_1I_2 + 2qkI_1I_2 + O(q^3)$	I_1I_2k $k <= q$	20.2825s
KPCA (Gaussian)	$q^2 + q^2K_0 + 4q^3 + 2qk^2 + qK_0 + 2q^2k + 2q^4 + 2q^3k + O(q^3)$	$qk + q$ $k <= q$	86.9237s
KPCA (Poly)	$q^2 + q^2K_0 + 4q^3 + 2qk^2 + qK_0 + 2q^2k + 2q^4 + 2q^3k + O(q^3)$	$qk + q$ $k <= q$	39.6125s
2D PCA	$qI_1I_2 + 2qI_1I_2^2 + qI_1^2 + 2qkI_1 + O(I_1^3)$	I_1k $k <= q$	6.0352s
Multilinear PCA	$qI_1I_2 + 2qI_1^2I_2 + 2qI_1I_2^2 + 2qk_{A_{(1)}}I_1 + 2qk_{A_{(2)}}I_2 + O(I_1^3) + O(I_2^3)$	$I_1k_{A_{(1)}} + I_2k_{A_{(2)}}$ $k_{A_{(1)}} <= I_1$ $k_{A_{(2)}} <= I_2$	9.5341s

Figure 2.5 shows the average run time for these methods. It can be initially observed that PCA, 2D PCA and multilinear PCA were comparatively faster than the other three algorithms. Moreover, 2D PCA and multilinear PCA were the two most computationally efficient methods more so than the PCA. Moreover, we can differentiate the two algorithms from the last column in Table 2.2, in that it shows that

the 2D PCA required only needs 6.0352s while the multilinear PCA required 9.5341s. As for the memory requirements to store the feature bases, it is difficult to distinguish from the third column of Table 2.2 which method would need less memory. But with the increase of q, the 2D PCA and the multilinear PCA would use less memory since they are independent of the parameter q. Moreover, in the experiments, the number of eigenvectors of the multilinear PCA was usually less than the number of eigenvectors for the 2D PCA when seeking high recognition rates.

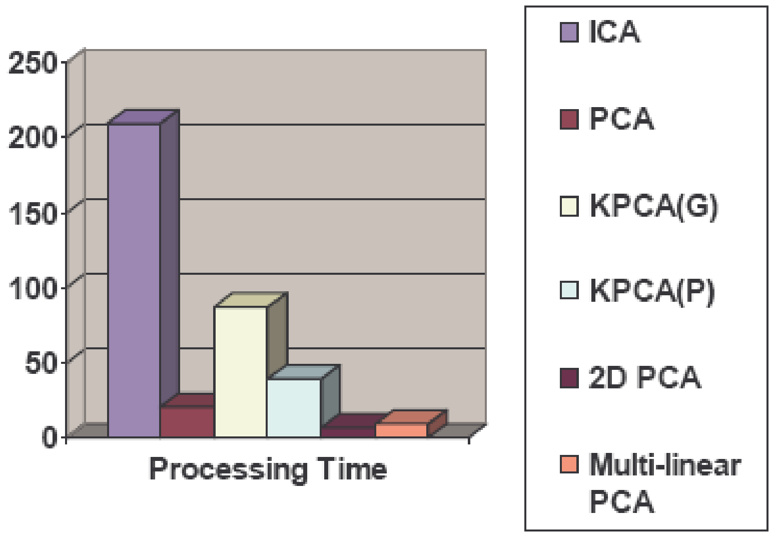


Figure 2.5 Processing time among contemporary methods

The second column of Table 2.2 also shows that for the ICA, the iterative computational process was time consuming. Concerning the KPCA, the most processing time was needed in computing the kernel matrix. The Gaussian kernel on the other hand requires more processing time than the polynomial kernel. Moreover, in this application, the multilinear PCA had to compute one more mode than the 2D PCA and both the theoretical derivations and experiment results demonstrated that.

2.6 Retrospective

In this chapter, several contemporary methods are contrasted in a thorough comparative evaluation in terms of both computational and memory requirements. The main accomplishment of this section is to show that the accuracy rate for face recognition has been optimized with little or no compromise on memory and computational requirements for multilinear PCA. Through real-world applications of face recognition using the AT&T database, this study has proven that the multilinear PCA is superior or equal to other methods in recognition accuracy. In terms of processing time, 2D PCA revealed to be the most computationally efficient method. But usually, smaller feature bases of the multilinear PCA can achieve the recognition accuracy of larger feature bases of the 2D PCA.

To achieve the goal of this dissertation, incremental procedure of multilinear PCA can be utilized in accordance to the preliminary results from this section. Although 2D PCA was the most computationally efficient method, it is however based on the mean covariance matrix, which is not suitable for the important process of incremental learning. Furthermore, the multilinear PCA, which does not depend on the mean covariance matrix, has the potential to achieve computational efficiency for incremental learning applications.

CHAPTER III

Modified Fast Principal Component Analysis

3.1 Introduction

Due to the randomly generated initial vector (which may converge to local minimum) in the fixed-point algorithm, the existing fast principal component analysis (fast PCA) has unstable performance in the order it generates eigenvectors. In this chapter, by modifying the fast PCA algorithm, the deficiency of fixed point algorithm is minimized. To evaluate the merit of the proposed modified algorithm, similarities between standard eigenvectors from eigenvalue decomposition (EVD), eigenvectors from fast PCA, and eigenvectors generated using the proposed modified algorithm are compared. The comparison indicates that the eigenvectors from the modified fast PCA has better similarity to the standard eigenvectors. In addition, the fast PCA and modified fast PCA are compared into the face recognition application to evaluate their performance.

3.2 Related work

Principal component analysis is used for approximating a set of vectors by a low dimension subspace that can still keep most of the information contained in all the vectors. Consequently, a minimum mean square error is achieved between the original vectors and the reconstructed ones. This concept is also the core of the fast PCA [Sharma and Paliwal, 2007].

As one of the most important standards for evaluating an algorithm, the computational complexity of the PCA has been studied through decades. In this field, researchers have made great improvements, such as the Cyclic Jacobi's method [Golub and Loan, 1996], its modification [Reddy and Herron, 2001] and power method [Schilling and Harris, 2000]. Moreover, the Fast PCA is a computationally fast technique for finding the leading eigenvectors. It is obvious that the fast PCA is computationally efficient (i.e. with data dimensionality 2000, Fast PCA needs 2.28s and EVD based PCA needs 153.26s. Another point worth addressing is the close similarity between eigenvectors from fast PCA and EVD based PCA.

3.3 Data and subjects

In this chapter, the AT&T database introduced earlier in chapter 2 is used throughout.

3.4 Modified fast principal component analysis

In this section, the fast PCA algorithm is first introduced and the necessity to enhance this given algorithm is analyzed. The modification that was introduced and the procedure of the algorithm are explained. Also, two experiments are implemented to verify that the modified method can indeed achieve higher similarity.

3.4.1 Fast PCA

The fast PCA is obtained by minimizing the mean square error between the original vectors and their reconstructed versions from a dimensionally-reduced principal component transform, while no or minimal concessions are made on accuracy.

Suppose that $x \in \Re^{m \times 1}$ represents a vector with a mean $\mu = E[x]$, the reduced dimensional feature vector is then denoted as $y \in \Re^{h \times 1}$. With the reconstructed vector of x being $\hat{x} \in \Re^{m\times 1}$, the mean square error can be presented as:

$$MSE = E[\| x - \hat{x} \|^2] \tag{3.1}$$

where $\|\bullet\|$ denotes the norm value and the function $E[\bullet]$ is defined as the expectation operation.

In the PCA transform, the reduced eigenbasis is supposed to be $U \in \Re^{m\times h}$, and then the reduced dimensional vector can be computed as $y = U^T(x-\mu)$, with zero empirical mean. With the reduced dimensional vector y, the reconstructed $\hat{x}$ of x can be computed as

$$\hat{x} = Uy + \mu = UU^T(x-\mu) + \mu \tag{3.2}$$

Therefore, equation 3.1 can be rewritten as follows:

$$MSE = E[\| (I - UU^T)(x-\mu) \|^2] \tag{3.3}$$

The scalar function $\| (I - UU^T)(x-\mu) \|^2$, which determines the norm of a vector, can thus be simplified as follows:

$$\begin{aligned}\left\|(I-UU^T)(x-\mu)\right\|^2 &= ((I-UU^T)(x-\mu))^T(I-UU^T)(x-\mu)\\ &= (x-\mu)^T(I-UU^T)^T(I-UU^T)(x-\mu)\\ &= (x-\mu)^T(I-UU^T)(I-UU^T)(x-\mu)\\ &= (x-\mu)^T(I-2UU^T+UU^TUU^T)(x-\mu)\end{aligned} \tag{3.4}$$

Since the eigenvectors are orthonormal, the following relation applies:

$$UU^TUU^T = UIU^T = UU^T \tag{3.5}$$

This further simplifies equation 3.4 to the following squared norm:

$$\left\|(I-UU^T)(x-\mu)\right\|^2 = (x-\mu)^T(I-UU^T)(x-\mu) \tag{3.6}$$

Then the derivative of MSE can be determined as in [Golub and Reinsch, 1970] to yield the following:

$$\frac{\partial}{\partial U} E[(x-\mu)^T (I-UU^T)(x-\mu)] = -2E[(x-\mu)(x-\mu)^T U] \tag{3.7}$$

The fixed-point algorithm [Hyvärinen and Oja, 1997] and Gram-Schmidt orthonormalization process [Golub and Loan, 1996] can then be used to estimate the leading eigenvectors.

3.4.2 Modified fast PCA

The eigenbasis $U \in \Re^{m \times h}$ of a matrix $D \in \Re^{m \times n}$ should evidently satisfy the standard relation $\lambda U = DU$, where λ represents the corresponding eigenvalues. The eigenbasis $U = [e_1, e_2, ..., e_h]$ should be composed of eigenvectors with the largest eigenvalues, and should be ordered as e_1, e_2, …, e_h column-wise in accordance to their respective eigenvalues such that $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_h$. Under usual circumstances, eigenvectors are computed in a descending order of their corresponding eigenvalues by the fast PCA. Unfortunately, the algorithm tends to be numerically unstable if the initial vector is generated randomly [Bakir et al., 2007; Berinde, 2007]. Due to this instability, the eigenvectors are not set in descending order. The intuitive approach is to repeat the iteration with various initial vectors. In this case, the additional step is to check if the new eigenvalue is smaller than the previous one. If so, proceed computing the next eigenvector; otherwise go back to the previous step, and recalculate that eigenvalue and respective eigenvector using a different initial vector. The pseudo-code implementation of the modified fast PCA (MFPCA) is provided as follows.

Algorithm procedure:

```
a. Set $ind = 1$.
b. Initialize eigenvector $U_{ind}$ randomly with size $m \times 1$.
```

c. Update $U'_{ind} = CovU_{ind}$.

d. Implement the Gram-Schmidt process $U'_{ind} = U'_{ind} - \sum_{i=1}^{ind-1} proj_{U_i} U'_{ind}$ where $proj_u v = \frac{\langle u,v \rangle}{\langle u,u \rangle} u$ and $\langle u,v \rangle$ denotes the inner product of vectors u and v.

e. Find the norm of U'_{ind} then normalize U'_{ind} by $U'_{ind} / \|U'_{ind}\|$.

f. Compare the $\|U'_{ind}\|$ with the $\|U'_{ind-1}\|$, if $\|U'_{ind}\| > \|U'_{ind-1}\|$ and ind > 1, set ind = ind-1 and go to step b, else go to step a.

g. If $abs(U'^{T}_{ind} U_{ind} - 1) < \varepsilon$, where ε is the tolerance error given, is not satisfied, go to step *c* with $U_{ind} = U'_{ind}$.

h. Set $U_{ind} = U'_{ind}$ and $ind = ind + 1$ and go to step b until $ind = k$.

The processing step *f* as shown in bold indicates the modification that was introduced in the MFPCA. The next section gives the experimental results for the proposed modified method.

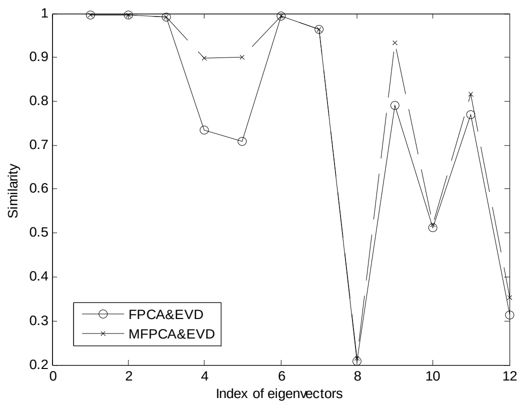
3.5 Experimental results

To test the modified method, images in the AT&T database are normalized into different dimensions 30×30, 40×40 and 50×50, respectively. Then, all images are reshaped into a vector, and matrices are generated. After centering the matrix and finding the covariance matrix, covariance matrices of dimensions 900×900, 1600×1600 and 2500×2500 are obtained. The comparative results, in terms of both accuracy in estimating the eigenvectors and computational requirements between the Eigenvalue Decomposition (EVD), FPCA and MFPCA, are given.

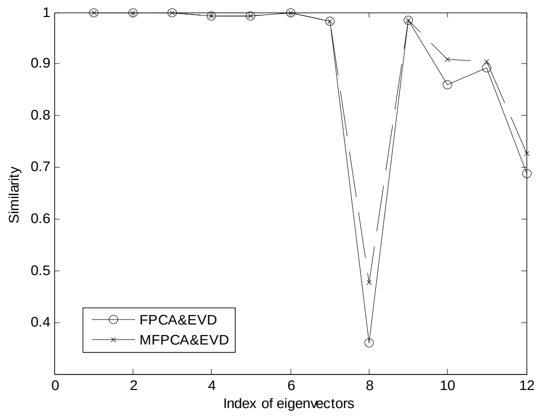
The first is to compare the similarity between eigenvectors from the aforementioned methods and those eigenvectors from the EVD-based PCA. Moreover, since the fixed-point method relies on random initial values which means the results will not be the same for different trials, the experiments are conducted in a statistical way, and the results are given as mean values of several trials.

The similarity between vectors is reflected through the dot product, and the closer to 1 is this dot product, the better is the similarity between the two vectors. In all the tests, the number of eigenvectors selected is randomly chosen to be the twelve top eigenvectors with the largest eigenvalues.

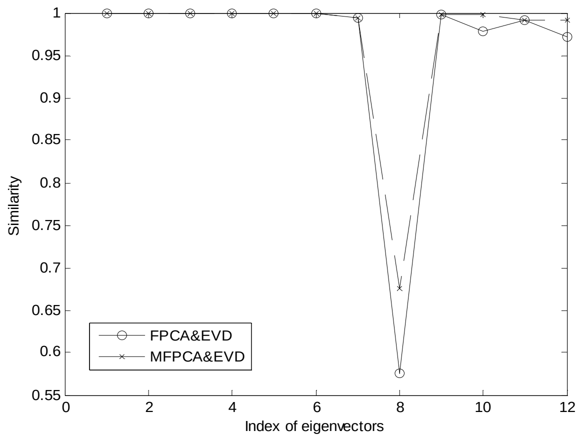
Figure 3.1 through 3.3 show that MFPCA provides better similarity than the fast PCA under different parameter settings. However, the defect of the fast PCA, which can be seen in the eight eigenvector index for all tests, cannot be overcome by the modified PCA. Although the eigenvalues are in descending order, the local minimum still shows up and gives a low similarity to EVD eigenvector. However, since the obtained eigenvector satisfied the iterative requirement, the eigenvectors after the eighth one still can have a good similarity with the EVD eigenvectors. Moreover, under the same error tolerance, with the increase of the dimension, the accuracy increases, too.



(a)

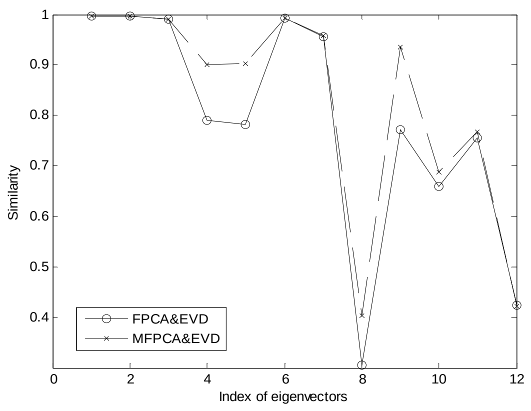


(b)

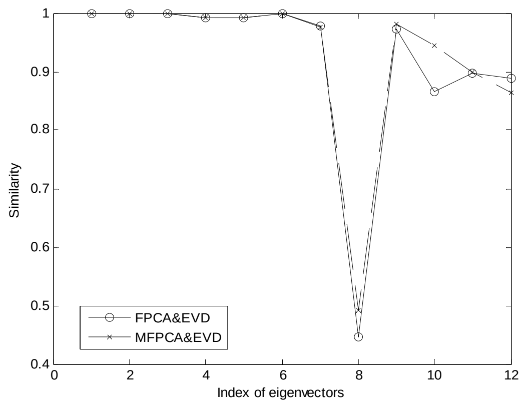


(c)

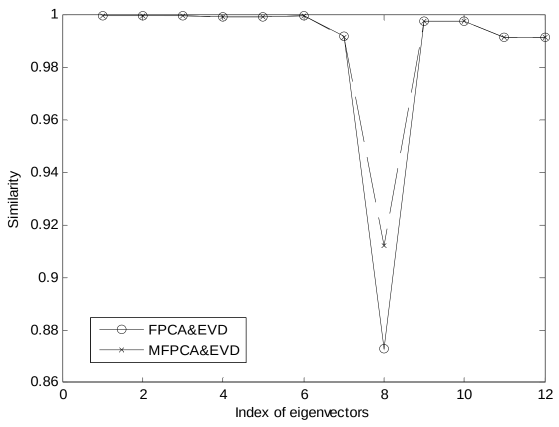
Figure 3.1 Similarity comparisons among eigenvectors from FPCA, MFPCA and EVD methods. FPCA&EVD shows the similarity between eigenvectors from fast PCA and from EVD method, and MFPCA&EVD shows the similarity between eigenvectors from MFPCA and from EVD method. (a) AT&T dataset, 30x30, ε =10e-3; (b) AT&T dataset, 30x30, ε =10e-4; and (c) 30x30, ε =10e-5.



