16

Bi-2DPCA: A Fast Face Coding Method for Recognition

Jian Yang [a], Yong Xu [b] and Jing-yu Yang [a]
[a] *Department of Computer Science, Nanjing University of Science and Technology, Nanjing 210094, China*
[b] *Shenzhen Graduate School, Harbin Institute of Technology, Shenzhen, China*

1. Introduction

Face recognition has received significant attention in the past decades due to its potential applications in biometrics, information security, law enforcement, etc. Numerous methods have been suggested to address this problem [1]. Among appearance-based holistic approaches, principal component analysis (PCA) turns out to be very effective. As a classical unsupervised learning and data analysis technique, PCA was first used to represent images of human faces by Sirovich and Kirby in 1987 [2, 3]. Subsequently, Turk and Pentland [4, 5] applied PCA to face recognition and presented the well-known Eigenfaces method in 1991. Since then, PCA has been widely investigated and has become one of the most successful approaches to face recognition [6-15].

PCA-based image representation and analysis technique is based on image vectors. That is, before applying PCA, the given 2D image matrices must be mapped into 1D image vectors by stacking their columns (or rows). The resulting image vectors generally lead to a high-dimensional image vector space. In such a space, calculating the eigenvectors of the covariance matrix is a critical problem deserving consideration. When the number of training samples is smaller than the dimension of images, the singular value decomposition (SVD) technique is useful for reducing the computational complexity [1-4]. However, when the training sample size becomes large, the SVD technique is helpless. To deal with this problem, an incremental principal component analysis (IPCA) technique has been proposed recently [16]. But, the efficiency of this algorithm still depends on the distribution of data.

Over the last few years, two PCA-related methods, independent component analysis (ICA) [17] and kernel principal component analysis (KPCA) [18, 19] have been of wide concern. Bartlett [20], Yuen [21], Liu [22], and Draper [23] proposed using ICA for face representation and found that it was better than PCA when cosine was used as the similarity measure (however, the performance difference between ICA and PCA was not significant if the Euclidean distance is used [23]). Yang [24] and Liu [25] used KPCA for face feature extraction and recognition and showed that KPCA outperforms the classical PCA. Like PCA, ICA and KPCA both follow the matrix-to-vector mapping strategy when they are used for image analysis and, their algorithms are more complex than PCA. So, ICA and KPCA are considered to be computationally more expensive than PCA. The experimental results in

[24] showed the ratio of the computation time required by ICA, KPCA and PCA is, on average, 8.7: 3.2: 1.0.

Recently, a straightforward image projection technique, coined two-dimensional principal component analysis (2DPCA) [26, 27], was developed for image representation. Differing from standard PCA, 2DPCA is based on 2D matrices rather than 1D vectors. That is, 2DPCA does not need to transform an image matrix into a vector in advance. Instead, it can construct an image covariance matrix directly using the original image matrices. In contrast with the covariance matrix of PCA, the size of the image covariance matrix of 2DPCA is much smaller. Thereby, 2DPCA has two remarkable advantages over PCA. Firstly, it is much easier to evaluate the covariance matrix accurately with a given number of training images. Secondly, it is computationally more efficient to determine the eigenvectors.

The disadvantage of 2DPCA, however, is also obvious [27]. 2DPCA-based image representation was not as economical as PCA in terms of storage requirements, since 2DPCA needs more coefficients than PCA for image representation. Although a feasible alternative to deal with this is to use PCA after 2DPCA for further dimensional reduction, it is still unclear how the dimension of 2DPCA could be reduced directly. In this paper, we will address this issue.

In image compression area, the classical 2D-KLT technique was ever adopted to implement KLT (KL transform) for computational efficiency, based on an assumed image model [28-30]. Assuming that the image random field has a separable covariance function (or autocorrelation function) and, the horizontal and vertical statistics satisfy the first-order Markov model, PCA can be equivalently implemented by a separable transform, i.e. 2D-KLT. Recently, Olmos et al. [31] borrowed the idea of classical 2D-KLT and suggested a method called ST-KLT to carry out spatial-temporal analysis on multi-channel signal. However, in Olmos's method, the horizontal and vertical covariance matrices are not generated by the assumed first-order Markov covariance functions but evaluated by the training samples. This implies that the separability assumption was actually abandoned in ST-KLT. The abandonment of this assumption will give rise to a series of problems; for example, no theory can guarantee that ST-KLT is a good approximation to KLT.

Motivated by the classical 2D-KLT, we develop a new feature extraction method coined Bi-2DPCA to overcome the weakness of 2DPCA. The initial idea of Bi-2DPCA is to perform 2DPCA twice sequentially: the first one is in horizontal direction and the second is in vertical direction. After the second compression, the resulting features are significantly reduced but still as powerful as the original ones (i.e. 2DPCA features). Differing from the classical 2D-KLT, Bi-2DPCA does not depend on an assumed image model. Instead, it can work as independently as PCA. More importantly, Bi-2DPCA lays a solid theoretical foundation for a separable transform without any assumed image model. It provides a sequentially optimal image representation mechanism in the sense of minimal mean-square error. In comparison, ST-KLT lacks theoretical justifications and is shown to be sub-optimal with respect to representation error. So, Bi-2DPCA is a better approximation to KLT than ST-KLT in theory.

The remainder of this paper is organized as follows. Section 2 outlines some relevant techniques, such as PCA (KLT), classical 2D-KLT, and ST-KLT. 2DPCA technique is outlined and its properties are presented in Section 3. Bi-2DPCA is proposed in Section 4 and Bi-2DPCA based representation error is analyzed. In Section 5, the proposed Bi-2DPCA is systematically compared to other PCA (KLT) techniques. In Section 6, Bi-2DPCA is applied

to face coding and recognition cases. The method is evaluated and compared to other methods using the AT&T and FERET face databases. Finally, conclusions are offered in Section 7.

2. Relevant methods

2.1 PCA (Holistic KLT)

The KL transform (KLT) was originally introduced as a series expansion for continuous random process by Karhunen [32] and Loeve [33]. Hotelling [34] first studied a method of principal components to deal with random sequences, which is actually the discrete formulation of the KL series expansion. Thereby, the KL transform is also called principal component analysis (PCA).

Given a set of M training samples (image vectors) $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_M$ in $\mathbb{R}^N$, the covariance matrix of PCA can be evaluated by

$$\mathbf{S}_t = \frac{1}{M}\sum_{j=1}^{M}(\mathbf{x}_j - \overline{\mathbf{x}})(\mathbf{x}_j - \overline{\mathbf{x}})^{\mathrm{T}} \tag{1}$$

where $\overline{\mathbf{x}}$ denotes the mean vector of all training samples.

The orthonormal eigenvectors $\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_d$ of $\mathbf{S}_t$ corresponding to d largest eigenvlaues are chosen as projection axes. If we calculate these eigenvectors directly, the computational complexity is $\mathcal{O}(N^3)$. In real-world applications like face recognition where the training sample size is smaller than the dimension of image vector, there is a more computationally-efficient way to solve this eigen-problem by virtues of SVD technique [35]. Specifically, let $\mathbf{Q} = \left[\mathbf{x}_1 - \overline{\mathbf{x}}, \cdots, \mathbf{x}_M - \overline{\mathbf{x}}\right]$ and form an $M \times M$ Gram matrix $\mathbf{K} = \mathbf{Q}^{\mathrm{T}}\mathbf{Q}$. We only need to solve $\mathbf{K}$'s eigenvectors, so the computational complexity is reduced to $\mathcal{O}(M^3)$.

After the projection of sample **x** onto these eigenvectors, we get the PCA-transformed feature vector

$$\mathbf{y} = \mathbf{\Psi}^{\mathrm{T}}(\mathbf{x} - \overline{\mathbf{x}}), \text{ where } \mathbf{\Psi} = (\mathbf{w}_1, \mathbf{w}_2, \cdots, \mathbf{w}_d), \tag{2}$$

PCA has a number of desirable properties. For example, the PCA transform coefficients are uncorrelated and, PCA-based image representation has minimal mean-square approximation error (that is, PCA packs most of the energy into a small number of principal components so that the error due to truncation is smaller than with other transforms). These properties make it optimal in many signal-processing applications.

It should be mentioned that PCA is a 1D vector based technique. That is, before we apply PCA to face image feature extraction, an initial step is to transform 2D image matrices into 1D image vectors. Generally, an image $\mathbf{A} = (\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \cdots, \boldsymbol{\alpha}_n)$ is converted into a column vector $\mathbf{x} = \mathrm{vec}(\mathbf{A}) = (\boldsymbol{\alpha}_1^{\mathrm{T}}, \boldsymbol{\alpha}_2^{\mathrm{T}}, \cdots, \boldsymbol{\alpha}_n^{\mathrm{T}})^{\mathrm{T}}$ by stacking the columns of $\mathbf{A}$.

2.2 Classical 2D-KLT (Separable KLT)

Different from face recognition, in image compression area, an ensemble of images rather than a category of images need to be processed using the same transform matrix. That is to say, KLT-based image compression must be functionally independent of the data [30]. Since it is not realistic to use the training samples to obtain a covariance matrix, an assumed image model is needed. An ensemble of images $\mathbf{A}(i,j)$ can be characterized by a two-dimensional random field (the mean is assumed to be zero without loss of generality), in which the total covariance function (or auto-correlation function) is assumed to be *separable* [29, 30], i.e.,

$$\mathrm{E}[\mathbf{A}(i,i')\,\mathbf{A}(j,j')] = r(i,j;i',j') = r_1(i,i')\,r_2(j,j'). \tag{3}$$

This means that the covariance function of the random field can be expressed by the product of covariance functions of two one-dimensional sequences. Generally, the two one-dimensional sequences are assumed to be first-order stationary Markov sequences, that is,

$$r_1(i,i') = \rho_1^{|i-i'|} = \rho_1^{\Delta i} \text{ and } r_2(j,j') = \rho_2^{|j-j'|} = \rho_2^{\Delta j} \tag{4}$$

where ρ_1 and ρ_2 are horizontal (column) and vertical (row) correlations and $|\rho_1|<1$, $|\rho_2|<1$.

Thus, the horizontal and vertical covariance matrices are

$$\mathbf{R}_h = \{\rho_1^{\Delta i}\}_{\Delta i=0,1,\cdots,n-1} \text{ and } \mathbf{R}_v = \{\rho_2^{\Delta j}\}_{\Delta j=0,1,\cdots,m-1}. \tag{5}$$

Due to the separability assumption of the total covariance function, the total covariance matrix $\mathcal{R}$ can be expressed by the Kronecker (outer) product of $\mathbf{R}_h$ and $\mathbf{R}_v$, i.e.,

$$\mathcal{R} = \mathbf{R}_h \otimes \mathbf{R}_v \tag{6}$$

The Kronecker product has the following property:

Lemma 1 [30] Suppose that **A** is $n\times n$ matrix and **B** is $m\times m$ matrix. If $\mathbf{C}=\mathbf{A}\otimes\mathbf{B}$, $\mathbf{C}\xi_k=\gamma_k\xi_k$, $\mathbf{A}\mathbf{x}_i=\lambda_i\mathbf{x}_i$, and $\mathbf{B}\mathbf{y}_j=\mu_j\mathbf{y}_j$, then, $\xi_k=\mathbf{x}_i\otimes\mathbf{y}_j$ and $\gamma_k=\lambda_i\mu_j$, where $1\le i\le n$, $1\le j\le m$, $1\le k\le mn$.

