Hindawi Publishing Corporation
EURASIP Journal on Advances in Signal Processing
Volume 2007, Article ID 91467, 7 pages
doi:10.1155/2007/91467

Research Article

Human Hand Recognition Using IPCA-ICA Algorithm

Issam Dagher, William Kobersy, and Wassim Abi Nader

Department of Computer Engineering, University of Balamand, Elkoura, Lebanon

Received 3 July 2006; Revised 21 November 2006; Accepted 2 February 2007

Recommended by Satya Dharanipragada

A human hand recognition system is introduced. First, a simple preprocessing technique which extracts the palm, the four fingers, and the thumb is introduced. Second, the eigenpalm, the eigenfingers, and the eigenthumb features are obtained using a fast incremental principal non-Gaussian directions analysis algorithm, called IPCA-ICA. This algorithm is based on merging sequentially the runs of two algorithms: the principal component analysis (PCA) and the independent component analysis (ICA) algorithms. It computes the principal components of a sequence of image vectors incrementally without estimating the covariance matrix (so covariance-free) and at the same time transforming these principal components to the independent directions that maximize the non-Gaussianity of the source. Third, a classification step in which each feature representation obtained in the previous phase is fed into a simple nearest neighbor classifier. The system was tested on a database of 20 people (100 hand images) and it is compared to other algorithms.

1. INTRODUCTION

Biometrics is an emerging technology [1, 2] that is used to identify people by their physical and/or behavioral characteristics and, so, inherently requires that the person to be identified is physically present at the point of identification. The physical characteristics of an individual that can be used in biometric identification/verification systems are fingerprint [3, 4], hand geometry [5, 6], palm print [7–9], face [4, 10], iris [11, 12], retina [13], and the ear [14].

The behavioral characteristics are signature [12], lip movement [15], speech [16], keystroke dynamics [1, 2], gesture [1, 2], and the gait [1, 2].

A single physical or behavioral characteristic of an individual can sometimes be insufficient for identification. For this reason, multimodal biometric systems—that is, systems that integrate two or more different biometrics characteristics—are being developed to provide an acceptable performance, and increase the reliability of decisions. The human hand contains a wide variety of features, for example, shape, texture, and principal palm lines—that can be used by biometric systems. Features extracted by projecting palm images into the subspace obtained by the PCA transform are called eigenpalm features, whereas those extracted by projecting images of fingers and thumb are called eigenfinger and eigenthumb features. This paper merges sequentially two techniques based on principal component analysis and independent component analysis.

The first technique is called incremental principal component analysis (IPCA) which is an incremental version of the popular unsupervised principal component technique. The traditional PCA [17] algorithm computes eigenvectors and eigenvalues for a sample covariance matrix derived from a well-known given image data matrix, by solving an eigenvalue system problem. Also, this algorithm requires that the image data matrix be available before solving the problem (batch method). The incremental principal component method updates the eigenvectors each time a new image is introduced.

The second technique is called independent component analysis (ICA) [18]. It is used to estimate the independent characterization of human hand vectors (palms, fingers, or thumbs). It is known that there is a correlation or dependency between different human hand vectors. Finding the independent basic vectors that form those correlated ones is a very important task. The set of human hand vectors is represented as a data matrix X where each row corresponds to a different human hand. The correlation between rows of matrix X can be represented as the rows of a mixing matrix A. The independent basic vectors are represented as rows of source matrix S. The ICA algorithm extracts these independent vectors from a set of dependent ones using

$$X = A \cdot S. \tag{1}$$

When the dimension of the image is high, both the computation and storage complexity grow dramatically. Thus, the idea of using a real time process becomes very efficient in order to compute the principal independent components for observations arriving sequentially. Each eigenvector or principal component will be updated, using FastICA algorithm, to a non-Gaussian component. In (1), if the source matrix S contains Gaussian uncorrelated elements then the resulting elements in the mixed matrix X will be also Gaussian but correlated elements.