(a)

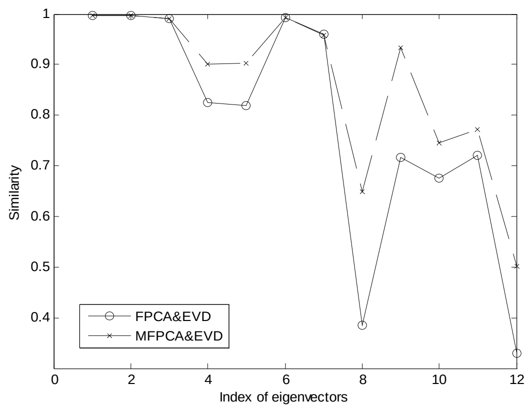


(b)

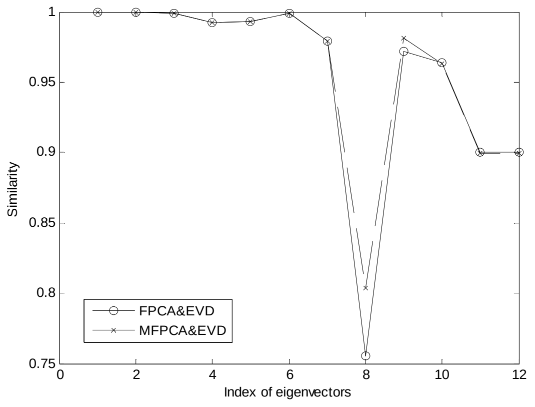


(c)

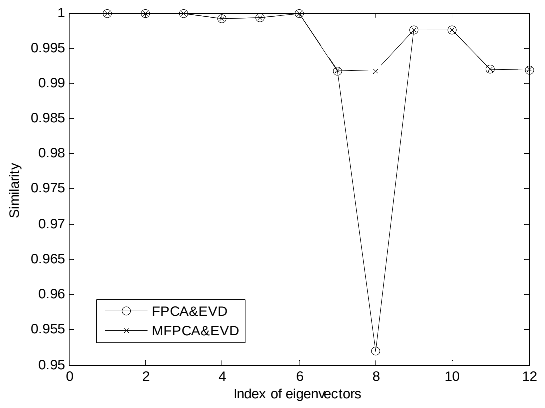
Figure 3.2 Similarity comparisons among eigenvectors from FPCA, MFPCA and EVD methods. FPCA&EVD shows the similarity between eigenvectors from fast PCA and from EVD method, and MFPCA&EVD shows the similarity between eigenvectors from MFPCA and from EVD method. (a) 40x40, ε =10e-3; (b) 40x40, ε =10e-4; (c) 40x40, ε =10e-5.



(a)



(b)



(c)

Figure 3.3 Similarity comparisons among eigenvectors from FPCA, MFPCA and EVD methods. FPCA&EVD shows the similarity between eigenvectors from FPCA and from EVD method, and MFPCA&EVD shows the similarity between eigenvectors from MFPCA and from EVD method. (a) 50x50, ε =10e-3; (b) 50x50, ε =10e-4; (c) 50x50, ε =10e-5.

Table 3.1 Processing time comparison among EVD, FPCA and MFPCA under different conditions (mean of 100 trials)

Image size	30×30	40×40	50×50
EVD	6.4860s	39.6555s	165.1443s
FPCA ($\varepsilon = 10^{-3}$)	0.6221s	1.8825s	4.9189s
MFPCA ($\varepsilon = 10^{-3}$)	0.7758s	2.4675s	6.5651s
FPCA ($\varepsilon = 10^{-4}$)	1.1759s	3.4587s	7.6038s
MFPCA ($\varepsilon = 10^{-4}$)	1.2575s	3.6353s	7.9852s
FPCA ($\varepsilon = 10^{-5}$)	1.8219s	5.2412s	13.0582s
MFPCA ($\varepsilon = 10^{-5}$)	1.7588 s	5.2497s	13.2501s

As can be seen in Table 3.1, FPCA and MFPCA have comparable processing times, both of which are significantly faster than EVD. The entry in Table 3.1, with the image size of 30x30 and error tolerance of $\varepsilon = 10^{-5}$, is the only case where the MFPCA is faster than FPCA. The reason that less iterations were required to achieve the error tolerance requirement is a purely coincidental case based on the 100 random trials considered.

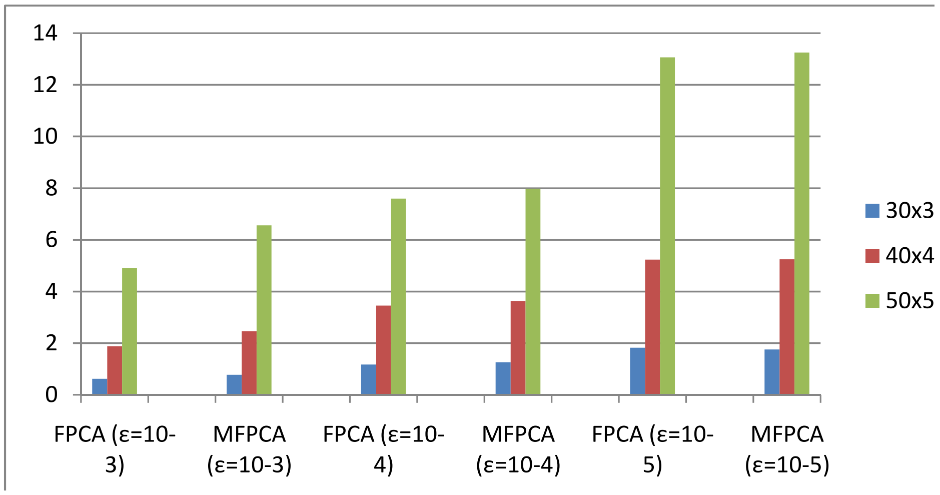


Figure 3.4 Processing time comparison between FPCA and MFPCA.

The second set of experiments involved applying these methods for face recognition to assess the different recognition accuracies. The number of eigenvectors kept is determined by the best recognition accuracy that the standard eigenvectors can achieve. For the first set of data, we randomly chose five images of one subject as feature bases and use the rest of images for testing. For the other set of data, we kept the first nine images out of ten for feature bases extraction and the one left was used as the testing image.

The results for this experiment are shown in Table 3.2. The results of both of these methods are based on mean values of 100 trials. The value between parenthesis (*) indicates the number of eigenvectors kept for face recognition and the two numbers shown in brackets [*, *] means the range of the recognition accuracy in terms of minimum and maximum values over 100 trials.

In Table 3.2, for the Leave-one-out dataset, nine is the number of minimum eigenvectors that gave the best accuracy using the standard PCA, and it is therefore chosen as the number of eigenvectors kept for both fast PCA and modified fast PCA.

Table 3.2 Comparison of recognition rates (%) for different methods

Method	Five-to-Five	Leave-one-out
Standard PCA	89%(53)	95%(9)
Fast PCA (error=10e-5)	88.7%(53) [88%, 89%]	94.7%(9) [90%, 95%]
Modified fast PCA (error=10e-5)	88.7%(53) [88%, 89%]	95%(9) [95%, 95%]
Fast PCA (error=10e-4)	88.56%(53) [88%, 89%]	93.90%(9) [90%, 95%]
Modified fast PCA (error=10e-4)	88.60%(53) [88%, 89%]	94.88%(9) [92.5%, 95%]
Fast PCA (error=10e-3)	88.61%(53) [87.5%, 90%]	93.33%(9) [85%, 95%]
Modified fast PCA (error=10e-3)	88.52%(53) [87.5%, 90%]	93.90%(9) [90%, 97.5%]

The results indicate that the modified method has overall a better recognition rate than fast PCA. But the results obtained from the Five-to-Five dataset, with fifty-three eigenvectors needed for the best accuracy using the standard PCA, reveal that no one single method outperformed the other, in view of the varied results that were obtained with different error tolerance rate. Note that when the error tolerance is set to be $\varepsilon = 10^{-5}$, the two methods have the same recognition accuracy. However, when setting the error tolerance to $\varepsilon = 10^{-4}$, the modified fast PCA now has a better performance than the fast PCA (88.56% for fast PCA and 88.60% for modified fast PCA). And if the error tolerance increases to $\varepsilon = 10^{-3}$, the fast PCA outperforms instead the modified fast PCA (88.61% for fast PCA and 88.52% for modified fast PCA). Obviously with such small variability in the results, we can hardly acquire a trend in terms of error tolerance and accuracy in the results. In fact, the results show that the fixed-point-based fast PCA is only useful for finding a few leading eigenvectors. As an iterative method, the eigenvectors of fixed-point-based fast PCA are calculated based on the previous ones, which means the more eigenvectors obtained, the bigger is the error accumulated.

CHAPTER IV

Incremental Multilinear Principal Component Analysis

4.1 Introduction

This chapter establishes the mathematical foundation for a fast incremental multilinear method which combines the traditional sequential Karhunen-Loeve (SKL) algorithm with the newly developed incremental modified fast PCA algorithm (IMFPCA). In accordance with the characteristics of the data structure, the proposed algorithm achieves both computational efficiency and high accuracy for incremental subspace updating. Moreover, the theoretical foundation is analyzed in detail as to the competing aspects of IMFPCA and SKL with respect to the different data unfolding schemes. Besides the general experiments designed to test the performance of the proposed algorithm, an incremental face recognition system was developed as a real-world application for the proposed algorithm.

4.2 Related works

The so-called appearance-based techniques, such as the Principal Component Analysis (PCA) and the Linear Discriminant Analysis (LDA), have been extensively used in the literature with a wide range of applications in fields such as computer vision, pattern classification, signal and image processing, among others. However, their computational complexity and their batch mode computational frameworks still impose practical constraints in applications that demand concurrently faster execution speed and higher accuracy in the results. A variation on the singular value decomposition

(R-SVD) [Golub and Reinsch, 1970] provides a faster approach for obtaining a specific subspace of a given data structure. Based on R-SVD, [Levy and Lindenbaum, 2000] developed the sequential Karhunen-Loeve (SKL) algorithm, which is characterized by a faster execution speed and higher suitability for dealing with image sequences. Many other applications were consequently reported utilizing the SKL algorithm. For instance, [Ross et al., 2007] proposed a visual tracking system based on an incremental subspace method with sample mean update. Also, [Zhao et al., 2006] developed a novel incremental PCA with specific application to face recognition. Moreover, [Chin and Suter, 2007] developed the incremental subspace method for kernel PCA, and applied it to offline and online face recognition as well as visual tracking. In [Hoegaertsa et al., 2007], the authors proposed a method which is similar to the research concept in [Zhao et al., 2006], but extended it into the kernel space and included both updating and down-dating procedure for tracking purposes. Another kind of fast principal component extraction method called Principal Component Orthogonal Projection Approximation and Subspace Tracking (PC-OPAST) was introduced by [Bartelmaos and Abed-Meraim, 2008] to be applied for incremental learning as well. The PC-OPAST method alleviates the computational burden for estimating the principal eigenvectors of the covariance matrix using Givens rotations for tri-diagonalization.

With the use of tensors in multilinear algebra being firmly established, great efforts have been devoted to their potential use for dimensionality reduction. In [Wang and Ahuja, 2008], the alternative least squares method was used to find a desired tensor with minimum cost. This method is applied on the multidimensional data directly. A new framework of multilinear PCA for dimensionality reduction and feature extraction was provided in [Lu et al., 2008] with an application to gait recognition. The iterative local optimization procedure was applied to find projection matrices. Moreover, there are some studies on the incremental learning of tensors. A visual tracking system proposed by [Li et al., 2007] was based on an incremental tensor subspace learning method, and the subspace update deployed the SKL algorithm. In [Sun et al, 2008], multilinear analysis and wavelets were combined for the analysis of time-evolving data. Alternating minimization was adopted for unfolding modes without including time dimension as a compression step, and then discrete wavelet transform was adopted on the results of the compression step. Moreover, in [Ozawa et al., 2008], the subspace update is based on the characteristic of chunk data input. Unlike most articles for incremental learning which keep a static number of eigenvectors, the approach in [Hall et al., 2002] was based on the reconstruction error, in which the number of eigenvectors used can change with each incremental update.

In all of these studies, the challenge remains in finding the appropriate balance between computational efficiency and high accuracy in estimating the eigenvectors. To come to terms with this challenge, this study proposes a modified fast PCA algorithm embedding an incremental multilinear method. Based on the characteristics of multilinear method, a new incremental subspace update method is described. Practical implementations of this new incremental procedure on different kinds of targets in image sequences are chosen to prove the validity of the incremental multilinear PCA

4.3 Data and subjects

For experimental evaluation purposes, two databases will be utilized in this chapter. One is the AT&T database, which has been used already in Chapter III, and the other is [the MNIST database of handwritten digits].

The MNIST database of handwritten digits has a total number of 70,000 examples. The digits have been size-normalized and centered in a fixed-size image of 28x28. Due to its large size, it allows for more iterations to be tested. Moreover, the image size of MNIST database is found to be more computationally suitable, since the dimension of the covariance matrix for mode-3 in the AT&T database (112x92) were considered unjustifiably large for the same tests that were considered. The figure below shows some sample images from the MNIST database.

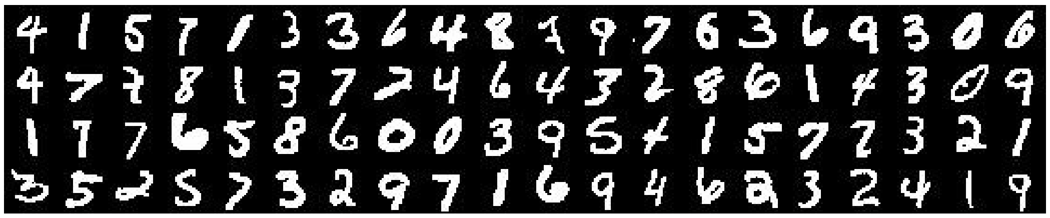


Figure 4.1 Sample images of digits from MNIST database.

4.4 Incremental algorithms for tensor objects

The conventional method used for incremental PCA is the SKL algorithm. In fact, most articles referenced earlier make use of the SKL algorithm mainly for its computational efficiency. For most image-as-vector systems, SKL is indeed very efficient. However, if the so-called "image-as-vector" systems are re-arranged as tensor objects with different data structures after different modes of unfolding, sometimes the covariance matrix itself provides new means for seeking additional computational benefits. In our study, the modified fast PCA and SKL algorithms are utilized for different unfolding modes in order to achieve better computational efficiency. The following sections introduce basic multilinear algebra, propose the new unfolding method, explain the incremental procedure and evaluate the computational complexity as it pertains to the incremental algorithm.

4.4.1 Unfolding methods

Traditional unfolding methods are called backward cyclic and forward cyclic. How to implement incremental learning based on the traditional unfolding method was addressed in detail in [Lathauwer et al., 2000]. In order to achieve incremental learning, an extra step of matrix computation is required. However, a new unfolding method is utilized in this study. Taking the backward cyclic unfolding for example, the elements in $A_{(n)}$ can be defined as $(A_{(n)})_{(index)} = \mathcal{A}_{i_1 i_2 \ldots i_N}$, where

$$index = [i_{n,} \sum_{p_1=n+1}^{N} (i_{p_1} - 1)(\prod_{p_2=p_1+1}^{N} I_{p_2})(\prod_{p_2=1}^{n-1} I_{p_2}) + \sum_{p_1=1}^{n-1} (i_{p_1} - 1)(\prod_{p_2=p_1+1}^{n-1} I_{p_2})] \tag{4.1}$$

with I_{p_2} being the length of the dimension p_2.

The elements obtained by the new unfolding method are defined with a different index,

$$index = [i_{n,} \sum_{p_2=N}^{n+1} (i_{p_2} - 1)(\prod_{p_1=p_2}^{n+1} I_{p_1})(\prod_{p=n-1}^{1} I_{p_1}) + \sum_{p_2=n-1}^{1} (i_{p_2} - 1)(\prod_{p_1=p_2-1}^{1} I_{p_1})] \tag{4.2}$$

Figure 4.2 shows the difference graphically between the new proposed unfolding method and the backward cyclic method of mode-1 unfolding for a 3rd-order tensor. Figure 4.3 illustrates the different unfolding procedures for mode-2 for both the forward cyclic unfolding method and the new unfolding method.

