Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 530251, 12 pages
http://dx.doi.org/10.1155/2014/530251

Research Article

Applications of PCA and SVM-PSO Based Real-Time Face Recognition System

Ming-Yuan Shieh, Juing-Shian Chiou, Yu-Chia Hu, and Kuo-Yang Wang

Department of Electrical Engineering, Southern Taiwan University of Science and Technology, Tainan City 710, Taiwan

Correspondence should be addressed to Juing-Shian Chiou; jschiou@mail.stust.edu.tw

Received 25 February 2014; Accepted 23 April 2014; Published 13 May 2014

Academic Editor: Rui Mu

This paper incorporates principal component analysis (PCA) with support vector machine-particle swarm optimization (SVM-PSO) for developing real-time face recognition systems. The integrated scheme aims to adopt the SVM-PSO method to improve the validity of PCA based image recognition systems on dynamically visual perception. The face recognition for most human-robot interaction applications is accomplished by PCA based method because of its dimensionality reduction. However, PCA based systems are only suitable for processing the faces with the same face expressions and/or under the same view directions. Since the facial feature selection process can be considered as a problem of global combinatorial optimization in machine learning, the SVM-PSO is usually used as an optimal classifier of the system. In this paper, the PSO is used to implement a feature selection, and the SVMs serve as fitness functions of the PSO for classification problems. Experimental results demonstrate that the proposed method simplifies features effectively and obtains higher classification accuracy.

1. Introduction

There are numerous approaches that develop useful schemes to detect and recognize the features of human faces in recent years. They are used to filter the background and detect the faces blocks from a digital image firstly, then to determine their features and generate characteristic vectors, to localize the faces continuously, and to recognize the main face finally. In general, the approaches of human face image processing consist of two fields as face detection and face recognition. For human-robot interaction, to recognize faces is much more difficult than to detect faces. It is because the facial expression and features are changeable and not easily recognized and predicted.

There are two standard linear subspace projections of the low-resolution facial images [1], the PCA and the linear discriminant analysis (LDA) [2], used to distinguish the different styles of images. The PCA is basically a compression procedure based on linear projection techniques on a subspace spanned by the principal eigenvectors (those corresponding to the largest eigenvalues) of the input covariance matrix. The LDA approach was proposed by Fisher firstly which identifies directions in space along which separation of the projections is maximized. While the LDA is not always superior to the PCA in terms of recognition accuracy, the PCA + LDA approach has been successfully applied in some face recognition applications.

However, the PCA and/or LDA based facial recognition may fail if the facial samples are captured in different directions. While the face images are captured from different sight depths and directions, the accuracy of the PCA based recognition will be reduced heavily. There are many face recognition methods provided to overcome such problems, some focus on the feature extraction. How to use the smallest dimensions to replace the most representation's feature and classification is the most important issue in this paper.

Since the facial feature selection process can be considered as a problem of global combinatorial optimization in machine learning, the PSO-SVM is usually used as an optimal classifier of the system. In the part of the classification, the PSO is used to implement feature selections, and the SVMs [3, 4] serve as fitness functions of the PSO for the classification problem. The advantage of the SVM in the multidimensional space is that it can quickly and correctly classify samples to find out the best support vector [5]. The PSO algorithm is a kind of imitation of birds clustering phenomenon algorithm [6–9], and in this paper, the PSO will be issued to correct the parameters of the SVM so that the image recognition process can be faster and more stable.

The rest of this paper is organized as follows: in Section 2, the image detection scheme is described which consists of smoothing filter, connected position, and ellipse detection. Section 3 describes the face recognition system which includes the adjustment of the dimensions, the PCA, and the PSO-SVM classifier. Section 4, experimental results are presented to demonstrate the feasibility of the proposed scheme. Finally, some conclusions are made in Section 5.