Suppose the eigenvector matrix of $\mathcal{R}$ is denoted by $\mathbf{\Psi}$, and the eigenvector matrices of $\mathbf{R}_h$ and $\mathbf{R}_v$ are denoted by $\mathbf{\Phi}_h$ and $\mathbf{\Phi}_v$. From Lemma 1, we have $\mathbf{\Psi}=\mathbf{\Phi}_h\otimes\mathbf{\Phi}_v$.

The KL transform of $\mathbf{x}=\mathrm{vec}(\mathbf{A})$ is

$$\mathbf{y}=\mathbf{\Psi}^T\mathbf{x}=(\mathbf{\Phi}_h^T\otimes\mathbf{\Phi}_v^T)\mathbf{x}, \tag{7}$$

which is equivalent to the separable transform

$$\mathbf{C}=\mathbf{\Phi}_v^T\mathbf{A}\mathbf{\Phi}_h. \tag{8}$$

The advantages of modeling the total covariance function by separable covariance functions of first-order stationary Markov sequences are twofold. First, this model makes it possible to implement KLT-based image compression for an ensemble of images because the model itself is independent of the data. Second, the process of image compression becomes computationally more efficient by virtue of a separable transform. The size of the eigen-

problem is significantly reduced after the decomposition of total covariance function and the transformation calculations are largely decreased [30] as well.
It should be emphasized that the assumptions of separable covariance function and the first-order stationary Markov models are crucial since they lay the foundations for the classical 2D-KLT. Without the assumptions, it makes no sense to discuss the classical 2D-KLT.
In addition to 2D-KLT, there are other traditional separable transforms such as two-dimensional discrete cosine transform (2D-DCT), Walsh-Hadamard transform, Slant transform and two-dimensional discrete Fourier transform (2D-DFT). Among these transforms, 2D-DCT has the optimal energy packing property for high correlated data. This is because DCT is very close to the KLT of a first-order stationary Markov sequence when the correlation parameter ρ is close to 1. This property of DCT combining with the fact that it is a fast transform make it a useful substitute for the KLT of high correlated first-order Markov sequences.

2.3 ST-KLT

Following the idea of the classical 2D-KLT, Olmos et al. [31] recently suggested a method called ST-KLT to carry out spatio-temporal analysis on multi-channel signals. Different from the classical 2D-KLT, in Olmos's method, the horizontal and vertical covariance matrices are not generated by covariance functions like Eq. (5) but evaluated by the training samples, i.e.,

$$\mathbf{R}_h = \frac{1}{M}\sum_{j=1}^{M}(\mathbf{A}_j - \overline{\mathbf{A}})^{\mathrm{T}}(\mathbf{A}_j - \overline{\mathbf{A}}), \tag{9}$$

$$\mathbf{R}_v = \frac{1}{M}\sum_{j=1}^{M}(\mathbf{A}_j - \overline{\mathbf{A}})(\mathbf{A}_j - \overline{\mathbf{A}})^{\mathrm{T}}, \tag{10}$$

where M is the number of training image samples, $\mathbf{A}_j$ is an $m \times n$ matrix denoting the j-th training samples, and $\overline{\mathbf{A}}$ is the mean image of all training samples. Thus, the image model (the assumption of images satisfying first-order Markov model with separable covariance function) is actually abandoned by ST-KLT. Without this model, the following relation

$$\mathbf{S}_t = \mathbf{R}_h \otimes \mathbf{R}_v \tag{11}$$

does not hold in general. In such a case, the authors [31] thought that the separable transform in Eq. (8) could be understood as an approximation to KLT with lower energy packing performance. But, this claim gives rise to a series of problems:
(i) Why can one say the separable transform in Eq. (8) is an approximation to KLT without the image model? What is the degree of the approximation?
(ii) The separable transform in Eq. (8) can be decomposed into two transforms: $\mathbf{B} = \mathbf{A}\boldsymbol{\Phi}_h$ and $\mathbf{C} = \boldsymbol{\Phi}_v^T\mathbf{B}$. What are their intuitive meanings without the separability assumption?
(iii) Does there exist a separable transform that is a better approximation to KLT without considering the separability assumption?
These problems are critical in theory but were not addressed in Olmos's paper [31]. In addition, if one uses training samples to evaluate the horizontal and vertical covariance

matrices, the intuitive meanings of $\mathbf{R}_h$ and $\mathbf{R}_v$ are not as clear as that in the classical 2D-KLT, since the assumption of the first-order stationary Markov statistics is abandoned.

From the image representation (or coding) point of view, the methods outlined above can be divided into two categories: image-data *dependent* methods and image-data *independent* methods. The image-data *dependent* methods need training image samples to learn the transform matrix, while image-data *independent* methods need a model to generate the transform matrix, without the training or learning process. The image-data *independent* methods include Classical 2D-KLT, 2D-DCT, and other separable transform based image coding methods like Walsh-Hadamard transform, Slant transform and so on. The image-data *dependent* methods include PCA, ST-KLT and 2DPCA [27]. In the following sections, we will outline 2DPCA and further derive a new image-data *dependent* coding method.

3. 2DPCA and Its Properties

3.1 Outline

The idea of 2DPCA was motivated by Liu's image side-projection technique [36]. Given image $\mathbf{A}$, an $m \times n$ random matrix, the aim of 2DPCA is to find a set orthogonal projection axes $\mathbf{X}_1, \cdots, \mathbf{X}_q$ so that the projected vectors $\mathbf{Y}_k = \mathbf{A}\mathbf{X}_k$ ($k = 1,2,\cdots,q$) achieve a maximal total scatter [27]. The *image covariance* (*scatter*) *matrix* of 2DPCA is introduced as follows:

$$\mathbf{G}_t = \mathrm{E}[(\mathbf{A} - \mathrm{E}\mathbf{A})^{\mathrm{T}}(\mathbf{A} - \mathrm{E}\mathbf{A})]. \tag{12}$$

It is easy to show $\mathbf{G}_t$ is an $n \times n$ non-negative definite matrix. The matrix can be evaluated by

$$\mathbf{G}_t = \frac{1}{M}\sum_{j=1}^{M}(\mathbf{A}_j - \overline{\mathbf{A}})^{\mathrm{T}}(\mathbf{A}_j - \overline{\mathbf{A}}) \tag{13}$$

where *M* is the number of training image samples, $\mathbf{A}_j$ is an $m \times n$ matrix denoting the *j*-th training samples, and $\overline{\mathbf{A}}$ is the mean image of all training samples.

The optimal projection axes $\mathbf{X}_1, \cdots, \mathbf{X}_q$ are chosen as the orthonormal eigenvectors of $\mathbf{G}_t$ corresponding to *q* largest eigenvalues [27]. After the projection of image samples onto these axes, i.e.,

$$\mathbf{Y}_k = (\mathbf{A} - \overline{\mathbf{A}})\mathbf{X}_k,\ k = 1,2,\cdots,q, \tag{14}$$

we obtain a family of *principal component vectors*, $\mathbf{Y}_1, \cdots, \mathbf{Y}_q$, which form an $m \times q$ *feature matrix* $\mathbf{B} = [\mathbf{Y}_1, \cdots, \mathbf{Y}_q]$. Letting $\mathbf{U} = [\mathbf{X}_1, \cdots, \mathbf{X}_q]$, Eq. (14) can be alternatively expressed by

$$\mathbf{B} = (\mathbf{A} - \overline{\mathbf{A}})\mathbf{U} \tag{15}$$

The *feature matrix* **B** are used to represent image $\mathbf{A}$ for classification. The similarity measure between two feature matrices, $\mathbf{B}_i = [\mathbf{Y}_1^{(i)}, \cdots, \mathbf{Y}_q^{(i)}]$ and $\mathbf{B}_j = [\mathbf{Y}_1^{(j)}, \cdots, \mathbf{Y}_q^{(j)}]$, is given

by

$$dist(\mathbf{B}^{(i)}, \mathbf{B}^{(j)}) = \sum_{k=1}^{q} \left\| \mathbf{Y}_k^{(i)} - \mathbf{Y}_k^{(j)} \right\| \tag{16}$$

where $\|\cdot\|$ is the notation of norm, which determines what measure is used.

It can be seen that 2DPCA is a 2D matrix based image analysis technique. That is, we do not need to transform an image matrix into a vector in advance. Instead, we can construct an *image covariance matrix* directly using the original image matrices, and then use its eigenvectors as projection axes to perform principal component analysis.

3.2 Correlation Property

First of all, let us give an intuitive explanation of the image covariance matrix $\mathbf{G}_t$. To this end, a generalized covariance definition should be given.

For 1-dimensional random variables ξ and η, we know that their covariance is defined by $\mathrm{E}\{(\xi - E\xi)(\eta - E\eta)\}$. Now, let us generalize this concept to the *n*-dimensional case. Suppose $\boldsymbol{\xi}$ and $\boldsymbol{\eta}$ are *n*-dimensional random column vectors, their covariance is defined by [37]

$$Cov(\boldsymbol{\xi}, \boldsymbol{\eta}) = \mathrm{E}\{(\boldsymbol{\xi} - \mathrm{E}\boldsymbol{\xi})^T (\boldsymbol{\eta} - \mathrm{E}\boldsymbol{\eta})\} \tag{17}$$

Note that the covariance of *n*-dimensional random vectors defined above is a scalar rather than a matrix. We can explain $Cov(\boldsymbol{\xi}, \boldsymbol{\eta})$ as the sum of the covariances of a set of 1-dimensional random variables. Specifically, let us denote $\boldsymbol{\xi} = (\xi_1, \xi_2, \cdots, \xi_n)$ and $\boldsymbol{\eta} = (\eta_1, \eta_2, \cdots, \eta_n)$, then

$$Cov(\boldsymbol{\xi}, \boldsymbol{\eta}) = \sum_{j=1}^{n} \mathrm{E}\{(\xi_j - \mathrm{E}\xi_j)(\eta_j - \mathrm{E}\eta_j)\} = \sum_{j=1}^{n} \mathrm{Cov}(\xi_j, \eta_j) \tag{18}$$

This means that the covariance of two *n*-dimensional random vectors defined in Eq. (17) is essentially the sum of the covariances of the corresponding components.

