A Novel Incremental Principal Component Analysis and Its Application for Face Recognition

Haitao Zhao, Pong Chi Yuen, *Member, IEEE*, and James T. Kwok, *Member, IEEE*

***Abstract*—Principal component analysis (PCA) has been proven to be an efficient method in pattern recognition and image analysis. Recently, PCA has been extensively employed for face-recognition algorithms, such as eigenface and fisherface. The encouraging results have been reported and discussed in the literature. Many PCA-based face-recognition systems have also been developed in the last decade. However, existing PCA-based face-recognition systems are hard to scale up because of the computational cost and memory-requirement burden. To overcome this limitation, an incremental approach is usually adopted. Incremental PCA (IPCA) methods have been studied for many years in the machine-learning community. The major limitation of existing IPCA methods is that there is no guarantee on the approximation error. In view of this limitation, this paper proposes a new IPCA method based on the idea of a singular value decomposition (SVD) updating algorithm, namely an SVD updating-based IPCA (SVDU-IPCA) algorithm. In the proposed SVDU-IPCA algorithm, we have mathematically proved that the approximation error is bounded. A complexity analysis on the proposed method is also presented. Another characteristic of the proposed SVDU-IPCA algorithm is that it can be easily extended to a kernel version. The proposed method has been evaluated using available public databases, namely FERET, AR, and Yale B, and applied to existing face-recognition algorithms. Experimental results show that the difference of the average recognition accuracy between the proposed incremental method and the batch-mode method is less than 1%. This implies that the proposed SVDU-IPCA method gives a close approximation to the batch-mode PCA method.**

***Index Terms*—Error analysis, face recognition, incremental principal component analysis (PCA), singular value decomposition (SVD).**

I. Introduction

FACE RECOGNITION has been an active research area in the computer-vision and pattern-recognition societies [1]–[6] in the last two decades. Since the original input-image space has a very high dimension [2], a dimensionality-reduction technique is usually employed before classification takes place. Principal component analysis (PCA) [1], [7] is one of the most popular representation methods for face recognition. It does not only reduce the image dimension, but also provides a compact feature for representing a face image. The well-known eigenface system was developed in 1991. In 1997, PCA was also employed for dimension reduction for linear discriminant analysis and a new algorithm, namely fisherface, was developed. After that, PCA has been extensively employed in face-recognition technology.

Usually, PCA is performed in batch mode. It means that all training data have to be ready for calculating the PCA projection matrix during training stage. The learning stops once the training data have been fully processed. If we want to incorporate additional training data into an existing PCA projection matrix, the matrix has to be retrained with all training data. In turn, it is hard to scale up the developed systems. To overcome this limitation, an incremental method is a straightforward approach.

Incremental PCA (IPCA) has been studied for more than two decades in the machine-learning community. Many IPCA methods have also been developed. Basically, existing IPCA algorithms can be divided into two categories. The first category computes the principal components (PCs) without computing the covariance matrix [13], [16], [17]. To the best of our knowledge, the candid covariance-free IPCA (CCIPCA) [13] algorithm developed by Weng *et al.* is the most recent work. Instead of estimating the PCs by stochastic gradient ascent (SGA) [17], the CCIPCA algorithm generates "observations" in a complementary space for the computation of the higher order PCs. Suppose that sample vectors are acquired sequentially, e.g., $u(1), u(2), \ldots$, possibly infinite. The first k dominant PCs $v_1(n), v_2(n), \ldots, v_k(n)$ are obtained as follows [13].

For $n = 1, 2, \ldots$, do the followings steps.

1) $u_1(n) = u(n)$.
2) For $i = 1, 2, \ldots, \min(k, n)$, do:
   a) if $i = n$, initialize the ith PC as $v_i(n) = u_i(n)$;
   b) otherwise

$$v_i(n) = \frac{n-1-l}{n} v_i(n-1) + \frac{1+l}{n} u_i(n) u_i^{\mathrm{T}}(n) \frac{v_i(n)}{\|v_i(n)\|}$$

$$u_{i+1}(n) = u_i(n) - u_i^{\mathrm{T}}(n) \frac{v_i(n)}{\|v_i(n)\|} \frac{v_i(n)}{\|v_i(n)\|}$$

where l is the amnesic parameter [13].

Since the PCs are obtained sequentially and the computation of the $(i+1)$th PC depends on the ith PC, the error will be propagated and accumulated in the process. Although the estimated vector $v_i(n)$ converges to the ith PC, no detailed error analysis was presented in [13]. Further, Weng *et al.* [13] defined

Manuscript received February 7, 2005; revised July 18, 2005. This project was supported in part by a Faculty research Grant of the Hong Kong Baptist University and in part by the Research Grant Council under Grant HKBU-2119/03E. This paper was recommended by Associate Editor H. Qiao.

H. Zhao is with the Institute of Aerospace Science and Technology, Shanghai Jiaotong University, China, and also with the Department of Computer Science, Hong Kong Baptist University, Hong Kong, SAR (e-mail: zhaoht@sjtu.edu.cn).

P. C. Yuen is with the Department of Computer Science, Hong Kong Baptist University, Hong Kong, SAR (e-mail: pcyuen@comp.hkbu.edu.hk).

J. T. Kwok is with the Department of Computer Science, The Hong Kong University of Science and Technology, Hong Kong, SAR (e-mail: jamesk@cs.ust.hk).

Digital Object Identifier 10.1109/TSMCB.2006.870645

a sample-to-dimension ratio n/d, where n is the number of training samples and d is the dimension of the sample space. If the ratio is small, it may be a problem from the statistical-estimation point of view. In the face-recognition problem, the high dimension of the face images is usually larger than 10 000. Even though 5000 training samples are used, the sample-to-dimension ratio $(5000/10\,000 = 0.5)$ is still very low [13]. Hence, methods in this category are not suitable for face-recognition technology.

The second category of the IPCA algorithms [14], [18]–[20], [26] reconstructs the significant PCs from the original training images and a newly added sample. When a new sample is added, the dimension of the subspace is increased by one. The updated eigenspace is obtained by using low-dimensional coefficient vectors of the original images. The reason is that the coefficient vectors and reconstructed images are only represented in different coordinate frames. Since the dimension of the eigenspace is small, this approach is computationally efficient [14]. Since the new samples are added one by one and the least significant PC is discarded to preserve the dimension of the subspace, this approach also suffers from the problem of unpredicted approximation error. Hall *et al.* [12] also demonstrated that, for the updating of the eigenspace with low dimension coefficient vectors, adding one datum each time is much less accurate than adding complete spaces.

On the other hand, there are powerful mathematical tools, namely SVD updating algorithms [8], [21]–[25], which were developed based on singular value decomposition (SVD). Zha and Simon [8] further enhance the SVD updating algorithm and applied it to latent semantic indexing (LSI) for information retrieval. They investigated matrices possessing the so-called low-rank-plus-shift structure, i.e., matrix A satisfying

$$A^{\mathrm{T}}A = \text{a low-rank matrix} + \text{a multiple of the identity matrix.}$$

If a matrix has the low-rank-plus-shift structure, the SVD updating algorithm will obtain the same best low-rank approximation as in the batch mode. This means that no error will be introduced. Theoretically, this approach is really good and interesting. However, in practice, it is difficult to prove that the data matrix has the low-rank-plus-shift structure. It is because the data matrix will be changed with the insertion of new sample(s), and the rank of the data matrix can be changed during the incremental-learning process. Another important missing item is that there is no analysis on error in case a matrix does not satisfy the low-rank-plus-shift structure.

In this paper, we develop a new IPCA method based on the SVD updating algorithm, namely the SVD updating-based IPCA (SVDU-IPCA) algorithm. Instead of working directly on the data matrix X, we apply our method on the matrix $\Sigma = (X - E(X))^{\mathrm{T}}(X - E(X))$, where $E(\cdot)$ is the expectation operator. This is often employed in implementing the PCA-based face-recognition methods, such as the eigenface method [1]. The dimensionality of the face image (usually higher than 10 000) is much higher than that of the data used in LSI (usually less than 1000). Working on the matrix Σ can result in computational savings. We also prove that if the data matrix has the low-rank-plus-shift structure, the error bound using our proposed SVDU-IPCA method is also equal to zero. However, as discussed, it is hard to prove that the input matrices have the low-rank-plus-shift structure. Therefore, we have presented a detailed error analysis and derived an error bound for the approximation between the incremental algorithm and the batch mode. Another characteristic of our proposed SVDU-IPCA method is that it can be easily extended to the kernel version. Since the conventional incremental methods [8], [13], [19] need to compute the additional vectors in the feature space (not in terms of inner products), they are difficult to adopt to the kernel trick.

The rest of this paper is organized as follows. Section II briefly reviews PCA and SVD updating algorithms. Section III presents our proposed SVDU-IPCA method. A mathematical analysis of our proposed SVDU-IPCA algorithm is presented in Section IV. Section V reports our incremental kernel PCA (IKPCA). Experimental results are shown in Section VI, and Section VII concludes this paper.

II. Brief Review on PCA and SVD Updating

In this section, we will briefly review the procedures of the PCA and SVD updating algorithms [8].

A. *Principal Component Analysis*

PCA [30] is one of the oldest and best known techniques in multivariate analysis. Let $x \in \mathbb{R}^n$ be a random vector, where n is the dimension of the input space. The covariance matrix of x is defined as $\Xi = E\{[x - E(x)][x - E(x)]^{\mathrm{T}}\}$. Let $u_1, u_2, \ldots, u_n$ and $\lambda_1, \lambda_2, \ldots, \lambda_n$ be eigenvectors and eigenvalues of Ξ, respectively, and $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. Then, PCA factorizes Ξ into $\Xi = U\Lambda U^{\mathrm{T}}$, with $U = [u_1, u_2, \ldots, u_n]$ and $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$. One important property of PCA is its optimal signal reconstruction in the sense of minimum mean squared error (mse). It follows that once the PCA of Ξ is available, the best rank-m approximation of Ξ can be readily computed. Let $P = [u_1, u_2, \ldots, u_m]$, where $m < n$. $y = P^{\mathrm{T}}x$ will be an important application of PCA in dimensionality reduction.

B. *SVD Updating*

Let $A \in \mathbb{R}^{m\times n}$, $A_{m\times n} = U\Lambda V^{\mathrm{T}}$, and its best rank-$k$ approximation $\hat{A}_{m\times n} \equiv U_k\Lambda_k V_k^{\mathrm{T}}$, where U_k and V_k are formed by the first k columns of U and V, respectively, and Λ_k is the kth leading principal submatrix of Λ. For any matrix $A \in \mathbb{R}^{m\times n}$, we will use $\mathrm{best}_k(A)$ to denote its best rank-k approximation.

The SVD updating algorithm [8], [15] provides an efficient way to carry out the SVD of a larger matrix $[A_{m\times n}, B_{m\times r}]$, where B is an $m \times r$ matrix consisting of r additional columns.

The procedure in obtaining the best rank-k approximation of $[A, B]$ can be summarized in Algorithm 1.

By exploiting the orthonormal properties and block structure, the SVD computation of $[A, B]$ can be efficiently carried out by using the smaller matrices, U_k, V_k, and the SVD of the smaller matrix $\begin{bmatrix} \Lambda_k & U_k^{\mathrm{T}}B \\ 0 & R \end{bmatrix}$. The computational-complexity analysis and details of the SVD updating algorithm are described in [8].

Algorithm 1—SVD Updating [8]:

1) Obtain the QR decomposition $(I - U_kU_k^{\mathrm{T}})B = QR$.
2) Obtain the rank-k SVD of the $(k+r') \times (k+r)$ matrix

$$\begin{bmatrix} \Lambda_k & U_k^{\mathrm{T}}B \\ 0 & R \end{bmatrix} = \hat{U}\hat{\Lambda}\hat{V}^{\mathrm{T}}$$

where r' is the rank of $(I - U_kU_k^{\mathrm{T}})B$.

