World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:2, No:4, 2008

A New Implementation of PCA for Fast Face Detection

Hazem M. El-Bakry

***Abstract*—** Principal Component Analysis (**PCA**) has many different important applications especially in pattern detection such as face detection / recognition. Therefore, for real time applications, the response time is required to be as small as possible. In this paper, new implementation of **PCA** for fast face detection is presented. Such new implementation is designed based on cross correlation in the frequency domain between the input image and eigenvectors (weights). Simulation results show that the proposed implementation of **PCA** is faster than conventional one.

Keywords—Fast Face Detection, PCA, Cross Correlation, Frequency Domain

I. Introduction

PRINCIPAL component analysis (or Kahunen – Loeve expansion) is applied to find the aspects of face which are important for identification. Facial eigenvectors, or, as they are sometimes called, eigenpictures or eigenfaces, provide a compact representation of whole faces, which is optimal for face reconstruction. Sirovich and Kirby [1-2] first applied the principal component analysis in efficient face representation. In this technique a new coordinate system is created for the faces where coordinates are part of the eigenvectors of a set of face images. New faces can be approximately reconstructed with only part of their projection onto the new low-dimensional space. Matthew Turk and Alex Pentland [3] expanded the idea to face recognition. Faces are encoded by a small set of weights corresponding to their projection onto the new coordinate system, and are recognized by comparing them with those of known individuals. Eigenfaces behaves well in case of different expression changes [22]. Improving the speed of PCA for face detection is very important and necessary for real time applications such as covert surveillance of criminals. In this paper, we concentrate on increasing the speed of PCA during the detection phase while its performance (detection rate) is the same as conventional implementation.

Manuscript received April 1, 2005.
H. M. El-Bakry, is assistant lecturer with Faculty of Computer Science and Information Systems – Mansoura University – Egypt. Now, he is PhD student in University of Aizu, Aizu Wakamatsu, Japan 965-8580 (phone +81-242-37-2760, fax. +81-242-37-2743, e-mail: d8071106@u-aizu.ac.jp).

For face detection, the **PCA** algorithm is applied to check the presence of a face at each pixel position in the input image. This searching problem is realized using cross correlation in the frequency domain. The cross correlation is preformed between the whole input image and the eigenvectors. This new idea increases the speed of the detection process compared to normal implementation of **PCA** algorithm in the spatial domain.

It was proved that performing cross correlation in the frequency domain is faster than time domain [19]. By the words "fast cross correlation", it is meant that cross correlation is performed in the frequency domain. A general fast pattern detection model using fast cross correlation was presented in [7-18]. Fast cross correlation was applied successfully for many different applications. Fast sub-image detection was achieved using fast cross correlation. A fast searching algorithm for face/object detection using neural networks and fast cross correlation was presented in [8,16,17]. Very fast iris detection using fast cross correlation was described in [14]. A faster algorithm for pattern detection using fast cross correlation and image decomposition was presented in [8,10,14,17]. The fastest pattern detection was achieved by using fast cross correlation, image decomposition and parallel processors. Furthermore, real time fast code detection for communication applications using fast cross correlation was introduced in [12]. In addition, a new time delay artificial neural network was invented using fast cross correlation as presented in [9]. As well as, an interesting mathematical application by using fast cross correlation was introduced [18]. Moreover, an Internet application for fast searching on web pages using fast cross correlation was presented in [15]. Finally, high speed data processing using fast cross correlation was introduced in [13].

Therefore, it is clear that in the previous work neural networks were used for pattern detection. The speed of pattern detection was achieved by applying cross correlation in the frequency domain between the input data and the weights of neural networks. Here, I make use of the idea of applying cross correlation in the frequency domain to increase the speed of **PCA** for face detection. The number of computation steps required by **PCA** is reduced by performing cross correlation in the frequency domain between the input image and eigenvectors.

The paper is organized as follows: in section II, the principles of **PCA** are discussed. Training and detection phases are

described. Fast **PCA** for pattern detection is presented in section III. The complexity of fast **PCA**, the speed up ratio for pattern detection, and simulation results are given.

II. Theory of PCA

This technique takes the advantage of face structure by proposing a scheme for recognition which is based on information theory approach, seeking to encode the most relevant information in a group of faces which will best distinguish them from one another. The approach transforms face images into a small set characteristic feature images (eigenfaces) which are the principle components of the initial training set of face images. Recognition is performed by projecting a new image into the subspace spanned by the eigenfaces (face space) and then classifying the face by comparing its position in face space with the positions of known individuals [3]. The technique of eigenspace representation is based on the earlier work presented in [5] on autoassociative memories. Autoassociative memories are a special case of associative memories in which the input patterns are associated with themselves. The goal of autoassociative memories is to find the values or weights for the connections between input units so that when a portion of an input is presented as a memory key, the memory retrieves the complete pattern, filling in missing components. Kohonen [6] used faces as stimuli to illustrate some properties of autoassociative memories. Specifically, he showed that an autoassociative memory can act as a content addressable memory of faces. Kohonen and Anderson et. al. pointed out that using an autoassociative memory to store a set of patterns is equivalent to comparing the eigen decomposition of the cross-product matrix created from the set of features describing these patterns, or, in other words, to performing the principle component analysis of the set of patterns. The model is presented first, followed by a discussion of the interpretation of eigenvectors as "macrofeatures".

