INCREMENTAL TWO-DIMENSIONAL TWO-DIRECTIONAL PRINCIPAL COMPONENT ANALYSIS (I(2D)²PCA) FOR FACE RECOGNITION

Yonghwa Choi¹, Takaomi Tokumoto², Minho Lee¹, and Setichi Ozawa²

¹ School of Electronics Engineering, Kyungpook National University, Korea
² Graduate School of Engineering, Kobe University, Japan

ABSTRACT

In this paper, we propose a new incremental two-directional two-dimensional principal component analysis (I(2D)²PCA) to efficiently recognize human faces. For implementing a real time face recognition system in an embedded system, the reduction of computational load as well as memory of a feature extraction algorithm is very important issue. The (2D)²PCA is faster than the conventional PCA. From memory capacity point of view, the incremental PCA is very efficient algorithm by adapting the eigenspace only using a new incoming sample data without memorizing all of previous trained data. In order to construct an efficient algorithm with less memory and small computational load, we propose a new feature extraction method by combining the IPCA and the (2D)²PCA. To evaluate the performance of the proposed I(2D)²PCA, a series of experiments were performed on two face image databases: ORL and Yale face databases. The experimental results show that the proposed feature extraction method is efficient by reducing the memory while computational load is nearly similar to (2D)²PCA.

Index Terms— Principal Component Analysis (PCA), Incremental two-directional two-dimensional principal component analysis (I(2D)²PCA), Face recognition, Feature extraction

1. INTRODUCTION

The principal component analysis (PCA) is a well-known feature extraction and data representation technique, which is widely used in the areas of pattern recognition and computer vision. Sirovich and Kirby [1][2] firstly used the PCA to efficiently represent the human faces. The 2D face image matrices were previously transformed into 1D image vectors column by column or row by row. However, concatenating 2D matrices into 1D vector often leads to a high-dimensional vector space which induces a difficulty in evaluating the covariance matrix accurately due to its larger size and the relatively small number of training samples [3]. Furthermore, computing of the eigenvectors with a large size covariance matrix is a very time-consuming task.

To overcome those problems, the (2D)²PCA is newly proposed by Daoqiang Zhang and Zhi-Hua Zhou [4]. The advantage of using the (2D)²PCA is so fast because covariance matrix size is very smaller than the traditional PCA. It was reported in [4] that the recognition accuracy on several face databases was higher using 2DPCA than PCA, and the extraction of image features is computationally more efficient using 2DPCA than PCA.

On the other hand, most of the conventional PCA is a kind of batch type learning, which means that all of training samples should be prepared. Also, it is not easy to adapt a feature space for time varying and/or unseen data. If we need to add a new sample, the conventional PCA needs to keep whole data to update the eigenvector. Resultantly, that causes to waste the memory capacity because it must have whole trained data up to now. In order to solve that problem, the incremental PCA (IPCA) is one of good candidate method, because the IPCA method can update current eigenvector using only new data and previous eigenvector without considering the whole data.

By combing the advantages of both the (2D)²PCA and IPCA, we developed a new feature extraction method, so called I(2D)²PCA with less computational load than the IPCA as well as smaller memory waste than the (2D)²PCA. In order to show the efficiency of the proposed method, we use two face image databases such as ORL and Yale face database.

The rest of this paper is organized as follows: Section 2 briefly reviews the conventional 2DPCA, (2D)²PCA and IPCA methods. Section 3 includes the proposed I(2D)²PCA method. In Section 4, the experiments on the face databases are shown to compare the performances of various methods. Finally, we describe the conclusion with further work at Section 5.

2. CONVENTIONAL 2DPCA, (2D)²PCA AND IPCA

2.1 2DPCA and (2D)²PCA

Suppose that there are N training samples, denoted by m by n matrices Aᵢ (k = 1, 2, ⋯, N). The covariance matrix is as follows:

978-1-4577-0539-7/11/$26.00 ©2011 IEEE 1493 ICASSP 2011
\[ C_{\text{row}} = \frac{1}{N} \sum_{j=1}^{N} (A_j - \bar{A})(A_j - \bar{A})^T. \]  

(1)

where the \( \bar{A} \) is a mean of sample images. 