3) The best rank-k approximation of $[A, B]$ is $([U_k, Q]\hat{U})\hat{\Lambda}\left(\begin{bmatrix} V_k & 0 \\ 0 & I \end{bmatrix}\hat{V}\right)^{\mathrm{T}}$.

III. Proposed IPCA Algorithm

This section presents our proposed SVDU-IPCA algorithm based on the SVD updating algorithm. A comprehensive theoretical analysis on SVDU-IPCA will be given in the next section.

Let $x_1, x_2, \ldots, x_m$ be the original face-image vectors, let $x_{m+1}, x_{m+2}, \ldots, x_{m+r}$ be the newly added face-image vectors, and let the data matrix $X_1 = [x_1, x_2, \ldots, x_m]$, the data matrix $X_2 = [x_{m+1}, x_{m+2}, \ldots, x_{m+r}]$, and $X = [X_1, X_2]$. Due to the high dimensionality of the face image and the difficulty in developing the incremental-learning method based on the covariance matrix Ξ, we proposed to develop the incremental-learning algorithm based on the matrix $\Sigma = (X - E(X))^{\mathrm{T}}(X - E(X))$.

Let

$$\Sigma_1 = (X_1 - E(X_1))^{\mathrm{T}}(X_1 - E(X_1)) \tag{1}$$

and

$$\Sigma = (X - E(X))^{\mathrm{T}}(X - E(X)) = \begin{bmatrix} \Sigma_1 & \Sigma_2 \\ \Sigma_2^{\mathrm{T}} & \Sigma_3 \end{bmatrix} \tag{2}$$

where Σ_2 is of size $m \times r$ and Σ_3 is of size $r \times r$. Due to the high dimensionality of the face image, wherein Σ_1 and Σ are often singular, we develop the incremental-learning method based on the presupposition of positive semidefinite matrices.[1] Then, Σ can also be written as

$$\Sigma = \begin{bmatrix} P & Q_1 \\ Q_2 & Q_3 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} P & Q_1 \\ Q_2 & Q_3 \end{bmatrix}.$$

Fig. 1 shows the basic idea of our proposed IPCA algorithm graphically. We adopt the notations from [22], in which the rotation sign in Fig. 1 represents the orthogonal transformation. The idea of the proposed incremental procedure is described as follows. The rank-k eigendecomposition of $\Sigma_1 = P^{\mathrm{T}}P$ corresponds to best rank-k approximation $\tilde{P}^{(1)}$ of P. We can then obtain the best rank-k approximation $\tilde{P}^{(2)}$ of $\begin{bmatrix} \tilde{P}^{(1)} \\ Q_2 \end{bmatrix}$. From the theoretical analysis in the next section, we will find that $Q_2 = 0$. Hence, no computational cost is required in this procedure. Then, we can obtain the best rank-k approximation of $[\tilde{P}^{(2)} \begin{bmatrix} Q_1 \\ Q_3 \end{bmatrix}]$. This will approximate the SVD of $\begin{bmatrix} P & Q_1 \\ Q_2 & Q_3 \end{bmatrix}$, from which we can get an approximated eigendecomposition of

$$\begin{bmatrix} P & Q_1 \\ Q_2 & Q_3 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} P & Q_1 \\ Q_2 & Q_3 \end{bmatrix} = \Sigma.$$

[1]In other applications, Σ_1 and Σ can be positive definite and the proposed SVDU-IPCA algorithm can be obtained by the same theoretical derivation.

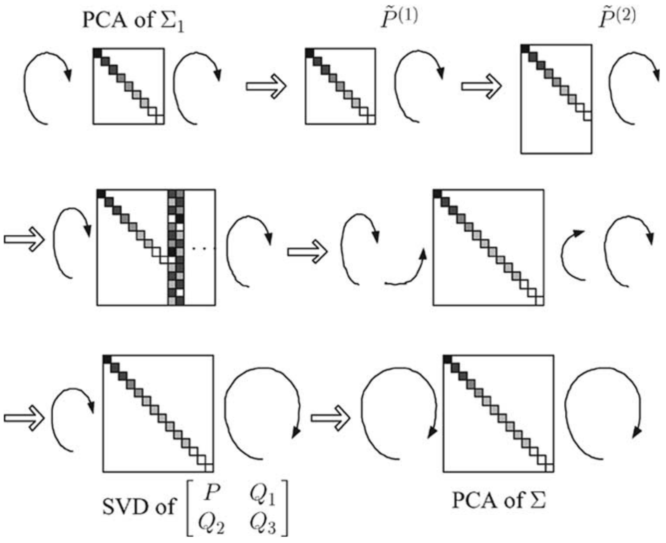

Fig. 1. Visualization of our novel IPCA algorithm. The orthogonal transformation is denoted by rotation.

Since Σ_1 is a positive semidefinite matrix, there exists an orthogonal matrix U, such that $\Sigma_1 = U\Lambda U^{\mathrm{T}}$, where $U = [u_1, u_2, \ldots, u_m]$ and $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_m)$, and $\lambda_1, \lambda_2, \ldots, \lambda_m$ being the eigenvalues of Σ_1. Suppose $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_l > 0$ and $\lambda_{l+1} = \lambda_{l+2} = \cdots = \lambda_m = 0$. Let $\tilde{\Lambda} = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_l)$, $\tilde{U} = [u_1, u_2, \ldots, u_l]$, and $P_{l\times m} = \tilde{\Lambda}^{1/2}[u_1, u_2, \ldots, u_l]^{\mathrm{T}}$. Then, $\Sigma_1 = P^{\mathrm{T}}P$.

Q_1, Q_2, and Q_3 are derived as follows (please refer to the Appendix for a detailed derivation)

$$(Q_1)_{l\times r} = \tilde{\Lambda}^{-\frac{1}{2}}\tilde{U}^{\mathrm{T}}\Sigma_2 \tag{3}$$

$$Q_2 = 0 \tag{4}$$

$$Q_3^{\mathrm{T}}Q_3 = \Sigma_3 - \Sigma_2^{\mathrm{T}}\tilde{U}\tilde{\Lambda}^{-1}\tilde{U}^{\mathrm{T}}\Sigma_2. \tag{5}$$

Since $P_{l\times m} = (\mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_l))^{1/2}[u_1, u_2, \ldots, u_l]^{\mathrm{T}}$, we have

$$\tilde{P}^{(1)} = \begin{bmatrix} I_k \\ 0 \end{bmatrix}_{l\times k} \Lambda_k V_k^{\mathrm{T}}$$

where $I_k \in \mathbb{R}^{k\times k}$, $I_k = \mathrm{diag}(1, 1, \ldots, 1)$, $\Lambda_k = (\mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_k))^{1/2}$, and $V_k = [u_1, u_2, \ldots, u_k]^{\mathrm{T}}$.

Because $Q_2 = 0$, thus

$$\tilde{P}^{(2)} = \begin{bmatrix} \tilde{P}^{(1)} \\ Q_2 \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} I_k \\ 0 \end{bmatrix} \Lambda_k V_k^{\mathrm{T}} \\ 0 \end{bmatrix}_{(l+r)\times m} = \begin{bmatrix} I_k \\ 0 \end{bmatrix}_{(l+r)\times k} \Lambda_k V_k^{\mathrm{T}}$$

which is of rank k already. Hence, the best rank-k approximation $\tilde{P}^{(2)}$ of $\begin{bmatrix} \tilde{P}^{(1)} \\ 0 \end{bmatrix}$ is simply $\begin{bmatrix} \tilde{P}^{(1)} \\ 0 \end{bmatrix}$ itself.

Based on these derivations, the proposed SVDU-IPCA algorithm is shown in Algorithm 2. Notice that although, conceptually, the incremental procedure involves the addition of rows, $Q_2 = 0$ means that no extra computation cost is required. Thus, only one round of approximation based on column addition is required. In the third and fourth steps of the incremental procedure, the computation cost of the matrices $\begin{bmatrix} 0_{k\times k} & \\ & I \end{bmatrix} \begin{bmatrix} Q_1 \\ Q_3 \end{bmatrix}$ and $[I_k\ 0] \begin{bmatrix} Q_1 \\ Q_3 \end{bmatrix}$ is quite low. Hence, the proposed incremental algorithm is computationally efficient. In the next section, we will show that the approximated error is also bounded.

Algorithm 2—Proposed SVDU-IPCA Algorithm: Given the original data $X_1 = [x_1, x_2, \ldots, x_m]$ and the rank-k eigendecomposition of $\Sigma_1 = P^{\mathrm{T}}P$, for the newly added data $[x_{m+1}, x_{m+2}, \ldots, x_{m+r}]$, do the following.

1) Compute the matrix Σ_2 and Σ_3 according to (2).
2) Obtain the best rank-k approximation of $P_{l\times m}$, which is
$$\tilde{P}^{(1)} = \begin{bmatrix} I_k \\ 0 \end{bmatrix}_{l\times k} \Lambda_k V_k^{\mathrm{T}}.$$
3) Compute Q_1 according to (3) and Q_3 as the square root of $\Sigma_3 - Q_1^{\mathrm{T}} Q_1$ in (5).
4) Obtain the QR decomposition $(I_{(l+r)\times(l+r)} - \left[\begin{smallmatrix} I_k \\ 0 \end{smallmatrix}\right][I_k\ 0])\left[\begin{smallmatrix} Q_1 \\ Q_3 \end{smallmatrix}\right] = JK$, i.e.,
$$\begin{bmatrix} 0_{k\times k} & \\ & I \end{bmatrix}_{(l+r)\times(l+r)} \begin{bmatrix} Q_1 \\ Q_3 \end{bmatrix} = JK.$$
5) Obtain the SVD of the smaller matrix
$$\begin{bmatrix} \Lambda_k & [I_k\ 0]\begin{bmatrix} Q_1 \\ Q_3 \end{bmatrix} \\ 0 & K \end{bmatrix} = \hat{U}\hat{\Lambda}\hat{V}^{\mathrm{T}}.$$
6) Obtain the best rank-k approximation of $\begin{bmatrix} P & Q_1 \\ Q_2 & Q_3 \end{bmatrix}$, which is
$$\begin{bmatrix} P & Q_1 \\ Q_2 & Q_3 \end{bmatrix} = \left([I_k\ J]\hat{U}\right) \hat{\Lambda} \left(\begin{bmatrix} V_k & 0 \\ 0 & I \end{bmatrix} \hat{V}\right)^{\mathrm{T}}.$$
7) Obtain the best rank-k approximation of
$$\Sigma = \begin{bmatrix} P & Q_1 \\ Q_2 & Q_3 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} P & Q_1 \\ Q_2 & Q_3 \end{bmatrix}.$$

IV. Analysis of the Proposed SVDU-IPCA Algorithm

This section presents a mathematical analysis of the proposed SVDU-IPCA algorithm. We prove that our incremental algorithm gives the same result as batch-mode PCA if the matrix has the low-rank-plus-shift structure. We have also presented an error-bound analysis if the low-rank-plus-shift structure is not satisfied. Finally, a computational-complexity analysis is also discussed.

Before going into detailed analysis, let us define some symbols as follows. $\hat{\Sigma}_1 = (\tilde{P}^{(1)})^{\mathrm{T}} \tilde{P}^{(1)}$ and

$$\hat{\Sigma} = \begin{bmatrix} \hat{\Sigma}_1 & \Sigma_2 \\ \Sigma_2^{\mathrm{T}} & \Sigma_3 \end{bmatrix}$$
$$\tilde{\Sigma}_1 = \Sigma_1 - \hat{\Sigma}_1$$
$$\tilde{\Sigma} = \Sigma - \hat{\Sigma}.$$

A. Use of $\hat{\Sigma}$

The eigendecomposition of $\hat{\Sigma}$ is as follows

$$\hat{\Sigma} = W \mathrm{diag}(\hat{\lambda}_1, \ldots, \hat{\lambda}_n) W^{\mathrm{T}} \tag{6}$$

where $W = [w_1, w_2, \ldots, w_n]$ is orthogonal, and the eigenvalues $\{\hat{\lambda}_k\}$ are arranged in nonincreasing order.