The training phase is performed in the spatial domain as follows. Let a face image Γ be a two-dimensional n by n array of intensity values. An image may also be considered as a vector of dimensions nxn, so that a typical image of size 20 by 20 (as in our experiments) becomes a vector of dimension 400. The main idea of the principle component analysis is to find the vectors that best account for the distribution of face images within the entire image space. These vectors determine the subspace of face images, which can be called "face space". Each vector is of length nxn, describes an n by n image, and is a linear combination of the original face images. Because these vectors are the eigenvectors of the covariance matrix corresponding to the original sub-images, and because they are face – like in appearance, we can refer to them as "eigenfaces". For a set of face images be $\Gamma_1, \Gamma_2, \Gamma_3, \ldots\ldots\ldots, \Gamma_M$. The average image β of the set is defined by [3]:

$$\beta = \frac{1}{M}\sum_{t=1}^{M}\Gamma_t \qquad (1)$$

Each image differs from the average by the vector [3]:

$$\phi_i = \Gamma_i - \beta \qquad (2)$$

This set of very large vectors is then subject to principle component analysis, which seeks a set of M orthonormal vectors, u_t , which best describes the distribution of the data. The k^{th} vector, u_k, is chosen such that [3]:

$$\lambda_k = \frac{1}{M}\sum_{t=1}^{M}(u_k^T \phi_t)^2 \qquad (3)$$

is a maximum, subject to:

$$u_\ell^T u_k = \delta_{\ell k} = \begin{cases} 1 & \text{if } \ell = k \\ 0 & \text{if } \ell < k \end{cases} \qquad (4)$$

The vectors u_k, and scalars λ_k are the significant M eigenvectors and eigenvalues, respectively, of the covariance matrix

$$C = \frac{1}{M}\sum_{t=1}^{M}\phi_t \phi_t^T = AA^T \qquad (5)$$

where the matrix $A=[\phi_1, \phi_2, \ldots\ldots\ldots, \phi_t]$. The matrix C, however, is n^2 by n^2, and determining the n^2 eigenvectors and eigenvalues is an intractable task for typical image sizes. We need a computationally feasible method to find these eigenvectors [3]. If the number of data points in the image space is less than the dimension of the space, (M<nxn), there will be only M-1, rather than n^2, meaning eigenvectors. The remaining eigenvectors will have associated eigenvalues of zero. Fortunately we can solve for the n^2 dimensional eigenvectors in this case by first solving for the eigenvectors of an M by M matrix. For example solving a 10x10 matrix rather than 400x400 matrix and taking appropriate linear combinations of the images ϕ_t. Consider the eigenvectors v_i of $A^T A$ such that [3]:

$$A^T A v_i = \lambda_i v_i \qquad (6)$$

Pre-multiplying both sides by A, from the left, we have [3]:

$$AA^T A v_i = \lambda_i A v_i \qquad (7)$$

from which we see that Av_i are the eigenvectors of $C = A^T A$.

Following this analysis, an M by M matrix called $L= A^T A$ may be constructed, and the M eigenvectors v_ℓ of L may be computed. These vectors determine the linear combination of the M training set face images to form the eigenspace u_ℓ [3]:

$$u_\ell = \sum_{k=1}^{M} v_{\ell k}\phi_k \qquad (8)$$

With this analysis the calculations are greatly reduced from the order of the number of pixels in the images (n^2) to the order of the number of images in the training set (M). In practice, the training set of face images will be relatively small ($M<<n^2$), and the calculations become quite manageable. The associated eigenvalues allow us to rank the eigenvectors according to their usefulness in characterizing the variation among the images [3].

In the detection phase, a new sub-image Γ is transformed into eigenspace components w_k (projected into "facespace") by a simple operation [3]:

$$w_k = u_k^t (\Gamma - \beta) \quad (9)$$

for k=1, ..., M. The average image β is subtracted and the reminder is projected onto the eigenspace u_k. The weights form a vector $\Omega^t = [w_1, w_2, \ldots\ldots\ldots, w_M]$ that describe the contribution of each eigenvalue in representing the sub-image, treating the eigenfaces as a basis set for face image.