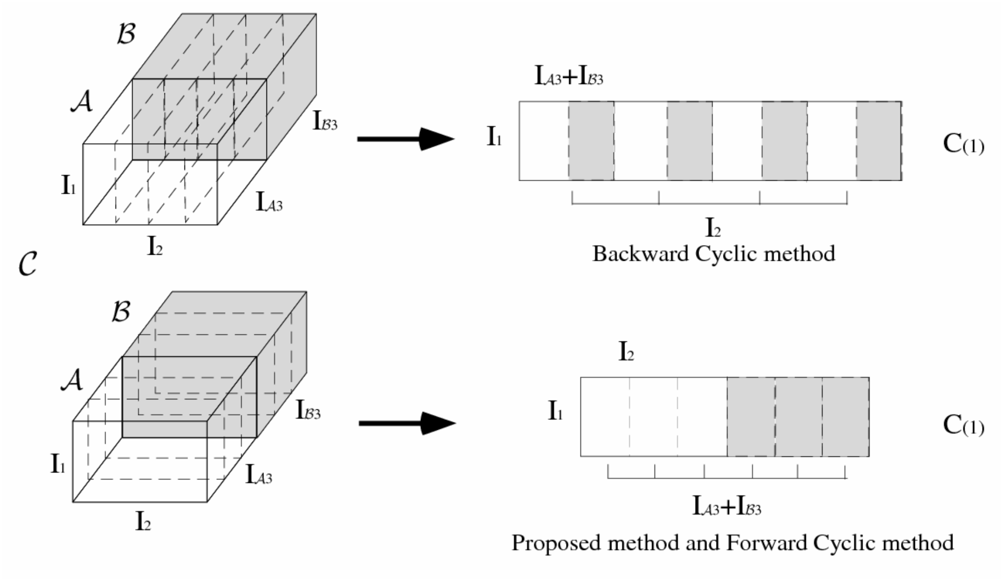


Figure 4.2 Unfolding procedures for mode-1.

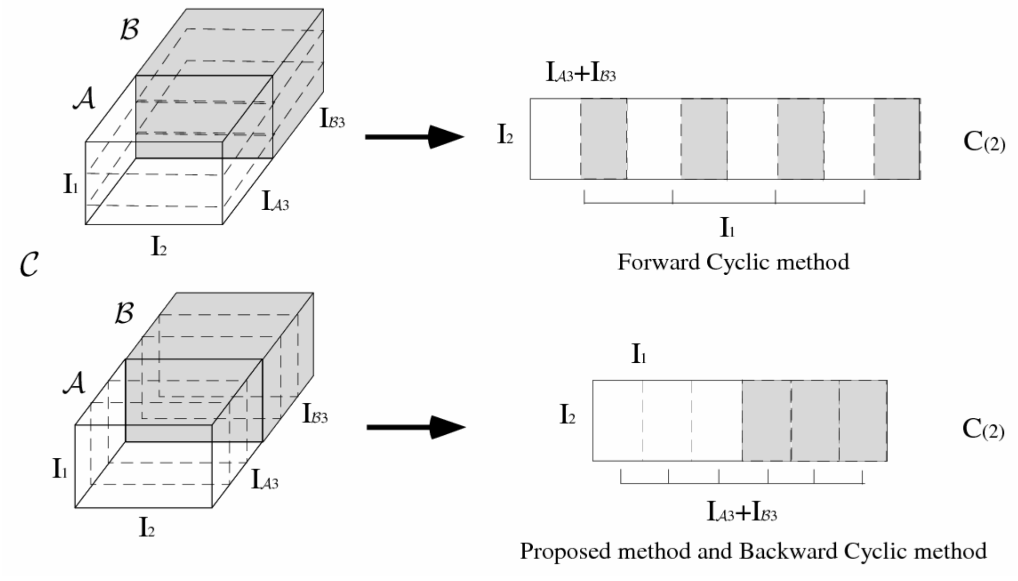


Figure 4.3 Unfolding procedures for mode-2.

The new method keeps the newly added data at the end of the matrix, which can be directly used in the incremental algorithm, instead of requiring additional matrix computations. The problem in the structuring of the unfolding between mode-1 and mode-2, as seen in Figure 4.2 and Figure 4.3, respectively is now resolved by the proposed algorithm where the structuring of the unfolding is unified in that the new data is placed at the end of the matrix for both mode-1 and mode-2.

Therefore, if A is the old tensor and B is the new tensor, their unfolding can be expressed as $C_{(n)} = [A_{(n)}, B_{(n)}]$ for $n = 1, \ldots, N-1$ and $C_{(N)} = [\begin{matrix} A_{(N)} \\ B_{(N)} \end{matrix}]$ for the last mode (n=N). Due to the multilinear property, the transpose of the data is utilized for the last mode.

4.4.2 Incremental Procedure

Both incremental processes of the SKL and MFPCA algorithms include mean update and total number of samples update, and these updates are defined as follows:

Mean update is given by

$$\mathcal{M}^{C} = (I_N^{\mathcal{A}} \mathcal{M}^{\mathcal{A}} + I_N^{\mathcal{B}} \mathcal{M}^{\mathcal{B}}) / (I_N^{\mathcal{A}} + I_N^{\mathcal{B}}) \tag{4.3}$$

where $\mathcal{M}^{\mathcal{A}}$ and $\mathcal{M}^{\mathcal{B}}$ represent the mean tensors for $\mathcal{A}$ and $\mathcal{B}$, $I_N^{\mathcal{A}}$ and $I_N^{\mathcal{B}}$ define the number of tensors in the old and new tensor sequences, respectively. The number of samples thus becomes

$$I_N^{C} = I_N^{\mathcal{A}} + I_N^{\mathcal{B}} \tag{4.4}$$

Moreover, when new samples are taken into account, the mean value changes, affecting as a consequence the old centered data in the sequence. Such a change should be taken into consideration. Mean value update was first provided in [Levy and Lindenbaum, 2000], and was then extended in [Ross et al., 2007], which not only provided explanation for mean update, but also included the concept of "forgetting factor". The so-called forgetting factor gives more weight to recent data over old data. In this study, these aforementioned concepts are extended to a tensor object. To facilitate the understanding of the mathematical foundation of the two incremental procedures, the mean update is described in the *Proposition* and the forgetting factor is defined in the *Corollary* below.

Proposition: Let $\mathcal{M}^{\mathcal{A}}$ be the mean tensor of $\mathcal{A}$, with $\hat{\mathcal{A}}$ being tensor $\mathcal{A}$ after centering. Let $U_{\hat{A}_{(n)}}(i)$ and $\lambda_{\hat{A}_{(n)}}(i)$, $i=1,...,k_{\hat{A}_{(n)}}$ be the largest $k_{\hat{A}_{(n)}}$ eigenvectors and eigenvalues of old unfolding data $\hat{A}_{(n)}$, respectively. A new tensor sequence is denoted as $\mathcal{B}$, with $\hat{\mathcal{B}}$ being tensor $\mathcal{B}$ after centering. Suppose further that $\mathcal{M}^{\mathcal{B}}$ is the mean tensor of $\mathcal{B}$. Then for the incremental modified fast PCA (IMFPCA) algorithm, the mode-*n* covariance matrix for the whole sequence with mean update can be expressed as:

$$\varphi_{\hat{C}_{(n)}} = \sum_{i=1}^{k_{\hat{A}_{(n)}}} \lambda_{\hat{A}_{(n)}}(i) U_{\hat{A}_{(n)}}(i) U_{\hat{A}_{(n)}}(i)^T + \hat{B}_{(n)} \hat{B}_{(n)}{}^T + \frac{I_N^A I_N^B}{I_N^A + I_N^B} (M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})(M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})^T \tag{4.5}$$

As for the SKL algorithm, the mode-*n* unfolding of $\hat{\mathcal{B}}$ with mean update is generated by

$$\tilde{B}_{(n)} = [\hat{B}_{(n)}, \sqrt{\frac{I_N^{\mathcal{A}} I_N^{\mathcal{B}}}{I_N^{\mathcal{A}} + I_N^{\mathcal{B}}}} (M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})] \tag{4.6}$$

and the matrix R is generated as

$$R = [\begin{matrix} diag(\lambda_{\hat{A}_{(n)}}) & U_{\hat{A}_{(n)}}^T \tilde{B}_{(n)} \\ 0 & \tilde{E}(\tilde{B}_{(n)} - U_{\hat{A}_{(n)}} U_{\hat{A}_{(n)}}^T \tilde{B}_{(n)}) \end{matrix}] \tag{4.7}$$

where $\tilde{E} = orth(\tilde{B}_{(n)} - U_{\hat{A}_{(n)}} U_{\hat{A}_{(n)}}^T \tilde{B}_{(n)})$ with the *orth* function being used to orthonormalize the column-wise vectors in the resulting matrix.

Corollary: For the same definitions provided in the aforementioned *Proposition* with the inclusion of the forgetting factor f , the covariance matrix can now be generated for the IMFPCA algorithm as follows:

$$\varphi_{\hat{C}_{(n)}} = f^2 \sum_{i=1}^{k_{\hat{A}_{(n)}}} \lambda_{\hat{A}_{(n)}}(i) U_{\hat{A}_{(n)}}(i) U_{\hat{A}_{(n)}}(i)^T + \hat{B}_{(n)} \hat{B}_{(n)}{}^T + \frac{I_N^{\mathcal{A}} I_N^{\mathcal{B}} (f^2 I_N^{\mathcal{B}} + I_N^{\mathcal{A}})}{(I_N^{\mathcal{A}} + I_N^{\mathcal{B}})^2} (M_{\hat{A}_{(n)}} - M_{\hat{B}_{(n)}})(M_{\hat{A}_{(n)}} - M_{\hat{B}_{(n)}})^T \tag{4.8}$$

As for the SKL algorithm, the mode-n unfolding of $\hat{B}$ with mean update is generated by

$$\tilde{B}_{(n)} = [\hat{B}_{(n)}, \sqrt{\frac{I_N^{\mathcal{A}} I_N^{\mathcal{B}} (f^2 I_N^{\mathcal{B}} + I_N^{\mathcal{A}})}{(I_N^{\mathcal{A}} + I_N^{\mathcal{B}})^2}} (M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})] \tag{4.9}$$

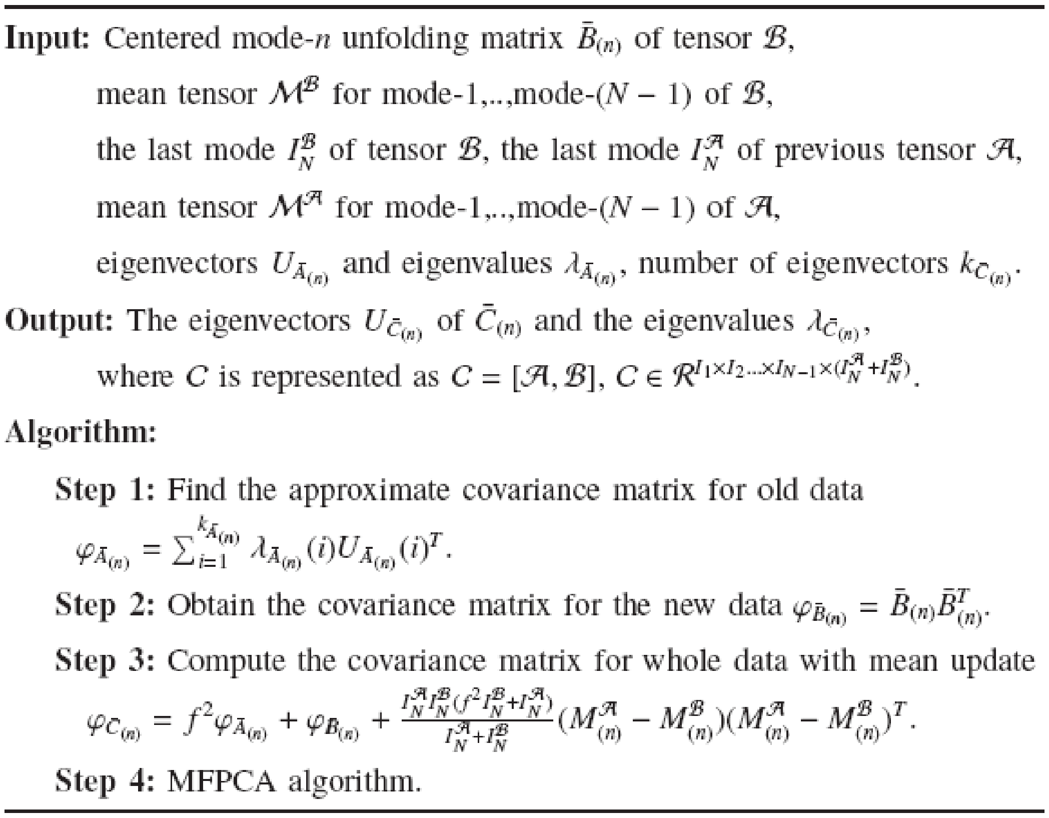
while matrix R is generated as

$$R = [\begin{matrix} f diag(\lambda_{\hat{A}_{(n)}}) & U_{\hat{A}_{(n)}}^T \tilde{B}_{(n)} \\ 0 & \tilde{E}(\tilde{B}_{(n)} - U_{\tilde{A}_{(n)}} U_{\tilde{A}_{(n)}}^T \tilde{B}_{(n)}) \end{matrix}] \tag{4.10}$$

Note that the proposition is a special case of the corollary, which considers the forgetting factor as 1. The proof for the proposition and corollary is provided in section 4.4.4. The proposition described earlier applies to steps 1 through 3 for the IMFPCA algorithm described in Table 4.1 and to only step 6 for the SKL algorithm described in Table 4.2. Tables 4.1 and 4.2 are the pseudo code for the steps considered for the two incremental algorithms SKL and IMFPCA.

In order to achieve better efficiency overall, the IMFPCA and SKL algorithms are applied in accordance to their computational requirements, which means that for mode-1 up to mode-(N-1), IMFPCA is applied; while for the specific mode-N, the SKL algorithm is used instead. Details on these computational requirements are explored next.

Table 4.1 Pseudocode for IMFPCA



```
Input: Centered mode-$n$ unfolding matrix $\bar{B}_{(n)}$ of tensor $\mathcal{B}$,
    mean tensor $\mathcal{M}^{\mathcal{B}}$ for mode-1,..,mode-$(N-1)$ of $\mathcal{B}$,
    the last mode $I_N^{\mathcal{B}}$ of tensor $\mathcal{B}$, the last mode $I_N^{\mathcal{A}}$ of previous tensor $\mathcal{A}$,
    mean tensor $\mathcal{M}^{\mathcal{A}}$ for mode-1,..,mode-$(N-1)$ of $\mathcal{A}$,
    eigenvectors $U_{\bar{A}_{(n)}}$ and eigenvalues $\lambda_{\bar{A}_{(n)}}$, number of eigenvectors $k_{C_{(n)}}$.
Output: The eigenvectors $U_{\bar{C}_{(n)}}$ of $\bar{C}_{(n)}$ and the eigenvalues $\lambda_{\bar{C}_{(n)}}$,
    where $C$ is represented as $C = [\mathcal{A}, \mathcal{B}]$, $C \in \mathcal{R}^{I_1 \times I_2 \ldots \times I_{N-1} \times (I_N^{\mathcal{A}} + I_N^{\mathcal{B}})}$.
Algorithm:
  Step 1: Find the approximate covariance matrix for old data
    $\varphi_{\bar{A}_{(n)}} = \sum_{i=1}^{k_{A_{(n)}}} \lambda_{\bar{A}_{(n)}}(i) U_{\bar{A}_{(n)}}(i)^T$.
  Step 2: Obtain the covariance matrix for the new data $\varphi_{\bar{B}_{(n)}} = \bar{B}_{(n)} \bar{B}_{(n)}^T$.
  Step 3: Compute the covariance matrix for whole data with mean update
    $\varphi_{C_{(n)}} = f^2 \varphi_{\bar{A}_{(n)}} + \varphi_{B_{(n)}} + \frac{I_N^{\mathcal{A}} I_N^{\mathcal{B}} (f^2 I_N^{\mathcal{B}} + I_N^{\mathcal{A}})}{I_N^{\mathcal{A}} + I_N^{\mathcal{B}}} (M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})(M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})^T$.
  Step 4: MFPCA algorithm.
```

Table 4.2 Pseudocode for SKL algorithm

```
Input: Centered mode-$n$ unfolding matrix $\bar{B}_{(n)}$ of tensor $\mathcal{B}$,
    mean tensor $\mathcal{M}^{\mathcal{B}}$ for mode-1,..,mode-$(N-1)$ of $\mathcal{B}$,
    the last mode $I_N^{\mathcal{B}}$ of tensor $\mathcal{B}$, the last mode $I_N^{\mathcal{A}}$ of previous tensor $\mathcal{A}$,
    mean tensor $\mathcal{M}^{\mathcal{A}}$ for mode-1,..,mode-$(N-1)$ of $\mathcal{A}$,
    eigenvectors $U_{\bar{A}_{(n)}}$ and eigenvalues $\lambda_{\bar{A}_{(n)}}$, number of eigenvectors $k_{C_{(n)}}$.
Output: The eigenvectors $U_{\bar{C}_{(n)}}$ of $\bar{C}_{(n)}$ and the eigenvalues $\lambda_{\bar{C}_{(n)}}$,
    where $C$ is represented as $C = [\mathcal{A}, \mathcal{B}]$,
    $C \in \mathcal{R}^{I_1 \times I_2 \ldots \times I_{N-1} \times (I_N^{\mathcal{A}} + I_N^{\mathcal{B}})}$.
Algorithm:
  Step 1: Generate data $\hat{B}_{(n)} = [\bar{B}_{(n)}, \sqrt{\frac{I_N^{\mathcal{A}} I_N^{\mathcal{B}}}{I_N^{\mathcal{A}} + I_N^{\mathcal{B}}}}(M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})]$.
  Step 2: Calculate $D = U_{\bar{A}_{(n)}}^T \hat{B}_{(n)}$.
  Step 3: Projection of new data $E = \hat{B}_{(n)} - U_{\bar{A}_{(n)}} D$.
  Step 4: Calculate the orthogonal basis by QR decomposition
    $\tilde{E} = qr(E)$, where $\tilde{E}$ is an orthogonal matrix.
  Step 5: Construct matrix $R = [\begin{matrix} f diag(\lambda_{\bar{A}_{(n)}}) & U_{\bar{A}_{(n)}} \\ 0 & \tilde{E}(\hat{B}_{(n)} - U_{\bar{A}_{(n)}} U_{\bar{A}_{(n)}}^T \hat{B}_{(n)}) \end{matrix}]$.
  Step 6: Calculate $SVD(R) = \tilde{U}\tilde{V}\tilde{Q}^T$.
  Step 7: Calculate $U = [U_{\bar{A}_{(n)}}, \tilde{D}^{(n)}]\tilde{U}$.
  Step 8: Define $\lambda_{C_{(n)}}(i) = \tilde{V}(i,i)$ and $U_{C_{(n)}}(i) = U(i)$,where $i = 1, \ldots, k_{C_{(n)}}$.
```

4.4.3 Computational Complexity

4.4.3.1 Computational Complexity of MFPCA

In the iterative procedure of the pseudocode of the MFPCA algorithm given in Table 1, the major processing steps and their respective computational requirements are as shown in Table 4.3. In these operations, i takes on the values from 1 to k. Therefore, with the given relation

$$1^2 + 2^2 + 3^2 + \ldots + k^2 = k(k+1)(2k+1)/6 \tag{4.11}$$

The order of complexity in the number of operations can thus be approximated for all L iterations as $O(I_n^2 L) + O(I_n k^3 L)$.