The FastICA method does not have a solution if the random variables to estimate are Gaussian random variables. This is due to the fact that the joint distribution of the elements of X will be completely symmetric and does not give any special information about the columns of A. In this paper, S is always a non-Gaussian vector. It should be noted that the central limit theorem states that the sum of several independent random variables, such as those in S, tends towards a Gaussian distribution. So $x_i = a_1 s_1 + a_2 s_2$ is more Gaussian than either s_1 or s_2. The central limit theorem implies that if we can find a combination of the measured signals in X with minimal Gaussian properties, then that signal will be one of the independent signals. Once W is determined it is a simple matter to invert it to find A.

Each image x, represented by an (n, m) matrix of pixels, will be represented by a high-dimensional vector of $n \times m$ pixels. These image vectors will be the rows of X and the resulting uncorrelated components will be the rows of S. Therefore, each column of A, called w, will be a direction that maximizes the non-Gaussianity of the projection of the dependent images x into w. The raw features that are sent to this algorithm are the grayscale levels of every pixel in the image without using any geometric features or wavelet-based features.

It should be noted that the batch method no longer satisfies an up coming new trend of signal processing research in which all visual filters are incrementally derived from very long online real-time video stream. Online development of visual filters requires that the system perform while new sensory signals flow in. When the dimension of the image is high, both the computation and storage complexity grow dramatically. Thus, the idea of using a real time process becomes very efficient in order to compute the principal independent components for observations (faces) arriving sequentially.

2. SYSTEM DESCRIPTION

The multimodal biometric identification system consists of the following phases.

(i) The image-acquisition phase: a hand image is taken using a low-cost scanner; the spatial resolution of the images is 180 dots per inch (dpi) and 256 gray levels.

(ii) The preprocessing phase: three regions of interest are localized: a palm region, four-finger region, and the thumb region.

(iii) The processing phase: the three normalized regions are transformed by the sequential PCA-ICA algorithm into three spaces called eigenpalm space, eigenfinger space, and eigenthumb space. The feature spaces are spanned by a certain number of the largest eigenvectors. The outputs of the feature-extraction modules, for the sample x, are three feature vectors.

(iv) The recognition phase: the matching between the corresponding vectors and the templates from a database is performed.

3. PREPROCESSING PHASE

Images of the right hand are scanned at 180 dpi/256 gray levels using a low-cost scanner. The user puts his/her hand on the scanner with the fingers spread naturally; there are no pegs, or any other hand-position constrainers.

Figure 1 shows the separation the hand image of all users into three different regions.

The regions are the thumb image (Figure 1(e)), the palm (Figure 1(d)), and the remaining four fingers (Figure 1(c)).

Figure 1(b) shows that the calculations of the regions is quite simple and requires no hard or long processing time, simply the four-fingers region is separated from the other two by taking a horizontal line at 45 percent of the original image. Then, the palm region is obtained by a vertical line at 70 percent of the subimage obtained before.

The method described above is applied to all users' images since the entire scanned are acquired having 1020 × 999 pixels.

After acquiring and obtaining three regions, a geometry normalization is also applied and the "four-fingers" region is normalized to 454×200 pixels, the "palm" region to 307×300 pixels, and the "thumb" region to 180 × 300 pixels.