III. Fast PCA for Face Detection Using Cross Correlation in the Frequency Domain

Here, we are interested in increasing the speed of the detection process using **PCA** during the detection phase. By the words "**Fast PCA**" we mean reducing the number of computation steps required by **PCA** in the detection phase. First a set of face examples are collected and its average image is obtained. This operation is performed in the spatial domain. Then, the eigenvalues are computed as described in the previous section. In the detection phase, each sub-image in the input image (under test) is tested for the presence or absence of the required face. At each pixel position in the input image each sub-image is multiplied by the eigenvectors (weights), which has the same size as the sub-image. This multiplication is done in the spatial domain. The outputs of processors in the hidden layer are multiplied by the weights of the output layer. When the final output is high this means that the sub-image under test contains the required face and vice versa. Thus, we may conclude that this searching problem is cross correlation in the spatial domain between the image under test and the eigenvectors.

In this section, a fast algorithm for face detection based on two dimensional cross correlations that take place between the tested image and a window of weights (eigenvectors) is described. Such window is represented by the eigenvectors computed during the learning phase. The convolution theorem in mathematical analysis says that a convolution of **f** with **h** is identical to the result of the following steps: let **F** and **H** be the results of the Fourier transformation of **f** and **h** in the frequency domain. Multiply **F** and **H** in the frequency domain point by point and then transform this product into spatial domain via the inverse Fourier transform [19]. As a result, these cross correlations can be represented by a product in the frequency domain. Thus, by using cross correlation in the frequency domain a speed up in an order of magnitude can be achieved during the detection process [7-11].

In the detection phase, a sub-image **X** of size nxn (sliding window) is extracted from the tested image, which has a size **PxT**, and fed to the hidden processors. Let **U** be the eigenvectors between the input sub-image and the processors in the hidden layer. This vector has a size of **nxn** and can be represented as **nxn** matrix. The output of hidden processor **h** can be calculated as follows:

$$h = \sum_{j=1}^{n} \sum_{k=1}^{n} U(j,k) X(j,k) - U\beta \quad (10)$$

Eq. (10) represents the output of the hidden neuron for a particular sub-image **X**. It can be computed for the whole image **Ψ** as follows:

$$h(e,v) = \sum_{j=-n/2}^{n/2} \sum_{k=-n/2}^{n/2} U(j,k)\, \Psi(e+j, v+k) \quad (11)$$

Eq. (11) represents a cross correlation operation. Given any two functions **f** and **d**, their cross correlation can be obtained by [7-10]:

$$f(x,y) \otimes d(x,y) = \left(\sum_{n=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(x+n, y+n)\, d(nn) \right) \quad (12)$$

Therefore, Eq. (11) can be written as follows:

$$h = \Psi \otimes U - U\beta \quad (13)$$

where **h** is the output of the hidden processor and **h (e,v)** is the activity of the hidden processor when the sliding window is located at position **(e,v)** in the input image Ψ and **(e,v)∈[P-n+1,T-n+1]**.

Now, the above cross correlation can be expressed in terms of the Fourier Transform:

$$\Psi \otimes U = F^{-1}\left(F(\Psi) \bullet F^*(U)\right) \quad (14)$$

(*) means the conjugate of the **FFT** for the weight matrix. Hence, by evaluating this cross correlation, a speed up ratio can be obtained comparable to conventional **PCA**.

In [3], the author stated that the hidden layer of eigenface units must be fed into a neural network which classify their outputs Therefore, the final output can be evaluated as follows:

$$O(e,v) = g\left(\sum_{i=1}^{q} W_O(i)\, h_i(e,v) + b_O \right) \quad (15)$$

where ***q*** is the number of neurons in the hidden layer. **O(e,v)** is the output of the neural network when the sliding window located at the position **(e,v)** in the input image **Ψ**. **b** is the bias, **g** is the activation function, and $\mathbf{W_o}$ is the weight matrix between hidden and output layer.

The complexity of the new implementation of **PCA** using cross correlation in the frequency domain can be analyzed as follows:

1. For a tested image of **NxN pixels**, the **2D-FFT** requires a number equal to $\mathbf{N^2 log_2 N^2}$ of complex computation steps. Also, the same number of complex computation steps is required for computing the **2D-FFT** of the eigenvectors matrix.

2. The inverse **2D-FFT** is computed. So, one backward and two forward transforms have to be computed. Therefore, for an image under test, the total number of the **2D-FFT** to compute is $\mathbf{3N^2 log_2 N^2}$.

3. The input image and eigenvectors matrix should be multiplied in the frequency domain. Therefore, a number of complex computation steps equal to $\mathbf{N^2}$ should be added.