4.4.3.2 Computational Complexity of IMFPCA

The major processing steps with their respective computational complexities are given in Table 4.4. These results followed the same reasoning used for finding the order of complexity in the number of operations for the MFPCA.

4.4.3.3 Computational Complexity of SKL

The major processing steps of SKL with their computational complexities are as shown in Table 4.5.

Table 4.3 Computational complexity of the MFPCA method for a single iteration

Processing steps for the i$^{\text{th}}$ eigenvector for MFPCA	Computational Complexity
Calculate $U(i) = \varphi * U(i)$	$O(I_n^2)$
Gram-Schmidt process	$O(I_n i^2)$
Calculate the norm $\lambda(i)$ of $U(i)$	$O(I_n^2)$

Table 4.4 Computational Complexity of the IMFPCA Method

Major processing steps in IMFPCA	Computational Complexity
Approximate covariance matrix for $\bar{A}_{(n)}$	$O(I_n^2 k_{\bar{A}_{(n)}})$
Covariance matrix for $\bar{B}_{(n)}$	$O(I_n I_1 I_2 \ldots I_{N-1} I_N^{\mathcal{B}})$
Covariance matrix for $\bar{C}_{(n)}$	$O(I_n I_1 I_2 \ldots I_{N-1})$
Computational steps in MFPCA	$O(I_n^2 L) + O(I_n k_{\bar{C}_{(n)}}^3 L)$

Table 4.5 Computational Complexity for SKL

Major steps in SKL	Computational Complexity
Calculate $D = U_{\bar{A}_{(n)}}^T \hat{B}_{(n)}$	$O(k_{\bar{A}_{(n)}} I_1 I_2 ... I_{N-1} I_N^{\mathcal{B}})$
Projection of new data $E = \hat{B}_{(n)} - U_{\bar{A}_{(n)}} D$	$O(k_{\bar{A}_{(n)}} I_1 I_2 ... I_{N-1} I_N^{\mathcal{B}})$
Calculate the orthogonal basis by QR decomposition $\tilde{E} = qr(E)$.	$O(I_n I_1 I_2 ... I_{N-1} I_N^{\mathcal{B}})$
Suppose $S = \min(I_n, I_1 I_2 ... I_{n-1} I_{n+1} ... I_N)$, Calculate $SVD(R) = \tilde{U}\tilde{V}\tilde{Q}^T$	$O(I_n I_1 I_2 ... I_{N-1} I_N^{\mathcal{B}}) + O((k_{\bar{A}_{(n)}} + S)^3)$

From Tables 4.4 and 4.5, we can observe that the total number of operations for the IMFPCA method can be approximated by $O(I_n{}^2 (k_{\bar{A}_{(n)}} + L)) + O(I_n I_1 I_2 ... I_N I_N^{\mathcal{B}})$, while the total number of operations for SKL can be approximated by $O((k_{\bar{A}_{(n)}} + S)^3) + O((k_{\bar{A}_{(n)}} + I_n) I_1 I_2 ... I_N I_N^{\mathcal{B}})$. Since the number $k_{\bar{A}_{(n)}}$ is usually less than 15, and L is empirically observed to be less than 10, the two computational complexities can be simplified into $O(I_n{}^2) + O(I_n I_1 I_2 ... I_N I_N^{\mathcal{B}})$ and $O(S^3) + O(I_n I_1 I_2 ... I_N I_N^{\mathcal{B}})$. Therefore, for mode-1 up to mode-(N-1), it is obvious that IMFPCA outperforms SKL (in this case, $S = I_n$).

It is important to note that for the last mode (n=N), the dimensional parameters need to be changed to estimate the computational complexity. Suppose the parameters for the dimensions are $I_1{}^{old}, I_2{}^{old}, ..., I_{N-1}{}^{old}, I_N{}^{old}$, then the new parameters that are used for evaluating the computational complexity are defined as $I_N^{new} = I_1{}^{old} I_2{}^{old} ... I_{N-1}{}^{old}$, $I_1^{new}, I_2^{new}, ..., I_{N-2}^{new} = 1$, and $I_{N-1}^{new} = I_N{}^{old}$. The proof for this required change is provided in section 4.4.4.3. The computational complexity for the IMFPCA is thus far more simplified than that of the SKL algorithm (in this case, $S = I_1 I_2 ... I_{n-1} I_{n+1} ... I_N$).

For incremental learning, these theoretical results support the use of IMFPCA for mode-1 up to mode-(N-1) and the use of SKL solely for mode-N. Following these theoretical findings, empirical results are provided in the next section to verify the effectiveness of IMFPCA for face recognition as a real-world application.

4.4.4 Proofs for Proposition, Corollary and the Required Change

4.4.4.1 Proof of the proposition

This proof is to show the theoretical derivation of the equation (4.5) for IMFPCA and equation (4.6) for SKL.

The mean tensor for a tensor sequence $\mathcal{A} \in \Re^{I_1 \times I_2 ... \times I_N^{\mathcal{A}}}$ is defined as $\mathcal{M}^{\mathcal{A}} = \frac{1}{I_N} \sum_{i_N=1}^{I_N} \mathcal{A}(i_N)$ where $\mathcal{A}(i_N) \in \Re^{I_1 \times I_2 ... \times I_{N-1}}$ and $\mathcal{M}^{\mathcal{A}} \in \Re^{I_1 \times I_2 ... \times I_{N-1}}$. Similarly, for the new tensor sequence $\mathcal{B} \in \Re^{I_1 \times I_2 ... \times I_N^{\mathcal{B}}}$, the mean computed in the same fashion as for $\mathcal{A}$ is $\mathcal{M}^{\mathcal{B}} \in \Re^{I_1 \times I_2 ... \times I_{N-1}}$.

The centered tensor sequences are given by $\hat{\mathcal{A}}(i_N) = \mathcal{A}(i_N) - \mathcal{M}^{\mathcal{A}}$ and $\hat{\mathcal{B}}(i_N) = \mathcal{B}(i_N) - \mathcal{M}^{\mathcal{B}}$, where $i_N = 1, ..., I_N^{\mathcal{A}} / I_N^{\mathcal{B}}$. If a new tensor sequence $\mathcal{C}$ is generated as $\mathcal{C} = [\mathcal{A}, \mathcal{B}]$ then $\mathcal{C} \in \Re^{I_1 \times ... \times (I_N^{\mathcal{A}} + I_N^{\mathcal{B}})}$.

Suppose further that the parameters provided are eigenvectors $U_{\hat{A}_{(n)}} \in \Re^{I_n \times k_{\hat{A}_{(n)}}}$, eigenvalues $\lambda_{\hat{A}_{(n)}} \in \Re^{k_{\hat{A}_{(n)}} \times 1}$ for the mode-n unfolding matrix $\hat{A}_{(n)}$, mean tensor $\mathcal{M}^{\mathcal{A}}$ for tensor $\mathcal{A}$ and the new tensor sequence $\mathcal{B}$ with mean $\mathcal{M}^{\mathcal{B}}$. Let $\mathcal{M}^{C}$ be the mean tensor of $\mathcal{C}$, and $\mathcal{X}(i)$ be one tensor object in the tensor sequence, where $\mathcal{X}$ can be $\mathcal{A}$, $\mathcal{B}$ or $\mathcal{C}$. We can then compute the covariance matrix for mode-n unfolding matrix $\hat{C}_{(n)}$ as follows:

$$\begin{aligned}
\varphi_{\hat{C}_{(n)}} &= \sum_{i=1}^{I_N^{\mathcal{A}}} (C(i)_{(n)} - M_{(n)}^{C})(C(i)_{(n)} - M_{(n)}^{C})^T + \sum_{j=I_N^{\mathcal{A}}+1}^{I_N^{\mathcal{A}}+I_N^{\mathcal{B}}} (C(j)_{(n)} - M_{(n)}^{C})(C(j)_{(n)} - M_{(n)}^{C})^T \\
&= \sum_{i=1}^{I_N^{\mathcal{A}}} (A(i)_{(n)} - M_{(n)}^{\mathcal{A}} + M_{(n)}^{\mathcal{A}} - M_{(n)}^{C})(A(i)_{(n)} - M_{(n)}^{\mathcal{A}} + M_{(n)}^{\mathcal{A}} - M_{(n)}^{C})^T \\
&+ \sum_{j=1}^{I_N^{\mathcal{B}}} (B(i)_{(n)} - M_{(n)}^{\mathcal{B}} + M_{(n)}^{\mathcal{B}} - M_{(n)}^{C})(B(i)_{(n)} - M_{(n)}^{\mathcal{B}} + M_{(n)}^{\mathcal{B}} - M_{(n)}^{C})^T \\
&= \varphi_{\hat{A}}^{(n)} + \varphi_{\hat{B}}^{(n)} + I_N^{\mathcal{A}}(M_{(n)}^{\mathcal{A}} - M_{(n)}^{C})(M_{(n)}^{\mathcal{A}} - M_{(n)}^{C})^T + I_N^{\mathcal{B}}(M_{(n)}^{\mathcal{B}} - M_{(n)}^{C})(M_{(n)}^{\mathcal{B}} - M_{(n)}^{C})^T
\end{aligned} \tag{4.12}$$

Since the relation among $\mathcal{M}^{\mathcal{A}}$, $\mathcal{M}^{\mathcal{B}}$ and $\mathcal{M}^{C}$ is given by

$$\mathcal{M}^{C} = \frac{1}{I_N^{\mathcal{A}} + I_N^{\mathcal{B}}}(I_N^{\mathcal{A}}\mathcal{M}^{\mathcal{A}} + I_N^{\mathcal{B}}\mathcal{M}^{\mathcal{B}}) \tag{4.13}$$

this formula can be simplified to yield

$$\varphi_{\hat{C}_{(n)}} = \varphi_{\hat{A}_{(n)}} + \varphi_{\hat{B}_{(n)}} + \frac{I_N^{\mathcal{A}} I_N^{\mathcal{B}}}{I_N^{\mathcal{A}} + I_N^{\mathcal{B}}}(M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})(M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})^T \tag{4.14}$$

Therefore, for the IMFPCA algorithm, the mode-*n* covariance matrix for the sequence with mean update is thus given by

$$\begin{aligned}
\varphi_{\hat{C}_{(n)}} &= \sum_{i=1}^{k_{\hat{A}_{(n)}}} \lambda_{\hat{A}_{(n)}}(i) U_{\hat{A}_{(n)}}(:,i) U_{\hat{A}_{(n)}}(:,i)^T + \hat{B}_{(n)} \hat{B}_{(n)}{}^T + \\
&\frac{I_N^{\mathcal{A}} I_N^{\mathcal{B}}}{I_N^{\mathcal{A}} + I_N^{\mathcal{B}}}(M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})(M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})^T
\end{aligned} \tag{4.15}$$

which completes the proof.

For the SKL algorithm as a second part of the *Proposition*, we have

$$\begin{aligned}
\varphi_{\hat{C}_{(n)}} &= \varphi_{\hat{A}_{(n)}} + \hat{B}_{(n)} \hat{B}_{(n)}{}^T + \frac{I_N^{\mathcal{A}} I_N^{\mathcal{B}}}{I_N^{\mathcal{A}} + I_N^{\mathcal{B}}}(M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})(M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})^T \\
&= \varphi_{\hat{A}_{(n)}} + [\hat{B}_{(n)}, \sqrt{\frac{I_N^{\mathcal{A}} I_N^{\mathcal{B}}}{I_N^{\mathcal{A}} + I_N^{\mathcal{B}}}}(M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})][\hat{B}_{(n)}, \sqrt{\frac{I_N^{\mathcal{A}} I_N^{\mathcal{B}}}{I_N^{\mathcal{A}} + I_N^{\mathcal{B}}}}(M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})]^T
\end{aligned} \tag{4.16}$$

And recall that this equation can also be expressed as

$$\varphi_{\hat{C}_{(n)}} = \varphi_{\hat{A}_{(n)}} + \tilde{B}_{(n)}\tilde{B}_{(n)}^T \tag{4.17}$$

Therefore, the mode-*n* unfolding of $\mathcal{B}$ with mean update is generated simply by using $\tilde{B}_{(n)}$ as derived below

$$\tilde{B}_{(n)} = [\hat{B}_{(n)}, \sqrt{\frac{I_N^{\mathcal{A}} I_N^{\mathcal{B}}}{I_N^{\mathcal{A}} + I_N^{\mathcal{B}}}}(M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})] \tag{4.18}$$

4.4.4.2 Proof of the corollary

This proof is to show the theoretical derivation of equation (4.8) for IMFPCA and equations (4.9) and (4.10) for SKL.

From proof of *Proposition*, the forgetting factor f is added, yielding

$$\begin{aligned}
\varphi_{\hat{C}_{(n)}} &= \sum_{i=1}^{I_N^{\mathcal{A}}} (f(C(i)_{(n)} - M_{(n)}^{C}))(f(C(i)_{(n)} - M_{(n)}^{C}))^T + \\
&\quad \sum_{j=I_N^{\mathcal{A}}+1}^{I_N^{\mathcal{A}}+I_N^{\mathcal{B}}} (C(j)_{(n)} - M_{(n)}^{C})(C(j)_{(n)} - M_{(n)}^{C})^T \\
&= f^2 \sum_{i=1}^{I_N^{\mathcal{A}}} (C(i)_{(n)} - M_{(n)}^{C})(C(i)_{(n)} - M_{(n)}^{C})^T + \\
&\quad \sum_{j=I_N^{\mathcal{A}}+1}^{I_N^{\mathcal{A}}+I_N^{\mathcal{B}}} (C(j)_{(n)} - M_{(n)}^{C})(C(j)_{(n)} - M_{(n)}^{C})^T \\
&= f^2 \varphi_{\hat{A}_{(n)}} + \varphi_{\hat{B}_{(n)}} + f^2 I_N^{\mathcal{A}} (M_{(n)}^{\mathcal{A}} - M_{(n)}^{C})(M_{(n)}^{\mathcal{A}} - M_{(n)}^{C})^T + \\
&\quad I_N^{\mathcal{B}} (M_{(n)}^{\mathcal{B}} - M_{(n)}^{C})(M_{(n)}^{\mathcal{B}} - M_{(n)}^{C})^T \\
&= f^2 \varphi_{\hat{A}_{(n)}} + \varphi_{\hat{B}_{(n)}} + \frac{I_N^{\mathcal{A}} I_N^{\mathcal{B}} (f^2 I_N^{\mathcal{B}} + I_N^{\mathcal{A}})}{(I_N^{\mathcal{A}} + I_N^{\mathcal{B}})^2} (M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})(M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})^T
\end{aligned} \tag{4.19}$$

So for IMFPCA,

$$\begin{aligned}
\varphi_{\hat{C}_{(n)}} &= f^2 \sum_{i_n=1}^{k_{\hat{A}_{(n)}}} \lambda_{\hat{A}_{(n)}}(i) U_{\hat{A}_{(n)}}(:,i) U_{\hat{A}_{(n)}}(:,i)^T + \hat{B}_{(n)}\hat{B}_{(n)}{}^T + \\
&\quad \frac{I_N^{\mathcal{A}} I_N^{\mathcal{B}} (f^2 I_N^{\mathcal{B}} + I_N^{\mathcal{A}})}{(I_N^{\mathcal{A}} + I_N^{\mathcal{B}})^2} (M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})(M_{(n)}^{\mathcal{A}} - M_{(n)}^{\mathcal{B}})^T
\end{aligned} \tag{4.20}$$

And for SKL algorithm, the matrix R is thus generated as

$$R = \begin{bmatrix} fdiag(\lambda_{\hat{A}_{(n)}}) & U^T_{\hat{A}_{(n)}} \tilde{B}_{(n)} \\ 0 & \tilde{E}(\tilde{B}_{(n)} - U_{\hat{A}_{(n)}} U^T_{\hat{A}_{(n)}} \tilde{B}_{(n)}) \end{bmatrix} \tag{4.21}$$

where $\tilde{B}_{(n)} = orth(\hat{B}_{(n)} - U_{\hat{A}_{(n)}} U^T_{\hat{A}_{(n)}} \hat{B}_{(n)})$.