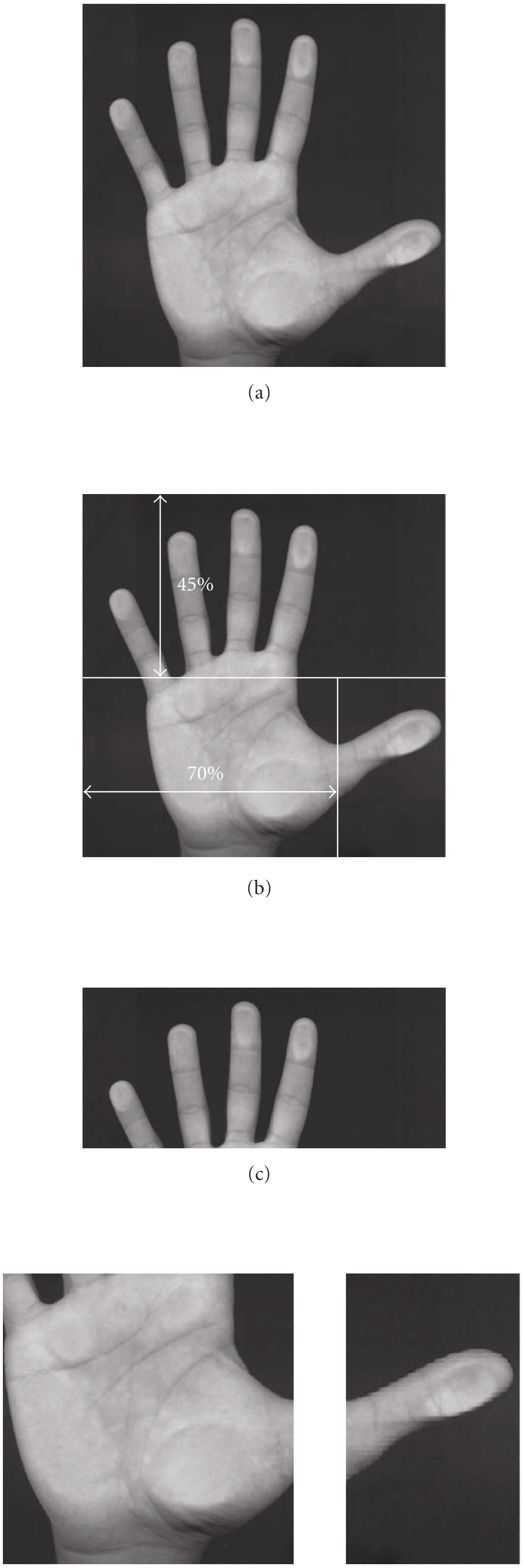
4. DERIVATION OF THE IPCA-ICA ALGORITHM

Each time a new image is introduced, the non-Gaussian vectors will be updated. They are presented by the algorithm in a decreasing order with respect to the corresponding eigenvalue (the first non-Gaussian vector will correspond to the largest eigenvalue). While the convergence of the first non-Gaussian vector will be shown in Section 4.1, the convergence of the other vectors will be shown in Section 4.2.

4.1. *The first non-Gaussian vector*

4.1.1. Algorithm definition

Suppose that the sample d-dimensional vectors, $u(1)$; $u(2)$; . . ., possibly infinite, which are the observations from a certain given image data, are received sequentially. Without loss of generality, a fixed estimated mean image is initialized in the beginning of the algorithm. It should be noted that a simple way of getting the mean image is to present sequentially all the images and calculate their mean. This mean



(a)

(b)

(c)

(d)

(e)

Figure 1: (a) original image. (b) original image with regions of interest marked. (c) "four-fingers" subimage. (d) "palm" subimage. (e) "thumb" subimage.

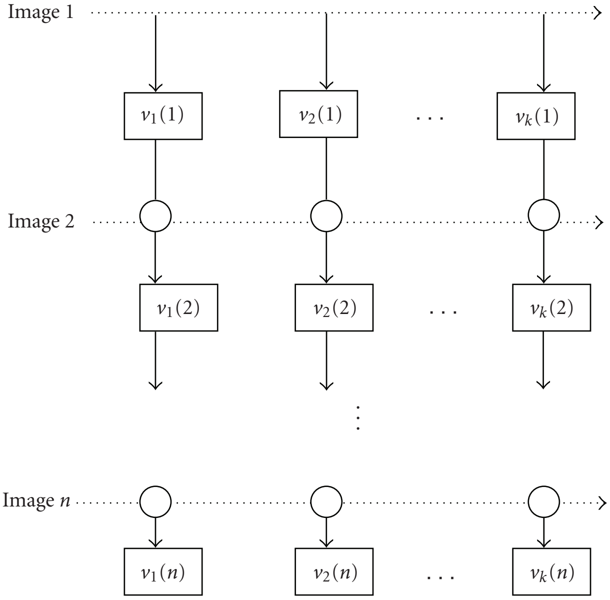


Figure 2: IPCA-PCA algorithm description.

can be subtracted from each vector $u(n)$ in order to obtain a normalization vector of approximately zero mean. Let $C = E[u(n)u^T(n)]$ be the $d \times d$ covariance matrix, which is not known as an intermediate result. The IPCA-ICA algorithm can be described as follows.