In the following, we show that $\mathrm{best}_k(\Sigma) = \mathrm{best}_k(\hat{\Sigma})$, if Σ has the low-rank-plus-shift structure.

Theorem 1: Assume that $\Sigma = \Omega + \sigma^2 I$, $\sigma > 0$, where Ω is symmetric and positive semidefinite with $\mathrm{rank}(\Omega) = k$ $(k \le m)$. Then, we have

$$\mathrm{best}_k(\Sigma) = \mathrm{best}_k(\hat{\Sigma}).$$

Proof: Let $\Sigma_1 = U\Gamma U^{\mathrm{T}} = U \mathrm{diag}(\gamma_1, \ldots, \gamma_m) U^{\mathrm{T}}$, where $\gamma_1 \ge \gamma_2 \ge \cdots \ge \gamma_m$. Then

$$\begin{aligned} \Sigma_1 &= \hat{\Sigma}_1 + \tilde{\Sigma}_1 \\ &= U\mathrm{diag}(\gamma_1, \ldots, \gamma_k, 0, \ldots, 0)U^{\mathrm{T}} \\ &\quad + U\mathrm{diag}(0, \ldots, 0, \gamma_{k+1}, \ldots, \gamma_m)U^{\mathrm{T}}. \end{aligned}$$

Let $\Omega = \begin{bmatrix} \Omega_1 & \Omega_2 \\ \Omega_2^{\mathrm{T}} & \Omega_3 \end{bmatrix}$, then we have $\Sigma_1 = \Omega_1 + \sigma^2 I_m$, i.e., $\Omega_1 = \Sigma_1 - \sigma^2 I_m$. Consider

$$\begin{aligned} U^{\mathrm{T}}\Omega_1 U &= U^{\mathrm{T}}\Sigma_1 U - \sigma^2 I_m \\ &= \Gamma - \sigma^2 I_m \\ &= \mathrm{diag}(\gamma_1 - \sigma^2, \ldots, \gamma_k - \sigma^2, \\ &\qquad \gamma_{k+1} - \sigma^2, \ldots, \gamma_m - \sigma^2). \end{aligned}$$

Since $\mathrm{rank}(\Omega_1) \le \mathrm{rank}(\Omega) \le k$, then $U^{\mathrm{T}}\Omega_1 U = \mathrm{diag}(\gamma_1 - \sigma^2, \ldots, \gamma_k - \sigma^2, 0, \ldots, 0)$, i.e.,

$$U^{\mathrm{T}}\Sigma_1 U = \mathrm{diag}(\gamma_1, \gamma_2, \ldots, \gamma_k, \sigma^2, \ldots, \sigma^2).$$

Consider

$$\Omega = \Sigma - \sigma^2 I = \begin{bmatrix} \Sigma_1 - \sigma^2 I_m & \Sigma_2 \\ \Sigma_2^{\mathrm{T}} & \Sigma_3 - \sigma^2 I_{n-m} \end{bmatrix}$$

we have

$$\begin{aligned} \Omega &= \begin{bmatrix} U\mathrm{diag}(\gamma_1 - \sigma^2, \ldots, \gamma_k - \sigma^2, \\ \quad 0, \ldots, 0)U^{\mathrm{T}} & \Sigma_2 \\ \Sigma_2^{\mathrm{T}} & \Sigma_3 - \sigma^2 I_{n-m} \end{bmatrix} \\ &= \begin{bmatrix} \hat{\Sigma}_1 & \Sigma_2 \\ \Sigma_2^{\mathrm{T}} & \Sigma_3 \end{bmatrix} - \begin{bmatrix} \sigma^2 I_k & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & \sigma^2 I_{n-m} \end{bmatrix}. \end{aligned}$$

Hence

$$\begin{aligned} W^{\mathrm{T}}\Omega W &= W^{\mathrm{T}}\Sigma W - \sigma^2 I \\ &= W^{\mathrm{T}}\hat{\Sigma} W - \begin{bmatrix} \sigma^2 I_k & & \\ & 0 & \\ & & \sigma^2 I_{n-m} \end{bmatrix} \\ &= \mathrm{diag}(\hat{\lambda}_1, \ldots, \hat{\lambda}_n) - \begin{bmatrix} \sigma^2 I_k & & \\ & 0 & \\ & & \sigma^2 I_{n-m} \end{bmatrix}. \end{aligned}$$

It follows that

$$W^{\mathrm{T}}\Sigma W = W^{\mathrm{T}}\hat{\Sigma} W + \begin{bmatrix} \sigma^2 I_m & 0 \\ 0 & 0 \end{bmatrix}.$$

Since $\{\hat{\lambda}_k\}$'s are arranged in nonincreasing order, we can conclude that

$$\mathrm{best}_k(\Sigma) = \mathrm{best}_k(\hat{\Sigma})$$

completing the proof. ■

In Theorem 1, we consider the case where Σ satisfies the low-rank-plus-shift property. Our second objective is to see the difference between eigenvectors of Σ and eigenvectors of $\hat{\Sigma}$, which indicates the approximated errors. Details are discussed as follows.

Theorem 2: From (6), let $\Sigma_1 = U\Gamma U^{\mathrm{T}} = U\mathrm{diag}(\gamma_1, \ldots, \gamma_m)U^{\mathrm{T}}$, where $\gamma_1 \geq \gamma_2 \geq \cdots \geq \gamma_m$, $\varepsilon = \gamma_{k+1}$, and

$$B = \begin{bmatrix} U & 0 \\ 0 & I_{n-m} \end{bmatrix} \times \mathrm{diag}\left(\underbrace{0, \ldots, 0}_{k}, 1, \frac{\gamma_{k+2}}{\varepsilon}, \ldots, \frac{\gamma_m}{\varepsilon}, 0, \ldots, 0\right) \begin{bmatrix} U & 0 \\ 0 & I_{n-m} \end{bmatrix}^{\mathrm{T}}.$$

Since

$$\begin{aligned} \Sigma_1 &= \hat{\Sigma}_1 + \tilde{\Sigma}_1 \\ &= U\mathrm{diag}(\gamma_1, \ldots, \gamma_k, 0, \ldots, 0)U^{\mathrm{T}} \\ &\quad + U\mathrm{diag}(0, \ldots, 0, \gamma_{k+1}, \ldots, \gamma_m)U^{\mathrm{T}} \\ \Sigma &= \begin{bmatrix} \Sigma_1 & \Sigma_2 \\ \Sigma_2^{\mathrm{T}} & \Sigma_3 \end{bmatrix} \\ &= \begin{bmatrix} \hat{\Sigma}_1 & \Sigma_2 \\ \Sigma_2^{\mathrm{T}} & \Sigma_3 \end{bmatrix} + \begin{bmatrix} \tilde{\Sigma}_1 & 0 \\ 0 & 0 \end{bmatrix} \\ &= \hat{\Sigma} + \tilde{\Sigma}. \end{aligned}$$

Assume that $\beta_{ji} = w_j^{\mathrm{T}} B w_i$. Then, for $\hat{\lambda}_i > 0$ and $\hat{\lambda}_i \neq \hat{\lambda}_j$, $(i \neq j)$, we have

$$\|v_i - w_i\| \leq 2\varepsilon \left(\sum_{j \neq i}^{n} \left| \frac{\beta_{ji}}{(\hat{\lambda}_i - \hat{\lambda}_j)} \right| + \frac{1}{2} \right) + O(\varepsilon^2)$$

where $V = [v_1, v_2, \ldots, v_n]$ is orthogonal and $V^{\mathrm{T}}\Sigma V = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$.

Proof: Since $\tilde{\Sigma}_1 = U\mathrm{diag}(0, \ldots, 0, \gamma_{k+1}, \ldots, \gamma_m)U^{\mathrm{T}}$, then

$$\tilde{\Sigma} = \varepsilon \begin{bmatrix} U & 0 \\ 0 & I_{n-m} \end{bmatrix} \times \mathrm{diag}\left(0, \ldots, 0, 1, \frac{\gamma_{k+2}}{\varepsilon}, \ldots, \frac{\gamma_m}{\varepsilon}, 0, \ldots, 0\right) \begin{bmatrix} U & 0 \\ 0 & I_{n-m} \end{bmatrix}^{\mathrm{T}}$$

we have $\tilde{\Sigma} = \varepsilon B$ and $\|B\| \leq 1$. For a fixed i, consider $\Sigma = \hat{\Sigma} + \varepsilon B$ with eigenvector $v_i = w_i(\varepsilon)$, and $\lambda_i = \hat{\lambda}_i(\varepsilon)$. Since $\hat{\Sigma}$ is symmetric, λ_i can be expressed as a power series in ε [29]

$$\lambda_i = \hat{\lambda}_i(\varepsilon) = \hat{\lambda}_i + k_1\varepsilon + k_2\varepsilon^2 + \cdots. \tag{7}$$

Similarly

$$v_i = w_i(\varepsilon) = w_i + \varepsilon z_1 + \varepsilon^2 z_2 + \cdots.$$

Since $\hat{\Sigma}$ is symmetric, we can express the vectors z_i's as linear combinations of the eigenvectors of $\hat{\Sigma}$ [29]. So

$$\begin{aligned} v_i &= w_i(\varepsilon) \\ &= w_i + \varepsilon \Sigma_{j=1}^{n} t_{j1} w_j + \varepsilon^2 \Sigma_{j=1}^{n} t_{j2} w_j + \cdots \\ &= (1 + \varepsilon t_{11} + \varepsilon^2 t_{12} + \cdots) w_1 + (\varepsilon t_{21} + \varepsilon^2 t_{22} + \cdots) \\ &\quad + \cdots + (\varepsilon t_{n1} + \varepsilon^2 t_{n2} + \cdots) w_n. \end{aligned} \tag{8}$$

Since this is an eigenvector, and normalizing (8) such that the coefficient of w_i is equal to 1, then

$$\begin{aligned} \tilde{v}_i = w_i &+ \left(\sum_{j=1}^{\infty} \varepsilon^j \tilde{t}_{1j}\right) w_1 + \cdots + \left(\sum_{j=1}^{\infty} \varepsilon^j \tilde{t}_{(i-1)j}\right) w_{i-1} + \cdots \\ &+ \left(\sum_{j=1}^{\infty} \varepsilon^j \tilde{t}_{(i+1)j}\right) w_{i+1} + \cdots + \left(\sum_{j=1}^{\infty} \varepsilon^j \tilde{t}_{nj}\right) w_n. \end{aligned} \tag{9}$$

Consider

$$(\hat{\Sigma} + \varepsilon B)\tilde{v}_i = \Sigma \tilde{v}_i = \lambda_i \tilde{v}_i = \hat{\lambda}_i(\varepsilon)\tilde{v}_i \quad \hat{\Sigma} w_i = \hat{\lambda}_i w_i. \tag{10}$$

Substituting (7) and (9) into (10), and considering the terms of order ε, then

$$\begin{aligned} &\varepsilon\hat{\Sigma}\left(\tilde{t}_{11}w_1 + \cdots + \tilde{t}_{(i-1)1}w_{i-1} \right. \\ &\quad \left. + \tilde{t}_{(i+1)1}w_{i+1} + \cdots + \tilde{t}_{n1}w_n\right) + \varepsilon B w_i \\ &\quad = \varepsilon\left(k_1 w_i + \hat{\lambda}_i \tilde{t}_{11} w_1 + \cdots + \hat{\lambda}_i \tilde{t}_{(i-1)1} w_{i-1} \right. \\ &\qquad \left. + \hat{\lambda}_i \tilde{t}_{(i+1)1} w_{i+1} + \cdots + \hat{\lambda}_i \tilde{t}_{n1} w_n\right). \end{aligned}$$

In other words

$$\begin{aligned} &\hat{\Sigma}\left(\sum_{j \neq i}^{n} \tilde{t}_{j1} w_j\right) + B w_i = \hat{\lambda}_i \left(\sum_{j \neq i}^{n} \tilde{t}_{j1} w_j\right) + k_1 w_i \\ &\sum_{j \neq i}^{n} \tilde{t}_{j1} \hat{\Sigma} w_j - \hat{\lambda}_i \left(\sum_{j \neq i}^{n} \tilde{t}_{j1} w_j\right) + B w_i = k_1 w_i \\ &\sum_{j \neq i}^{n} \tilde{t}_{j1} \left(\hat{\lambda}_j - \hat{\lambda}_i\right) w_j + B w_i = k_1 w_i. \end{aligned} \tag{11}$$

Since $w_l^{\mathrm{T}} w_l = 1$, and $w_l^{\mathrm{T}} w_i = 0 \quad (l \neq i)$, multiplying w_l on the left of (11), then we have

$$(\hat{\lambda}_l - \hat{\lambda}_i)\tilde{t}_{l1} w_l^{\mathrm{T}} w_l + w_l^{\mathrm{T}} B w_i = 0$$

for $l \neq i$. Therefore, $|\tilde{t}_{l1}| = |(\beta_{li}/(\hat{\lambda}_i - \hat{\lambda}_l))|$.