4.4.4.3 Need for the change in the dimensional parameters

Using the same definitions provided in the *Proposition*, for the last mode unfolding, we have $\hat{A}_{(N)} \in \Re^{I_N^{\mathcal{A}} \times I_1 \ldots I_{N-1}}$, $\hat{B}_{(N)} \in \Re^{I_N^{\mathcal{B}} \times I_1 \ldots I_{N-1}}$ and $\hat{C}_{(N)} \in \Re^{(I_N^{\mathcal{A}} + I_N^{\mathcal{B}}) \times I_1 \ldots I_{N-1}}$. Since we need to work on their transpose, then $\hat{A}^T_{(N)} \in \Re^{I_1 \ldots I_{N-1} \times I_N^{\mathcal{A}}}$, $\hat{B}^T_{(N)} \in \Re^{I_1 \ldots I_{N-1} \times I_N^{\mathcal{B}}}$ and $\hat{C}^T_{(N)} \in \Re^{I_1 \ldots I_{N-1} \times (I_N^{\mathcal{A}} + I_N^{\mathcal{B}})}$. In order to use the formulae provided in the *Proposition*, we set $\hat{I}_N^{\mathcal{A}} = \hat{I}_N^{\mathcal{B}} = I_1 \ldots I_{N-1}$, then let $\hat{I}_1, \ldots, \hat{I}_{N-2} = 1$, $\hat{I}_{N-1}^{\mathcal{B}} = I_N^{\mathcal{B}}$ and $\hat{I}_{N-1}^{\mathcal{A}} = I_N^{\mathcal{A}}$ to obtain $\hat{A}^T_{(N)} \in \Re^{\hat{I}_N^{\mathcal{A}} \times \hat{I}_1 \ldots \hat{I}_{N-1}^{\mathcal{A}}}$, $\hat{B}^T_{(N)} \in \Re^{\hat{I}_N^{\mathcal{B}} \times \hat{I}_1 \ldots \hat{I}_{N-1}^{\mathcal{B}}}$ and $\hat{C}^T_N \in \Re^{\hat{I}_N^{\mathcal{C}} \times \hat{I}_1 \ldots (\hat{I}_{N-1}^{\mathcal{A}} + \hat{I}_{N-1}^{\mathcal{B}})}$.

4.5 Experimental results

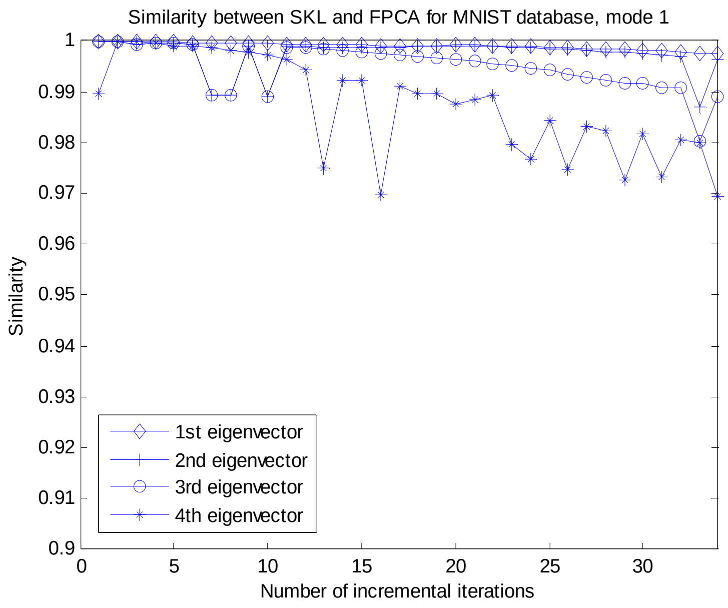
Several experiments were designed to assess the merit of the modified fast PCA (MFPCA) with its efficient incremental procedure. The key aspects of the algorithm investigated include:

1) Accuracy obtained from the incremental learning procedure, contrasting the results of both IMFPCA and SKL methods, and
2) Processing time required for the incremental subspace update under different unfolding modes between IMFPCA and SKL.

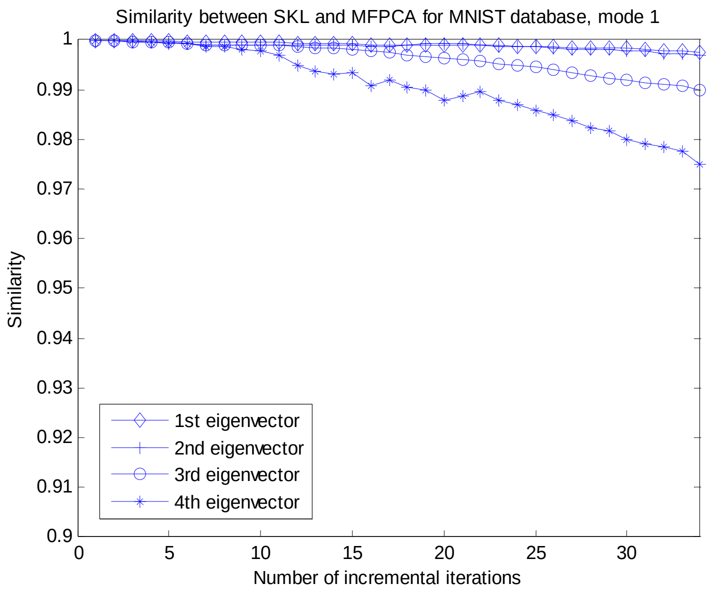
4.5.1 Accuracy and computational requirements for incremental procedure

There are two sets of experiments conducted in this section: (1) ascertaining similarity between the first four eigenvectors from SKL and IMFPCA, which is obtained by dot product; and (2) determining subspace distance among SKL, IMFPCA and batch mode PCA. The forgetting factor is set to be one in this test. The parameters for the dataset are set as follows: (a) PCA: $k=4$, (b) SKL: $k=8$, (c) IMFPCA: $k=8$ and (d) MFPCA: $k=8$. Only the first four eigenvectors are used in the test, the remaining four eigenvectors for the incremental algorithms are used to minimize the error. The batch number for AT&T is 10 and for MNIST is 30. A different error tolerance of $\varepsilon=10^{-7}$ is chosen to ensure that the incremental algorithm used will provide more accurate results.

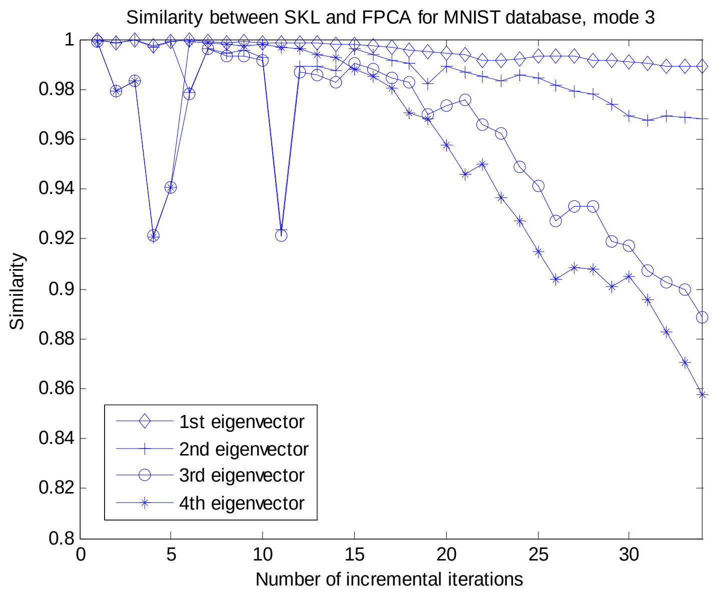
Since mode-2 is similar to mode-1, only mode-1 results were shown for simplicity sake. From the similarity results in Figure 4.4, it is obvious that IMFPCA performs better than IFPCA, and with smoother curves or transitions. Some similarities of the FPCA algorithm, as described in section 3.5, drop to a low value where a local minimum happens; however, it satisfied the criteria of minimizing the mean square error, which can still allow for the eigenvectors estimated after the local minimum to achieve a higher similarity. Moreover, although the final similarity values are close among the different algorithms, the IMFPCA is the more consistent in estimating the eigenvectors over many trials, as can be clearly seen in Figure 4.4. This outcome is significant and will yield better accuracy in real-world applications, such as incremental face recognition provided in this study. It is also essential to carry out experiments involving the use of IMFPCA with different error tolerance settings, as evidenced in the results provided below.


Similarity between SKL and FPCA for MNIST database, mode 1
Similarity
Number of incremental iterations
1st eigenvector
2nd eigenvector
3rd eigenvector
4th eigenvector

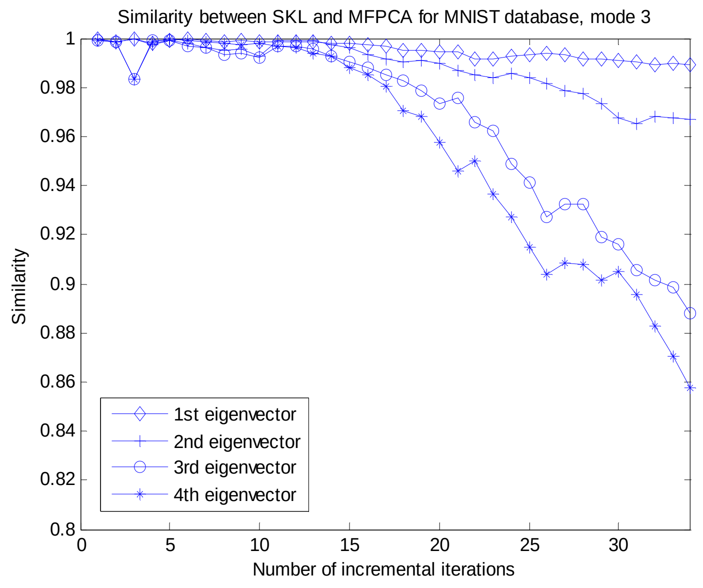
(a)


Similarity between SKL and MFPCA for MNIST database, mode 1
Similarity
Number of incremental iterations
1st eigenvector
2nd eigenvector
3rd eigenvector
4th eigenvector

(b)



(c)

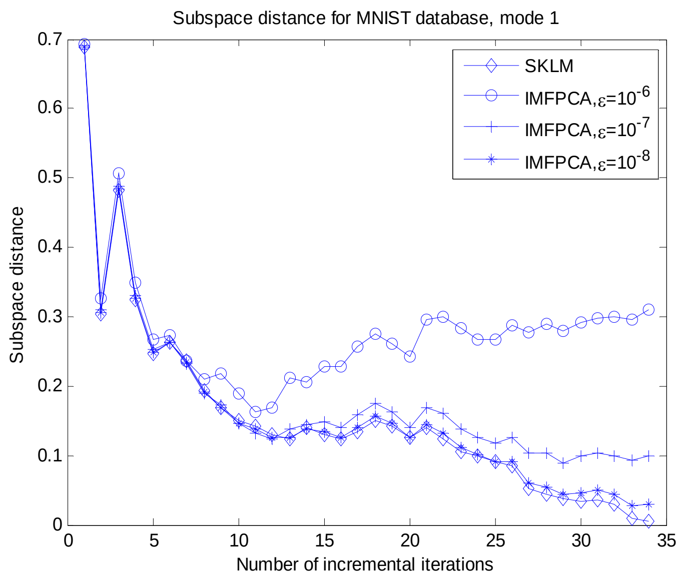


(d)

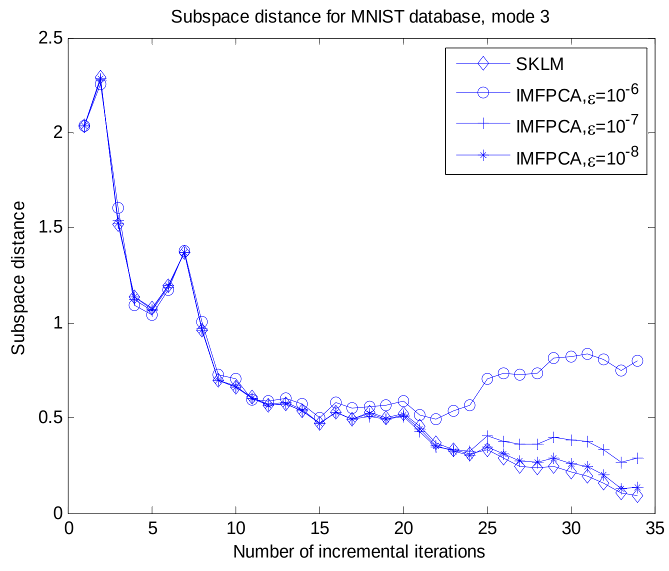
Figure 4.4 Similarity between eigenvectors from SKL and IMPCA and from SKL and IMFPCA using the MNIST database with mode-1 unfolding and mode-3 unfolding, respectively. The similarities of the top four eigenvectors are compared. The x-axis is used for the number of updates, and the y-axis is used for the similarity measure. For better visualization, different scales were adopted for different tests; however, the same test with different algorithms is visualized with the same scale.

The error accumulated along the incremental procedure brings up the discrepancy between the incremental results and the true results (from batch methods). This defect is unavoidable for incremental procedures. A distance measure based on the principal angles in [Knyazev and Argentati, 2002] between subspaces was used here to examine the nature of this discrepancy. The distance is expressed by $d(U_0, U_1) = \sqrt{\sum_{i=1}^{k} \theta_i^2}$ where U_0 and U_1 are k-dimensional subspaces and $\theta_1 \ldots \theta_k$ are the principal angles between them. Given $\theta_i \in [0, \frac{\pi}{2}]$, the distance satisfies $d \in [0, \frac{\sqrt{k}\pi}{2}]$. If the two subspaces are identical, then $d \to 0$. It can be seen that in Figure 4.5 the subspace distance converges to the true result gradually.

In Figure 4.5, the SKL algorithm is proven to perform better in this case than IMFPCA in all tests. With error tolerance of $\varepsilon = 10^{-6}$ for both mode-1 and mode-3, IMFPCA had the distance increased after ten iterations, which means the error tolerance cannot be adopted. However, if the error tolerance is set at a lower scale, such as $\varepsilon = 10^{-7}$ and $\varepsilon = 10^{-8}$, then the distance can be considered as a stable discrepancy from the true result (which is the result from eigen decomposition based on the whole dataset). The lower error tolerance leads to better convergence. It is worth noticing that the subspace distance for mode-1 is smaller than mode-3 (take error tolerance $\varepsilon = 10^{-7}$, for example, where the final subspace distance for mode-1 was 0.0991 and for mode-3 was 0.2888), which does not conform to the conclusion made from the experiment described in section 3.5, "under same error tolerance, with the increase of the dimension, the accuracy increases too." The reason is that in the incremental procedure, the dimension for mode-1 is 28, while the dimension for mode-3 is 784, and both of them used the first eight eigenvectors. Obviously, for the larger dimension mode, the error was higher. From this test, it can be concluded that both SKL and IMFPCA methods can provide acceptable discrepancy distance in the incremental procedure with proper parameter settings.



(a)



(b)

Figure 4.5 Subspace distance examination on MNIST database. (a) Mode 1: SKL, IMFPCA with $\varepsilon = 10^{-6}$, IMFPCA with $\varepsilon = 10^{-7}$ and IMFPCA with $\varepsilon = 10^{-8}$. (b) Mode 3: SKL, IMFPCA with $\varepsilon = 10^{-6}$, IMFPCA with $\varepsilon = 10^{-7}$ and IMFPCA with $\varepsilon = 10^{-8}$. And the curve is the average results from 100 trials.

4.5.2 Processing time requirements for the incremental procedure

The processing times of SKL and IMFPCA for the incremental procedure are compared next. It can be observed that for mode-1 unfolding, IMFPCA outperforms SKL; while for mode-3 unfolding, it is the SKL algorithm that outperforms IMFPCA.