\( U_{\text{opt}}^{\text{row}} \) is a matrix whose column vectors correspond to the eigen vectors of the above covariance. \( U_{\text{opt}}^{\text{row}} \) is composed by the orthonormal eigenvectors \( U_1^{\text{row}}, \ldots, U_d^{\text{row}}, \) of \( C_{\text{row}} \) corresponding to the \( d \) largest eigenvalues. Because of that covariance is smaller than the conventional PCA, solving eigen problem is very efficient than conventional PCA. Also, \( d \) can be selected by threshold \( \theta \) as follows:

\[
\sum_{i=1}^{d} \lambda_i \geq \theta, 
\]

(2)

where \( \lambda_1, \lambda_2, \ldots, \lambda_n \) is the \( n \) largest eigenvalues of \( C_{\text{row}} \) and \( \theta \) is a pre-set threshold [3].

The 2DPCA has usually larger coefficient than the PCA. To reduce the number of coefficients, there exits the 2DPCA which projects one more time in another direction [4]. The 2DPCA only works for row or column direction. That is, the 2DPCA learns the eigen vector from a set of training images reflecting the information between the rows of images, and then projects an \( m \) by \( n \) image \( A \) onto \( U_{\text{opt}}^{\text{row}} \), yielding an \( m \) by \( d \) matrix \( Y = (U_{\text{opt}}^{\text{row}})^T A \). Similarly, the 2DPCA learns \( U_{\text{opt}}^{\text{col}} \) reflecting the information between columns of images, and then project \( A \) onto \( U_{\text{opt}}^{\text{col}} \), yielding an \( q \) by \( n \) matrix \( Y = (U_{\text{opt}}^{\text{col}})^T A \). Finally, we can get the two-directional 2DPCA, so called ((2D)²PCA) [4].

Both the 2DPCA and the (2D)²PCA need to prepare whole of training data to get an optimal eigen space in a second order statistics point of view. But, it is difficult to adapt a feature space for time varying and/or unseen data. If we need to add a new sample, those approaches need to keep whole data to update the eigenvector, which means that large memory capacity should be prepared.

2.2 IPCA

The IPCA method proposed by Hall and Martin [5] can update current eigenvector using only new data and previous eigenvector without considering whole data, which can provide an efficient way to solve the previous problems of the 2DPCA and the (2D)²PCA.

Suppose that \( N \) training samples have been presented onto eigen space. An eigen space model consist of calculating the eigenvectors and eigenvalues from the covariance matrix of \( \bar{x} \), where \( \bar{x} \) is a mean input vector, \( U \) is a \( n \times k \) matrix whose column vectors correspond to the eigenvectors, and \( A \) is a \( k \times k \) eigenvalues matrix. Here, \( k \) is the number of dimensions of the current eigen space.

Let’s the \((N+1)\)-th training sample be \( y \). The addition of new sample may lead to the changes in both of a mean vector and covariance matrix. The mean input vector \( \bar{x} \) is easily updated as follows:

\[
\hat{x} = \frac{1}{N+1}(N\bar{x} + y) 
\]

(3)

Covariance change by a new sample affects to the eigen space model. Therefore, the new eigen space should be recalculated. The problem is how to update the new eigen space. When the eigen space is reconstructed to adapt to a new sample, we must check whether a new eigen axis should be augmented or not. If energy of a new sample can be represented by the current eigen space, the dimensional augmentation is not needed in reconstructing the eigen space. However, if it has some energy in the complementary space to the current eigen space, the dimensional augmentation cannot be avoided. The following accumulation ratio is commonly used as its criterion to identify whether a new axis should be augmented [6] :

\[
A(k) = \frac{\sum_{i=1}^{k} \lambda_i}{\sum_{i=1}^{n} \lambda_i} \geq \theta 
\]

(4)

where \( \lambda_i \) is the \( i \)-th largest eigenvalue, \( k \) and \( n \) are the numbers of dimensions of the current feature space and input space, respectively. By specifying an appropriate threshold value \( \theta \), we can determine the feature space dimensions by searching for a minimum \( k \) such that \( A(k) > \theta \) holds. We need accumulation ratio including the energy of a new sample. Then, new accumulation ratio \( A'(k) \) is the following:

\[
A'(k) = \frac{N(N+1)\sum_{i=1}^{k} \lambda_i + N\|y - \bar{x}\|^2}{N(N+1)\sum_{i=1}^{n} \lambda_i + N\|y - \bar{x}\|^2} 
\]

(5)

Using the Eq. (5), we can incrementally update the eigen space only using a new sample without keeping whole training samples.