Accordingly, we can define the correlation coefficient between $\boldsymbol{\xi}$ and $\boldsymbol{\eta}$ as follows:

$$\rho(\boldsymbol{\xi}, \boldsymbol{\eta}) = \frac{Cov(\boldsymbol{\xi}, \boldsymbol{\eta})}{\sqrt{Cov(\boldsymbol{\xi}, \boldsymbol{\xi}) \cdot Cov(\boldsymbol{\eta}, \boldsymbol{\eta})}} \tag{19}$$

Now, let us analyze the image covariance matrix $\mathbf{G}_t$. Suppose image $\mathbf{A}$ is formed by a set of column vectors (random vectors), i.e. $\mathbf{A} = (\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2, \cdots, \boldsymbol{\alpha}_n)$, then

$$\begin{aligned} \mathbf{G}_t &= \mathrm{E}[(\mathbf{A} - \mathrm{E}\mathbf{A})^T (\mathbf{A} - \mathrm{E}\mathbf{A})] \\ &= \\ &\mathrm{E}[(\boldsymbol{\alpha}_1 - \mathrm{E}\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2 - \mathrm{E}\boldsymbol{\alpha}_2, \cdots, \boldsymbol{\alpha}_n - \mathrm{E}\boldsymbol{\alpha}_n)^T (\boldsymbol{\alpha}_1 - \mathrm{E}\boldsymbol{\alpha}_1, \boldsymbol{\alpha}_2 - \mathrm{E}\boldsymbol{\alpha}_2, \cdots, \boldsymbol{\alpha}_n - \mathrm{E}\boldsymbol{\alpha}_n)] \\ &= \left[\mathrm{E}\{(\boldsymbol{\alpha}_i - \mathrm{E}\boldsymbol{\alpha}_i)^T (\boldsymbol{\alpha}_j - \mathrm{E}\boldsymbol{\alpha}_j)\}\right]_{n \times n} \end{aligned}$$

$$= \left\lfloor Cov(\boldsymbol{\alpha}_i, \boldsymbol{\alpha}_j) \right\rfloor_{n \times n}$$

This derivation shows that the element on the *i*-th line and the *j*-th column of $\mathbf{G}_t$ are essentially the covariance of column *i* and column *j* of image **A**. Now, the physical meaning of $\mathbf{G}_t$ is clear. It characterizes the correlation between column vectors (rather than the elements) of image matrices. From this viewpoint, $\mathbf{G}_t$ can be called *image column (or horizontal) covariance matrix*.

Let us consider the correlation between the principal component vectors after 2DPCA transform. From Eq. (14), we have $\mathbf{Y}_k = (\mathbf{A} - \mathrm{E}\mathbf{A})\mathbf{X}_k$, $(i = 1, 2, \cdots, q)$. The covariance between two principal component vectors $\mathbf{Y}_i$ and $\mathbf{Y}_j$ is

$$\begin{aligned} Cov(\mathbf{Y}_i, \mathbf{Y}_j) &= \mathrm{E}\{(\mathbf{Y}_i - \mathrm{E}\mathbf{Y}_i)^T (\mathbf{Y}_j - \mathrm{E}\mathbf{Y}_j)\} \\ &= \mathrm{E}\{[\mathbf{A}\mathbf{X}_i - \mathrm{E}(\mathbf{A}\mathbf{X}_i)]^T [\mathbf{A}\mathbf{X}_j - \mathrm{E}(\mathbf{A}\mathbf{X}_j)]\} \\ &= \mathbf{X}_i^T \{\mathrm{E}[(\mathbf{A} - \mathrm{E}\mathbf{A})^T (\mathbf{A} - \mathrm{E}\mathbf{A})]\}\mathbf{X}_j \end{aligned}$$

It follows from Eq. (12) that

$$Cov(\mathbf{Y}_i, \mathbf{Y}_j) = \mathbf{X}_i^T \mathbf{G}_t \mathbf{X}_j \tag{20}$$

So, the correlation coefficient between two projected feature vectors $\mathbf{Y}_i$ and $\mathbf{Y}_j$ is

$$\rho(\mathbf{Y}_i, \mathbf{Y}_j) = \frac{\mathbf{X}_i^T \mathbf{G}_t \mathbf{X}_j}{\sqrt{\mathbf{X}_i^T \mathbf{G}_t \mathbf{X}_i} \sqrt{\mathbf{X}_j^T \mathbf{G}_t \mathbf{X}_j}} \tag{21}$$

Since the projection vectors of 2DPCA are selected as $\mathbf{X}_1, \cdots, \mathbf{X}_q$, a set of orthonormal eigenvectors of $\mathbf{G}_t$, it is easy to draw the following conclusion:

Proposition 1 The principal component vectors of 2DPCA, $\mathbf{Y}_i = (\mathbf{A} - \mathrm{E}\mathbf{A})\mathbf{X}_i$ ($i = 1, 2, \cdots, q$), satisfy $Cov(\mathbf{Y}_i, \mathbf{Y}_j) = 0$, $i \neq j$, $i, j = 1, \cdots, q$, which means $\rho(\mathbf{Y}_i, \mathbf{Y}_j) = 0$, $i \neq j$, $i, j = 1, \cdots, q$.

Proposition 1 indicates that the projected feature vectors resulting from 2DPCA are mutually uncorrelated. In other words, 2DPCA transform can eliminate the correlation between column vectors (rather than the elements) of image matrices.

3.3 Minimal mean square error representation property

Assume that $\mathbf{A}$ is an $m \times n$ random image matrix. Without loss of generality, the expectation of image samples generated from $\mathbf{A}$ is supposed to be zero, i.e. $E\mathbf{A} = 0$, in the following discussion since it is easy to centralize image $\mathbf{A}$ by $\mathbf{A} - E\mathbf{A}$ if $E\mathbf{A} \neq 0$.

Suppose that in $\mathbb{R}^n$, we are given an arbitrary set of vector system $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n$ which satisfy

$$\mathbf{u}_i^T \mathbf{u}_j = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \tag{22}$$

Projecting $\mathbf{A}$ onto these orthonormal basis vectors $\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_n$, we have

$$\mathbf{A}\mathbf{u}_j = \boldsymbol{\beta}_j, \ j = 1, 2, \cdots, n \tag{23}$$

Then, the image can be completely expanded by

$$\mathbf{A} = \sum_{j=1}^{n} \boldsymbol{\beta}_j \mathbf{u}_j^T \tag{24}$$

If we use the first *d* components to represent $\mathbf{A}$, the reconstructed approximation is

$$\hat{\mathbf{A}} = \sum_{j=1}^{d} \boldsymbol{\beta}_j \mathbf{u}_j^T \tag{25}$$

And, the reconstruction mean-square error (MSE) can be characterized by

$$\varepsilon^2 = \mathrm{E} \| \mathbf{A} - \hat{\mathbf{A}} \|_F^2 \tag{26}$$

where $\| \cdot \|$ denotes the Frobenius norm of a matrix.

Theorem 1 [38] Suppose $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_n$ are the eigenvectors of $\mathbf{G}_t$ corresponding to eigenvalues $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$. If we use the first *q* eigenvectors as projection axes and the resulting component vectors $\boldsymbol{\beta}_1, \boldsymbol{\beta}_2, \cdots, \boldsymbol{\beta}_q$ to represent $\mathbf{A}$, the reconstruction mean-square error can be minimized in the sense of the matrix Frobenius norm, and $\varepsilon^2 = \sum_{j=q+1}^{n} \lambda_j$.

It should be noted that the minimal MSE property of 2DPCA (the result of Theorem 1) is with respect to the expansion form in Eq. (24) (or the transform form in Eq. (23)), which is different from that of PCA. Regarding this, we will give more detailed explanation in Section 5.1.

Theorem 1 provides a theoretical justification for choosing of the principal eigenvectors of $\mathbf{G}_t$ to expand the images. This eigenvectors construct an optimal coordinate system for image expansion in *n*-dimensional space. Under this coordinate system, the expansion coefficients (a set of projected vectors) form the optimal representation of images in the sense of minimal mean-square error.

We can also comprehend the physical meanings of 2DPCA based image transform from the *image energy* point of view. For a given image **A**, after a complete 2DPCA transform (all eigenvectors of $\mathbf{G}_t$ are used), we obtain its image **B**, which is the same size as **A**. By this transform, the average image energy is reassigned on image columns. The energy allocated to the first *p* columns is $\sum_{j=1}^{q} \lambda_j$, while that allocated to the remaining columns is $\sum_{j=q+1}^{n} \lambda_j$.

Since the eigenvalues $\lambda_1, \lambda_2, \cdots, \lambda_n$ are sorted in decreasing order, most of the image energy is packed into a small number of column vectors of **B**. In a word, 2DPCA realizes the optimal image energy compression in horizontal direction.

4. Bi-2DPCA

4.1 Idea

2DPCA can eliminate the correlations between image columns and compress the image energy optimally in horizontal direction. But, it disregards the correlations between image rows and the data compression in vertical direction. So, its compression rate is far lower than PCA and more coefficients are needed for the representation of images. This must lead to a slow classification speed and large storage requirements for large-scaled databases [27, 38].

In this section, we will suggest a way to overcome the weakness of 2DPCA. Our idea is very simple, just to perform 2DPCA compression twice: the first one is in horizontal direction and the second is in vertical direction (*Note that any operation in vertical direction can be equivalently implemented by an operation in horizontal direction by virtue of the transpose operation of matrix*). Specifically, given image **A**, we obtain the feature matrix **B** after the first 2DPCA compression. Then, we transpose **B** and input $\mathbf{B}^T$ into 2DPCA, and determine the transform matrix **V**. Projecting $\mathbf{B}^T$ onto **V**, we obtain $\mathbf{C}^T = \mathbf{B}^T\mathbf{V}$. The resulting feature matrix is $\mathbf{C} = \mathbf{V}^T\mathbf{B}$. This process is illustrated in Figure 1.

In the whole process, the first 2DPCA transform $\mathbf{B} = \mathbf{AU}$ performs the compression of 2D-data in horizontal direction, making the image energy pack into a small number of columns. While the second 2DPCA transform $\mathbf{C} = \mathbf{V}^T\mathbf{B}$ performs the compression of 2D-data in vertical direction, eliminating the correlations between columns of image **B** and making its energy further compact into a small number of rows. Ultimately, the energy of the whole image is packed into the up-left corner of the image matrix.