Recalling (9), we have

$$\|\tilde{v}_i - w_i\| \leq \varepsilon \sum_{j\neq i}^{n} \left| \frac{\beta_{ji}}{(\hat{\lambda}_i - \hat{\lambda}_j)} \right| + O(\varepsilon^2). \tag{12}$$

Therefore

$$\|\tilde{v}_i\| \leq 1 + \varepsilon \sum_{j\neq i}^{n} \left| \frac{\beta_{ji}}{(\hat{\lambda}_i - \hat{\lambda}_j)} \right| + O(\varepsilon^2).$$

Let $\|\tilde{v}_i\| = 1 + \varepsilon s + O(\varepsilon^2)$. If ε is small enough, then we can get

$$|s| \leq \sum_{j\neq i}^{n} \left| \frac{\beta_{ji}}{(\hat{\lambda}_i - \hat{\lambda}_j)} \right| + 1.$$

Thus

$$\tilde{v}_i = v_i \left(1 + \varepsilon s + O(\varepsilon^2)\right). \tag{13}$$

Substituting (13) in (12), we have

$$\left\|v_i \left(1 + \varepsilon s + O(\varepsilon^2)\right) - w_i\right\| \leq \varepsilon \sum_{j\neq i}^{n} \left| \frac{\beta_{ji}}{(\hat{\lambda}_i - \hat{\lambda}_j)} \right| + O(\varepsilon^2).$$

Since $\|v_i\| = 1$, then

$$\begin{aligned} \|v_i - w_i\| - \varepsilon|s| + O(\varepsilon^2) &\leq \varepsilon \sum_{j\neq i}^{n} \left| \frac{\beta_{ji}}{(\hat{\lambda}_i - \hat{\lambda}_j)} \right| + O(\varepsilon^2) \\ \|v_i - w_i\| &\leq \varepsilon \sum_{j\neq i}^{n} \left| \frac{\beta_{ji}}{(\hat{\lambda}_i - \hat{\lambda}_j)} \right| + \varepsilon|s| + O(\varepsilon^2). \end{aligned}$$

Considering $|s| \leq \sum_{j\neq i}^{n} |(\beta_{ji}/(\hat{\lambda}_i - \hat{\lambda}_j))| + 1$, we have

$$\|v_i - w_i\| \leq 2\varepsilon \left(\sum_{j\neq i}^{n} \left| \frac{\beta_{ji}}{(\hat{\lambda}_i - \hat{\lambda}_j)} \right| + \frac{1}{2} \right) + O(\varepsilon^2)$$

for $\hat{\lambda}_i > 0$ and $\hat{\lambda}_i \neq \hat{\lambda}_j$, $(i \neq j)$.

Thus, completing the proof. ■

Corollary: Let $\|\cdot\|_{\mathrm{F}}$ denote the Frobenius norm, then

$$\|V - W\|_{\mathrm{F}} \leq 2\sqrt{n}\varepsilon \left(\max_{1\leq i\leq n} \sum_{j\neq i}^{n} \left| \frac{\beta_{ji}}{(\hat{\lambda}_i - \hat{\lambda}_j)} \right| + \frac{1}{2} \right) + O(\varepsilon^2).$$

Proof: The proof is straightforward and therefore omitted. ■

According to Corollary, when $\varepsilon \to 0$, $W \to V$. It means that when $\tilde{\Sigma}_1$ is closely approximated with Σ_1, the error between the PCs of the batch method and those of SVDU-IPCA will be very small.

B. Computational Complexity of the SVDU-IPCA Algorithm

Considering the first step of the proposed SVDU-IPCA algorithm in Algorithm 2, the matrix Σ_1 is seldom positive definite in face recognition. If we make $\tilde{P}^{(1)} = P$, then $\mathrm{rank}(\tilde{P}^{(1)}) = l \ (l < m)$, and no information will be lost in this step. Recall that $Q_2 = 0$, thus $\begin{bmatrix} \tilde{P}^{(1)} \\ Q_2 \end{bmatrix} = \begin{bmatrix} \tilde{P}^{(1)} \\ 0 \end{bmatrix}$, which is of rank k already. The best rank-k approximation $\tilde{P}^{(2)}$ of $\begin{bmatrix} \tilde{P}^{(1)} \\ Q_2 \end{bmatrix}$ is simply $\begin{bmatrix} \tilde{P}^{(1)} \\ 0 \end{bmatrix}$ itself. Then, no computation cost is required and no information is lost. Hence, only one round of approximation based on column addition is required. The SVDU-IPCA algorithm requires $O(r^3)$ flops in the second step and $O((l + r - k)r^2)$ flops for the QR decomposition in the third step. Then, $O((k + r)^2 k)$ flops are needed in the fourth step. Since $Q_2 = 0$, $[\tilde{P}^{(2)} \left[\begin{smallmatrix} Q_1 \\ Q_3 \end{smallmatrix}\right]]$ is a sparse matrix. Hence, the computation time of the SVDU-IPCA can be further reduced. Consider $X = [x_1, x_2, \ldots, x_m]$, where $\{x_i\}$ denotes a sequence of face-image vectors by row concatenation of the two-dimensional images. If we perform the incremental procedure (SVD updating) directly on the matrix X, it needs at least $O(n^2 k)$ flops [22]. In face-recognition applications, very often, only two to six images are available for training while the image dimension is higher than 10 000. That is to say, $\{l, r, k\} \ll n$. Hence, the computational complexity will be much higher than the proposed SVDU-IPCA. Consider that we perform the incremental procedure (SVD updating) directly on the matrix Σ. Since $\Sigma_2 \neq 0$, we need obtain two best rank-k approximations of $\tilde{\Sigma}^{(2)} = \begin{bmatrix} \tilde{\Sigma}^{(1)} \\ \Sigma_2^{\mathrm{T}} \end{bmatrix}$ and $\left[\tilde{\Sigma}^{(2)} \begin{bmatrix} \Sigma_2 \\ \Sigma_3 \end{bmatrix}\right]$, respectively. Then, two rounds of approximation will be required.

C. Remarks

In this paper, our method assumes that the Σ_1 submatrix is not changed when the matrix is expanded. This implies that the mean of the training samples is assumed to be constant, which may not hold especially when a large number of samples are added. In such cases, we need to perform noncentered PCA [30], which does not require centering (Σ is then the outer product matrix). Noncentered PCA is also a well-established technique in ecology, chemistry, geology [30], and pattern recognition [31], especially face recognition [32]. Noncentered PCA [30] uses the autocorrelation matrix, instead of the covariance matrix, to get the PCs.

Empirical results in Section VI also show that our incremental method under large addition of face images outperforms batch-mode noncentered PCA and centered PCA.

V. Incremental KPCA (IKPCA)

In this section, we will first briefly review KPCA, and then report our proposed IKPCA algorithm.

A. KPCA

This section gives a short review on KPCA [9]–[11]. Given a nonlinear mapping Φ, the input data space $\mathbb{R}^n$ can be mapped into the feature space $\mathbb{F}$:

$$\begin{aligned}\Phi:\quad & \mathbb{R}^n \to \mathbb{F}\\ & x \mapsto \Phi(x).\end{aligned}$$

Correspondingly, a pattern in the original input space $\mathbb{R}^n$ is mapped into a potentially much higher dimensional feature vector in the feature space $\mathbb{F}$.

An initial motivation of KPCA is to perform PCA in the feature space $\mathbb{F}$. However, it is difficult to do so directly because computation of dot product in a high-dimensional feature space is expensive. Fortunately, the algorithm can be implemented in the input space by virtue of the kernel trick. The explicit mapping process is not required at all.

PCA performed in the feature space $\mathbb{F}$ can be formulated as the diagonalization of the covariance matrix

$$\hat{C} = \frac{1}{n}\sum_{i=1}^{n}\Phi(x_i)\Phi(x_i)^{\mathrm{T}}$$

where $\{x_1, x_2, \ldots, x_n\}$ are the given training samples in the input space $\mathbb{R}^n$. For simplicity, we assume that the mapped data need not be centered.[2] We can find the eigenvalues and eigenvectors of $\hat{C}$ via solving the eigenvalue problem

$$n\lambda\alpha = K\alpha$$

where K is a symmetric $(n \times n)$ Gram matrix with the elements

$$K_{ij} = (\Phi(x_i), \Phi(x_j)) := K(x_i, x_j).$$

Consider the eigendecomposition $K = G\Lambda G^{\mathrm{T}}$, where $G = [\alpha_1, \alpha_2, \ldots, \alpha_n]$, with $\alpha_i = [\alpha_{i1}, \alpha_{i2}, \ldots, \alpha_{in}]^{\mathrm{T}}$, is orthogonal and $\Lambda = \mathrm{diag}(\lambda_1, \lambda_2, \ldots, \lambda_n)$. Denote the projection of the Φ image of a pattern x unto the kth component by φ_k. Finally, we have

$$\varphi_k = \frac{1}{\sqrt{\lambda_k}}\sum_{i=1}^{n}\alpha_{ki}K(x_i, x). \tag{14}$$

The existence of such a kernel function is guaranteed by Mercer's Theorem [9]. The Gaussian kernel [or radial basis function (RBF) kernel] $K(x, y) = \exp(-\|x - y\|/\sigma^2)$ has been widely studied in the literature and is also used in this paper.

B. IKPCA

The extension of existing incremental methods to KPCA is not straightforward. Most, if not all, incremental algorithms, such as those in [13], [14], [16], and [19], are iterative methods that need to compute the samples in the feature space directly (not in terms of inner products). They are difficult to adopt to the kernel trick. To deal with the high computational complexity, the sampling [27] and greedy methods [28] have been proposed to update KPCA. These algorithms can speed up KPCA very well when the Gram matrix is sparse. To the best of our knowledge, no incremental method for KPCA is available.

From the theory of KPCA, using the kernel trick to deal with the nonlinear mapping, the key issue is an eigenvalue problem. Our new incremental method is easy to adopt to the kernel trick and extend to the kernel version.

Consider the eigen equation

$$n\lambda\alpha = K\alpha.$$

Let K_1 be an $m \times m$ Gram matrix and

$$K = \begin{bmatrix} K_1 & K_2 \\ K_2^{\mathrm{T}} & K_3 \end{bmatrix}$$

be an expanded $(m+r) \times (m+r)$ Gram matrix, where K_2 is of size $m \times r$ and K_3 is of size $r \times r$. From Mercer's Theorem [9], both K_1 and K are positive semidefinite matrices.