The proposed algorithm takes the number of input images, the dimension of the images, and the number of desired non-Gaussian directions as inputs and returns the image data matrix, and the non-Gaussian vectors as outputs. It works like a linear system that predicts the next state vector from an input vector and a current state vector. The non-Gaussian components will be updated from the previous components values and from a new input image vector by processing sequentially the IPCA and the FastICA algorithms. While IPCA returns the estimated eigenvectors as a matrix that represents subspaces of data and the corresponding eigenvalues as a row vector, FastICA searches for the independent directions w where the projections of the input data vectors will maximize the non-Gaussianity. It is based on minimizing the approximate negentropy function [19] given by $J(x) = \sum_i k_i\{E(G_i(x)) - E(G_i(v))\}^2$ using Newton's method. Where $G(x)$ is a nonquadratic function of the random variable x and E is its expected value.

The obtained independent vectors will form a basis which describes the original data set without loss of information. The face recognition can be done by projecting the input test image onto this basis and comparing the resulting coordinates with those of the training images in order to find the nearest appropriate image.

Assume the data consists of n images and a set of k non-Gaussian vectors are given, Figure 2 illustrates the steps of the algorithm.

Initially, all the non-Gaussian vectors are chosen to describe an orthonormal basis. In each step, all those vectors

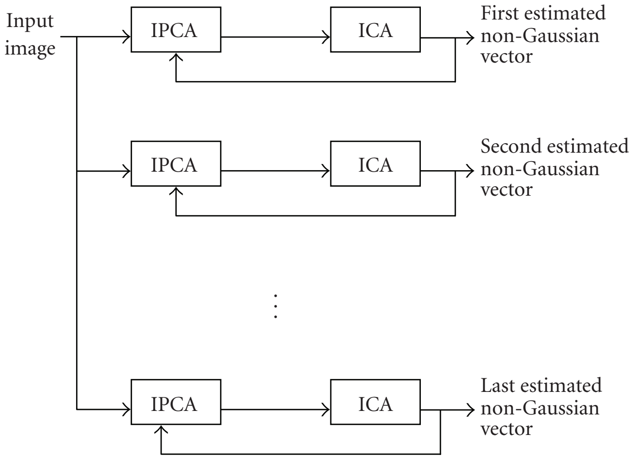

Input image
IPCA
ICA
First estimated non-Gaussian vector
IPCA
ICA
Second estimated non-Gaussian vector
⋮
IPCA
ICA
Last estimated non-Gaussian vector

Figure 3: IPCA-ICA algorithm block diagram.

will be updated using an IPCA updating rule presented in (7). Then each estimated non-Gaussian vector will be an input for the ICA function in order to extract the corresponding non-Gaussian vector from it (Figure 3).

4.1.2. *Algorithm equations*

By definition, an eigenvector x with a corresponding eigenvalue λ of a covariance matrix C satisfies

$$\lambda \cdot x = C \cdot x. \tag{2}$$

By replacing in (2) the unknown C with the sample covariance matrix $(1/n)\sum_{i=1}^{n} u(i) \cdot u^T(i)$ and using $v = \lambda \cdot x$, the following equation is obtained:

$$v(n) = \frac{1}{n}\sum_{i=1}^{n} u(i) \cdot u^T(i) \cdot x(i), \tag{3}$$

where $v(n)$ is the nth step estimate of v after entering all the n images.