4. The number of computation steps required by the fast **PCA** is complex and must be converted into a real version. It is known that the two dimensions Fast Fourier Transform requires $(N^2/2)\log_2 N^2$ complex multiplications and $N^2 \log_2 N^2$ complex additions [20,21]. Every complex multiplication is realized by six real floating point operations and every complex addition is implemented by two real floating point operations. So, the total number of computation steps required to obtain the **2D-FFT** of an **NxN** image is:

$$\rho = 6((N^2/2)\log_2 N^2) + 2(N^2 \log_2 N^2) \quad (16)$$

which may be simplified to:

$$\rho = 5N^2 \log_2 N^2 \quad (17)$$

Performing complex dot product in the frequency domain also requires $6N^2$ real operations.

5. In order to perform cross correlation in the frequency domain, the eigenvectors matrix must have the same size as the input image. Assume that the input sub-image has a size of ***(nxn)*** dimensions. So, the search process will be performed over sub-images of **(nxn)** dimensions and the weight matrix will have the same size. Therefore, a number of zeros = (N^2-n^2) must be added to the weight matrix. This requires a total real number of computation steps = (N^2-n^2) for all neurons. Moreover, after computing the **2D-FFT** for the eigenvectors matrix, the conjugate of this matrix must be obtained. So, a real number of computation steps = N^2 should be added in order to obtain the conjugate of the eigenvectors matrix for all neurons. Also, a number of real computation steps equal to **N** is required to create butterflies complex numbers $(e^{-jk(2\Pi n/N)})$, where **0<K<L**. These **(N/2)** complex numbers are multiplied by the elements of the input image or by previous complex numbers during the computation of the **2D-FFT**. To create a complex number requires two real floating point operations. So, the total number of computation steps required for the fast **PCA** becomes:

$$\sigma = (3*5N^2 \log_2 N^2) + 6N^2 + (N^2-n^2) + N^2 + N \quad (18)$$

which can be reformulated as:

$$\sigma = (15N^2 \log_2 N^2) + 8N^2 + N - n^2 \quad (19)$$

6. Using a sliding window of size **nxn** for the same image of **NxN** pixels, $(2n^2-1)(N-n+1)^2$ computation steps are required when using traditional **PCA** for face detection process. The theoretical speed up factor η can be evaluated as follows:

$$\eta = \frac{(2n^2 - 1)(N-n+1)^2}{15N^2 \log_2 N^2 + 8N^2 + N - n^2} \quad (20)$$

The theoretical speed up ratio (Eq. 11) with different sizes of the input image and different in size eigenvectors matrices is listed in Table I. Practical speed up ratio for manipulating images of different sizes and different in size eigenvectors matrices is listed in Table II using 700 MHz processor and ***MATLAB Ver 7.0.4*.365 *(R!4 SP2***) An interesting property with fast **PCA** is that the number of computation steps does not depend either on the size of the input sub-image or the size of the eigenvectors matrix **(n)**. The effect of **(n)** on the number of computation steps is very small and can be ignored. This is incontrast to conventional networks in which the number of computation steps is increased with the size of both the input sub-image and the eigenvectors matrix **(n)**.

In practical implementation, the multiplication process consumes more time than the addition one. The effect of the number of multiplications required for conventional **PCA** in the speed up ratio (Eq. 11) is more than the number of multiplication steps required by the fast **PCA**. In order to clear this, the following equation (η_m) describes relation between the number of multiplication steps required by conventional and fast **PCA**:

$$\eta_m = \frac{n^2 (N-n+1)^2}{(3)*(3N^2 \log_2 N^2) + 6N^2} \quad (21)$$

The results listed in Table III prove that the effect of the number of multiplication steps in case of conventional **PCA** is more than fast **PCA** and this the reason why practical speed up ratio is larger than theoretical speed up ratio.

For general fast cross correlation the speed up ratio (η_g) is in the following form:

$$\eta_g = \frac{(2n^2 - 1)N^2}{(3(5(N+\tau)^2 \log_2 (N+\tau)^2) + 8(N+\tau)^2 + (N+\tau) - n^2} \quad (22)$$

where τ is a small number depends on the size of the eigenvectors matrix. General cross correlation means that the process starts from the first element in the input matrix. The theoretical speed up ratio for general fast cross correlation (η_g) defined by Eq. (22) is shown in Table V. Compared with ***MATLAB*** cross correlation function **(xcorr2)**, experimental results show that our proposed algorithm is faster than this function as shown in Table IV.

IV. Conclusion

A new approach for fast **PCA** has been presented for face detection. It has been proved mathematically and practically that the speed of the detection process becomes faster than conventional **PCA**. This has been accomplished by applying cross correlation in the frequency domain between the input image and the eigenvectors. Simulation results have confirmed theoretical computations by using MATLAB. As a result, the speed of **PCA** has been increased while its performance (detection rate) is the same as conventional implementation. Such approach can be considered as a completion of that work presented in [3] to build fast and efficient face detection system. Moreover, this algorithm can be used to fast detect and object or subimage in a given input image.