Let the eigendecomposition $K_1 = V\Lambda^2V^{\mathrm{T}} = P^{\mathrm{T}}P$, where $P = \Lambda V^{\mathrm{T}}$. Following the same derivation as in Section III, IKPCA is developed and shown in Algorithm 3.

Algorithm 3—Proposed IKPCA: Given the original data $X_1 = [x_1, x_2, \ldots, x_m]$ and the rank-k eigendecomposition of the original Gram matrix $K_1 = P^{\mathrm{T}}P$, for the newly added data $[x_{m+1}, x_{m+2}, \ldots, x_{m+r}]$, do the following.

1) Compute the matrices K_2 and K_3.
2) Obtain the best rank-k approximation of $P_{l\times m}$, which is $\tilde{P}^{(1)} = \left[\begin{smallmatrix} I_k \\ 0 \end{smallmatrix}\right]_{l\times k}\Lambda_k V_k^{\mathrm{T}}$.
3) Compute $Q_1 = \Lambda^{-1}V^{\mathrm{T}}K_2$ and Q_3 as the square root of $K_3 - Q_1^{\mathrm{T}}Q_1$.
4) Obtain the QR decomposition $(I_{(l+r)\times(l+r)} - \left[\begin{smallmatrix} I_k \\ 0 \end{smallmatrix}\right][I_k\ 0])\left[\begin{smallmatrix} Q_1 \\ Q_3 \end{smallmatrix}\right] = JL$, i.e.,
$$\begin{bmatrix} 0_{k\times k} & \\ & I \end{bmatrix}_{(l+r)\times(l+r)} \begin{bmatrix} Q_1 \\ Q_3 \end{bmatrix} = JL.$$
5) Obtain the SVD of the smaller matrix
$$\begin{bmatrix} \Lambda_k & [I_k\ 0]\begin{bmatrix} Q_1 \\ Q_3 \end{bmatrix} \\ 0 & L \end{bmatrix} = \hat{U}\hat{\Lambda}\hat{V}^{\mathrm{T}}.$$
6) Obtain the best rank-k approximation of $\begin{bmatrix} P & Q_1 \\ 0 & Q_3 \end{bmatrix}$, which is
$$\begin{bmatrix} P & Q_1 \\ 0 & Q_3 \end{bmatrix} = \left([I_k\ J]\hat{U}\right)\hat{\Lambda}\left(\begin{bmatrix} V_k & 0 \\ 0 & I \end{bmatrix}\hat{V}\right)^{\mathrm{T}}.$$
7) Obtain the best rank-k approximation of
$$K = \begin{bmatrix} P & Q_1 \\ 0 & Q_3 \end{bmatrix}^{\mathrm{T}}\begin{bmatrix} P & Q_1 \\ 0 & Q_3 \end{bmatrix}.$$
8) Obtain the nonlinear projection of KPCA from (14).

[2]This can be viewed as a Karhunen–Loeve (K–L) transformation in the feature space. Actually, all calculations can be reformulated to deal with centering [9].

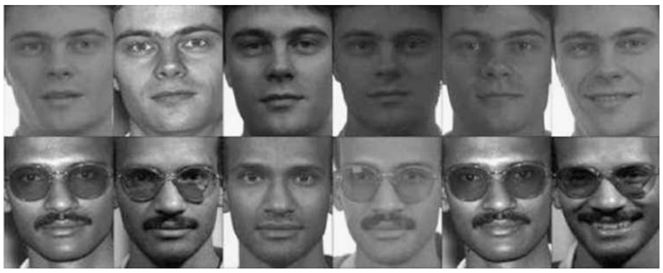
Fig. 2. Samples from the FERET dataset.

VI. Experiment Results

Three experiments will be presented in this section. In the first experiment, we evaluate and compare the performance of the proposed SVDU-IPCA with batch-mode PCA (both centered and noncentered) by eigenface and fisherface face-recognition methods using the FERET database. The second experiment is to compare the performance of SVDU-IPCA with the most recent IPCA algorithm, namely CCIPCA [13], using the AR face database. The third experiment compares the performance of the proposed IKPCA with KPCA using the Yale B database. Details of each experiment are discussed as follows.

A. Experiment 1: Performance Evaluation Using FERET Face Database

The objective of this experiment is to evaluate the performance of the proposed SVDU-IPCA algorithm by replacing the batch-mode PCA algorithm in the eigenface and fisherface face-recognition methods.

The FERET database is used for evaluation in this experiment. We selected 72 subjects from the FERET database, with six images for each subject. The six images are extracted from four different sets, namely Fa, Fb, Fc, and duplicate [2]. The images are selected to bear with more differences in lighting, facial expressions, and facial details. All images are aligned at the centers of the eyes and mouth. Histogram equalization is applied to the face images for photometric normalization, and the images also convert to the intensity images that contain values in the range 0.0 (black) to 1.0 (full intensity or white). The images from the two subjects are shown in Fig. 2.

1) Results of the Eigenface Method: Initially, we used 144 images (36 subjects, 4 per subject) to form the initial matrix Σ_1. Training images are then added in increments of 24 images (six subjects, four per subject) up to a maximum total number of 288 training images (72 subjects, 4 per subject). The remaining images are used for recognition. Minimum distance classifier is used in this experiment. The experiments are repeated 50 times and the average accuracy is recorded. Figs. 3 and 4 show the performance of the proposed IPCA method, batch-mode centered PCA, and batch-mode noncentered PCA when 30 and 50 PCs are used. The computational time, when 30 PCs are used, is recorded and shown in Table I, which indicates that our proposed IPCA algorithm is more efficient than the batch-mode PCA algorithm.

It is found that the recognition rate decreases when more training samples are added. This may be due to the fact that

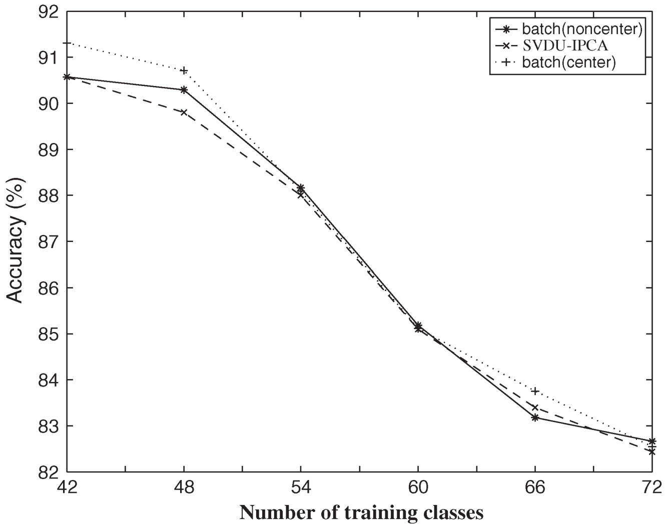

Fig. 3. Recognition accuracy using 30 PCs.

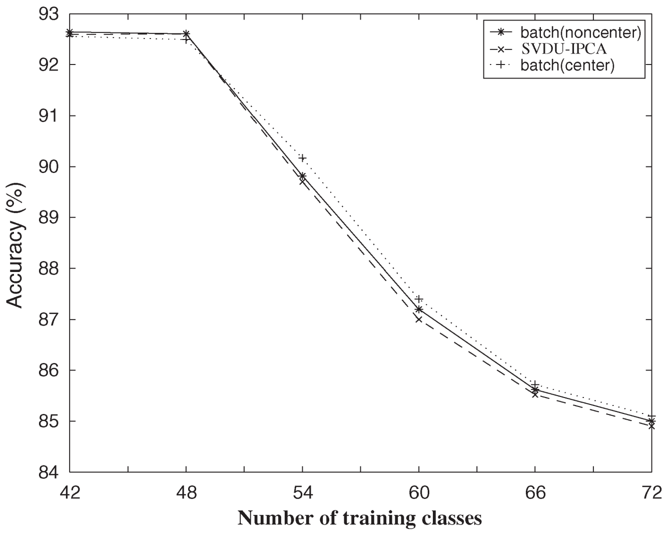

Fig. 4. Recognition accuracy using 50 PCs.

when more images are added, the (fixed) number of PCs may not be enough for the recognition tasks. For example, if 50 classes need 50 PCs to get a good result, 100 classes may need more PCs. If still 50 PCs are used, the accuracy will drop. From the experiment results, we can also find that the recognition accuracy of 50 PCs is higher than that of 30 PCs because of the same argument. Choosing the best dimension of the feature space is still an open problem and beyond the scope of this paper.

To demonstrate the benefit of using block updating, we perform an experiment using our SVDU-IPCA algorithm with increment by single and block. In the experiment, first, 144 images (36 subjects, 4 per subject) are used. Training images are then added in increments of 24 images (six subjects, four per subject) up to a maximum total number of 288 training images, and the remaining images are used for testing. To have a benchmark, experiments are performed using the eigenface method (batch-mode PCA) on the initial 144 images to get the projection matrix. This means the projection matrix does not

TABLE I
COMPARISON OF CPU TIME (s) ON FERET DATA SET (2.4-GHz PENTIUM-4 CPU WITH 512-MB RAM)

Number of Training Samples	168	192	216	240	264	288
Batch	0.062	0.1090	0.1570	0.2190	0.2970	0.4070
Incremental	0.016	0.031	0.047	0.064	0.094	0.1540

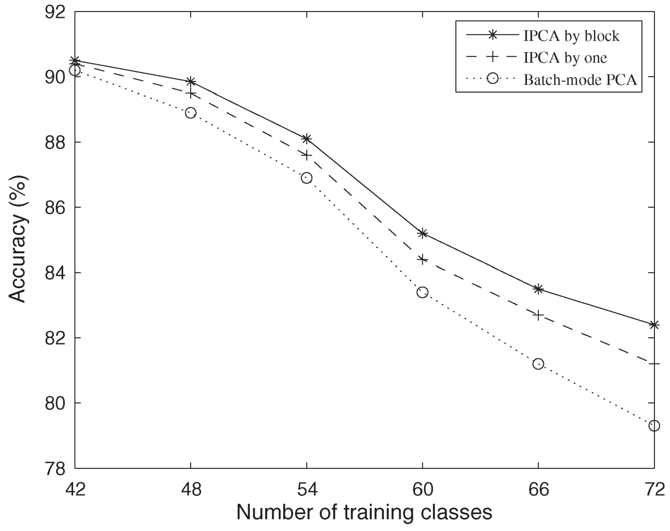

Fig. 5. Comparing the accuracy of block updating and single updating.

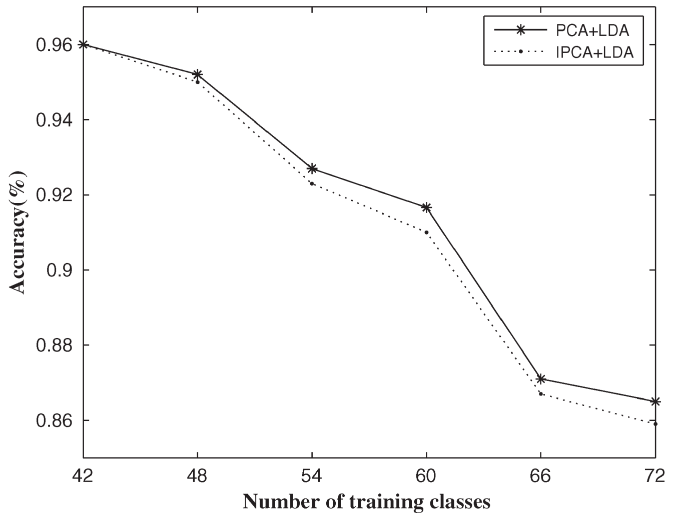

Fig. 6. Recognition accuracy of PCA+LDA and IPCA+LDA.

change when some new face images are added. Fig. 5 compares the performance when 30 PCs are used. It can be shown that incremental by block gives better performance than that by single.