When the accumulation ratio \( A'(k) \) is larger than a threshold value \( \theta \), the current eigen space must be expanded with a new axis. Otherwise, the number of dimensions remains the same. To find augmented eigen axis, we calculate residual vector \( h \) as follows:

\[
h = (y - \bar{x}) - U g. 
\]

(6)

\[
g = U^T (y - \bar{x}). 
\]

(7)

It has been shown that the eigenvectors and eigen values should be updated based on the solution of the following intermediate eigen problem:

\[
\begin{bmatrix}
\begin{bmatrix}
\lambda  \\
0
\end{bmatrix}
\begin{bmatrix}
0  \\
\gamma
\end{bmatrix}
\end{bmatrix}
\begin{bmatrix}
\begin{bmatrix}
U \\
0
\end{bmatrix}
\begin{bmatrix}
0 \\
\gamma
\end{bmatrix}
\end{bmatrix}
\end{bmatrix}
R = RA', 
\]

(8)

where \( \gamma = h^T (y - \bar{x}) \), and \( R \) is a \((k+1) \times (k+1)\) matrix whose column vectors correspond to the eigenvectors obtained from the above intermediate eigen problem, \( A' \) is the new eigenvalue matrix, and \( \Theta \) is a \( k \)-dimensional zero vector. Using the solution \( R \), we can calculate the new \( n \times (k+1) \) eigenvector matrix \( U' \) as follows:

1494
\[ U' = [U, \hat{h}] R \] (9)

where

\[ \hat{h} = \begin{cases} 
\frac{h}{\|h\|} & \text{if } A'(k) < \theta \\
0 & \text{otherwise}
\end{cases} \] (10)

Here, \( \theta \) is a threshold value.

3. I(2D)\(^3\)PCA

In this section, we explain the proposed I(2D)PCA and its extending to I(2D)\(^3\)PCA methods. After batch-type training based on the conventional 2DPCA or (2D)\(^3\)PCA method using a prepared training data set, the proposed method incrementally updates an eigen space to a new sample.

3.1 I(2D)PCA

After the 2DPCA is processed, the addition of new training sample may lead to the changes in both mean and covariance. Mean is easily updated as follows:

\[ \tilde{x'} = \frac{1}{N+1}(N\tilde{x} + y). \] (11)

where \( y \) is new training sample.

Changing the covariance means that eigen vector and eigen value are also changed. For updating the eigen space, we need to check whether an augment concept is necessary or not. In order to do, we modify the Eq. (5) for accumulation ratio as follows:

\[ A'(k) = \frac{N(N+1)\sum x_i^2 + N \cdot \text{tr}([U_i^T(y-x)][U_i^T(y-x)]^T)}{N(N+1)\sum x_i^2 + N \cdot \text{tr}((y-x)(y-x)^T)} \] (12)

The square of second norm \( \|y-x\|^2 \) in Eq. (5) is changed to the \( \text{tr}((y-x)(y-x)^T) \) in Eq. (12) because the \( X \) and \( y \) are not vectors but matrices.

When the new accumulation ratio \( A'(k) \) is smaller than a threshold value \( \theta \), it must allow the number of dimensions to be increased from \( k \) to \( k+1 \), and the current eigen space must be expanded. Otherwise, the number of dimensions is same.

If an axis is augmented, each column of a new training sample matrix considers as a column vector sample, and is used to applying for the IPCA. In order to select an efficient eigen axis, we adopt the chunking concept of IPCA [7] which can select some of eigen spaces among all of possible eigen axes. In the proposed I(2D)PCA, we just augment only one new axis if the eigen space have to be augmented.

Let us consider that random image \( x \) is following:

\[ x = [x_1, \ldots, x_n] \] (13)

where \( n \) is the number of column. Using eq. (6), each column can calculate a residual vector \( h_i \).

\[ h_i = (y_i - \overline{x}) - U g_i, \] (14)

where \( g_i = U^T(y_i - \overline{x}). \) (15)

Residual vector set \( h = [h_1, \ldots, h_n] \) is a candidate for a new axis. We have to select one vector in residual vector set \( h \), using following equation:

\[ l = \arg \max A'([U, h]), \] (16)

Based on Eq. (16), we can select the most appropriate axis which maximizes the accumulation ratio in Eq. (12). Now we can find intermediate eigen problem as follows:

\[ \begin{pmatrix} N & 0 \\ 0 & N \end{pmatrix} \begin{bmatrix} \Lambda & \mathbf{0} \\ \mathbf{0} & \gamma \end{bmatrix} \begin{pmatrix} \mathbf{g} \gamma \\ \mathbf{g} \end{pmatrix} R = \Lambda \mathbf{r} \] (17)

where \( \gamma = h_i^T(y_i - \overline{x}), \mathbf{g} \) is projected matrix onto eigen vector \( U \), we can calculate the new \( n \times (k+1) \) eigenvector matrix \( U' \) as follows:

\[ U' = [U, \hat{h}] R \] (18)

where

\[ \hat{h} = \begin{cases} 
\frac{h_i}{\|h_i\|} & \text{if } A'(n) < \theta \\
0 & \text{otherwise}
\end{cases} \] (19)

3.2 I(2D)\(^3\)PCA

The I(2D)PCA only works for column direction. We can simply extend the I(2D)PCA to an I(2D)\(^3\)PCA by applying the same procedure to the row direction of training sample.