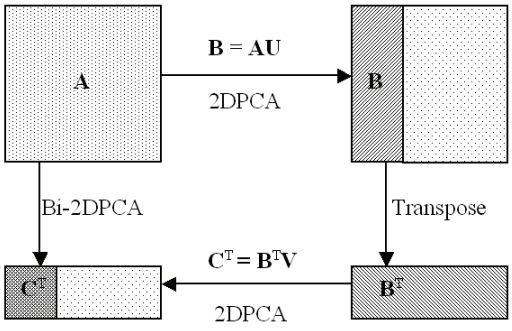


Fig. 1. Illustration of Bi-2DPCA

4.2 Algorithm

Now, let us present the detailed implementation of Bi-2DPCA. Based on the given training sample image **A**, we can construct the image covariance matrix $\mathbf{G}_t$ using Eq. (12). Suppose $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_q$ are the orthonormal eigenvectors of $\mathbf{G}_t$ corresponding to q largest eigenvalues $\lambda_1, \lambda_2, \cdots, \lambda_q$. Let $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_q)$. Then, we get the 2DPCA feature matrix of **A**, i.e.,

$$\mathbf{B} = (\mathbf{A} - \mathrm{E}\mathbf{A})\mathbf{U} \tag{27}$$

Constructing the image covariance matrix $\mathbf{H}_t$ based on $\mathbf{B}^T$, we have

$$\mathbf{H}_t = \mathrm{E}[(\mathbf{B} - \mathrm{E}\mathbf{B})(\mathbf{B} - \mathrm{E}\mathbf{B})^T] \tag{28}$$

From Eq. (27), we know $\mathrm{E}\mathbf{B} = 0$. Thus $\mathbf{H}_t = \mathrm{E}[\mathbf{B}\mathbf{B}^T]$. This matrix can be evaluated by

$$\mathbf{H}_t = \frac{1}{M}\sum_{j=1}^{M} \mathbf{B}_j^{\mathrm{T}}\mathbf{B}_j, \text{ where } \mathbf{B}_j = (\mathbf{A}_j - \overline{\mathbf{A}})\mathbf{U}, \tag{29}$$

Suppose $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_p$ are the orthonormal eigenvectors of $\mathbf{H}_t$ corresponding to p largest eigenvalues $\mu_1, \mu_2, \cdots, \mu_p$. Let $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_p)$. We get the 2DPCA feature matrix of $\mathbf{B}^T$ by

$$\mathbf{C}^T = (\mathbf{B}^T - \mathrm{E}\mathbf{B}^T)\mathbf{V} = \mathbf{B}^T\mathbf{V} \tag{30}$$

Thus

$$\mathbf{C} = \mathbf{V}^T\mathbf{B} = \mathbf{V}^T(\mathbf{A} - \mathrm{E}\mathbf{A})\mathbf{U} \tag{31}$$

The resulting feature matrix **C** is a $p \times q$ matrix, which is much smaller than the 2DPCA feature matrix **B** and the original image **A** since p and q are always selected much smaller than m and n. We can use **C** to represent **A** for recognition purpose.

In summary of the discussion so far, the Bi-2DPCA algorithm is given below:

Bi-2DPCA Algorithm

Step 1. Construct the image column covariance matrix $\mathbf{G}_t$ using Eq. (13). Calculate $\mathbf{G}_t$'s orthonormal eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_q$ corresponding to q largest eigenvalues.

Step 2. Construct the image row covariance matrix $\mathbf{H}_t$ using Eq. (29). Calculate $\mathbf{H}_t$'s orthonormal eigenvectors $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_p$ corresponding to p largest eigenvalues.

Step 3. Let $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_q)$ and $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_p)$. Use the transform $\mathbf{C} = \mathbf{V}^T(\mathbf{A} - \overline{\mathbf{A}})\mathbf{U}$ to get the feature matrix of the given image sample **A**.

4.3 Bi-2DPCA based image reconstruction

Bi-2DPCA allows the reconstruction of the original image pattern. Since $\mathbf{G}_t$'s eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_q$ and $\mathbf{H}_t$'s eigenvectors $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_p$ are both orthonormal, from Eq. (31), it is easy to obtain the reconstructed image of $\mathbf{A}$:

$$\widetilde{\mathbf{A}} = \mathrm{E}\mathbf{A} + \mathbf{V}\mathbf{C}\mathbf{U}^T, \tag{32}$$

where $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_q)$, $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_p)$.

Denoting $\mathbf{C} = (c_{ij})_{p\times q}$, Eq. (32) can be rewritten by

$$\widetilde{\mathbf{A}} = \overline{\mathbf{A}} + \sum_{i=1}^{p}\sum_{j=1}^{q} c_{ij}\mathbf{v}_i\mathbf{u}_j^T \tag{33}$$

Let us denote $\boldsymbol{\varphi}_{ij} = \mathbf{v}_i\mathbf{u}_j^T$ ($i = 1, \cdots, p$; $j = 1, \cdots, q$). Obviously, $\boldsymbol{\varphi}_{ij}$ is rank-1 matrix, which is of the same size as original image $\mathbf{A}$ and called the *basis image*. Any image $\mathbf{A}$ can be approximately reconstructed by adding up the weighted *basis images* and the mean image.

4.4 Reconstruction error evaluation

Without loss of generality, the expectation of image samples generated from $\mathbf{A}$ is also supposed to be zero, i.e. $E\mathbf{A} = 0$, in the following discussion.

If we use the feature matrix **C** to represent $\mathbf{A}$, the reconstruction error of image $\mathbf{A}$ can be expressed by

$$\boldsymbol{\Delta} = \mathbf{A} - \widetilde{\mathbf{A}} = \sum_{i=p+1}^{m}\sum_{j=q+1}^{n} c_{ij}\mathbf{v}_i\mathbf{u}_j^T \tag{34}$$

And, the total reconstruction mean-square error (MSE) can be characterized by

$$\varepsilon_{\mathrm{t}}^2 = \mathrm{E}\,\|\boldsymbol{\Delta}\|_F^2 = \mathrm{E}\,\|\mathbf{A} - \widetilde{\mathbf{A}}\|_F^2 \tag{35}$$

Theorem 2 Suppose $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_n$ are the eigenvectors of $\mathbf{G}_t$ corresponding to $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ and $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_m$ are the eigenvectors of $\mathbf{H}_t$ corresponding to $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_m$. Let $\mathbf{U} = (\mathbf{u}_1, \cdots, \mathbf{u}_q)$ and $\mathbf{V} = (\mathbf{v}_1, \cdots, \mathbf{v}_p)$. If we use the feature matrix $\mathbf{C} = \mathbf{V}^T(\mathbf{A} - \mathrm{E}\mathbf{A})\mathbf{U}$ to represent $\mathbf{A}$, the total mean square error is $\varepsilon_{\mathrm{t}}^2 = \sum_{j=q+1}^{n}\lambda_j + \sum_{j=p+1}^{m}\mu_j$.

The proof of Theorem 2 is given in Appendix A.

Theorem 2 tells us the total MSE of Bi-2DPCA based image representation is the sum of MSEs corresponding to the two involved 2DPCA. It is easy to understand this result from the *image energy loss* point of view. After the first 2DPCA compression on **A**, we obtain **B**

and know that the corresponding image energy loss is $\sum_{j=q+1}^{n}\lambda_j$ from Theorem 1. After the second 2DPCA compression on $\mathbf{B}^T$, we obtain **C** and know the image energy loss is $\sum_{j=p+1}^{m}\mu_j$ also from Theorem 1. So, the total *energy loss* should be $\varepsilon_t^2 = \sum_{j=q+1}^{n}\lambda_j + \sum_{j=p+1}^{m}\mu_j$.

5. Relationship to other PCA (KLT) Techniques

5.1 Relationship to PCA and 2DPCA

Let us begin our discussion from the viewpoint of mean-square error. Unquestionably, PCA (KLT) is optimal for 1D data representation (compression) in the sense of minimal mean-square error. For 1D data **x** (vector), if the transform form $\mathbf{y} = \mathbf{\Psi}^T\mathbf{x}$ is chosen, PCA-based transform is optimal. For 2D data **A** (matrix), we can transform the data into 1D vectors by stacking the columns of $\mathbf{A}$ and then use PCA to obtain a *holistically* optimal representation. For 2D data **A**, however, we can also choose another transform form $\mathbf{B} = \mathbf{AU}$. With respect to this transform form, 2DPCA turns out to be optimal; it realizes an optimal compression in horizontal direction. Bi-2DPCA provides a sequentially optimal compression mechanism with respect to the transform form $\mathbf{C} = \mathbf{V}^T\mathbf{AU}$. That is to say, if we choose such a transform form to compress the image data from horizontal to vertical, Bi-2DPCA is optimal in the sense of minimal mean-square error.

PCA, 2DPCA and Bi-2DPCA are all *image-data dependent* coding method. In contrast to PCA, a remarkable advantage of Bi-2DPCA is its low computation requirement. The computational advantage of Bi-2DPCA mainly embodies in the following three aspects:

First, Bi-2DPCA needs less computation than PCA in the construction of covariance matrices. Suppose the image size is $m \times n$ and the training sample size is *M*. Denote $N = m \times n$ and $l = \min\{M, N\}$. To form the covariance matrix (or the corresponding Gram matrix), PCA needs $M \times N \times l$ multiplications, while Bi-2DPCA needs $M \times m \times n^2$ multiplications to construct $\mathbf{G}_t$ and $M \times q \times m^2$ multiplications to construct $\mathbf{H}_t$. So, the total computation of Bi-2DPCA is $M \times (m \times n^2 + q \times m^2)$, which is less than $M \times N \times (m+n)$ since *q* is much smaller than *n*. Generally, $(m+n) < l$ in face recognition problems, so Bi-2DPCA needs less computation for constructing covariance matrices.

Second, Bi-2DPCA has a lower computational complexity than PCA on solving the eigen-problem. From the discussion in Section 2.1, we know that the computational complexity of PCA is $\mathcal{O}(l^3)$. Since Bi-2DPCA only needs to calculate the eigenvectors of $\mathbf{G}_t$ and $\mathbf{H}_t$, its computational complexity is $\mathcal{O}(m^3 + n^3)$. Since l is much larger than *m* or *n* in real-world applications, PCA is computationally more intensive than Bi-2DPCA.

Third, the transformation calculation of Bi-2DPCA in Eq. (31) is also smaller than that of PCA in Eq. (2). The transformation calculation of PCA is $m \times n \times d$, while the calculation of

Bi-2DPCA are $m \times n \times q + m \times q \times p$, which is smaller than $m \times n \times (p+q)$. Since $d = pq$ (if the same amounts of features are required by both methods), PCA generally needs more calculation than Bi-2DPCA in image transformation.

The foregoing discussions show PCA is computationally more intensive than Bi-2DPCA. Besides, another advantage of Bi-2DPCA is that it needs less memory requirement than PCA in face recognition systems. This is because PCA needs to save a much larger transform matrix for feature extaction. The transform matrix of PCA is sized of $N \times d = m \times n \times d$, which amounts to the size of d orginal images. While, the two transform matrices of Bi-2DPCA are only sized of $m \times p + n \times q$, which is generally less than one orginal image in size.

A comparison between PCA and Bi-2DPCA is summarized in Table 1. Here, it should be stressed that the computational advantages of Bi-2DPCA over PCA is independent of the algorithms that are adopted to calculate the eigenvectors. If an algorithm can speed up the eigenvector computation of PCA, it is certain to speed up the eigenvector computation of Bi-2DPCA in the same way.

Method	Computation Requirements			Memory Requirements
	Construction of covariance matrix	Solving eigen-problem	Image Transformation	
PCA	More	More	More	More
Bi-2DPCA	Less	Less	Less	Less

Table 1. A comparison between PCA and Bi-2DPCA in computation and memory requirements

Compared to 2DPCA, the compression rate of Bi-2DPCA is significantly improved. That is, Bi-2DPCA needs much less coefficients than 2DPCA for image representation. The advantages of Bi-2DPCA over 2DPCA are twofold. First, the storage requirements can be significantly reduced. Second, the classification speed will be increased since less computation is needed in distance (similarity) calculation.

5.2 Relationship to classical 2D-KLT

Bi-2DPCA is an *image-data dependent* coding method while 2D-KLT is *image-data independent*. The underlying difference between these two methods is that the classical 2D-KLT is based on an assumed image model while Bi-2DPCA not. The implementation of the classical 2D-KLT depends on the assumption that an ensemble of images satisfies the first-order Markov model with separable autocorrelation function. Without this assumption, the method cannot exist independently because its covariance matrices are constructed by the separable autocorrelation function rather than training samples. In contrast, Bi-2DPCA can work independently without any assumed image model. Like PCA, it relies on training samples to evaluate its covariance matrices.