2) Results of the Fisherface Method: Besides the eigenface method (PCA), the fisherface [33] method (PCA+LDA) is another well-known method in appearance-based approach.

In the fisherface method, PCA is used for dimensional reduction to ensure the full rank of the within-class scatter matrix. In doing so, the direct computation of linear discriminant analysis (LDA) [31] becomes feasible.

In this experiment, we initially use 144 images for training. Training images are then added in increments of 24 images up to a maximum total number of 288 training samples. The remaining images are used for recognition. In the PCA step, in order to avoid the singularity of the within-class scatter matrix, 100 PCs are selected, and then in the LDA step, $c-1$ features are used for face recognition, where c is the number of classes. The results are recorded and plotted in Fig. 6. It can be seen that the proposed incremental method gives a very close performance to batch-mode PCA when it is applied to fisherface.

B. Experiment 2: Comparison Between SVDU-IPCA and CCIPCA Algorithms

We have shown in Experiments 1 and 2 that our proposed SVDU-IPCA gives a very close approximation to the batch-mode PCA. In this section, we would like to show that our proposed SVDU-IPCA outperforms the existing IPCA methods. To do the comparison, we select the most recent method, namely the CCIPCA algorithm [13]. The AR database is selected [34]. Face-image variations in the AR database include illumination, facial expression, and occlusion. For most of the individuals in the AR database, images were taken in two sessions (separated by two weeks). In our experiment, 119 individuals (64 male and 55 female) who participated in both sessions were selected. We manually cropped the face portion of the images. All the cropped images are aligned at the centers of the eyes and mouth and then normalized with resolution 112×92. Due to the presence of sunglasses and scarves, the face images in AR contain a large area of occlusion. We discard these images and 952 images (119×8) are selected in our experiment. In order to get a lager sample-to-dimension ratio, which is important for the CCIPCA algorithm, wavelet transform is applied on all the images two times, and the low-frequency images are used as input vectors. After two times of wavelet transform, the resolution of the input image is 30×25. Note that, after the wavelet transform, some information of the original image is lost.

In the experiment, initially, we used 238 images (119 persons, 2 images per person). Training images are then added in increments of 119 images (119 persons, 1 image per person) up to a maximum total number of 714 images (119 persons, 6 images per person). The remaining images are used for recognition. Minimum distance classifier is used in this experiment. The experiments are repeated 50 times and the average accuracy is recorded. Experiments on batch-mode PCA are performed as a benchmark. Figs. 7 and 8 plot the performance when 50 and 100 PCs are used, respectively. It can be found that, in all the conditions that the sample-to-dimension ratio changes from 0.476 to 0.952, the SVDU-IPCA algorithm outperforms the CCIPCA algorithm.

In order to give a comprehensive comparison, we also use the measurement as suggested in [13]. Initially 238 training images are used and then adds up to a maximum total number

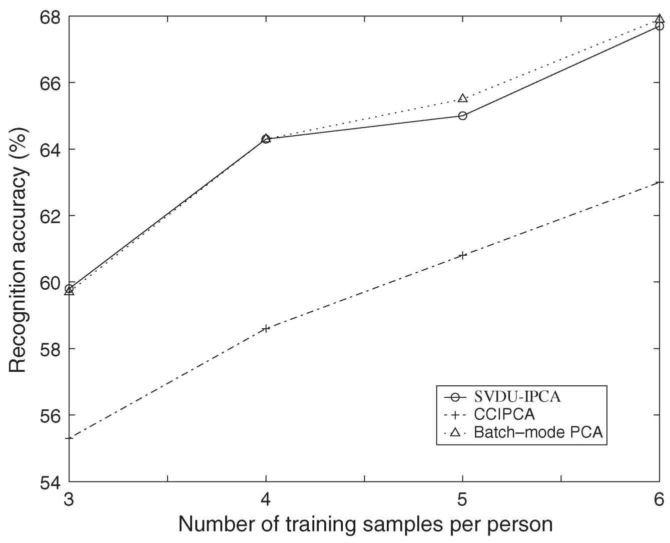

Fig. 7. Recognition accuracy using 50 PCs.

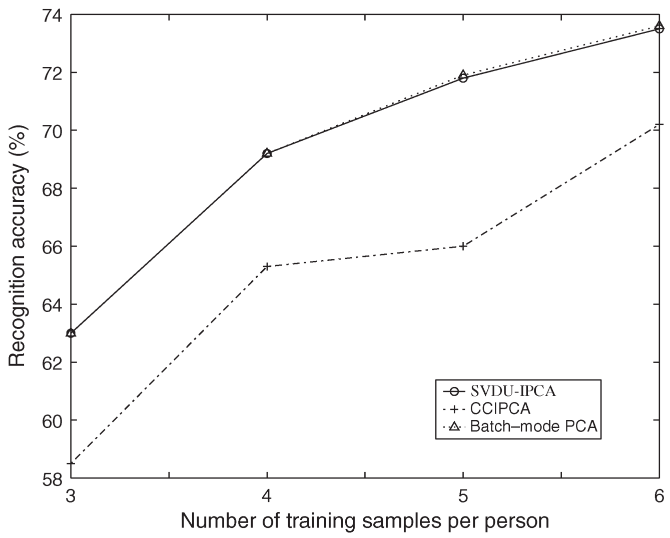

Fig. 8. Recognition accuracy using 100 PCs.

of 714 training images (119 persons, 6 images per person). Let the PCs of our SVDU-IPCA be $\hat{V} = [\hat{v}_1, \hat{v}_2, \ldots, \hat{v}_m]$, the PCs of CCIPCA be $\tilde{V} = [\tilde{v}_1, \tilde{v}_2, \ldots, \tilde{v}_m]$, and the PCs of the batch-mode PCA on the total training samples be $V = [v_1, v_2, \ldots, v_m]$. We have computed the correlation $V^{\mathrm{T}}\hat{V}$ and $V^{\mathrm{T}}\tilde{V}$. The correlation between two unit eigenvectors v and $\bar{v}$ is represented by their inner product $v\bar{v}^{\mathrm{T}}$. Figs. 9 and 10 show the correlation (dot products) of the PCs. In both figures, the upper row [Figs. 9(a), (b) and 10(a), (b)] shows the results of our SVDU-IPCA while the lower row [Figs. 9(c), (d) and 10(c), (d)] shows the results of the CCIPCA algorithm. It can be seen that the proposed SVDU-IPCA can accurately estimate the eigenvectors for both the first 20 and the last 20. In this experiment, the sample-to-dimension ratio is greater than 0.9, and CCIPCA estimates the first several PCs with high correlation with the actual ones. However, the other eigenvectors, especially the last eigenvectors, cannot be estimated accurately. The error in estimation may be due to the fact that, in CCIPCA, the computation of the $(i+1)$th PC is based on the ith PC, and the error will be propagated.

C. Experiment 3: Results on IKPCA

The objective of this experiment is to demonstrate the effectiveness of the proposed IKPCA. The Yale Group B face database is selected for evaluation because images have larger variations in pose and illumination.

The Yale Group B face database contains 5850 source images of ten subjects each captured under 585 viewing conditions (9 poses $\times$ 65 illumination conditions). In our experiments, we use images under 45 illumination conditions and these 4050 images (9 poses $\times$ 45 illumination conditions) have been divided into four subsets according to the angle the light-source direction makes with the camera axis: subset 1 (up to 25°, 7 images per pose), subset 2 (up to 12°, 12 images per pose), subset 3 (up to 50°, 12 images per pose), and subset 4 (up to 77°, 14 images per pose) [35]. All frontal-pose images are aligned by the centers of the eyes and mouth and the other images are aligned by the center points of the faces. Then, all images are normalized with the same resolution of 57 $\times$ 47. The images also convert to the intensity images that contain values in the range 0.0 (black) to 1.0 (full intensity or white). Some images from one individual are shown in Fig. 11.

For each subset, we show images of nine poses under one illumination condition. In the first row are images of nine poses under one illumination condition from subset 1. In the second row are images of nine poses under one illumination condition from subset 2, etc.

First, we fix the pose variation to evaluate the performance of IKPCA. For each pose, we randomly select 80 images (two images per subset). Training images are then added in increments of 40 images (one image per subset) up to a maximum total number of 240 images. The remaining images are used for recognition. Minimum distance classifier is used. In the experiment, 50 PCs are used in the eigendecomposition updating and testing. The experiments are repeated 50 times and the average accuracy is recorded and plotted in Fig. 12. It can be seen that the two curves are almost overlapping. This implies that our proposed IKPCA gives a close approximation to batch-mode KPCA.

Besides the recognition accuracy, we have compared the gram matrix and PC vectors from IKPCA and batch-mode KPCA. The results are recorded and shown in Tables II and III. In Table II, K and $\hat{K}$ are the Gram matrices used in the batch method and the approximate Gram matrix used in the incremental method, respectively. $\|V - W\|_{\mathrm{F}}$ is the Frobenius norm of the difference between the PCs of the batch method and those of the incremental method. Table III records the difference between the first PC of the batch method and that of the incremental method. Using Theorem 2, we can also obtain the error bound of the angle between the first PC of the batch method and that of the incremental method.

Finally, we make the training samples include pose and illumination variations. We select two images from each illumination subset and each pose. That is to say, we select 720 images (10 persons $\times$ 9 poses $\times$ 4 subsets $\times$ 2 images) for

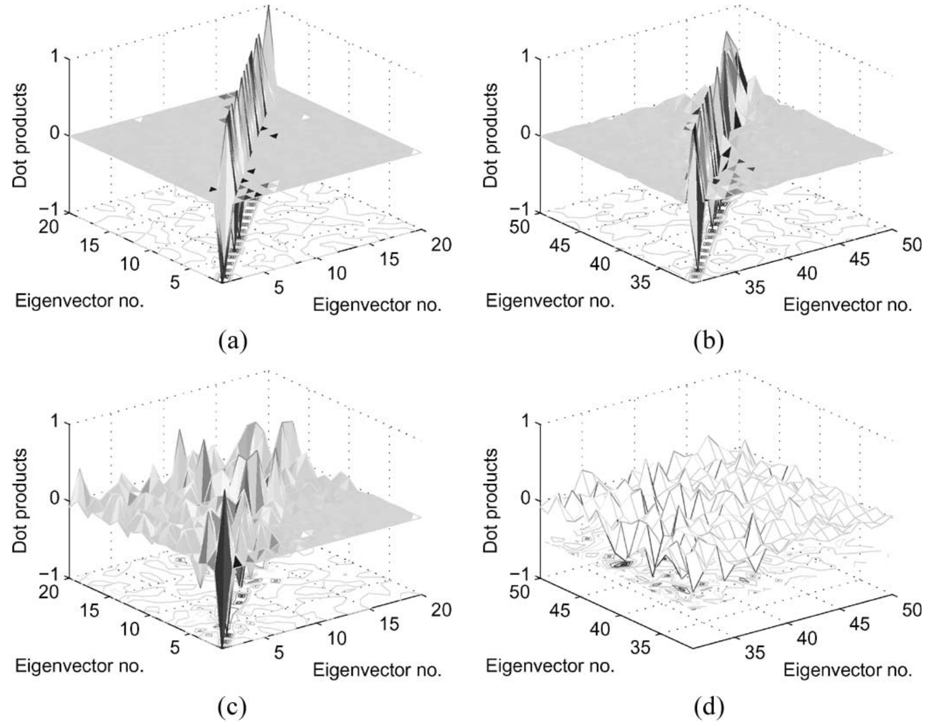

Fig. 9. Correctness, or the correlation in the experiment using 50 PCs. (a) and (c) The correlation, represented by products $V^{\mathrm{T}}\hat{V}$, of the first 20 PCs. (b) and (d) The correlation, represented by products $V^{\mathrm{T}}\hat{V}$, of the last 20 PCs. (a) and (b) show the results for SVDU-IPCA algorithm while (c) and (d) for CCIPCA.