Table 4.6 Incremental Learning Processing Time for Different Modes of Unfolding

Figure #	Total processing time for incremental procedure	
	IMFPCA	SKL
4.5(a), $\varepsilon = 10^{-6}$	0.3340s	11.1481s
4.5(a), $\varepsilon = 10^{-7}$	0.3927s	
4.5(a), $\varepsilon = 10^{-8}$	0.4607s	
4.5(b), $\varepsilon = 10^{-6}$	14.1518s	0.4938s
4.5(b), $\varepsilon = 10^{-7}$	18.3362s	
4.5(b), $\varepsilon = 10^{-8}$	23.7908s	

In Table 4.6, it shows that for mode-1, IMFPCA performs faster while for mode-3, the SKL is faster regardless of the error tolerance. These results validate the discussion on the computational complexity as provided in section 3.3 and it highlights the merit of the proposed algorithm.

CHAPTER V

Human face learning and recognition

5.1 Introduction

This chapter describes the process required for human face learning and recognition. The multilinear principal component analysis has already been proved in chapter two to have better accuracy than other conventional methods. The incremental algorithm based on multilinear principal component analysis provided in chapter four is utilized with the purpose of spending less time and still yield high accuracy.

5.2 Related works

Traditional face recognition methods rely on training set to obtain the classifier that will be used for the testing images. The mechanism for determining if the subject in one testing image is not in the training set exits; however, as more images from the subject are tested, there is no way for the system to recognize the subject. To resolve this problem, [Castrillón-Santana et al., 2007] provide a system of learning to recognize faces based on the incremental principal component analysis. Moreover, [Raducanu and Vitrià, 2007] also explored a system for a cognitive vision process using face recognition as a case study based on the non-parametric discriminant analysis. With the inspiration of their initial work, a system based on the incremental multilinear principal component analysis is described in this chapter.

5.3 Data and subjects

The face image dataset used in this set of experiments was collected from the face recognition database of University of Essex, the Georgia Tech face database, and the

AT&T database of faces.

The face recognition database of University of Essex has 395 subjects, with 20 images per subject. It contains images of people of various racial origins. Since most of them were first year undergraduate students, the majority of subjects are between 18 to 20 years old. Some of the pictures were taken with different background, different lighting condition and extreme variation of expressions. Some of the subjects have the pictures with or without glasses and with or without beards.

The Georgia Tech Face database contains images of 50 individuals. For each subject, there are 15 images captured between 06/01/99 and 11/15/99, with the resolution of 640x480. Most of the images present frontal faces with different illumination conditions, facial expression, and appearance.

For the experiments considered in this chapter, we have used 40 subjects from AT&T database, 150 subjects from the face recognition database of University of Essex (randomly chosen) and 50 subjects from the Georgia Tech Face database. And for each subject, if they have more than 10 images, only 10 images are chosen randomly to compose the dataset to be utilized in the experiments. Therefore, there are 240 subjects in total with 10 images each in the dataset. Since the three databases have different resolution, the pre-process step is necessary. All the images are preprocessed to include only the face and normalized into 168x118. Figure 5.1 provides sample images after pre-processing as illustrative examples.

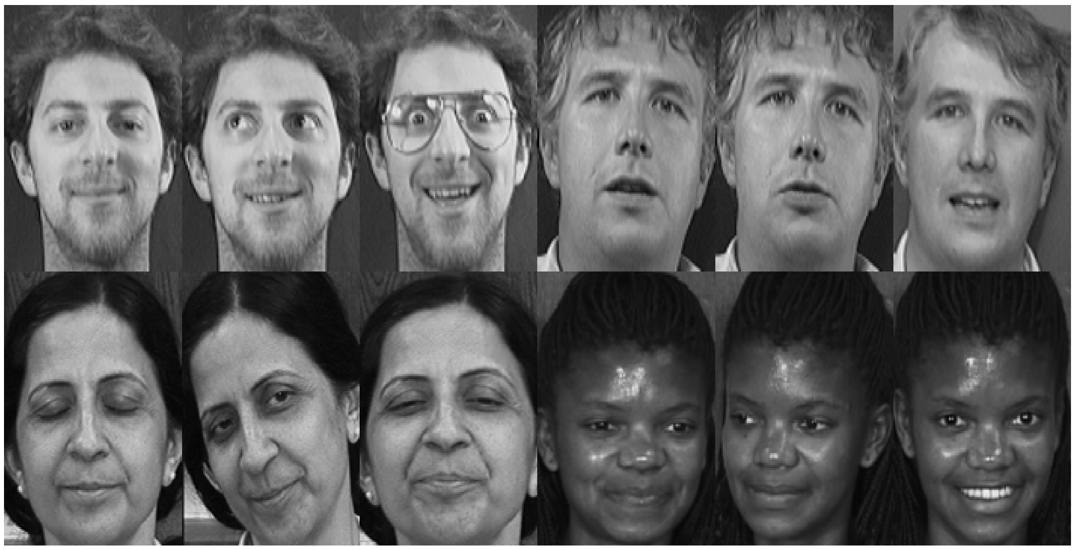


Figure 5.1 Sample images from the dataset used for the incremental face recognition experiment

5.4 Method

Different from the traditional face recognition system with a certain training set, the training set of the online face recognition increases as more faces are introduced. As one application of the proposed algorithm, a simple system is designed with an arbitrary threshold and without the verification step. To make the system more robust, adaptive thresholding and verification are usually suggested. The system for incremental face recognition is described through a flowchart in Figure 5.2.

The L-2 norm distance mentioned before is used in the recognition process. In the test, a threshold is given to evaluate if this image is already in the accumulated data. The index from the distance method gives the preliminary recognition result in the accumulated data for the testing image. Then it is assumed that when the distance is less than the threshold, the face is associated to a person in the accumulated data. Otherwise, this person will be considered as a new subject who is not yet included in the accumulated images. Since there are too many combinations among k_1, k_2 and k_3, in our case, we just select $k_1 = 2$, $k_2 = 3$ and $k_3 = 15$ randomly among those many other combinations that can be selected. And the batch number is set at 100. This batch number is used to indicate that after every 100 images, the eigenvectors can be updated incrementally (a different number could also be used, keeping in mind that the larger this number is the less iterations there would be and the less meaningful would be the comparison between the two methods).

Assuming that the first 200 images are used as the basis for the incremental learning with their identification, the threshold is chosen based on the following steps:

1. compute the eigenvectors of the first 100 images,
2. perform the recognition process on the other 100 images using their previous images as the training pool,
3. record the distance of the 100 images from the recognition process, and determine the threshold that will be used for deciding whether the new image is in the previous image data.

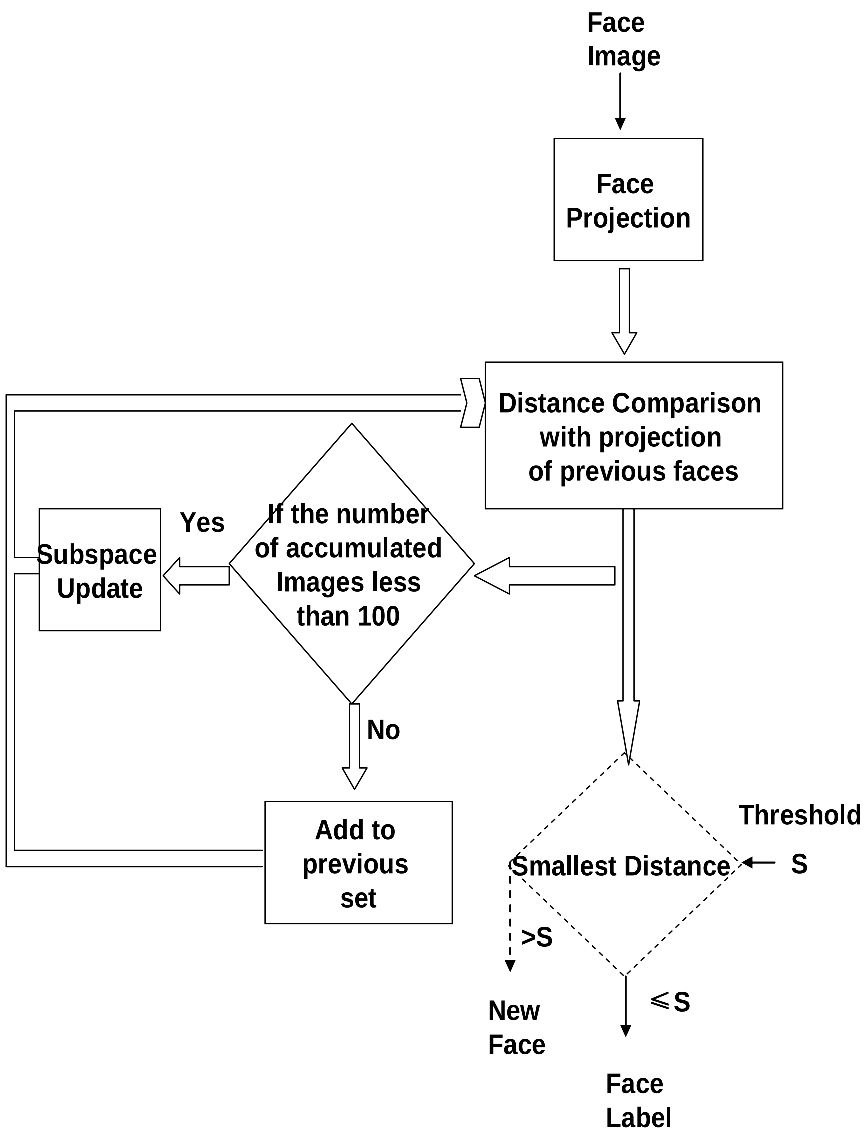


Figure 5.2 Flow chart for the incremental face recognition procedure

Since we have two classes, the optimal approach to select a threshold in bimodal histogram as illustrated in Figure 5.3 is one that maximizes the intra-class variance.

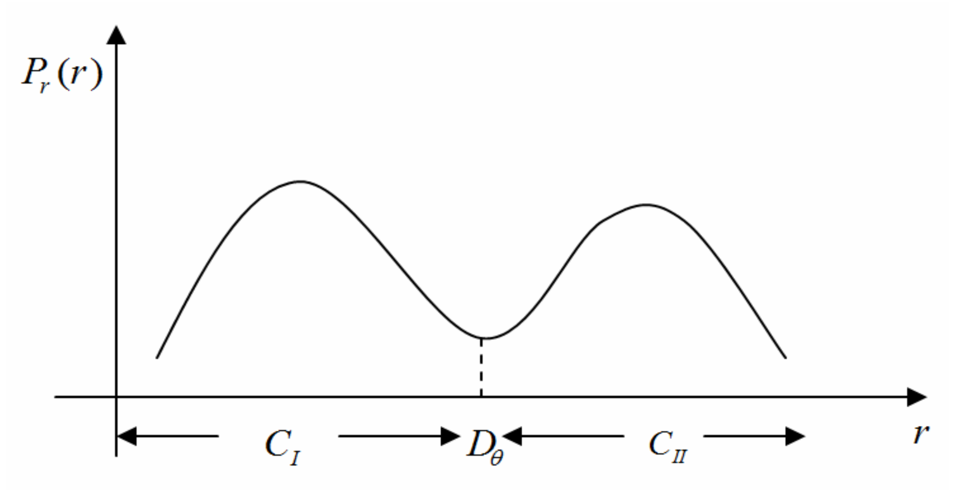


Figure 5.3 Bimodal histogram example.

Considering the bimodal histogram example in Figure 5.3, we have

$$P_I = P_r(C_I) = \sum_{i=1}^{\theta} P_i \tag{5.1}$$

$$P_{II} = P_r(C_{II}) = \sum_{i=\theta+1}^{ND} P_i \tag{5.2}$$

where θ is considered as the index of the threshold and D_θ the threshold that separates the two classes, P_I is the probability density of distances smaller than the threshold D_θ and P_{II} is the probability density of distances bigger than the threshold D_θ, and ND is the number of the distances in the recognition process.

The mean of the distances of all the images is

$$\mu = \sum_{i=1}^{ND} D_i P_i \tag{5.3}$$

And the mean distance of correctly recognized images and the mean distance of the wrongly recognized images are defined respectively by

$$\mu_I = \frac{1}{P_I} \sum_{i=1}^{\theta} D_i P_i \tag{5.4}$$

$$\mu_{II} = \frac{1}{P_{II}} \sum_{i=\theta+1}^{ND} D_i P_i \tag{5.5}$$

The threshold D_θ is optimal when the interclass variance given by

$$V(D_\theta) = P_I(\mu - \mu_I)^2 + P_{II}(\mu - \mu_{II})^2 \tag{5.6}$$

is maximized.

In this example, the parameters chosen are also $k_1 = 2$, $k_2 = 3$ and $k_3 = 15$ to obtain the histogram shown in Figure 5.4 which was needed to find D_θ.

With the conditions set above, the threshold is determined as 2.26×10^7. And the incremental accuracy is determined by $\frac{N_c}{N}$, where N_c is the number of images that are recognized correctly and N is the accumulated number of images that were used for incremental recognition.

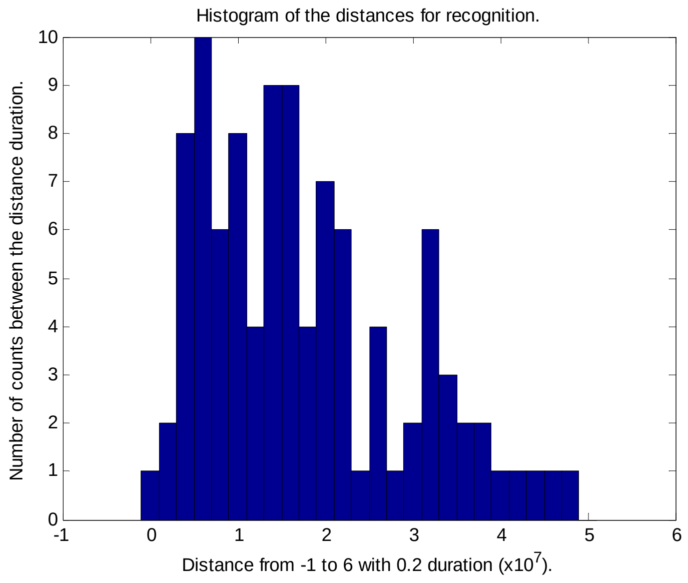


Figure 5.4 Histogram to determine the threshold

5.5 Results

From the recognition accuracy results given in Figure 5.5, the two methods provide identical performance in terms of recognition rate. However, when comparing the processing time, the proposed subspace update method has faster processing speed of ***85.3325s*** than the SKL-based IMPCA of ***285.9654s***.

Table 5.1 below provides evidence that the selection of the values of k_1, k_2 and k_3 can be randomly made and the results will always show that the proposed IMFPCA method is faster for the same accuracy than SKL-based method. It should be noted that the proposed method focuses on improving the computational efficiency for the incremental multilinear PCA. There are already many algorithms provided that can improve the recognition accuracy such as LDA, ICA and other PCA related algorithms [Chang and Hsu, 2009; Lin et al., 2009]. For those cases, when the incremental multilinear process is necessary, the proposed method can also be adopted to reduce the computational complexity without loss of accuracy.

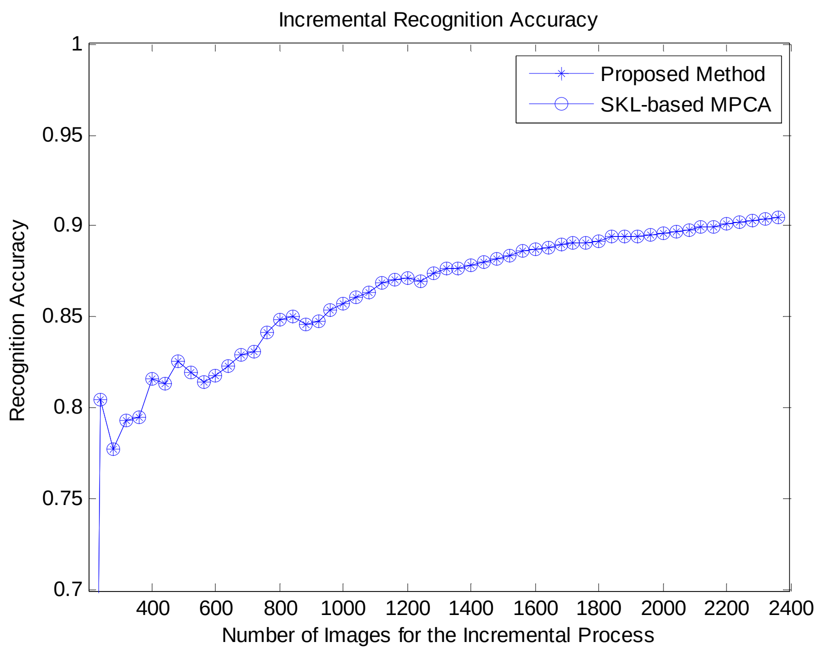


Figure 5.5 Incremental face recognition accuracy.

Table 5.1 Processing time and accuracy with different parameter settings. And for the processing time and accuracy, [* / *] means [Proposed / SKL].