Actually, the classical 2D-KLT and Bi-2DPCA have different utilities. The classical 2D-KLT is suitable for an ensemble of images and generally applied to image compression. Bi-2DPCA is suitable for a category of images that have some similar characteristics. Bi-2DPCA

can be used for image representation and recognition, such as face recognition, palm identification, etc.

5.3 Relationship to ST-KLT

As discussed in Section 2.3, without the image model, the total covariance matrix is generally not equal to the outer product of the horizontal and vertical covariance matrices. Thus, the separable transform in Eq. (8) is not equivalent to KL transform. The minimal MSE property of ST-KLT cannot be guaranteed and the degree of approximation cannot be evaluated in theory. These problems are critical and not addressed by the authors in their paper [31].

2DPCA provides us theoretical insights to see through the series of problems left by ST-KLT. First of all, by the correlation analysis in Section 3.2, the intuitive meanings of the horizontal and vertical covariance matrices become clear. The horizontal covariance matrix shows the correlation between column vectors of image samples, while the vertical covariance matrix shows the correlation between image row vectors. Secondly, the transform (compression) in horizontal or vertical direction has a clear explanation. 2DPCA-based transform is the optimal transform in horizontal direction in the sense of minimal mean square error. The image energy is compacted into a small number of columns after 2DPCA transform. Similarly, if we use the transpose of image matrices as the input data, 2DPCA can realize the optimal compression in vertical direction.

Thirdly, Bi-2DPCA is a sequentially optimal technique but ST-KLT is not. It is easy to see this by analyzing their transform processes. The transform $\mathbf{C} = \mathbf{V}^T\mathbf{B}\mathbf{U}$ can be decomposed into two transforms: the column (horizontal) transform $\mathbf{B} = \mathbf{A}\mathbf{U}$ and the row (vertical) transform $\mathbf{C} = \mathbf{V}^T\mathbf{B}$. The first transform is same for the two methods but the second one is different. For the second transform, the transform matrix **V** of Bi-2DPCA is formed by the eigenvectors of the matrix $\mathbf{H}_t = \mathrm{E}[(\mathbf{B}-\mathrm{E}\mathbf{B})(\mathbf{B}-\mathrm{E}\mathbf{B})^T]$, while the transform matrix of ST-KLT is formed by the eigenvectors of the matrix $\mathbf{R}_v = \mathrm{E}[(\mathbf{A}-\mathrm{E}\mathbf{A})(\mathbf{A}-\mathrm{E}\mathbf{A})^T]$. These two matrices are obviously different. Since the second transform is independent of the first one, we only need to find an optimal transform to recompress the current image **B** (rather than the original image **A**) after the first 2DPCA transform. From Theorem 1, it follows that the eigenvector system of $\mathbf{H}_t$ should be optimal for compressing **B**. Other vector systems, for example, the eigenvector system of $\mathbf{R}_v$, must be sub-optimal and lead to a larger mean-square error. So, as a separable transform, Bi-2DPCA is better than ST-KLT in terms of energy packing performance.

In summary, Bi-2DPCA and ST-KLT are both *image-data dependent* coding method. Bi-2DPCA lays a solid theoretical foundation for a separable transform without any assumed image model. It also provides a sequentially optimal mechanism to implement this transform. In comparison, ST-KLT is sub-optimal in theory.

Besides the theoretical advantages, Bi-2DPCA is also computationally more efficient than ST-KLT. The reason is that the construction of $\mathbf{H}_t$ needs less computation than the

construction of $\mathbf{R}_v$, since the image **B** (i.e., the feature matrix after 2DPCA compression on **A**) is much smaller than the original image **A.**

6. Experiments and Analysis

6.1 Experiments using the AT&T Database

The AT&T database contains images from 40 individuals, each providing 10 different images. For some subjects, the images were taken at different times. The facial expressions (open or closed eyes, smiling or non-smiling) and facial details (glasses or no glasses) also vary. The images were taken with a tolerance for some tilting and rotation of the face of up to 20 degrees. Moreover, there is also some variation in the scale of up to about 10%. All images are grayscale and normalized to a resolution of 92×112 pixels.

6.1.1 Image Reconstruction Analysis

In this section, we will examine the mechanism of Bi-2DPCA based image reconstruction. The first five images of each individual are drawn from AT&T database to form a training sample set. Thus, the training sample size is 200. Based on these training samples, let us form the image horizontal covariance matrix $\mathbf{G}_t$ and calculate its eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_q$ corresponding to q largest eigenvalues. Then, we construct the image vertical covariance matrix $\mathbf{H}_t$ and get its eigenvectors $\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_p$ corresponding to p largest eigenvalues. Letting $\boldsymbol{\varphi}_{ij} = \mathbf{v}_i \mathbf{u}_j^T$ ($i = 1, \cdots, 9$; $j = 1, \cdots, 10$), we obtain 90 basis images, which are shown in Figure 2(a). As a comparison, 90 principal basis images of 2D-DCT are shown in Figure 2(b).

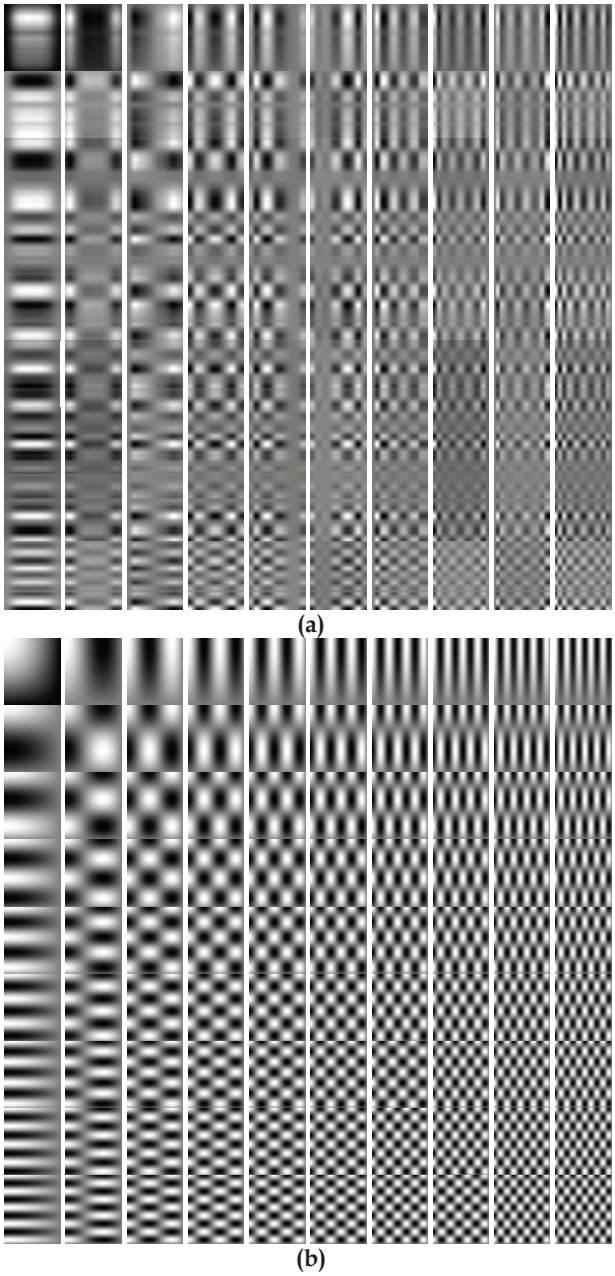


Fig. 2. Examples of basis images of Bi-2DPCA and 2D-DCT. (a) Basis images of Bi-2DPCA; (b) basis images of 2D-DCT, from up to down, $i = 1, \cdots, 9$; from left to right, $j = 1, \cdots, 10$.

From Figure 2, we can see that the basis images of Bi-2DPCA and 2D-DCT have some common properties. The lower-index (*i*, *j* are smaller) basis images φ_{ij} contain lower-frequency information and, the higher-index (*i*, *j* are larger) basis images φ_{ij} contain higher-frequency information. If we fix the column index *j*, the information in horizontal direction becomes more and more conspicuous with the increase of the row index *i*. Similarly, if we fix the row index *i*, the information in vertical direction becomes more and more conspicuous with the increase of the column index *j*. Besides, we can also see some remarkable differences between them. The basis images of Bi-2DPCA appear to be more face-image-data dependent, while those of 2D-DCT not. This is because that the basis images of Bi-2DPCA are obtained by training based on the given face images, while those of 2D-DCT are generated by a statistical model. It can be seen from Figure 2 that the first basis image φ_{11} of Bi-2DPCA is a prototype of a face, which shows some inherent information of face images. In contrast, the basis images of 2D-DCT do not embody any face-related information.

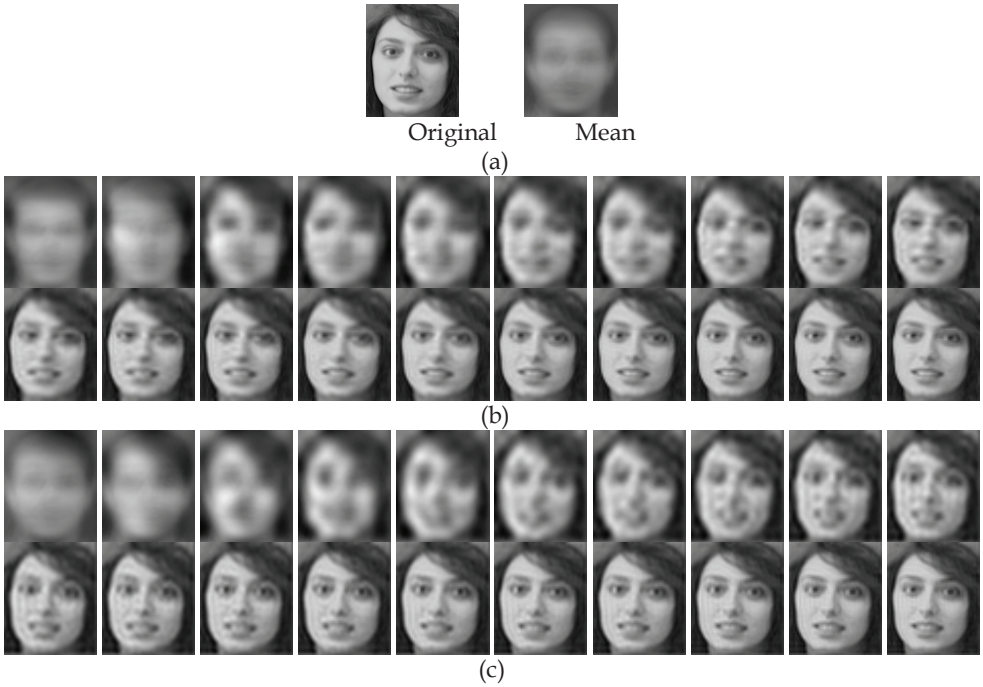


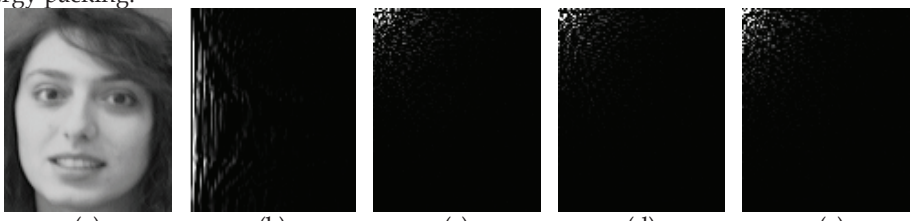
Fig. 3. An original image and its reconstructed images based on Bi-2DPCA and 2D-DCT. (a) An original image and the mean image; (b) the reconstructed images based on Bi-2DPCA; (c) the reconstructed images based on 2D-DCT, when $p, q = 2,4,\cdots,40$ (from left to right, up to down).