References

[1] L. Sirovich and M. Kirby, "Low-dimensional procedure for the characterization of human faces", Journal of the Optical Society of America A, 4(3), 519-524, 1987.

[2] M. Kirby and L. Sirovich, "Application of the Karhunen-Loeve procedure for the characterization of human faces", IEEE Transaction on Pattern Analysis and Machine Intelligence, 12(1), 1990.

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

[4] www. Cs. Unchicagio.edu/~qingj/thesis/index.htm , " Principle Component Analysis and Neural Network Based Face Recognition"
[5] J. A. Anderson, J. W. Silverstein, S. A. Ritz, and R. S. Jones and T. Kohonen, " Discriminative features, categorical perception, and probability learning: some applications of a neural model, " Psychological Review, No. 84, pp. 413-451, 1977.
[6] T. Kohonen, " Associative Memory": A system theoretic approach. Berlin : Springer – Velag, 1977.
[7] Hazem M. El-Bakry, and Qiangfu Zhao, "Fast Normalized Neural Processors For Pattern Detection Based on Cross Correlation Implemented in the Frequency Domain," Accepted and Under Publication in the Journal of Research and Practice in Information Technology.
[8] Hazem M. El-Bakry, and Qiangfu Zhao, "Speeding-up Normalized Neural Networks For Face/Object Detection," Machine Graphics & Vision Journal (MG&V), vol. 14, No.1, 2005, pp. 29-59.
[9] Hazem M. El-Bakry, and Qiangfu Zhao, "Fast Time Delay Neural Networks," the International Journal of Neural Systems, vol. 15, No.6, December 2005, pp.445-455.
[10] Hazem M. El-Bakry, and Qiangfu Zhao, "A New Technique for Fast Pattern Recognition Using Normalized Neural Networks," WSEAS Transactions on Information Science and Applications, Issue 11, Vol. 2, November 2005, pp. 1816-1835.
[11] Hazem M. El-Bakry, and Qiangfu Zhao, "Fast Pattern Detection Using Normalized Neural Networks and Cross Correlation in the Frequency Domain," EURASIP Journal on Applied Signal Processing, Special Issue on Advances in Intelligent Vision Systems: Methods and Applications—Part I, Vol. 2005, No. 13, 1 August 2005, pp. 2054-2060.
[12] Hazem M. El-Bakry, and Qiangfu Zhao, "A Fast Neural Algorithm for Serial Code Detection in a Stream of Sequential Data," International Journal of Information Technology, vol.2, no.1, pp. 71-90, 2005.
[13] Hazem M. El-bakry, and Qiangfu Zhao, "Modified Time Delay Neural Networks For Fast Data Processing," Proc. of IEEE Eighth International Symposium on Signal Processing and its Applications, Sydney, Australia, August 28-31, 2005.
[14] Hazem M. El-Bakry, "Human Iris Detection Using Fast Cooperative Modular Neural Nets and Image Decomposition," Machine Graphics & Vision Journal (MG&V), vol. 11, no. 4, 2002, pp. 498-512.
[15] Hazem M. El-Bakry, "New High Speed Normalized Neural Networks for Fast Pattern Discovery on Web Pages," the International Journal of Computer Science and Network Security, vol.6, No. 2A, Feb. 2006, pp.142-152.
[16] Hazem M. El-Bakry, "Face detection using fast neural networks and image decomposition," Neurocomputing Journal, vol. 48, 2002, pp. 1039-1046.
[17] Hazem M. El-Bakry, "Automatic Human Face Recognition Using Modular Neural Networks," *Machine Graphics & Vision Journal (MG&V)*, vol. 10, no. 1, 2001, pp. 47-73.
[18] Hazem M. El-Bakry, and Qiangfu Zhao, "New Faster Normalized Neural Networks For Sub-Matrix Detection Using Cross Correlation in the Frequency Domain and Matrix Decomposition," International Journal of Signal Processing, vol.2, no.3, 2005, pp. 183-202.
[19] R. Klette, and Zamperon, "Handbook of image processing operators," John Wiley & Sons, Ltd, 1996.
[20] James W. Cooley and John W. Tukey, "An algorithm for the machine calculation of complex Fourier series," *Math. Comput*. 19, 297–301 (1965).
[21] J.P. Lewis, "Fast Normalized Cross Correlation", Available from http://www.idiom.com/~zilla/
[22] www.Cs.Unchicagio.edu/~qingj/thesis/index.htm , " Principle Component Analysis and Neural Network Based Face Recognition".

Eng. Hazem Mokhtar El-Bakry (Mansoura, EGYPT 20-9-1970) received B.Sc. degree in Electronics Engineering, and M.Sc. in Electrical Communication Engineering from the Faculty of Engineering, Mansoura University – Egypt, in 1992 and 1995 respectively. Since 1997, he has been an assistant lecturer at the Faculty of Computer Science and Information Systems – Mansoura University – Egypt. Currently, he is a doctoral student at the Multimedia device laboratory, University of Aizu - Japan. In 2004, he got a Research Scholarship from Japanese Government based on a recommendation from University of Aizu.