Since $\lambda = \|v\|$ and $x = v/\|v\|$, $x(i)$ is set to $v(i-1)/\|v(i-1)\|$ (estimating $x(i)$ according to the given previous value of v). Equation (3) leads to the following equation:

$$v(n) = \frac{1}{n}\sum_{i=1}^{n} u(i) \cdot u^T(i) \cdot \frac{v(i-1)}{||v(i-1)||}. \tag{4}$$

Equation (4) can be written in a recursive form:

$$v(n) = \frac{n-1}{n} v(n-1) + \frac{1}{n} u(n) u^T(n) \frac{v(n-1)}{||v(n-1)||}, \tag{5}$$

where $(n-1)/n$ is the weight for the last estimate and $1/n$ is the weight for the new data.

To begin with, let $v(0) = u(1)$ the first direction of data spread. The IPCA algorithm will give the first estimate of the first principal component $v(1)$ that corresponds to the maximum eigenvalue:

$$v(1) = u(1) u^T(1) \frac{v(0)}{||v(0)||}. \tag{6}$$

Then, the vector will be the initial direction in the FastICA algorithm:

$$w = v(1). \tag{7}$$

The FastICA algorithms will repeat until convergence the following rule:

$$w_{\text{new}} = E\big[v(1) \cdot G'\big(w^T \cdot v(1)\big)\big] - E\big[G''\big(w^T \cdot v(1)\big)\big] \cdot w, \tag{8}$$

where $G''(x)$ is the derivative of the function $G'(x)$ (equation (10)). It should be noted that this algorithm uses an approximation of negentropy in order to assure the non-Gaussianity of the independent vectors. Before starting the calculation of negentropy, a nonquadratic function G should be chosen, for example,

$$G(u) = -\exp\Big(-\frac{u^2}{2}\Big), \tag{9}$$

and its derivative:

$$G'(u) = u \cdot \exp\Big(-\frac{u^2}{2}\Big). \tag{10}$$

In general, the corresponding non-Gaussian vector w, for the estimated eigenvectors $v_k(n)$, will be estimated using the following repeated rule:

$$w_{\text{new}} = E\big[v_k(n) \cdot G'\big(w^T \cdot v_k(n)\big)\big] - E\big[G''\big(w^T \cdot v_k(n)\big)\big] \cdot w. \tag{11}$$

4.2. *Higher order non-Gaussian vectors*

The previous discussion only estimates the first non-Gaussian vector. One way to compute the other higher order vectors is following what *stochastic gradient ascent* (SGA) does: start with a set of orthonormalized vectors, update them using the suggested iteration step, and recover the orthogonality using *Gram-Schmidt orthonormalization* (GSO). For real-time online computation, avoiding time-consuming GSO is needed. Further, the non-Gaussian vectors should be orthogonal to each other in order to ensure the independency. So, it helps to generate "observations" only in a complementary space for the computation of the higher order eigenvectors. For example, to compute the second order non-Gaussian vector, first the data is subtracted from its projection on the estimated first-order eigenvector $v_1(n)$, as shown in:

$$u_2(n) = u_1(n) - u_1^T(n)\frac{v_1(n)}{||v_1(n)||}\frac{v_2(n)}{||v_2(n)||}, \tag{12}$$

where $u_1(n) = u(n)$. The obtained residual, $u_2(n)$, which is in the complementary space of $v_1(n)$, serves as the input data to the iteration step. In this way, the orthogonality is always enforced when the convergence is reached, although not exactly so at early stages. This, in effect, better uses the sample available and avoids the time-consuming GSO.

After convergence, the non-Gaussian vector will also be enforced to be orthogonal, since they are estimated in complementary spaces. As a result, all the estimated vectors w_k will be

```
For $i = 1 : n$
        img = input image from image data matrix;
u(i) = img;
        for $j = 1 : k$
                if $j == i$, initialize the $j$th non-Gaussian
                vector as
                        $v_j(i) = u(i)$;
                else
                        $v_j(i) = \frac{i-1}{i} v_j(i-1) + \frac{1}{i} u(i) u^T(i) \frac{v_j(i-1)}{||v_j(i-1)||}$; (vector update)
                        $u(i) = u(i) - u^T(i) \cdot \frac{v_j(i)}{||v_j(i)||} \frac{v_j(i)}{||v_j(i)||}$; (to ensure orthogonality)
                end
          $w = v_j(i)$;
        Repeat until convergence (wnew = w)
        $w_{\text{new}} = E[v_j(i) \cdot G'(w^T \cdot v_j(i))] - E[G''(w^T \cdot v_j(i))] \cdot w$;
                                (searching for the direction that
                                 maximizes non-Gaussianity)
        end
$v_j(i) = w^T \cdot v_j(i)$; (projection on the direction of non-Gaussianity $w$)
end
end
```