		*k*3=10	*k*3=20
*k*1=2,*k*2=2	Processing time	86.3s / 286.0s	86.6s / 288.7s
	Accuracy	88.1% / 88.1%	89.2% / 89.2%
*k*1=2,k2=4	Processing time	84.8s / 289.5s	87.5s / 291.3s
	Accuracy	89.7% / 89.7%	90.1% / 90.1%
*k*1=4,k2=2	Processing time	86.2s / 289.6s	88.2s / 294.5s
	Accuracy	89.3% / 89.3%	89.8% / 89.8%
*k*1=4,k2=4	Processing time	87.2s / 298.4s	87.0s / 292.0s
	Accuracy	90.2% / 90.2%	90.6% / 90.6%

CHAPTER VI

Object tracking application

6.1 Introduction

As a challenging problem that it is, object tracking [Yilmaz et al., 2006] has been studied over years as face recognition [Zhao et al., 2003]. A lot of progress has been made since, however, critical problems such as object motion abruption, object appearance change and non-rigid object structures are still a cause for serious concern as they severely degrade the process of tracking. This chapter is to assess the performance of the method that was used for face recognition as it is now applied for object tracking even under the presence of noise and other unforeseen situations.

6.2 Data and subjects

[David indoor sequence] is provided by David Ross for researchers to test their algorithms. The sequence has a very clear object, David's face, with appearance change and background changes. One video in the [CAVIAR Test Case Scenarios] is utilized, which was taken by the surveillance camera. The subject in that sequence is small and the background is unchanged. The last sequence is a section of a tennis game, which has changed background and subject with vast scale movement.

6.3 Method

The proposed method introduced in chapter four is utilized here for the incremental update of the subspace. Besides the subspace update, another two important issues that should be concerned are the motion estimation and the likelihood determination. In this section, the motion estimation and the likelihood computation are introduced, and then the pseudocode of the tracking algorithm that address these issues is given.

6.3.1 Motion estimation

In our case, a Markov model with hidden state variables is used, in which the target motion between two successive frames is evaluated by the affine motion parameters. Suppose $Z_t = (x_t, y_t, \theta_t, s_t, \alpha_t, \beta_t)$ describes the affine motion parameters of a target at time t, where x_t denotes the x translation, y_t denotes the y translation, θ_t denotes the rotation angle, s_t denotes the scale, α_t denotes the aspect ratio, and β_t denotes the skew direction at time t. Given a set of observed images $\mathcal{G}_t = \{G_1, G_2, ..., G_t\}$, the Z_t can be estimated through Bayes' theorem as

$$p(Z_t \mid \mathcal{G}_t) \propto p(G_t \mid Z_t) \int p(Z_t \mid Z_{t-1}) p(Z_{t-1} \mid \mathcal{G}_{t-1}) dZ_{t-1} \tag{6.1}$$

where $p(G_t \mid Z_t)$ denotes the likelihood function, and $p(Z_t \mid Z_{t-1})$ denotes the dynamic model.

Based on its definition, Z_t is modeled with all parameters being independent through Gaussian distribution estimation, which gives the relation as

$$p(Z_t \mid Z_{t-1}) = \mathcal{N}(Z_t; Z_{t-1,} \Sigma) \tag{6.2}$$

where Σ denotes a diagonal covariance matrix whose diagonal elements are σ_x^2, σ_y^2, σ_θ^2, σ_s^2, σ_α^2 and σ_β^2.

6.3.2 Likelihood determination

The likelihood determination is related to the tensor algebra, which has already been introduced in chapter four. Therefore, only brief explanations are given.

Given a tensor $\mathcal{A} \in \Re^{I_1 \times I_2 \times I_3}$, a test image $\mathcal{T} \in \Re^{I_1 \times I_2 \times 1}$, a mean image

$\mathcal{M} \in \Re^{I_1 \times I_2 \times 1}$ the mode-i eigenbasis $U_{A_{(i)}} \in \Re^{I_i \times k_{A_{(i)}}}$ ($i = 1, 2$) and the mode-3 eigenbasis $U_{A_{(3)}} \in \Re^{I_1 I_2 \times k_{A_{(3)}}}$ of $\mathcal{A}$, the likelihood can be determined by the sum of the reconstruction error norms of the three modes as:

$$RE = \sum_{i=1}^{2} \left\| (J_{(i)} - M_{(i)}) - (J_{(i)} - M_{(i)}) \prod_{j=1}^{2} \times_j U_{A_{(i)}} U_{A_{(i)}}^T \right\| + \left\| (J_{(3)} - M_{(3)}) - (J_{(3)} - M_{(3)}) U_{A_{(3)}} U_{A_{(3)}}^T \right\| \tag{6.3}$$

The smaller is RE the larger is the likelihood determination.

The likelihood function can thus be estimated as:

$$p(G_t \mid Z_t) \propto \exp(-RE) \tag{6.4}$$

6.3.3 Tracking algorithm

The tracking algorithm can be summarized using the following steps.

(1) Locate the target object in the first frame, either manually or automatically.

(2) Initialize the eigenbasis, the mean and number of observations.

(3) Go to the next frame and find potential windows of subjects and do the interpolation to reconstruct the windows with the specific size setting through the motion estimation.

(4) Compute the likelihood for each reconstructed window.

(5) Store the parameters for the most likely window.

(6) Update the subspace if the condition satisfies the requirement, otherwise go to next step.

(7) If this frame is not the last one, go to step 3. Otherwise, stop.

There are tradeoffs among subspace update frequency, number of particles and the processing speed. Those parameters should be chosen according to the characteristic of the object in the sequence.

6.4 Results

For a grayscale video sequence with I_3 frames and of image size $I_1 \times I_2$, a tensor $A \in \Re^{I_1 \times I_2 \times I_3}$ is established. The parameter settings for different videos are given in Table 6.1. k is the number of eigenvectors retained for mode-1 and mode-2. Sometimes $k_{1,2}$ is bigger than $k_{A_{(1)}}$ and $k_{A_{(2)}}$ to compensate the discrepancy in the subspace update, but only $k_{A_{(1)}}$ and $k_{A_{(2)}}$ are used for likelihood determination, while for mode-3, they are k_3 and $k_{A_{(3)}}$. And ε is the error tolerance introduced in MFPCA algorithm. The region size defines the size of normalized tracking results. Different tracking tests with unique reasons to verify the effectiveness of the proposed algorithm are considered.

The first data sequence was a person walking in an indoor environment with face expression variation, pose change, illumination variation and the background change, and the face served as the tracking object. Obviously, the algorithm works fine to track the object through the video as shown in Figure 6.1.

The second data sequence was the same as the first one except that the Gaussian random noise with variance 0.05 was added. The purpose of this test is to determine if the proposed method is robust to noise disturbance, so the corresponding parameters are set to be the same as the first sequence, except for the size of the normalized region. And the result in Figure 6.2 shows that the proposed method is robust for noise prone situations.

The third one is a smaller object tracking. It can be observed in Figure 6.3 that the proposed method can track the object in the sequence. It should be mentioned that since the background of this sequence is unchanged, tracking algorithm special for still background can provide better results. The fourth sequence recorded a tennis player during a match who has a larger scale for movement, and the tennis player is the target. The tracking results in Figure 6.4 prove again the robustness of the proposed method.

Table 6.1 Tracking parameter settings

sequences	David (462 frames)	Noise David (462 frames)	Walk (135 frames)	Tennis (1175 frames)
$k_{1,2}$	2	3	2	2
$k_{A_{(1)}}$	2	2	2	2
$k_{A_{(2)}}$	2	2	2	2
k_3	5	5	5	10
$k_{A_{(3)}}$	5	5	4	5
ε	10^{-8}	10^{-8}	10^{-8}	10^{-8}
Region Size	12×12	20×20	12×12	12×12
Number of samples	300	300	300	500

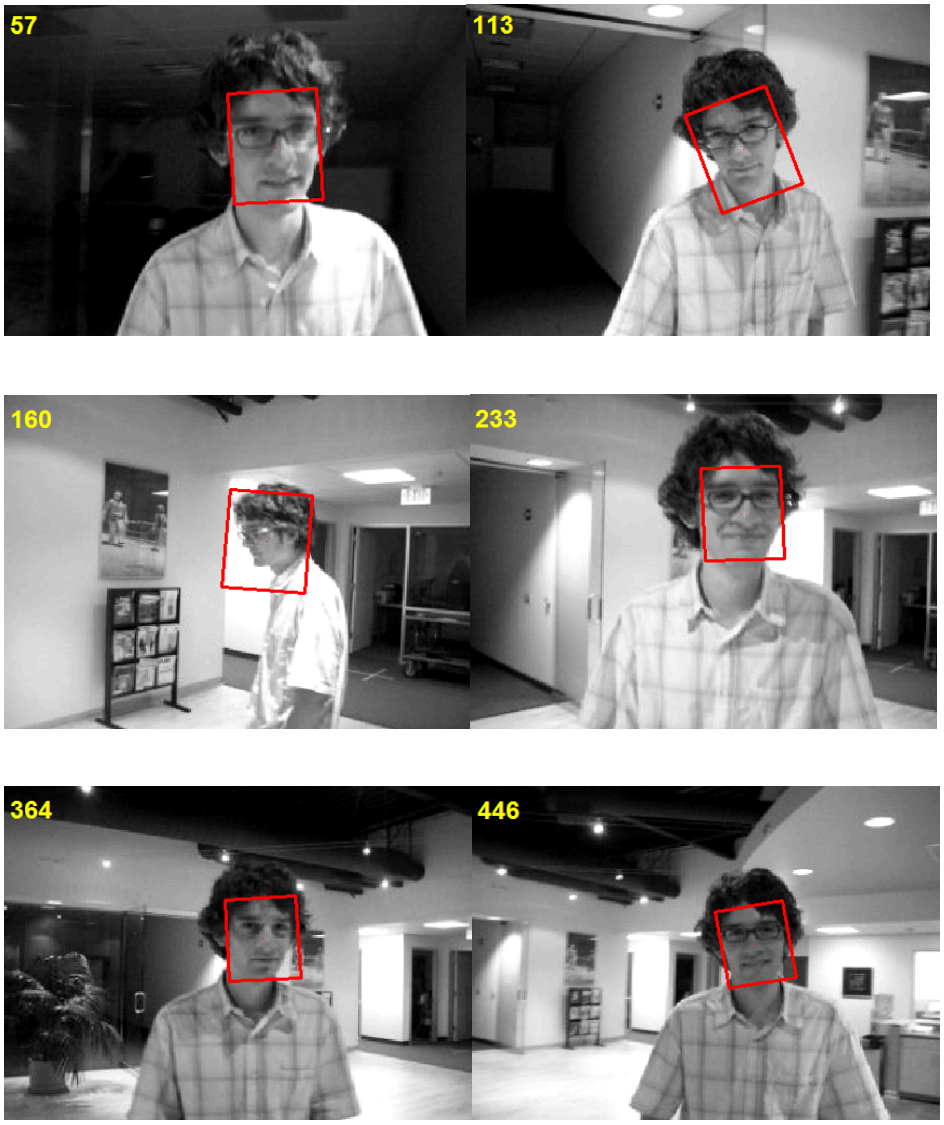


Figure 6.1 Tracking results of normal data sequence.

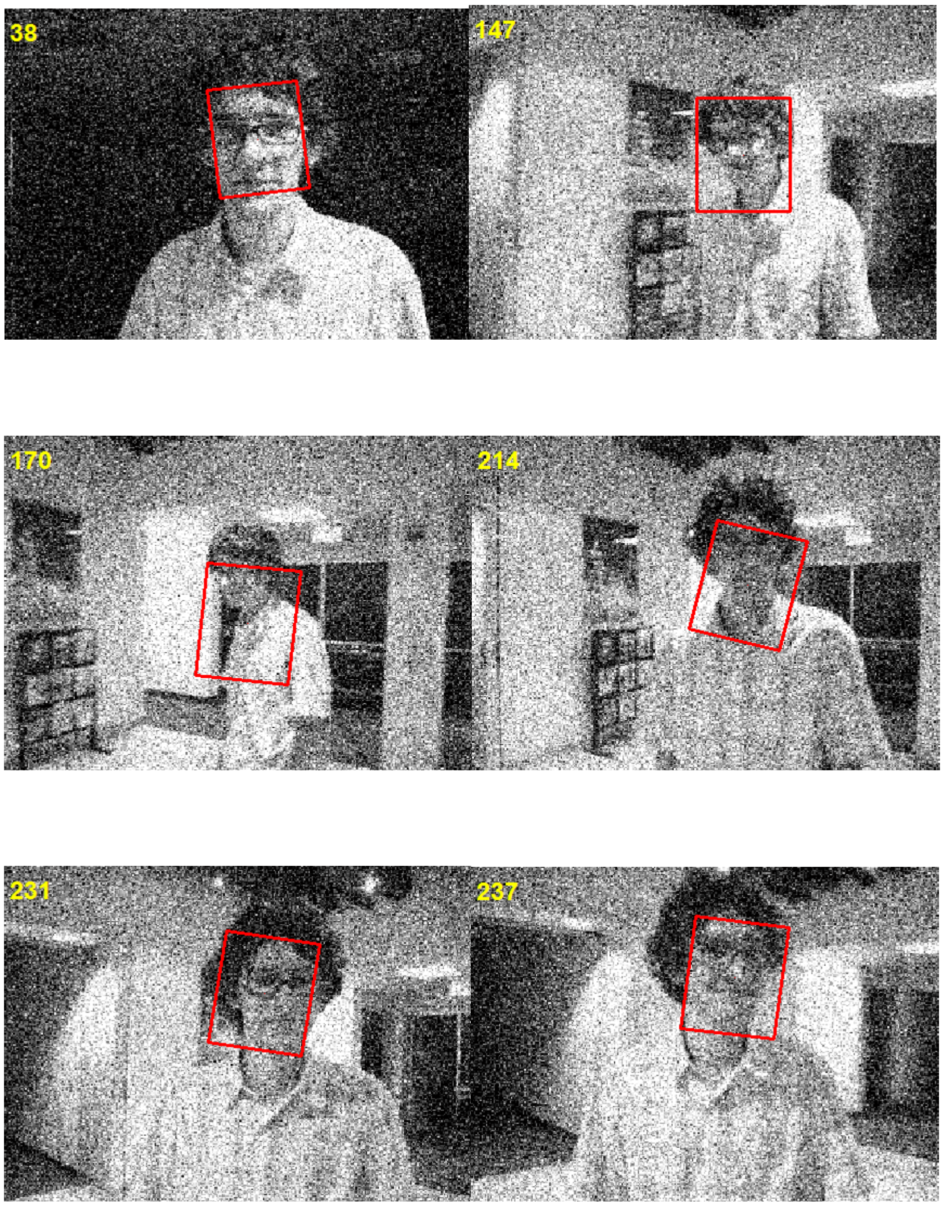


Figure 6.2 Tracking results of data sequence with noise.

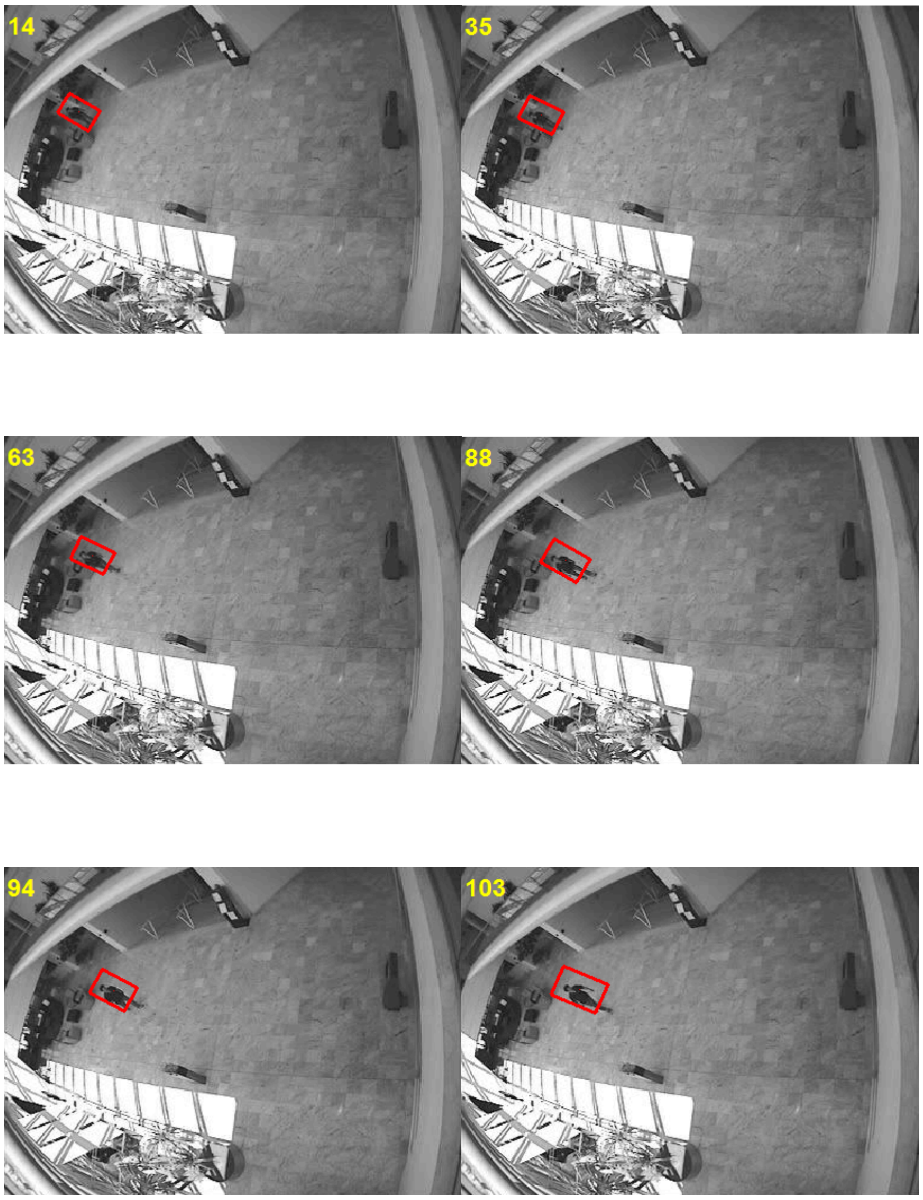


Figure 6.3 Tracking results of small target.