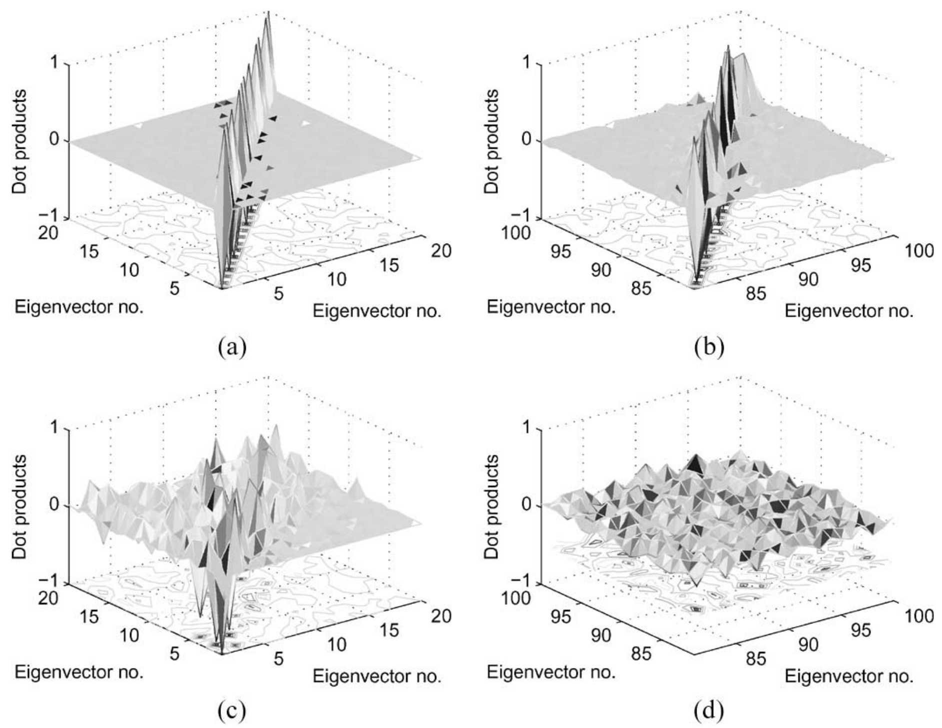

Fig. 10. Correctness, or the correlation in the experiment using 100 PCs. (a) and (c) The correlation, represented by products $V^{\mathrm{T}}\tilde{V}$, of the first 20 PCs. (b) and (d) The correlation, represented by products $V^{\mathrm{T}}\tilde{V}$, of the last 20 PCs. (a) and (b) show the results for SVDU-IPCA algorithm while (c) and (d) for CCIPCA.

training. Then, other training images are added in increments of 360 images $(10 \text{ persons} \times 9 \text{ poses} \times 4 \text{ subsets} \times 1 \text{ image})$ up to total number of 2160 images. The remaining images are used for recognition. In the experiment, 50 PCs are used in the eigendecomposition updating and testing. Experiment results, together with the computational time, are recorded and shown in Tables IV–VI. Table VI indicates that KPCA by using incremental learning is computational efficient.

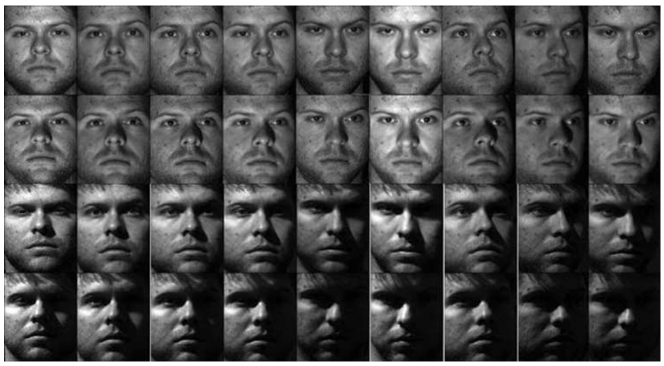

Fig. 11. Images of one person from the Yale B database.

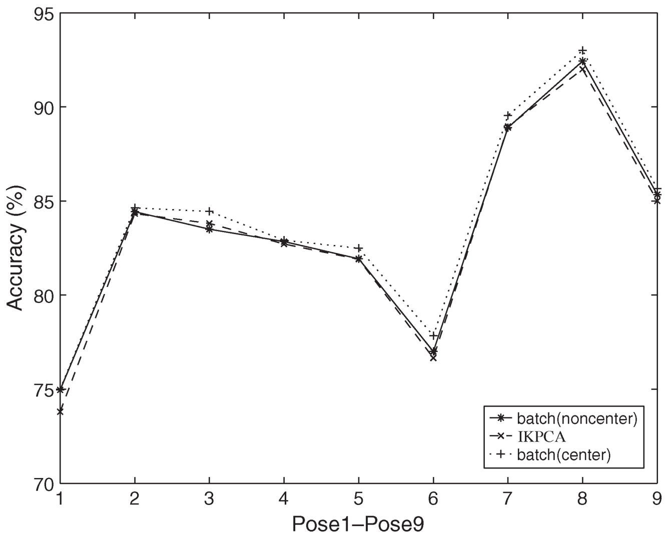

Fig. 12. Recognition accuracy on the Yale B database.

TABLE II
COMPARISON OF THE GRAM MATRIX AND PCs ON THE YALE B DATA SET WITH FIXED POSE AND VARIANT ILLUMINATION

Training Samples	120	160	200	240
$\|K - \hat{K}\|$	0.012	0.020	0.030	0.041
$\|V - W\|_F$	2.19	2.49	2.50	3.31

TABLE III
COMPARISON OF THE FIRST PC ON THE YALE B DATA SET WITH FIXED POSE AND VARIANT ILLUMINATION

Training Samples	120	160	200	240
$\|v_1 - w_1\|$	1.59e-005	2.34e-005	2.91e-005	3.05e-005
error bound of the angle	0.08°	0.27°	0.46°	0.74°

TABLE IV
COMPARISON OF THE GRAM MATRIX AND PCs ON THE YALE B DATA SET WITH VARIATIONS ON POSE AND ILLUMINATION

Training Samples	1080	1440	1800	2160
$\|K - \hat{K}\|$	0.042	0.073	0.10	0.13
$\|V - W\|_F$	2.79	3.33	3.69	3.82

The experiment on the large database, Yale Group B, shows that our new incremental algorithm not only is computationally efficient, but also has small approximated error.

TABLE V
COMPARISON OF THE FIRST PC ON THE YALE B DATA SET WITH VARIATIONS ON POSE AND ILLUMINATION

Training Samples	1080	1440	1800	2160
$\|v_1 - w_1\|$	4.80e-005	5.22e-005	6.22e-005	1.66e-004
error bound of the angle	0.73°	1.79°	3.55°	6.04°

TABLE VI
COMPARISON OF CPU TIME (s) ON THE YALE B DATA SET WITH VARIATIONS ON POSE AND ILLUMINATION (2.4-GHz PENTIUM-4 CPU WITH 512-MB RAM)

Number of Training Samples	1080	1440	1800	2160
Batch	41.52	100.23	217.14	450.5
Incremental	19.77	48.94	103.63	214.72

VII. CONCLUSION

A new IPCA, namely SVDU-IPCA, is derived and reported in this paper. The proposed SVDU-IPCA algorithm adopts the concept of an SVD updating algorithm, in which we do not need to recompute the eigendecomposition from scratch. The main contribution of this paper is that we perform an error analysis of the proposed IPCA algorithm. A complete mathematical derivation is presented, and we prove that the error to be introduced in our SVDU-IPCA algorithm is bounded. In other words, the proposed SVDU-IPCA algorithm guarantees the maximum error when approximating the batch-mode PCA. Another characteristic of the proposed SVDU-IPCA algorithm is that it can be easily extended to kernel space. An IKPCA algorithm is also presented. To the best of our knowledge, this is the first IKPCA algorithm.

Extensive experiments using available public databases have been performed in order to evaluate the performance of our proposed SVDU-IPCA algorithm. It is found that the approximation error is quite small. The proposed algorithm is also applied to face recognition. Two well-known PCA-based face-recognition methods, namely eigenface and fisherface, are used for evaluation. The experimental results show that the proposed SVDU-IPCA algorithm can be used in both methods with a very small degradation in recognition accuracy. This implies that the existing eigenface- and fisherface-based algorithms/systems can be scaled up easily by using the proposed SVDU-IPCA method as we do not need the previous training data.

APPENDIX

Since

$$\Sigma = \begin{bmatrix} \Sigma_1 & \Sigma_2 \\ \Sigma_2^{\mathrm{T}} & \Sigma_3 \end{bmatrix} = \begin{bmatrix} P^{\mathrm{T}}P & \Sigma_2 \\ \Sigma_2^{\mathrm{T}} & \Sigma_3 \end{bmatrix}$$

and

$$\Sigma = \begin{bmatrix} P & Q_1 \\ Q_2 & Q_3 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} P & Q_1 \\ Q_2 & Q_3 \end{bmatrix}$$
$$= \begin{bmatrix} \left(P^{\mathrm{T}}P + Q_2^{\mathrm{T}}Q_2\right)_{m\times m} & \left(P^{\mathrm{T}}Q_1 + Q_2^{\mathrm{T}}Q_3\right)_{m\times r} \\ \left(Q_1^{\mathrm{T}}P + Q_3^{\mathrm{T}}Q_2\right)_{r\times m} & \left(Q_1^{\mathrm{T}}Q_1 + Q_3^{\mathrm{T}}Q_3\right)_{r\times r} \end{bmatrix} \quad (15)$$

we have

$$P^{\mathrm{T}}P = P^{\mathrm{T}}P + Q_2^{\mathrm{T}}Q_2$$

that is, $Q_2^{\mathrm{T}}Q_2 = 0$.