Algorithm 1

(i) Non-Gaussian according to the learning rule in the algorithm,

(ii) independent according to the complementary spaces introduced in the algorithm.

4.3. Algorithm summary

Assume n different images $u(n)$ are given; let us calculate the first k dominant non-Gaussian vectors $v_j(n)$. Assuming that $u(n)$ stands for nth input image and $v_j(n)$ stands for nth update of the jth non-Gaussian vector.

Combining IPCA and FastICA algorithms, the new algorithm can be summarized as shown in Algorithm 1.

4.4. Comparison with PCA-ICA batch algorithm

The major difference between the IPCA-ICA algorithm and the PCA-ICA batch algorithm is the real-time sequential process. IPCA-ICA does not need a large memory to store the whole data matrix that represents the incoming images. Thus in each step, this function deals with one incoming image in order to update the estimated non-Gaussian directions, and the next incoming image can be stored over the previous one. The first estimated non-Gaussian vectors (corresponding to the largest eigenvalues) in IPCA correspond to the vectors that carry the most efficient information. As a result, the processing of IPCA-ICA can be restricted to only a specified number of first non-Gaussian directions. On the other side, the decision of efficient vectors in PCA can be done only after calculating all the vectors, so the program will spend a certain time calculating unwanted vectors. Also, ICA works usually in a batch mode where the extraction of independent components of the input eigenvectors can be done only when these eigenvectors are present simultaneously at the input. It is very clear that from the time efficiency concern, IPCA-ICA will be more efficient and requires less execution time than PCA-ICA algorithm. Finally, IPCA-ICA gives a better recognition performance than batch PCA-ICA by taking only a small number of basis vectors. These results are due to the fact that applying batch PCA on all the images will give the m noncorrelated basis vectors. Applying ICA on the n out of these m vectors will not guarantee that the obtained vectors are the most efficient vectors. The basis vectors obtained by the IPCA-ICA algorithm will have more efficiency or contain more information than those chosen by the batch algorithm.

5. EXPERIMENTAL RESULTS AND DISCUSSIONS

To demonstrate the effectiveness of the IPCA-ICA algorithm on the human hand recognition problem, a database consisting of 100 templates (20 users, 5 templates per user) was utilized. Four recognition experiments were made. Three of them were made using features from only one hand part (i.e., recognition based only on eigenpalm features, recognition based only on eigenfinger features, and recognition based only on eigenthumb features). The final experiment was done using a majority vote on the three previous recognition results.

The IPCA-ICA algorithm is compared against three feature selection methods, namely, the LDA algorithm, the PCA algorithm, and the batch PCA-ICA. For each of the three methods, the recognition procedure consists of (i) a feature extraction step where two kinds of feature representation of each training or test sample are extracted by projecting the sample onto the two feature spaces generalized by the PCA, the LDA, respectively, (ii) a classification step in which each

Table 1: Comparison between different algorithms.

	Palm	Finger	Thumb	Majority	time (s)
PCA vectors = 60	82.5 %	80 %	47.5 %	84 %	40.52
PCA vectors = 20	80 %	75 %	42.5 %	82 %	40.74
LDA vectors	90.5%	88.5%	65%	91.5%	52.28
Batch PCA-ICA	88%	85.5%	60%	88%	54.74
IPCA-ICA (users = 20)	92.5%	92.5%	77.5%	94%	45.12

feature representation obtained in the first step is fed into a simple nearest neighbor classifier. It should be noted at this point that, since the focus in this paper is on feature extraction, a very simple classifier, namely, nearest Euclidean distance neighbor, is used in step (ii). In addition, IPCA-PCA is compared to batch FastICA algorithm that applies PCA and ICA one after the other. In FastICA, the reduced number of eigenvectors obtained by PCA batch algorithm is used as input vectors to ICA batch algorithm in order to generate the independent non-Gaussian vectors. Here FastICA process is not a real time process because the batch PCA requires a previous calculation of covariance matrix before processing and calculating the eigenvectors. Notice here that PCA, batch PCA-ICA, and PCA-ICA algorithms are experimented using the same method of introducing and inputting the training images and tested using also the same nearest neighbor procedure.