His research interests include neural networks, pattern recognition, image processing, biometrics, cooperative intelligent systems and electronic circuits. In these areas, he has published more than 42 papers as a single author in major international journals and conferences. He is the first author in 28 refereed international journal papers and more than 72 refereed international conference papers.

Eng. El-Bakry has the patent No. 2003E 19442 DE HOL / NUR, Magnetic Resonance, SIEMENS Company, Erlangen, Germany, 2003. Furthermore, he is associate editor for journal of computer science and network security (IJCSNS). In addition, he is a referee for IEEE Transactions on Signal Processing, the International Journal of Machine Graphics & Vision, the International Journal of Computer Science and Network Security, Enformatika Journals and many different international conferences organized by IEEE. Moreover, he has been awarded the Japanese Computer & Communication prize in April 2006.

He was selected as a chairman for the Facial Image Processing Session in the 6th International Computer Science Conference, Active Media Technology (AMT) 2001, Hong Kong, China, December 18-20, 2001 and for the Genetic Programming Session, in ACS/IEEE International Conference on Computer Systems and Applications Lebanese American University Beirut, Lebanon, June 25-29, 2001. He was invited for a talk in the Biometric Consortium, Orlando, Florida, USA, 12-14 Sep. 2001, which co-sponsored by the United States National Security Agency (NSA) and the National Institute of Standards and Technology (NIST).