Finally, we just use a simple classifier which is nearest-neighbor to recognize a human face using the extracted features.

\[ \text{dist} = \sqrt{\sum_{i=1}^{row} \sum_{j=1}^{col} (Y_{\text{train}}(i,j) - Y_{\text{test}}(i,j))^2} \] (20)

where \( Y \) is projected image onto eigen space, \( row \) and \( col \) are the number of reduced row and column sizes, respectively.

4. EXPERIMENTAL RESULT

The proposed I(2D)\(^3\)PCA method is used for face recognition and tested on two well-known face image databases such as ORL and Yale. All images are resized to 32x32 pixels.

4.1 Yale Database

Yale database consist of 15 individual class and 10 image per each individual. Fig. 1 shows a sample date in the Yale database (http://cvc.yale.edu/projects/yalefaces/yalefaces.html).

For the training, we use randomly selected 3 images per each individual. Among the selected images, randomly selected 20% images are used for initial batch type learning of IPCA, I(2D)PCA and I(2D)\(^3\)PCA, and remaining data is used for incremental learning. The remaining 7 images of each individual are used as test.
Fig. 1 Yale(top) and ORL(bottom) database for face recognition

We check the performance of proposed method by accuracy, number of coefficient and computational load. In test, each method is repeated by 20 times with different selection of training samples. Table 1 shows the average performance for Yale database.

Table 1. Performance comparison using Yale data base

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy (%)</th>
<th>Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA</td>
<td>80.80</td>
<td>29.0</td>
</tr>
<tr>
<td>2DPCA</td>
<td>82.05</td>
<td>473.6 (14.8x32)</td>
</tr>
<tr>
<td>(2D)\PCA</td>
<td>82.13</td>
<td>99.8 (10.5x9.5)</td>
</tr>
<tr>
<td>IPCA</td>
<td>78.47</td>
<td>35.0</td>
</tr>
<tr>
<td>I(2D)PCA</td>
<td>81.19</td>
<td>614.4 (19.2x32)</td>
</tr>
<tr>
<td>I(2D)\PCA</td>
<td>81.39</td>
<td>290.14 (17.8x16.3)</td>
</tr>
</tbody>
</table>

In table 1, accuracy means how many faces are successfully recognized for test data, and dimension is the number of coefficients that are projected to an optimal eigen space at one image.

4.2 ORL Database

The ORL database consists of 40 individual class and 10 image per each individual Fig. 1 shows a sample data for ORL database (http://www.cl.cam.ac.uk/research/dtg/attarchive/ facedatabase.html).

All of condition is same with the experiment using Yale database. Table 2 shows the average performance of each method using the ORL database.

Table 2. Performance comparison using ORL data base

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy (%)</th>
<th>Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>PCA</td>
<td>85.14</td>
<td>64.5</td>
</tr>
<tr>
<td>2DPCA</td>
<td>86.29</td>
<td>512 (16x32)</td>
</tr>
<tr>
<td>(2D)\PCA</td>
<td>86.64</td>
<td>180.3 (14.2x12.7)</td>
</tr>
<tr>
<td>IPCA</td>
<td>84.75</td>
<td>84.8</td>
</tr>
<tr>
<td>I(2D)PCA</td>
<td>86.16</td>
<td>560 (17.5x32)</td>
</tr>
<tr>
<td>I(2D)\PCA</td>
<td>86.28</td>
<td>292.8 (18.3x16)</td>
</tr>
</tbody>
</table>

5. CONCLUSION

We proposed a new I(2D)\PCA for face recognition, which can not only efficient to adapt a new environment containing unseen data with small memory capacity but also reduce the computational load comparing with the convention IPCA. As a further work, we are incorporating a kernel concept with the I(2D)\PCA, and implementing a real time personal authentication system.

6. ACKNOWLEDGEMENT

This work was supported by the Converging Research Center Program funded by the Ministry of Education, Science and Technology (2010K001130) (50%) and was also supported by the National Research Foundation of Korea(NRF) Grant(NRF-2010-616-D00096) (50%).

7. REFERENCES