Based on the basis images, Bi-2DPCA can realize the reconstruction of a given image using Eq. (33). The reconstructed image can be expressed as a superposition of a small fraction of

basis images weighted by the corresponding transform coefficients. Figure 3(b) shows a series of Bi-2DPCA based reconstructed images of the original image in Figure 3(a) when p and q ($p = q$) vary from 2 to 40 with an interval of 2. In contrast, Figure 3(c) shows a series of 2D-DCT based reconstructed images as p and q ($p = q$) vary in the same way. It is obvious that the reconstructed images become clearer when more basis images are involved in the superposition. This is because the higher-index (higher-frequency) basis images $\boldsymbol{\varphi}_{ij}$ contain more detailed image information. It also can be seen that for each p and q, Bi-2DPCA based reconstructed image is always better than 2D-DCT based reconstructed images. This is because Bi-2DPCA transform keeps more image energy and less reconstruction loss than 2D-DCT. We will discuss this in detail in the following subsection.

6.1.2 Image Energy and Representation Error Analysis

Let us work out the horizontal covariance matrix $\mathbf{G}_t$'s all eigenvectors $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_{92}$ and use them to form a complete 2DPCA transform. After this transform, the original image in Figure 4 (a) is transformed into the image in Figure 4(b). Figure 4(b) shows that the image energy is compacted into a small number of columns. If we select the first $q = 10$ columns of the transformed image and compress them further using a second complete 2DPCA (the transform is determined by all eigenvectors of $\mathbf{H}_t$), we obtain an image in Figure 4 (c). Obviously, the energy of the first 10 columns is re-compacted into the up-left corner of the image. This indicates Bi-2DPCA based transform does have good energy packing property. Figure 4 (d) and (e) shows that ST-KLT and 2D-DCT based transforms are also effective for energy packing.



(a) (b) (c) (d) (e)

Fig. 4. Illustration of energy packing property of 2DPCA, Bi-2DPCA, ST-KLT and 2D-DCT. (a) An original image, (b) 2DPCA transformed image, (c) Bi-2DPCA transformed image (q=10), (d) ST-KLT transformed image, (e) 2D-DCT transformed image

To gain more insights into the energy packing performance of Bi-2DPCA, ST-KLT and 2D-DCT, let us analyze their mean square errors (MSEs) in image reconstruction. If we use a $p \times q$ feature matrix $\mathbf{C}$ to represent an image, the MSE of Bi-2DPCA based image reconstruction can be evaluated using the tail eigenvalues of $\mathbf{G}_t$ and $\mathbf{H}_t$ according to Theorem 2. Since there is no similar property with ST-KLT and 2D-DCT, we have to employ the definition of mean square errors, Equation (35), to calculate the MSEs of ST-KLT and 2D-DCT. The MSEs of Bi-2DPCA, ST-KLT and 2D-DCT when p and q ($p = q$) vary from 1 to 10 are shown in Table 2. Table 2 indicates the MSE of Bi-2DPCA based image representation is smaller than that of ST-KLT based. This is consistent with our theoretical analysis in Section

5.3. Also, the MSEs of the two image-data *dependent* coding methods, Bi-2DPCA and ST-KLT, are much less than that of the image-data *independent* method 2D-DCT.

Does the MSE have an impact on the recognition performance? To answer this question, let q and p ($p = q$) vary from 1 to 10. In each case, we test Bi-2DPCA, ST-KLT and 2D-DCT and list their recognition rates in Table 3. Table 3 shows, on the whole, the performance difference between Bi-2DPCA and ST-KLT is not as significant as the performance difference between Bi-2DPCA and 2D-DCT. From Table 2, we know that the MSE difference between Bi-2DPCA and ST-KLT is not as significant as the MSE difference between Bi-2DPCA and 2D-DCT. So, we can conclude that the significant MSE difference between two methods does affect their recognition rates and, the image-data *dependent* methods are more suitable for representing faces for recognition purpose. The insignificant MSE difference, however, almost has no effect on the recognition results. The recognition performances of Bi-2DPCA and ST-KLT are very close when q and p are over 3. In practice, since we always choose a larger q and p for achieving the best recognition rate, the MSE difference between ST-KLT and Bi-2DPCA has no substantial effect on their performance.

p, q	1	2	3	4	5	6	7	8	9	10
Bi-2DPCA	207.13	170.42	136.21	118.05	103.93	90.77	79.19	70.48	63.26	57.78
ST-KLT	211.69	171.03	136.89	121.86	104.84	91.06	79.39	70.56	63.35	57.96
2D-DCT	216.17	181.20	159.91	145.15	126.42	107.60	90.75	80.86	73.42	66.69

Table 2. Mean Square errors of Bi-2DPCA, ST-KLT and 2D-DCT based image representation

p, q	1	2	3	4	5	6	7	8	9	10
Bi-2DPCA	13.5	66.5	90.5	93.5	95.5	95.0	95.5	95.5	95.5	96.0
ST-KLT	12.0	64.0	90.0	93.0	95.5	95.0	95.5	95.5	95.5	96.0
2D-DCT	13.5	57.0	87.0	89.5	94.0	93.5	93.5	94.5	94.5	93.5

Table 3. Recognition rates (%) of Bi-2DPCA, ST-KLT and 2D-DCT with the variation of p and q

6.2 Experiments using the FERET Database

The FERET database is a result of the FERET program, which was sponsored by the Department of Defense through the DARPA Program [39, 40]. It has become a standard database to test and evaluate state-of-the-art face recognition algorithms.

In the FERET 1996 standard subset, the basic gallery contains 1,196 face images. There are four sets of probe images compared to this gallery: the *fafb* probe set contains 1,195 images of subjects taken at the same time as the gallery images but with different facial expression; the *fafc* probe set contains 194 images of subjects under significantly different lighting conditions; the *Duplicate I* probe set contains 722 images of subjects taken between one minute and 1,031 days after the gallery image was taken; the *Duplicate II* probe set is a subset of the *duplicate I* set, containing 234 images taken at least 18 months after the gallery images. In our experiments, the face portion of each original image is automatically cropped based on the location of eyes and resized to an image of 80×80 pixels. The resulting image is then

pre-processed by a histogram equalization algorithm. Some example images after pre-processing are shown in Figure 5.

Fig. 5. Some example images of cropped images that were pro-processed by histogram equalization.

6.2.1 Performance Comparison Analysis

For consistency with other studies [23, 39], in our test, 500 images are selected from the gallery to form the training sample set. In order to reduce the effect that might be induced by the choice of the training sample set, we run the system ten times. In each time, the training sample set (containing 500 images) is randomly selected from the gallery so that the training sample sets are different for ten tests. PCA, 2DPCA and Bi-2DPCA are, respectively, employed for image representation. For PCA, 200 principal components are extracted to represent a face (this is consistent with the PCA-based baseline system in [39]). For 2DPCA, 10 principal component vectors (containing 800 features) are extracted for image representation. While for Bi-2DPCA, a 15×15 feature matrix is first obtained after image transform and then is converted into a 225-dimensional feature vector by stacking the columns in turn. Note that in our test, only the first 200 features are used by Bi-2DPCA for representation and classification purpose in accordance with the dimension of PCA. Finally, a nearest-neighbor classifier with three common distance metrics is employed for classification. These distance metrics includes: L2 (Euclidean) distance, L1 (city-block) distance, and cosine distance [23, 25]. For each method and each probe set, the average recognition rate and standard deviation (std) across ten tests with three distance metrics are listed in Tables 4-6. Taking the four probe sets as a whole testing set, the total recognition rate of each method is also calculated and listed in these tables.

Method	*fafb* (1195)	*fafc* (194)	*Duplicate I* (722)	*Duplicate II* (234)	Total (2345)
PCA	77.18 ± 0.38	14.84 ± 1.30	32.06 ± 0.43	10.15 ± 0.61	51.442
2DPCA	*79.93* ± 0.29	*19.35* ± 0.49	*34.90* ± 0.18	*11.51* ± 0.21	*54.227*
Bi-2DPCA	79.15 ± 0.08	16.45 ± 0.16	33.24 ± 0.12	11.42 ± 0.17	53.069

Table 4. Recognition rate (%) of PCA, 2DPCA and Bi-2DPCA with L2 distance metric

Method	*fafb* (1195)	*fafc* (194)	*Duplicate I* (722)	*Duplicate II* (234)	Total (2345)
PCA	76.49 ± 0.52	*38.42* ± 1.54	33.89 ± 0.79	*13.03* ± 1.55	53.892
2DPCA	*81.04* ± 0.22	13.30 ± 0.46	*35.90* ± 0.37	12.92 ± 0.27	54.740
Bi-2DPCA	79.65 ± 0.40	27.89 ± 1.70	35.41 ± 0.35	12.80 ± 0.53	*55.076*

Table 5. Recognition rate (%) of PCA, 2DPCA and Bi-2DPCA with L1 distance metric

Method	*fafb* (1195)	*fafc* (194)	*Duplicate I* (722)	*Duplicate II* (234)	Total (2345)
PCA	76.67 ± 0.42	11.06 ± 0.45	*33.80* ± 0.44	*12.81* ± 0.48	51.671
2DPCA	72.57 ± 0.37	16.03 ± 4.48	32.09 ± 0.66	10.13 ± 1.03	49.198
Bi-2DPCA	*79.10* ± 0.06	*16.50* ± 0.00	33.24 ± 0.08	11.50 ± 0.00	*53.056*

Table 6. Recognition rate (%) of PCA, 2DPCA and Bi-2DPCA with cosine distance metric

From Tables 4-6, we can draw the following conclusions. **1)** While the L2 distance metric is used, Bi-2DPCA is better than PCA for all probe sets with respect to the recognition accuracy and standard deviation. While the L1 and cosine distance metrics are employed, the recognition results are in relation to probe sets. For some probe sets, Bi-2DPCA performs better and for others, PCA perform better. However, as far as the total recognition rate is concerned, Bi-2DPCA is consistently superior to PCA, no matter what metric is used. **2)** 2DPCA outperforms Bi-2DPCA when the L2 metric is used, but its total recognition rate is worse than Bi-2DPCA with other two metrics. **3)** Every method achieves its best performance when the L1 distance metric is used. With this metric, Bi-2DPCA outperforms PCA and 2DPCA with respect to the total recognition rate.
In fact, the advantage of Bi-2DPCA over PCA is not only on its recognition accuracy, but also on its computational efficiency. In the next subsection, we will demonstrate that Bi-2DPCA is faster than PCA for face recognition system.