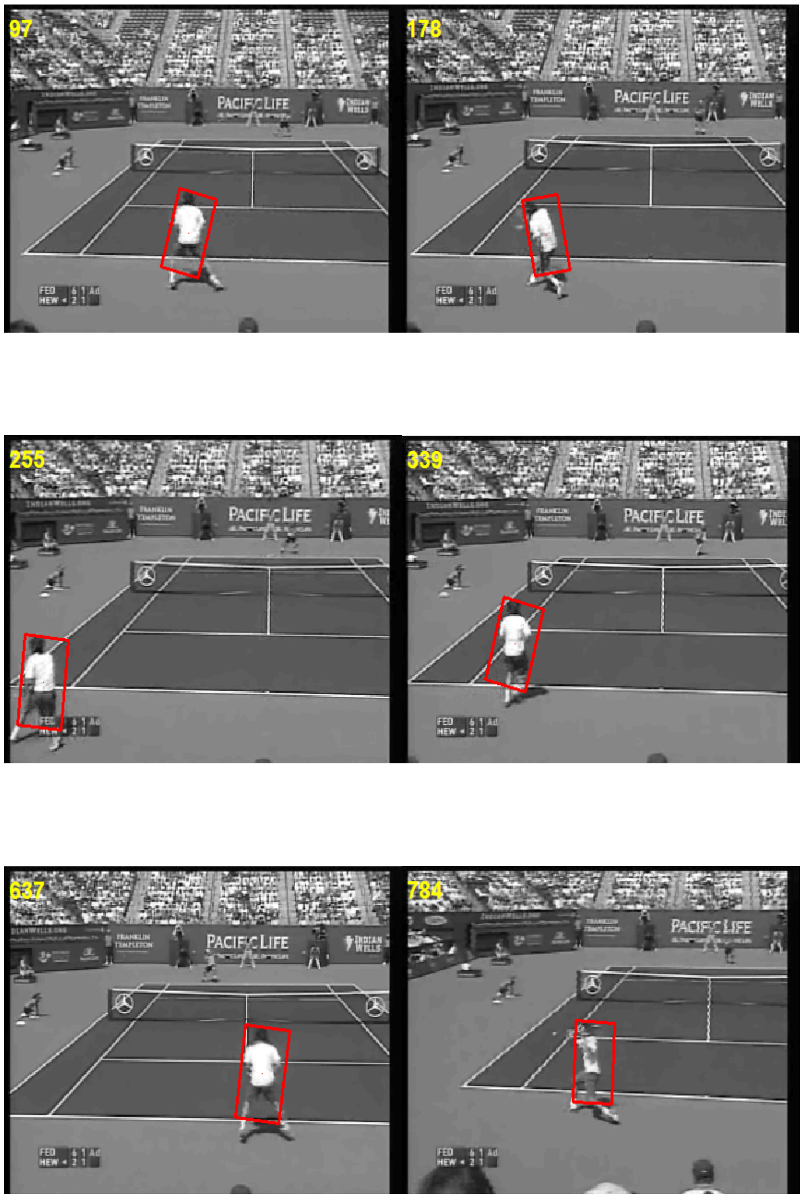


Figure 6.4 Tracking results of object with vast scale movement.

CHAPTER VII

Conclusions and future work

This dissertation established a novel system for human face learning and recognition based on incremental multilinear learning process based on the theoretical foundation of the Principal Component Analysis (PCA) and its leading eigenvectors.

As a consequence, a modified fast PCA method is introduced for estimating the leading eigenvectors with better accuracy than the fast PCA. This accomplishment is made while retaining the computational efficient of the fast PCA algorithm. The results show that the eigenvectors from the modified fast PCA has better similarity than the eigenvectors from fast PCA when compared to the Eigenvalue Decomposition-based PCA method as a benchmark. The second experiment provided in this study also confirms that if only few leading eigenvectors are required, the modified fast PCA does provide better accuracy than the fast PCA. This second experiment also shows that if a large number of eigenvectors is required, the results of both methods deteriorate slightly. It is also important to note with regards to the results provided in the first experiment, the assumption that by increasing the number of leading eigenvectors the results will improve is not always true in practice. This was demonstrated by comparing two testing methods on face recognition in that the number of eigenvectors that yielded the best accuracy with standard PCA would not necessarily lead to best accuracy when using either the fast PCA or the modified fast PCA when this number of eigenvectors is large. It is therefore recommended that the modified fast PCA be used with few leading eigenvectors, with the certainty that at least the accuracy obtained with these leading eigenvectors when using the standard PCA will be maintained.

The modified PCA was then tested in conjunction with the incremental process. First, the similarity measure is used to compare fast PCA and modified PCA in the incremental process, with the results indicating that the modified PCA has smoother curves than fast PCA, which means that the modified PCA is more stable. Second, the subspace distance for modified PCA in different modes of the incremental process is tested to prove that the proposed algorithm can estimate the leading eigenvectors with similar accuracy than SKL, while spending less time. These results are found to support the fact that the proposed method can be used for the incremental face recognition without compromising on accuracy and yet having a faster subspace update.

The overall strategy of the method also showed the ability to integrate the human face learning and the recognition process. The incremental multilinear method can thus be utilized to continuously update the subspace representation with the availability of new images. Moreover, with the comparison between the SKL and the proposed approach in chapter four, the proposed method is proven to be superior to SKL in the processing time of the subspace update process without any compromise on accuracy. The subspace updates method in this work is thus focused more on computational efficiency. In order to improve the accuracy of the system, methods with better recognition accuracy can be utilized, such as incremental multilinear linear discriminant analysis and incremental multilinear independent analysis. Both of them can use the concept provided in the proposed approach to have less computational complexity. For the sake of system robustness, an adaptive threshold and verification step were also suggested.

With the theoretical modifications firmly established in terms of their computational and accuracy merits, this dissertation also presented implementations of the proposed incremental learning method in challenging real world applications beyond face recognition. These included object tracking under different practical scenarios and under the presence of noise and other unforeseen situations inherent to real-world situations. Among the main issues that were addressed are change in appearance, change in background, and large scale movement. The results obtained under these challenges proved the practical merit and theoretical soundness of the proposed method in terms of both high accuracy and faster processing speed.

LIST OF REFERENCES

AT&T, the Database of Faces, available online, http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html

Bakir, G., Scholkopf, B., and Weston, J., "On the Pre-image Problem in Kernel Methods", *Kernel methods in Bioengineering, Signal and Image Processing*, Idea group publishing, pp. 284-302.

Bartelmaos, S., and Abed-Meraim, K., "Fast Principal Component Extraction Using Givens Rotations", *IEEE Signal Processing Letters*, Vol. 15, pp. 369-372, 2008.

Bartlett, M., Movellan, J., and Sejnowski, T., "Face Recognition by Independent Component Analysis", *IEEE Transactions on Neural Networks*, Vol. 13, No. 6, November 2002.

Berinde, V., "Iterative Approximation of Fixed points", *Springer*, 2007.

Castrillón-Santana, M., Déniz-Suárez, O., Lorenzo-Navarro, J., and Hernández-Tejera, M., "Learning to Recognize Faces by Successive Meetings", *Journal of Multimedia*, Vol. 1, No. 7, November/December 2006.

CAVIAR Test Case Scenarios, available online, http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/

Chin, T., and Suter, D., "Incremental Kernel Principal Component Analysis", *IEEE Transactions on Image Processing*, Vol. 16, No. 6, June 2007.

Chang, C., and Hsu, H., "Application of Principal Component Analysis to a Radial-basis Function Committee Machine for Face Recognition", *International Journal of Innovative Computing, Information and Control*, Vol. 5, No. 11(B), pp.4145-4154, 2009.

David Indoor sequence, available online, http://www.cs.toronto.edu/~dross/ivt/

Draper, B.A., Baek, K., Bartlett, M. S., and Beveridge, J.R., "Recognizing Faces with PCA and ICA", *Computer vision and image understanding: special issue on face recognition*, Vol. 91, issue 1-2, pp. 115-137, July-August 2003.

Duin, R., Juszczak, P., Ridder, D., Paclík, P., and Kalska, E., PR-Tools, a Matlab toolbox for pattern recognition, http://www.prtools.org, 2004.

Face Recognition Data, University of Essex, UK. Online available:

http://cswww.essex.ac.uk/mv/allfaces/index.html.

Hall, P., Marshall, D., and Martin, R., "Adding and Subtracting Eigenspaces with Eigenvale Decomposition and Singular Value Decomposition", *Image and Vision Computing*, 20(13-14), pp. 1009-1016, 2002.

Hoegaertsa, L., Lathauwerb, L., Goethalsa, I., Suykensa, J., Vandewallea, J., and Moora, B., "Efficiently Updating and Tracking the Dominant Kernel Principal Components", *Neural Networks*, Vol. 20, Issue 2, pp. 220-229, March 2007.

Hyvärinen, A., and Oja, E., "A Fast fixed-Point Algorithm for Independent Component Analysis", *Neural Computation*, Vol. 9, No. 7, pp. 1483-1492, October 1997.

Georgia Tech face database. Online available: http://www.anefian.com/face reco.htm.

Golub, G. H., and Reinsch, C., "Singular Value Decomposition and Least Squares Solutions", *Numerische Mathematik*, Vol. 14, No. 5, pp. 403-420, 1970.

Golub, G. H., and Loan, C., *Matrix Computation*, Johns Hopkins University Press, 1996.

Kim, K., Jung, K., and Kim, H., "Face recognition using Kernel Principal Component Analysis", *IEEE Signal Processing Letters*, Vol. 9, No. 2, February 2002.

Knyazev, A., and Argentati, M., "Principal Angels Between Subspaces in an A-based Scalar Product: Algorithms and Perturbation Estimates", *SIAM Journal of Scientific Computing*, Vol. 23, No. 6, pp. 2009-2041, 2002.

Lathauwer, L., Moor, B., and Vandewalle, J., "A Multilinear Singular Value Decomposition", *SIAM Journal on Matrix Analysis and Application*, Vol. 12, No. 4, pp. 1253-1278, March-May 2000.

Levy, A., and Lindenbaum, M., "Sequential Karhunen-Loeve Basis Extraction and its Application to Images", *IEEE Transactions on Image Processing*, Vol. 9, No. 8, August 2000.

Li, X., Hu, W., Zhang, Z., Zhang, X., and Luo, G., "Robust Vvisual Tracking Based on Incremental Tensor Subspace Learning", *IEEE 11th International Conference on Computer Vision*, 2007.

Lin, S., Lin, J., and Chiang, C., "Combining Scale Invariant Feature Transform with Principal Component Analysis in Face Recognition", *ICIC Express Letters*, Vol.3, No.4 (A), pp.927-932, 2009.

Lu, H., Plataniotis, K., and Venetsanopoulos, A., “MPCA: Multilinear Principal Component Analysis for Tensor Objects”, *IEEE Transactions on Neural Networks*, Vol. 19, Issue 1, pp. 18-39, January 2008.

MNIST Database of Handwritten Digits, available online, http://yann.lecun.com/exdb/mnist

Ozawa, S., Pang, S., and Kasabov, N., “Incremental Learning of Chunk Data for Online Pattern Classification Systems”, *IEEE Transactions on Neural Networks*, Vol. 19, Issue 6, pp. 1061-1074, June 2008.

Raducanu, B., and Vitrià, J., “Incremental Subspace Learning for Cognitive Visual Processes”, *Proceedings of the 2nd international conference on Advances in brain, vision and artificial intelligence*, pp. 214-223 , 2007.

Reddy, K., and Herron, T., “Computing the Eigen Decomposition of a Symmetric Matrix in Fixed-point Algorithms”, *IEEE Bangalore Section, Tenth Annual Symposium on Multimedia Communications and Signal Processing*, Nov 23-24, 2001.

Ross, D., Lim, J., Lin, R., and Yang, M., “Incremental Learning for Robust Visual tracking”, *International Journal of Computer Vision, Special Issue: Learning for Vision*, 2007.

Schilling, R. J., and Harris, S. L., “Applied Numerical Methods for Engineers using Matlab and C”, *Brooks/Cole Publishing Company*, 2000.

Schölkopf, B., Smola, A., and Müller, K.R., “Nonlinear Component Analysis as a Kernel Eigenvalue Problem”, *Neural Computing*, Vol. 10, pp. 1299-1319, 1998.

Sharma, A., and Paliwal, K. K., “Fast Principal Component Analysis using Fixed-point Algorithm”, *Pattern Recognition Letters*, 28, pp. 1151-1155, 2007.

Struc, V., The INface toolbox for illumination invariant face recognition, http://www.mathworks.com/matlabcentral/fileexchange/26523-the-inface-toolbox-for-illumination-invariant-face-recognition, January 2010.

Sun, J. , Tsourakakis, C. E., Hoke, E., Faloutsos, C. and Eliassi-Rad, T., “Two Heads Better than One: Pattern Discovery in Time-evolving Multi-aspect Data”, *Data Mining and Knowledge Discovery*, Vol. 17, No. 1, pp. 111-128, August 2008.

Turk, M. and Pentland, A., “Eigenfaces for Recognition”, *Journal of Cognitive Neuroscience*, Vol. 3, No. 1, pp. 71-86, 1991.

Wang, H. and Ahuja, N., “ A Tensor Approximation Approach to Dimensionality Reduction”, *International Journal of Computer Vision*, Vol. 76, No. 3, pp. 217-229, March 2008.

Yang, J., Zhang D., Frangi, A. F., and Yang, J., “Two-Dimensional PCA: a New Approach to Appearance-based Face Representation and Recognition”, *IEEE Transactions Pattern Analysis and Machine Intelligence*, Vol. 26, No. 1, pp. 131-137, January 2004.

Yilmaz, A., Javed, O., and Shah, M., “Object Tracking: A Survey”, *ACM computing Surveys*, Vol. 38, No. 4, article 13, Dec. 2006.

Yu, H., and Bennamoun, M., “1D-PCA, 2D-PCA to nD-PCA”, *Proceeding of the 18th International Conference on Pattern Recognition*, 2006.

Yuen, P. C., and Lai, J. H., “Face Representation Using Independent Component Analysis”, *Pattern Recognition*, Vol. 35, No. 6, pp. 1247-1257, 2002.

Zhao, H., Yuen, P. C., and Kwok, J. T., “A Novel Incremental Principal Component Analysis and Its Application for Face Recognition”, *IEEE Transactions on Systems, Man and Cybernetics—Part B: Cybernetics*, Vol. 36, No. 4, August 2006.

Zhao, W., Chellappa, R., Rosenfeld, A., and Phillips, P. J., “Face recognition: A literature survey”, *ACM Computing Surveys*, pp. 399-458, 2003.

VITA

JIN WANG

1983	Born, Wuhan, China
2005	B.S., Telecommunication Engineering Wuhan University of Technology Wuhan, China
2010	Ph.D. candidate in Electrical Engineering Florida International University Miami, Florida

HONORS AND AWARDS

Florida International University, Teaching Assistant (2005 - 2006)

Florida International University, Research Assistant (2007 - 2009)

Florida International University, Dissertation Year Fellowship (2010)

PUBLICATIONS AND PRESENTATIONS

Journal Publications

1. J. Wang, A. Barreto, N. Rishe, J. Andrian and M. Adjouadi, "A fast Incremental Multilinear Principal Component Analysis for Object Tracking", Second Revision, *International Journal of Innovative Computing, Information and Control*, 2010.

2. J. Wang, A. Barreto, L. Wang, Y. Chen, N. Rishe, J. Andrian, M. Adjouadi, "Multilinear Principal Component Analysis for Face Recognition with Fewer Features", **Neurocomputing**, (73): 1550-1555, June 2010.

3. Chen, Y; Adjouadi, M; Han, CA; Wang, J; Barreto, A; Rishe, N, et al., "A highly accurate and computationally efficient approach for unconstrained iris segmentation", **Image and Vision Computing**, 28 (2): 261-269 Sp. Iss. SI FEB 2010.

Conference Publications

4. J. Wang and M. Adjouadi, “Modified fast Principal Component Analysis”, The 2009 International Conference on Image Processing, Computer Vision, and Pattern Recognition (IPCV 09), Vol. I, pp. 374-377, Las Vegas, Nevada, USA, July 13-16, 2009.

5. Y. Chen, J. Wang, C. Han, L. Wang, and M. Adjouadi, “A robust segmentation approach to iris recognition based on video”, Applied Imagery Pattern Recognition (AIPR), Washington DC, 37th IEEE, Digital Object Identifier 10.1109/AIPR.2008.4906441, Oct. 15-17, 2008.

6. J. Wang, Y. Chen, and M. Adjouadi, “A Comparative Study of the Multi-linear PCA for Face Recognition”, Applied Imagery Pattern Recognition (AIPR), Washington DC, 37th IEEE, Digital Object Identifier 10.1109/AIPR.2008.4906476, Oct. 15-17, 2008.