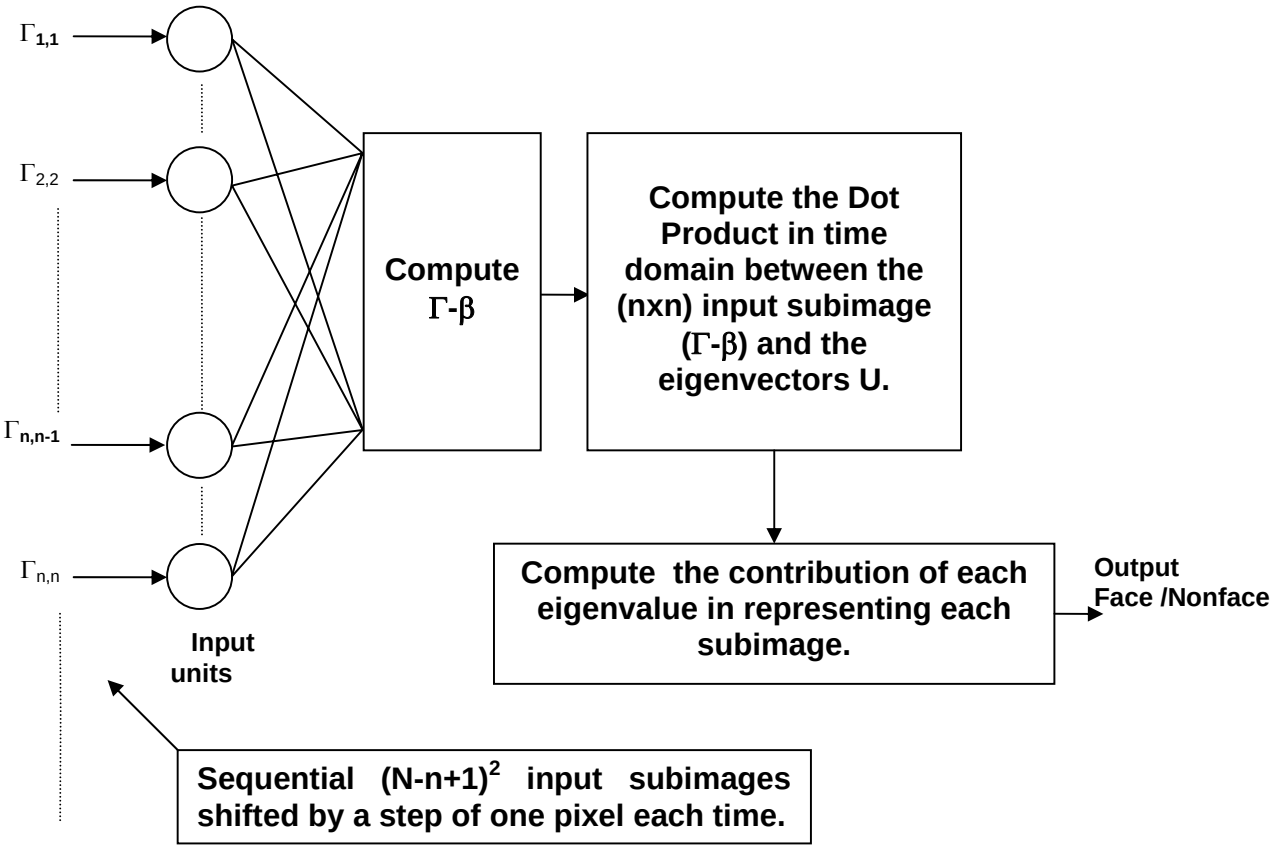

Fig.1 Classical implementation of PCA

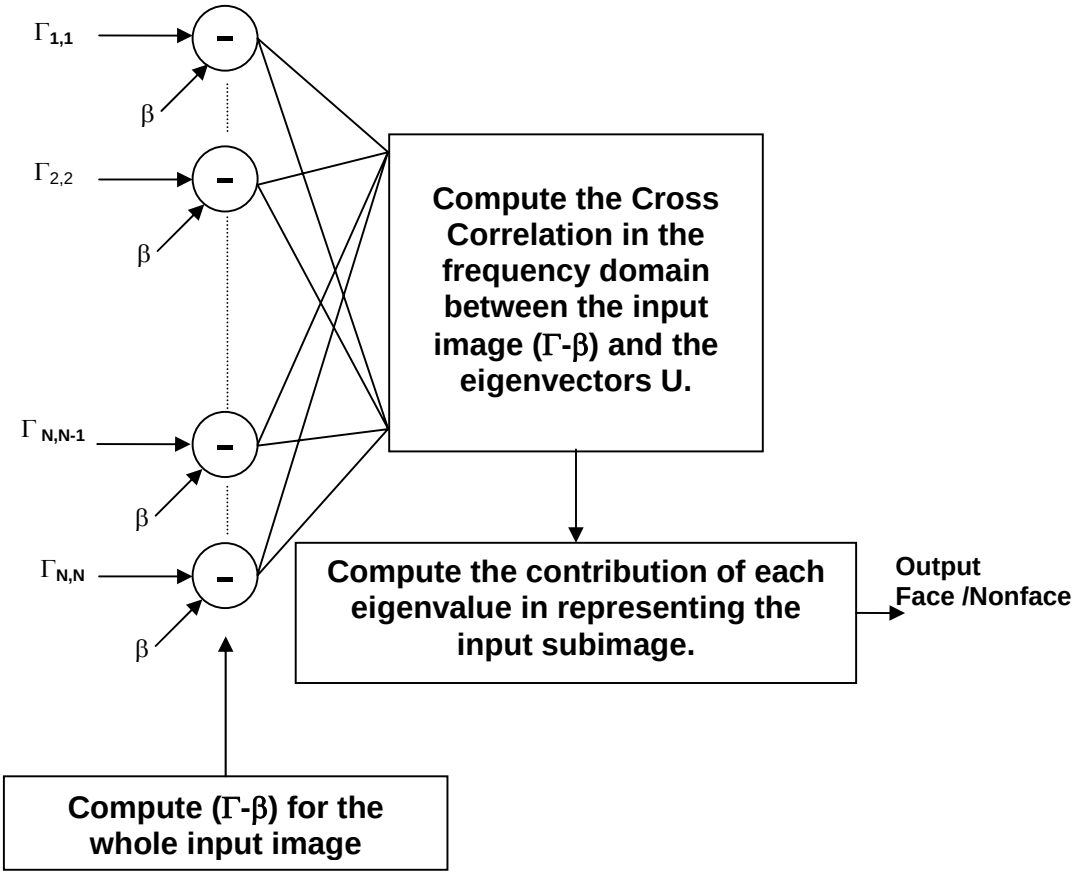

Fig. 2 Fast implementation of PCA

TABLE I
THE THEORETICAL SPEED UP RATIO FOR IMAGES WITH DIFFERENT SIZES

Image size	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	2.5290	3.4807	4.3761
200x200	2.7576	4.0759	5.5420
300x300	2.7505	4.1480	5.7601
400x400	2.7118	4.1285	5.7895
500x500	2.6697	4.0869	5.7636
600x600	2.6300	4.0408	5.7196
700x700	2.5939	3.9955	5.6701
800x800	2.5612	3.9526	5.6201
900x900	2.5317	3.9128	5.5716
1000x1000	2.5049	3.8758	5.5255
1100x1100	2.4804	3.8415	5.4818
1200x1200	2.4579	3.8097	5.4407
1300x1300	2.4371	3.7801	5.4021
1400x1400	2.4180	3.7524	5.3656
1500x1500	2.4001	3.7266	5.3312
1600x1600	2.3834	3.7023	5.2988
1700x1700	2.3678	3.6794	5.2681
1800x1800	2.3532	3.6578	5.2390
1900x1900	2.3393	3.6374	5.2113
2000x2000	2.3263	3.6181	5.1850