6.2.2 Computational Efficiency Analysis

In our experiments, we use Matlab for coding and the Matlab function "*eigs*" to calculate the eigenvectors in the implementation of PCA, 2DPCA and Bi-2DPCA. For each method, the average CPU time for training and testing (with L2 metric) across 10 random tests are listed in Table 8. It is apparent that Bi-2DPCA is much faster than PCA either for training or for testing. Although 2DPCA needs less time than Bi-2DPCA for training, it requires more time for the whole process: training and testing. To gain more insights into this, we will provide a detailed analysis on computation and memory requirements of PCA, 2DPCA and Bi-2DPCA based face recognition systems.

Method	Time for Training	Time for Testing (2345 samples)	Total
PCA	291.70	940.39	1232.09
2DPCA	97.38	912.60	1009.98
Bi-2DPCA	140.86	659.07	799.93

Table 8. The CPU time (s) for training and testing on FERET 1996 subset (CPU: Pentium IV 2.4GHz, RAM: 512Mb)

Method	Constructing covariance matrices (C_1)	Solving eigen-problems (C_2)	Transformation of an image (C_3)	Calculating distance (C_4)
PCA	$500^2 \times 80^2$	500^3	$80^2 \times 200$	$C_4 = 1196 \times 200$

	$=\mathbf{1.6\times10^9}$	$=\mathbf{1.25\times10^8}$	$=\mathbf{1.28\times10^6}$	$=\mathbf{2.392\times10^5}$
2DPCA	500×80^3 $=\mathbf{2.56\times10^8}$	80^3 $=\mathbf{5.12\times10^5}$	$80^2\times10$ $=\mathbf{6.4\times10^4}$	1196×800 $=\mathbf{9.568\times10^5}$
Bi-2DPCA	$500\times(80^3+80^2\times15)$ $=\mathbf{3.04\times10^8}$	$80^3\times2$ $=\mathbf{1.024\times10^6}$	$80^2\times15+80\times15\times14$ $=\mathbf{1.128\times10^5}$	1196×200 $=\mathbf{2.392\times10^5}$

Table 9. Calculation items of PCA, 2DPCA and Bi-2DPCA in training and testing process

Method	Training $C_1+C_2+1196\times C_3$	Testing $(C_3+C_4)\times2345$	Total
PCA	$\mathbf{3.2559\times10^9}$	$3.5625\times\mathbf{10^9}$	$6.8184\times\mathbf{10^9}$
2DPCA	$3.3306\times\mathbf{10^8}$	$2.3938\times\mathbf{10^9}$	$2.7269\times\mathbf{10^9}$
Bi-2DPCA	$4.3993\times\mathbf{10^8}$	$8.2544\times\mathbf{10^8}$	$1.2654\times\mathbf{10^9}$

Table 10. Comparisons of computation requirements of PCA, 2DPCA and Bi-2DPCA

Method	Projector Size	Gallery Size	Total
PCA	$80^2\times200$ **= 1,280,000**	1196×200 = **239,200**	***1,519,200***
2DPCA	80×10 = **800**	$1196\times80\times10$ = **956,800**	***957,600***
Bi-2DPCA	$80\times15+80\times14$ = **2,320**	1196×200 = **239,200**	***241,520***

Table 11. Comparisons of memory requirements of PCA, 2DPCA and Bi-2DPCA

The computation requirements can be measured by the amount of multiplications involved in the training and testing processes. The training process includes the following calculations: (C_1) constructing covariance matrices, (C_2) learning the projector (transform matrix) by solving eigen-problems and, (C_3) transformation of an image in gallery. And, the testing process includes: (C_3) transformation of an image in probe sets and, (C_4) calculating the distances between a probe and gallery for classification. The computations involved in these items are listed in Table 9. Table 10 exhibits the computation requirements in the training, testing and the whole process.

From Tables 9 and 10, we can see that Bi-2DPCA needs much less computations than PCA in the first three items and the same computations for item C_4. So, Bi-2DPCA is computationally more efficient than PCA either for training or for testing. 2DPCA consumes less computations than Bi-2DPCA in the first three items (so that it is faster than Bi-2DPCA for training), because it deals with one covariance matrix while Bi-2DPCA deals with two. However, 2DPCA spends much more computations on item C_4 since it requires four times of features than Bi-2DPCA (or PCA) for image representation. This leads to a larger amount of computations in its testing process. In a word, Bi-2DPCA has the least computation requirements among three methods for the whole process: training and testing.

The memory requirements of a face recognition system depend on the size of projector and the total size of data in gallery. Table 11 shows these items corresponding to each method. Bi-2DPCA has the least total memory requirements among three methods. PCA has the largest total memory requirements because its projector is very large, which amounts to a total size of 200 original face images. In contrast, the projector of Bi-2DPCA or 2DPCA is much smaller; its size is less than the size of one original image. Since 2DPCA has a much

lower compression rate than Bi-2DPCA, it requires more memory to save the feature vectors in gallery.
The above comparison demonstrates that Bi-2DPCA based image recognition system has advantages over PCA or 2DPCA based system on computation and memory requirements. This characteristic makes Bi-2DPCA to be a fast tool for face coding and recognition.

7. Conclusions

In this paper, two-dimensional principal component analysis (2DPCA) is first re-examined and its two properties are revealed. 2DPCA can eliminate the correlation between column vectors of image and compact the image energy onto a small number of column vectors (these vectors are used for image representation). In other words, 2DPCA realizes an optimal compression in horizontal direction. These properties are desirable and provide some theoretical supports for 2DPCA-based image representation. However, 2DPCA does not consider the correlation in vertical direction. This leads to a relative lower compression rate compared to PCA.
Bi-2DPCA technique is developed to overcome the weakness of 2DPCA. Basically, Bi-2DPCA is to perform 2DPCA twice sequentially, i.e., a first compression in horizontal direction followed by a second one in vertical direction. In this way, the correlations in both directions are eliminated and, the image energy is compacted into the up-left corner of image. The elements in this corner are chosen as features. So, Bi-2DPCA needs fewer coefficients than 2DPCA for image representation. This results in lower storage requirements and a remarkable speedup in classification. Actually, Bi-2DPCA based representation is not only economical in storage but also effective for discrimination. Our experiments on FERET database show Bi-2DPCA is comparable with 2DPCA.
In addition, the theoretical justification for Bi-2DPCA based image representation is provided. This representation mechanism is sequentially optimal in the sense of minimal mean-square error. In comparison, ST-KLT lacks this justification and is shown to be sub-optimal in theory. That is, the mean-square error (MSE) of ST-KLT is always larger than that of Bi-2DPCA. Besides, we also show that the MSEs of the image-data dependent coding methods such as Bi-2DPCA and ST-KLT are much less than the image-data independent method like 2D-DCT. Our experiments indicate that the significant MSE difference between two methods does affect their recognition performances and, the image-data dependent methods are more suitable for representing faces for recognition purpose. The insignificant MSE difference, however, almost has no effect on the recognition results.
In contrast to PCA, the most prominent advantage of Bi-2DPCA is its low computational complexity. Actually, Bi-2DPCA has lower computation requirement than PCA on almost all aspects involved, including the construction of covariance matrices, calculating the eigenvectors of these covariance matrices, and image transformation. This characteristic makes Bi-2DPCA faster than PCA in both training and testing processes. It should be mentioned that the speed advantage of Bi-2DPCA would become more remarkable with the increase of database scale and training sample size. Besides, our experiments on the FERET database also demonstrate that Bi-2DPCA is comparable with PCA in recognition performance.

8. Acknowledgements

This work was partially supported by the National Science Foundation of China under Grants No. 60632050, 60973098 and 60602038, the NUST Outstanding Scholar Supporting Program and the Program for New Century Excellent Talents in University of China.

Appendix A: The proof Theorem 2

To prove Theorem 2, a useful Lemma is first given:

Lemma A1 [41] If $\mathbf{A} \in \mathbb{R}^{m \times n}$, then $\| \mathbf{A} \|_F^2 = \mathrm{tr}(\mathbf{A}^T \mathbf{A}) = \mathrm{tr}(\mathbf{A}\,\mathbf{A}^T)$.

Proof of Theorem 2:

Let us denote $\mathbf{U} = (\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_q)$ and $\mathbf{V} = (\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_p)$. Then, $\mathbf{U}^T \mathbf{U} = \mathbf{I}_q$ and $\mathbf{V}^T \mathbf{V} = \mathbf{I}_p$.

The Bi-2DPCA based compression process contains two sequential operations, i.e. column-based (horizontal) compression $\mathbf{B} = \mathbf{A}\mathbf{U}$ and row-based (vertical) compression $\mathbf{C} = \mathbf{V}^T \mathbf{B}$.

Correspondingly, the column- and row-based reconstructions can be respectively represented by

$$\hat{\mathbf{A}} = \mathbf{B}\mathbf{U}^T, \tag{A.1}$$

$$\hat{\mathbf{B}} = \mathbf{V}\mathbf{C}. \tag{A.2}$$

From Theorem 1, we know their reconstruction MSEs are

$$\varepsilon_{\mathrm{v}}^2 = \mathrm{E} \| \mathbf{A} - \hat{\mathbf{A}} \|_F^2 = \sum_{j=q+1}^{n} \lambda_j \tag{A.3}$$

$$\varepsilon_{\mathrm{h}}^2 = \mathrm{E} \| \mathbf{B} - \hat{\mathbf{B}} \|_F^2 = \sum_{j=p+1}^{m} \mu_j \tag{A.4}$$

The Bi-2DPCA based reconstruction image of $\mathbf{A}$ is $\widetilde{\mathbf{A}} = \mathbf{V}\mathbf{C}\mathbf{U}^T = \hat{\mathbf{B}}\mathbf{U}^T$. The total reconstruction MSE is $\varepsilon_{\mathrm{t}}^2 = \mathrm{E} \| \mathbf{A} - \widetilde{\mathbf{A}} \|_F^2$. We will prove that $\varepsilon_{\mathrm{t}}^2 = \varepsilon_{\mathrm{v}}^2 + \varepsilon_{\mathrm{h}}^2$ as follows.