A summary of the experiment is shown below.

Number of users = 20.
Number of images per user = 5.
Number of trained images per user = 3.
Number of tested images per user = 2.

The recognition results are shown in Table 1.

For the PCA, LDA, the batch PCA-ICA, the maximum number of vectors is $20 * 3 = 60$ (number of users times the number of trained images).

Taking the first 20 vectors for the PCA will decrease the recognition rate as shown in the second row of Table 1.

The IPCA-ICA with 20 independent vectors (number of users) yields better recognition rate than the other 3 algorithms.

It should be noted that the execution training time in seconds on our machine Pentium IV is shown in the last column of Table 1.

Figure 4 shows the recognition results for the palm as a function of the number of vectors.

It should be noted that as the algorithm precedes in time, more number of non-Gaussian vectors are formed and the recognition results get better.

6. CONCLUSION AND FUTURE WORK

In this paper, a prototype of an online biometric identification system based on eigenpalm, eigenfinger, and eigenthumb features was developed. A simple preprocessing technique was introduced. It should be noted here that introducing constraints on the hand (using pegs especially at the thumb) will definitely increase the system performance. The use of a multimodal approach has improved the recognition rate.

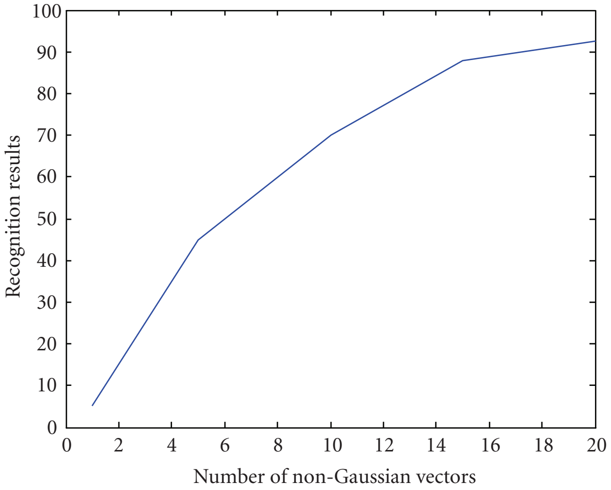


Figure 4: Recognition results for the palm as a function of the number of vectors.

The IPCA-ICA method based on incremental update of the non-Gaussian independent vectors has been introduced. The method concentrates on a challenging issue of computing dominating non-Gaussian vectors from an incrementally arriving high-dimensional data stream without computing the corresponding covariance matrix and without knowing the data in advance.

It is very efficient in memory usage (only one input image is needed at every step) and it is very efficient in the calculation of the first basis vectors (unwanted vectors do not need to be calculated). In addition to these advantages, this algorithm gives an acceptable recognition success rate in comparison with the PCA and the LDA algorithms.

In Table 1, it is clear that IPCA-ICA achieves higher average success rate than the LDA, the PCA, and the FastICA methods.

REFERENCES

[1] A. K. Jain, R. Bolle, and S. Pankanti, Eds., *Biometrics: Personal Identification in Networked Society*, Kluwer Academic, Boston, Mass, USA, 1999.

[2] D. Zhang, *Automated Biometrics: Technologies & Systems*, Kluwer Academic, Boston, Mass, USA, 2000.

[3] A. K. Jain, L. Hong, S. Pankanti, and R. Bolle, “An identity-authentication system using fingerprints,” *Proceedings of the IEEE*, vol. 85, no. 9, pp. 1365–1388, 1997.