TABLE II
PRACTICAL SPEED UP RATIO FOR IMAGES WITH DIFFERENT SIZES USING MATLAB VER 5.3

Image size	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	7.88	10.75	14.69
200x200	6.21	9.19	13.17
300x300	5.54	8.43	12.21
400x400	4.78	7.45	11.41
500x500	4.68	7.13	10.79
600x600	4.46	6.97	10.28
700x700	4.34	6.83	9.81
800x800	4.27	6.68	9.60
900x900	4.31	6.79	9.72
1000x1000	4.19	6.59	9.46
1100x1100	4.24	6.66	9.62
1200x1200	4.20	6.62	9.57
1300x1300	4.17	6.57	9.53
1400x1400	4.13	6.53	9.49
1500x1500	4.10	6.49	9.45
1600x1600	4.07	6.45	9.41
1700x1700	4.03	6.41	9.37
1800x1800	4.00	6.38	9.32
1900x1900	3.97	6.35	9.28
2000x2000	3.94	6.31	9.25

TABLE III
A COMPARISON BETWEEN THE NUMBER OF MULTIPLICATION STEPS REQUIRED FOR CONVENTIONAL AND FAST PCA TO MANIPULATE IMAGES WITH DIFFERENT SIZES (n=20, q=30)

.Image size	Conventional Neural Nets	Faster Neural Nets	Speed up ratio (η_m)
100x100	7.8732e+007	2.6117e+007	3.0146
200x200	3.9313e+008	1.1911e+008	3.3007
300x300	9.4753e+008	2.8726e+008	3.2985
400x400	1.7419e+009	5.3498e+008	3.2560
500x500	2.7763e+009	8.6537e+008	3.2083
600x600	4.0507e+009	1.2808e+009	3.1627
700x700	5.5651e+009	1.7832e+009	3.1209
800x800	7.3195e+009	2.3742e+009	3.0830
900x900	9.3139e+009	3.0552e+009	3.0486
1000x1000	1.1548e+010	3.8275e+009	3.0172
1100x1100	1.4023e+010	4.6921e+009	2.9886
1200x1200	1.6737e+010	5.6502e+009	2.9622
1300x1300	1.9692e+010	6.7026e+009	2.9379
1400x1400	2.2886e+010	7.8501e+009	2.9154
1500x1500	2.6320e+010	9.0935e+009	2.8944
1600x1600	2.9995e+010	1.0434e+010	2.8748
1700x1700	3.3909e+010	1.1871e+010	2.8564
1800x1800	3.8064e+010	1.3407e+010	2.8392
1900x1900	4.2458e+010	1.5041e+010	2.8229
2000x2000	7.8732e+007	2.6117e+007	3.0146

TABLE IV
THE THEORETICAL SPEED UP RATIO FOR THE GENERAL FAST PCA ALGORITHM

Image size	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	5.59	8.73	11.95
200x200	4.89	7.64	10.75
300x300	4.56	7.12	10.16
400x400	4.35	6.80	9.68
500x500	4.20	6.56	9.37
600x600	4.08	6.38	9.13
700x700	4.00	6.24	8.94
800x800	3.92	6.12	8.77
900x900	3.85	6.02	8.63
1000x1000	3.79	5.93	8.51
1100x1100	3.74	5.85	8.43
1200x1200	3.70	5.78	8.33
1300x1300	3.66	5.72	8.24
1400x1400	3.62	5.66	8.16
1500x1500	3.59	5.61	8.08
1600x1600	3.56	5.57	8.02
1700x1700	3.53	5.52	7.95
1800x1800	3.51	5.48	7.89
1900x1900	3.48	5.44	7.84
2000x2000	5.59	8.73	11.95

TABLE V
SIMULATION RESULTS OF THE SPEED UP RATIO FOR THE GENERAL FASTER CROSS CORRELATION COMPARED WITH THE MATLAB CROSS CORRELATION FUNCTION (XCORR2)

Image size	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	10.14	13.05	16.49
200x200	9.17	11.92	14.33
300x300	8.25	10.83	13.41
400x400	7.91	9.62	12.65
500x500	6.77	9.24	11.77
600x600	6.46	8.89	11.19
700x700	5.99	8.47	10.96
800x800	5.48	8.74	10.32
900x900	5.31	8.43	10.66
1000x1000	5.91	8.66	10.51
1100x1100	5.77	8.61	10.46
1200x1200	5.68	8.56	10.40
1300x1300	5.62	8.52	10.35
1400x1400	5.58	8.47	10.31
1500x1500	5.54	8.43	10.26
1600x1600	5.50	8.39	10.22
1700x1700	5.46	8.33	10.18
1800x1800	5.42	8.28	10.14
1900x1900	5.38	8.24	10.10
2000x2000	5.34	8.20	10.06