From Lemma A1, we have $\varepsilon_{\mathrm{t}}^2 = \mathrm{E} \| \mathbf{A} - \widetilde{\mathbf{A}} \|_F^2 = \mathrm{E}\{\mathrm{tr}[(\mathbf{A} - \widetilde{\mathbf{A}})(\mathbf{A} - \widetilde{\mathbf{A}})^T]\}$, where

$$\begin{aligned}
(\mathbf{A} - \widetilde{\mathbf{A}})(\mathbf{A} - \widetilde{\mathbf{A}})^T &= (\mathbf{A} - \hat{\mathbf{A}} + \hat{\mathbf{A}} - \widetilde{\mathbf{A}})(\mathbf{A} - \hat{\mathbf{A}} + \hat{\mathbf{A}} - \widetilde{\mathbf{A}})^T \\
&= [(\mathbf{A} - \mathbf{B}\mathbf{U}^T) + (\mathbf{B}\mathbf{U}^T - \hat{\mathbf{B}}\mathbf{U}^T)][(\mathbf{A} - \mathbf{B}\mathbf{U}^T) + (\mathbf{B}\mathbf{U}^T - \hat{\mathbf{B}}\mathbf{U}^T)]^T \\
&= (\mathbf{A} - \mathbf{B}\mathbf{U}^T)(\mathbf{A} - \mathbf{B}\mathbf{U}^T)^T + [(\mathbf{B} - \hat{\mathbf{B}})\mathbf{U}^T][(\mathbf{B} - \hat{\mathbf{B}})\mathbf{U}^T]^T \\
&\quad + (\mathbf{A} - \mathbf{B}\mathbf{U}^T)[(\mathbf{B} - \hat{\mathbf{B}})\mathbf{U}^T]^T + [(\mathbf{B} - \hat{\mathbf{B}})\mathbf{U}^T](\mathbf{A} - \mathbf{B}\mathbf{U}^T)^T.
\end{aligned}$$

Since $\mathbf{U}^T \mathbf{U} = \mathbf{I}_q$, the sum of the first two terms equals to

$$(\mathbf{A} - \hat{\mathbf{A}})(\mathbf{A} - \hat{\mathbf{A}})^T + (\mathbf{B} - \hat{\mathbf{B}})(\mathbf{B} - \hat{\mathbf{B}})^T$$

Since $\mathbf{U}^T\mathbf{U} = \mathbf{I}_q$ and $\mathbf{B} = \mathbf{AU}$, the sum of the last two terms equals to

$$(\mathbf{A} - \mathbf{B}\mathbf{U}^T)\mathbf{U}(\mathbf{B} - \hat{\mathbf{B}})^T + (\mathbf{B} - \hat{\mathbf{B}})\mathbf{U}^T(\mathbf{A}^T - \mathbf{U}\mathbf{B}^T) =$$

$$(\mathbf{AU} - \mathbf{B})(\mathbf{B} - \hat{\mathbf{B}})^T + (\mathbf{B} - \hat{\mathbf{B}})\mathbf{U}^T(\mathbf{AU} - \mathbf{B})^T = 0.$$

So, $(\mathbf{A} - \widetilde{\mathbf{A}})(\mathbf{A} - \widetilde{\mathbf{A}})^T = (\mathbf{A} - \hat{\mathbf{A}})(\mathbf{A} - \hat{\mathbf{A}})^T + (\mathbf{B} - \hat{\mathbf{B}})(\mathbf{B} - \hat{\mathbf{B}})^T$.

Thus, $\varepsilon_t^2 = \mathrm{E}\{\mathrm{tr}[(\mathbf{A} - \widetilde{\mathbf{A}})(\mathbf{A} - \widetilde{\mathbf{A}})^T]\}$

$$= \mathrm{E}\{\mathrm{tr}(\mathbf{A} - \hat{\mathbf{A}})(\mathbf{A} - \hat{\mathbf{A}})^T\} + \mathrm{E}\{\mathrm{tr}(\mathbf{B} - \hat{\mathbf{B}})(\mathbf{B} - \hat{\mathbf{B}})^T\}$$

$$= \mathrm{E}\,\|\,\mathbf{A} - \hat{\mathbf{A}}\,\|_F^2 + \mathrm{E}\,\|\,\mathbf{B} - \hat{\mathbf{B}}\,\|_F^2$$

$$= \varepsilon_v^2 + \varepsilon_h^2 = \sum_{j=q+1}^{n} \lambda_j + \sum_{j=p+1}^{m} \mu_j .$$

9. References

[1] W. Zhao, R. Chellappa, A. Rosenfeld, and P. Phillips, "Face recognition: A literature survey." Technical Report CAR-TR-948, UMD CS-TR-4167R, August, 2002.

[2] L. Sirovich and M. Kirby, "Low-dimensional procedure for characterization of human faces", J. Optical Soc. Of Am., 1987, vol.4, pp. 519-524.

[3] M. Kirby and L. Sirovich, "Application of the KL procedure for the characterization of human faces", IEEE Trans. Pattern Anal. Machine Intell., 1990,12(1), pp. 103-108.

[4] M. Turk and A. Pentland, "Eigenfaces for recognition", J. Cognitive Neuroscience, 1991, 3(1), pp. 71-86.

[5] M. Turk and A. Pentland, "Face recognition using Eigenfaces", Proc. IEEE Conf. On Computer Vision and Pattern Recognition, 1991, pp. 586-591.

[6] A. Pentland, B. Moghaddam and T. Starner, "View-based and mododular eigenspaces for face recognition", Proc. IEEE Conf. On Computer Vision and Pattern Recognition, 1994, pp. 84-91.

[7] A. Pentland, "Looking at people: sensing for ubiquitous and wearable computing", IEEE Trans. Pattern Anal. Machine Intell., 2000, 22(1), pp. 107-119.

[8] M.A. Grudin, "On internal representations in face recognition systems", Pattern Recognition, 2000, 33 (7), pp. 1161-1177.

[9] P.S. Penev and L. Sirovich, "The global dimensionality of face space", Proceedings of Fourth IEEE International Conference on Automatic Face and Gesture Recognition, 2000, pp. 264-270.

[10] T. Shakunaga and K. Shigenari, "Decomposed eigenface for face recognition under various lighting conditions", Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2001, Vol. 1, pp. 864-871.

[11] L. Zhao and Y. Yang, "Theoretical analysis of illumination in PCA-based vision systems", Pattern Recognition, 1999, 32(4), pp. 547-564.

[12] B. Moghaddam, "Principal manifolds and probabilistic subspaces for visual recognition", IEEE Trans. Pattern Anal. Machine Intell., 2002, June, 24(6), pp.780 – 788.

[13] J. Wu and Z.H. Zhou, "Face recognition with one training image per person", Pattern Recognition Letter, 2002, 23(14), 1711-1719.
[14] X. Chen, P.J. Flynn, and K.W. Bowyer, "PCA-based face recognition in infrared imagery: baseline and comparative studies", IEEE International Workshop on Analysis and Modeling of Faces and Gestures, Oct. 2003, pp. 127-134.
[15] H.C. Kim, D. Kim, S.Y. Bang and S.Y. Lee, Face recognition using the second-order mixture-of-eigenfaces method, Pattern Recognition, 2004, 37(2), pp. 337-349.
[16] J. Weng, Y. Zhang and W.-S. Hwang, Candid covariance-free incremental principal component analysis, IEEE Trans. Pattern Anal. Machine Intell., 2003, 25 (8), pp.1034-1040.
[17] A. Hyvärinen, J. Karhunen, E. Oja, Independent Component Analysis, Wiley, New York, 2001.
[18] B. Scholkopf, A. Smola, and K. R. Muller, "Nonlinear component analysis as a kernel eigenvalue problem", Neural Computation, 1998, 10(5), pp. 1299-1319.
[19] Bernhard Schölkopf and Alex Smola. Learning with kernels, MIT Press, Cambridge, MA, 2002.
[20] M.S. Bartlett, J.R. Movellan and T.J. Sejnowski, "Face recognition by independent component analysis", IEEE Trans. Neural Networks, 2002, 13(6), pp. 1450-1464.
[21] Pong C. Yuen, and J.H. Lai, Face representation using independent component analysis, Pattern Recognition, 2002, 35 (6), 1247-1257.
[22] C. Liu and H. Wechsler: "Independent component analysis of Gabor features for face recognition", IEEE Trans. Neural Networks, 2003, 14 (4), pp. 919-928.
[23] B.A. Draper, K. Baek, M.S. Bartlett, J.R. Beveridge, "Recognizing faces with PCA and ICA", Computer vision and image understanding, Special issue on Face Recognition, 2003, July, 91(1/2), 115-137.
[24] M. H. Yang, "Kernel Eigenfaces vs. kernel Fisherfaces: face recognition using kernel methods", Proceedings of the Fifth IEEE International Conference on Automatic Face and Gesture Recognition (RGR'02).
[25] C. Liu, "Gabor-based kernel PCA with fractional power polynomial models for face recognition", IEEE Transaction on Pattern Analysis and Machine Intelligence, 2004, 26(5), pp. 572-581.
[26] J. Yang, J. Y. Yang, "From image vector to matrix: a straightforward image projection technique — IMPCA vs. PCA", Pattern Recognition, 2002, 35(9), pp. 1997-1999.
[27] J. Yang, D. Zhang, A. F. Frangi, J.-Y. Yang, "Two-Dimensional PCA: a New Approach to Face Representation and Recognition", IEEE Transaction on Pattern Analysis and Machine Intelligence, 2004, 26(1), 131-137.
[28] N. S. Jayant and P. Noll, Digital Coding of Waveforms: Principles and Applications to Speech and Video. Prentice-Hall, Englewood Cliffs, New Jersey, 1984.
[29] A. Habibi and P.A. Wintz, "Image coding by linear transformation and block quantization", IEEE Trans. Communication Technology, 1971, 19 (1), pp. 50-62.
[30] A. K. Jain, Fundamentals of digital image processing, Prentice-Hall, Englewood Cliffs, New Jersey, 1989.
[31] S. Olmos, J.P. Martínez and L. Sörnmo, "Spatio-temporal linear expansions for repolarization analysis", Computers in Cardiology. IEEE Computer Society Press, Memphis, USA, September, 2002, pp. 689-692.

[32] H. Karhunen, "Uber lineare methoden in der Wahrscheinlich-Keitsrechnung", Ann. Acad. Science Fenn, Ser. A.I. 37, Helsmki, 1947.
[33] M. Loeve, "Fonctions aleatoires de seconde ordre", in P. Levy, Processus Stochastiques et Mouvement Brownien. Paris, France: Hermann, 1948.
[34] H. Hotelling, "Analysis of a complex of statistical variables into principle components", J. educ. Psychology, 1933, 24, pp. 417-441 and 498-520.
[35] Fukunaga K. Introduction to Statistical Pattern Recognition (2nd ed), Academic Press, Boston, 1990.
[36] K. Liu, Y.Q. Cheng, J.Y. Yang, et al., "Algebraic feature extraction for image recognition based on an optimal discriminant criterion", Pattern Recognition, 1993, 26 (6), 903-911.
[37] J. Yang, J.-Y. Yang, A.F. Frangi, and D. Zhang, "Uncorrelated Projection Discriminant Analysis and Its Application to Face Image Feature Extraction", International Journal of Pattern Recognition and Artificial Intelligence, 2003, 17(8), 1325-1347.
[38] D. Zhang, X. Jing, J. Yang, Biometric Image Discrimination Technologies, Idea Group Publishing, Hershey, USA, 2006
[39] P. J. Phillips, H. Moon, S. A. Rizvi, and P. J. Rauss, "The FERET Evaluation Methodology for Face-Recognition Algorithms," IEEE Transactions on Pattern Analysis and Machine Intelligence, 2000, 22 (10), pp.1090-1104.
[40] P. J. Phillips, The Facial Recognition Technology (FERET) Database, http://www.itl.nist.gov/iad/humanid/feret
[41] G. H. Golub and C. F. Van Loan, Matrix Computations, The Johns Hopkins University Press, Baltimore and London, Third edition, 1996.