[4] L. C. Jain, U. Halici, I. Hayashi, and S. B. Lee, *Intelligent Biometric Techniques in Fingerprint and Face Recognition*, CRC Press, New York, NY, USA, 1999.

[5] A. K. Jain, A. Ross, and S. Pankanti, “A prototype hand geometry-based verification system,” in *Proceedings of the 2nd International Conference on Audio- and Video-Based Biometric Person Authentication (AVBPA ’99)*, pp. 166–171, Washington, DC, USA, March 1999.

[6] R. Sanchez-Reillo, C. Sanchez-Avila, and A. Gonzalez-Marcos, "Biometric identification through hand geometry measurements," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, pp. 1168–1178, 2000.

[7] W. Shu and D. Zhang, "Automated personal identification by palmprint," *Optical Engineering*, vol. 37, no. 8, pp. 2359–2362, 1998.

[8] D. Zhang and W. Shu, "Two novel characteristics in palmprint verification: datum point invariance and line feature matching," *Pattern Recognition*, vol. 32, no. 4, pp. 691–702, 1999.

[9] D. Zhang, W.-K. Kong, J. You, and M. Wong, "Online palmprint identification," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 2, pp. 1041–1050, 2003.

[10] J. Zhang, Y. Yan, and M. Lades, "Face recognition: eigenface, elastic matching, and neural nets," *Proceedings of the IEEE*, vol. 85, no. 9, pp. 1423–1435, 1997.

[11] R. P. Wildes, "Iris recognition: an emerging biometric technology," in *Automated Biometrics: Technologies & Systems*, pp. 1348–1363, Kluwer Academic, Boston, Mass, USA, 2000.

[12] M. Negin, T. A. Chmielewski Jr., M. Salganicoff, et al., "An iris biometric system for public and personal use," *Computer*, vol. 33, no. 2, pp. 70–75, 2000.

[13] R. Hill, "Retina identification," in *Biometrics: Personal Identification in Networked Society*, pp. 123–141, Kluwer Academic, Boston, Mass, USA, 1999.

[14] M. Burge and W. Burger, "Ear biometrics," in *Biometrics: Personal Identification in Networked Society*, pp. 273–286, Kluwer Academic, Boston, Mass, USA, 1999.

[15] C. Bregler and Y. Konig, "Eigenlips for robust speech recognition," in *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '94)*, vol. 2, pp. 669–672, Adelaide, SA, Australia, April 1994.

[16] J. P. Campbell, "Speaker recognition: a tutorial," in *Automated Biometrics: Technologies & Systems*, pp. 1437–1462, Kluwer Academic, Boston, Mass, USA, 2000.

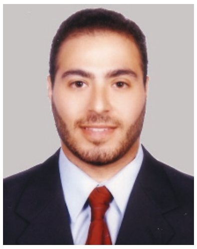
[17] J. Karhunen and J. Joutsensalo, "Representation and separation of signals using nonlinear PCA type learning," *Neural Networks*, vol. 7, no. 1, pp. 113–127, 1994.

[18] P. Comon, "Independent component analysis. a new concept?" *Signal Processing*, vol. 36, no. 3, pp. 287–314, 1994.

[19] A. Hyvärinen, "New approximations of differential entropy for independent component analysis and projection pursuit," in *Advances in Neural Information Processing Systems 10*, pp. 273–279, MIT Press, Cambridge, Mass, USA, 1998.

Issam Dagher is an Associate Professor at the Department of Computer Engineering in the University of Balamand, Elkoura, Lebanon. He finished his M.S degree at FIU, Miami, USA, and his Ph.D. degree at the University of Central Florida, Orlando, USA. His research interests include neural networks, fuzzy logic, image processing, signal processing.

William Kobersy finished his M.S degree at the Department of Computer Engineering in the University of Balamand, Elkoura, Lebanon.



Wassim Abi Nader finished his M.S degree at the Department of Computer Engineering in the University of Balamand, Elkoura, Lebanon.