Considering $Q_2 = 0$, (15) simplifies to

$$\begin{bmatrix} (P^{\mathrm{T}}P)_{m\times m} & (P^{\mathrm{T}}Q_1)_{m\times r} \\ \left(Q_1^{\mathrm{T}}P\right)_{r\times m} & \left(Q_1^{\mathrm{T}}Q_1 + Q_3^{\mathrm{T}}Q_3\right)_{r\times r} \end{bmatrix}.$$

On equating terms for Σ_2, we have

$$(P^{\mathrm{T}}Q_1)_{m\times r} = \Sigma_2$$

that is, $[u_1, u_2, \ldots, u_l]\tilde{\Lambda}^{1/2}Q_1 = \Sigma_2$. Hence

$$[u_1, u_2, \ldots, u_l]^{\mathrm{T}}[u_1, u_2, \ldots, u_l]\tilde{\Lambda}^{\frac{1}{2}}Q_1 = [u_1, u_2, \ldots, u_l]^{\mathrm{T}}\Sigma_2$$

and then

$$(Q_1)_{l\times r} = \tilde{\Lambda}^{-\frac{1}{2}}\tilde{U}^{\mathrm{T}}\Sigma_2.$$

Equating terms for Λ_3, we have

$$Q_3^{\mathrm{T}}Q_3 = \Sigma_3 - Q_1^{\mathrm{T}}Q_1.$$

Since $Q_1^{\mathrm{T}}Q_1 = \Sigma_2\tilde{U}\tilde{\Lambda}^{-1}\tilde{U}^{\mathrm{T}}\Sigma_2$, then

$$Q_3^{\mathrm{T}}Q_3 = \Sigma_3 - \Sigma_2^{\mathrm{T}}\tilde{U}\tilde{\Lambda}^{-1}\tilde{U}^{\mathrm{T}}\Sigma_2.$$

Consider

$$R = \begin{bmatrix} \tilde{U}\tilde{U}^{\mathrm{T}} & 0 \\ 0 & I \end{bmatrix}$$

$$S = \begin{bmatrix} I & -\tilde{U}\tilde{\Lambda}^{-1}\tilde{U}^{\mathrm{T}}\Sigma_2 \\ 0 & I \end{bmatrix}$$

where I is the identity matrix of appropriate size. We have

$$\begin{aligned} &S^{\mathrm{T}}R^{\mathrm{T}}\begin{bmatrix} \Sigma_1 & \Sigma_2 \\ \Sigma_2^{\mathrm{T}} & \Sigma_3 \end{bmatrix}RS \\ &= \begin{bmatrix} I & 0 \\ -\Sigma_2^{\mathrm{T}}\tilde{U}\tilde{\Lambda}^{-1}\tilde{U}^{\mathrm{T}} & I \end{bmatrix}\begin{bmatrix} \tilde{U}\tilde{U}^{\mathrm{T}} & 0 \\ 0 & I \end{bmatrix}\begin{bmatrix} \tilde{U}\tilde{\Lambda}\tilde{U}^{\mathrm{T}} & \Sigma_2 \\ \Sigma_2^{\mathrm{T}} & \Sigma_3 \end{bmatrix} \\ &\quad\times\begin{bmatrix} \tilde{U}\tilde{U}^{\mathrm{T}} & 0 \\ 0 & I \end{bmatrix}\begin{bmatrix} I & -\tilde{U}\tilde{\Lambda}^{-1}\tilde{U}^{\mathrm{T}}\Sigma_2 \\ 0 & I \end{bmatrix} \\ &= \begin{bmatrix} \tilde{U}\tilde{\Lambda}\tilde{U}^{\mathrm{T}} & 0 \\ 0 & \Sigma_3 - \Sigma_2^{\mathrm{T}}\tilde{U}\tilde{\Lambda}^{-1}\tilde{U}^{\mathrm{T}}\Sigma_2 \end{bmatrix}. \end{aligned}$$

As

$$\Sigma = \begin{bmatrix} \Sigma_1 & \Sigma_2 \\ \Sigma_2^{\mathrm{T}} & \Sigma_3 \end{bmatrix}$$

is positive semidefinite, by the definition of the positive semidefinite matrix, $\Sigma_3 - \Sigma_2^{\mathrm{T}}\tilde{U}\tilde{\Lambda}^{-1}\tilde{U}^{\mathrm{T}}\Sigma_2$ is also positive semidefinite. Hence, Q_3 can be obtained as the square root of $\Sigma_3 - \Sigma_2^{\mathrm{T}}\tilde{U}\tilde{\Lambda}^{-1}\tilde{U}^{\mathrm{T}}\Sigma_2$.

ACKNOWLEDGMENT

The authors would like to thank the research laboratories who contributed the FERET, AR, and Yale B databases for experiments. They would also like to thank the five anonymous reviewers for precious comments.

REFERENCES

[1] M. Turk and A. Pentland, "Eigenfaces for recognition," *J. Cognit. Neurosci.*, vol. 3, no. 1, pp. 71–86, 1991.
[2] P. J. Phillips, H. Moo, S. A. Rizvi, and P. J. Rauss, "The FERET evaluation methodology for face recognition algorithms," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 10, pp. 1090–1104, Oct. 2000.
[3] R. Chellappa, C. L. Wilson, and S. Sirohey, "Human and machine recognition of faces: A survey," *Proc. IEEE*, vol. 83, no. 5, pp. 705–740, May 1995.
[4] J. Daugman, "Face and gesture recognition: Overview," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 675–676, Jul. 1997.
[5] R. Brunelli and T. Poggio, "Face recognition: Features versus templates," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 15, no. 10, pp. 1042–1053, Oct. 1993.
[6] A. Samal and P. A. Iyengar, "Automatic recognition and analysis of human faces and facial expression: A survey," *Pattern Recognit.*, vol. 25, no. 1, pp. 65–77, Jan. 1992.
[7] M. Kirby and L. Sirovich, "Application of the Karhunen–Loeve procedure for the characterization of human faces," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 1, pp. 103–108, Jan. 1990.
[8] H. Zha and H. D. Simon, "On updating problems in latent semantic indexing," *SIAM J. Sci. Comput.*, vol. 21, no. 2, pp. 782–791, 1999.
[9] B. Schölkopf and A. Smola, *Learning With Kernels: Support Vector Machines, Regularization, Optimization and Beyond*. Cambridge, MA: MIT Press, 2002.
[10] B. Schölkopf, A. Smola, and K. Muller, "Nonlinear component analysis as a kernel eigenvalue problem," *Neural Comput.*, vol. 10, no. 5, pp. 1299–1319, Jul. 1998.
[11] M. Yang, "Kernel eigenfaces versus kernel fisherfaces: Face recognition using kernel methods," in *Proc. 5th IEEE Int. Conf. Automatic Face and Gesture Recognition (RGR)*, Washington, DC, 2002, pp. 215–220.
[12] P. M. Hall, A. D. Marshall, and R. R. Martin, "Merging and splitting eigenspace models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 9, pp. 1042–1049, Sep. 2000.
[13] J. Weng, Y. Zhang, and W. S. Hwang, "Candid covariance-free incremental principal component analysis," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 8, pp. 1034–1040, Aug. 2003.
[14] D. Skocaj and A. Leonardis, "Weighted and robust incremental method for subspace learning," in *Proc. 9th IEEE Int. Conf. Computer Vision*, Nice, France, 2003, vol. 2, pp. 1494–1500.
[15] J. T. Kwok and H. Zhao, "Incremental eigendecomposition," in *Proc. ICANN*, Istanbul, Turkey, Jun. 2003, pp. 270–273.
[16] T. D. Sanger, "Optimal unsupervised learning in a single-layer linear feedforward neural network," *Neural Netw.*, vol. 2, pp. 459–473, 1989.
[17] E. Oja and J. Karhunen, "On stochastic approximation of the eigenvectors and eigenvalues of the expectation of a random matrix," *J. Math. Anal. Appl.*, vol. 106, no. 1, pp. 69–84, 1985.
[18] Y. Li, "On incremental and robust subspace learning," *Pattern Recognit.*, vol. 37, no. 7, pp. 1509–1518, Jul. 2004.
[19] M. Artač, M. Jogan, and A. Leonardis, "Incremental PCA for on-line visual learning and recognition," in *Proc. Int. Conf. Pattern Recognition*, Quebec City, QC, Canada, 2002, vol. 3, pp. 781–784.
[20] A. Levy and M. Lindenbaum, "Sequential Karhunen–Loeve basis extraction and its application to images," *IEEE Trans. Image Process.*, vol. 9, no. 8, pp. 1371–1374, Oct. 2000.
[21] J. Bunch and C. Nielsen, "Updating the singular value decomposition," *Numer. Math.*, vol. 32, no. 2, pp. 131–152, 1978.
[22] M. Brand, "Incremental singular value decomposition of uncertain data with missing values," in *Proc. Eur. Conf. Computer Vision*, Copenhagen, Denmark, May 2002, vol. 2350, pp. 707–720.
[23] B. N. Parlett, *The Symmetric Eigenvalue Problem*. Englewood Cliffs, NJ: Prentice-Hall, 1980.
[24] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay, "Clustering in large graphs and matrices," in *Proc. ACM SODA Conf.*, Baltimore, MD, 1999, pp. 291–299.

[25] A. Frieze, R. Kannan, and S. Vempala, "Fast Monte-Carlo algorithms for finding low rank approximations," in *Proc. ACM FOCS Conf.*, Palo Alto, CA, 1998, pp. 370–378.

[26] D. Ross, J. Lim, and M. H. Yang, "Adaptive probabilistic visual tracking with incremental subspace update," in *Proc. 8th ECCV*, Prague, Czechoslovakia, May 2004, pp. 470–482.

[27] D. Achlioptas, F. Mcsherry, and B. Schölkopf, "Sampling techniques for kernel methods," in *Advances in Neural Information Processing Systems*, vol. 14. Cambridge, MA: MIT Press, 2002.

[28] A. J. Smola and B. Schölkopf, "Sparse greedy matrix approximation for machine learning," in *Proc. 17th Int. Conf. Machine Learning*, Stanford, CA, 2000, pp. 911–918.

[29] G. H. Golub, *Lectures on Matrix Computations 2004*, 2004. [Online]. Available: http://www.mat.uniroma1.it/~bertaccini/seminars/

[30] I. T. Jolliffe, *Principal Component Analysis*, 2nd ed. New York: Springer-Verlag, 2002.

[31] K. Fukunaga, *Introduction to Statistical Pattern Recognition*. New York: Academic, 1990.

[32] Z. Jin, J. Y. Yang, Z. S. Hu, and Z. Lou, "Face recognition based on the uncorrelated discriminant transform," *Pattern Recognit.*, vol. 34, no. 7, pp. 1405–1416, 2001.

[33] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces versus Fisherfaces: Recognition using class specific linear projection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 711–720, Jul. 1997.

[34] A. M. Martinez and R. Benavente, "The AR face database," Purdue Univ., West Lafayette, IN, CVC Tech. Rep. 24, Jun. 1998.

[35] A. S. Georghiades, P. N. Belhumeur, and D. J. Kriegman, "From few to many: Illumination cone models for face recognition under variable lighting and pose," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 6, pp. 643–660, Jun. 2001.

Haitao Zhao received the M.Sc. degree in applied mathematics in 2000 and the Ph.D. degree in pattern recognition and artificial intelligence in 2003, both from Nanjing University of Science and Technology, Nanjing, China.

Currently, he is an Assistant Professor in the Institute of Aerospace Science and Technology, Shanghai Jiaotong University, Shanghai, China. His major research interests include human face recognition, image processing, and data fusion.

Pong Chi Yuen (S'92–M'93) received the B.Sc. degree in electronic engineering with first class honors from City Polytechnic of Hong Kong, Hongkong, SAR, in 1989, and the Ph.D. degree in electrical and electronic engineering from The University of Hong Kong, Hongkong, SAR, in 1993.

Currently, he is a Professor in the Department of Computer Science, Hong Kong Baptist University, Hongkong, SAR. He was a recipient of the university fellowship to visit the University of Sydney in 1996. He was associated with the Department of Electrical Engineering, Laboratory of Imaging Science and Engineering. In 1998, he spent a six-month sabbatical leave in the University of Maryland Institute for Advanced Computer Studies, University of Maryland at College Park. He was also associated with the Computer Vision Laboratory, Center for Automation Research. He was the director of Croucher Advanced Study Institute on Biometric Authentication 2004 and is the track Co-Chair of the International Conference on Pattern Recognition 2006. From July 2005 to January 2006, he was a Visiting Professor in the Institute National de Recherche en Informatique et en Automatique, Rhone Alpes, France. His major research interests include human-face recognition, signature recognition, and medical image processing. He has published more than 80 scientific articles in these areas.

James T. Kwok (M'98) received the B.Sc. degree in electrical and electronic engineering from the University of Hong Kong, Hongkong, SAR, and the Ph.D. degree in computer science from the Hong Kong University of Science and Technology, Hongkong, SAR.

He was a Consultant at Lucent Technologies Bell Laboratories in Murray Hill, NJ. He then joined the Department of Computer Science, Hong Kong Baptist University, as an Assistant Professor. He returned to the Hong Kong University of Science and Technology, in 2000 and is an Assistant Professor in the Department of Computer Science. He is currently on the editorial board of *Neurocomputing*. His major research interests are kernel methods, machine learning, pattern recognition, and artificial neural networks.