2. The Face Detection System

In order to detect the face from an image, there are several key steps necessarily processed. The first is to detect the area of skin color; the second is to reduce the noise; the others are to distinguish which one or ones are the face by ellipse detection and to separate the face block from the image background. Figure 1 illustrates the flowchart of the proposed face detection scheme. The images captured by the webcam in sequence will be sent to the face detection system firstly, and then the area of the human face is separated from the complex background by the skin-color detection. Next, the noise is going to be filtered by the noise reduction. Finally, the captured contours will aid in locating the position of the human face. The block of the human face will only remain in the image after the face detection.

2.1. Skin-Color Detection. The RGB image is always with chromatism because of changeable illumination in every capture. For skin-color detection, a reliable color range defined as the skin-color is principal. Since the color values in RBG are so sensitive to varying illumination, most approaches adopt the color model of YCbCr replacing the RGB because the value Y is relative to luminance and the values of Cb and Cr are relative to chroma.

YCbCr color space can be regarded as a modified YUV color model. However, YCbCr is not an absolute color space which is a way of encoding RGB information. In YCbCr color space, Y is the luma component; Cb and Cr represent the blueness and the redness chroma components, respectively. The transform defined in the ITU-R BT.601 standard for digital component video the YCbCr color space can be translated from the RGB color space by

$$\begin{bmatrix} \mathrm{Y} \\ \mathrm{Cb} \\ \mathrm{Cr} \end{bmatrix} = \begin{bmatrix} 65.481 & 128.553 & 24.966 \\ -37.797 & -74.203 & 112.0 \\ 112.0 & -93.786 & -18.214 \end{bmatrix} \begin{bmatrix} \mathrm{R} \\ \mathrm{G} \\ \mathrm{B} \end{bmatrix} + \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix}. \tag{1}$$

The resultant Y is just between 16 and 235 because the values from 0 to 15 are called footroom and the values from 236 to 255 are called headroom. Besides, in this paper, the skin-color region of Cb calculated is between 71 and 127; the skin-color region of Cr is between 130 and 170 from the 150 sample images. By the skin-color regions, the human face block will be easily distinguished from the image according to the values of Cb and Cr.

The low-pass filter (LPF) was used to eliminate the small noises and connect the part of the incomplete image. The LPF makes the image for filtering become more smooth and uniform. In general, the start of the mask is set to be at the top left pixel of the preprocess image, and it will scan the whole image from left to right. It is also referred to the neighboring 8 pixels for processing. Generally the sizes of the masks are 3×3, 5×5, and 7×7. The larger the mask, the larger the filtering effect, but the calculation becomes relatively large. The 3×3 mask was used for low-pass filtering in this paper.

Besides, the opening operation of the morphology is usually used to eliminate noises in an image. The opening operation included two operands with erosion and dilation. First, it will use the erosion for the binary image and then use the dilation for its result. After this procedure, the noises will be removed from the image.

The opening operation of the morphology can not only remove a part of the noise pixels of the binary image but also make more complete region of the skin-color.

After finishing the procedure of the noise reduction, the connected component labeling (CCL) was used to find the location of the human face. This method was mainly used to find the connected pixels of the same object in the image. It marks each block by different labels and counts size, height, and width of each independent object in the image. The 4-connected was used to label pixels in this paper. It starts to scan the prelabeled binary image from top left. In the coordinates (x, y) of the pixel, determine the presence of 255 on a pixel before checking the right $(x + 1, y)$, left $(x - 1, y)$, top $(x, y+1)$, and bottom $(x, y-1)$, for any other 255 values. If yes, it will record its coordinates and the pixel is set to 0. Then there is the recursive process of checking all the pixels for presence of 255 valued pixels until none is present. Then the group with the most number of 255 valued pixels is searched and labeled.

While finishing the recursive scanning of the image, the group objects for image labeling were calculated. The biggest area of the region was found from all numbered regions, that is, the region of the human face. This region was scanned to find the black background of the facial boundary. Then the relative coordinates $(X_{\min}, Y_{\min})$ and $(X_{\max}, X_{\max})$ were employed, to capture the facial region from the original image. The biggest region for labeling of the binary image is shown in Figure 2.

The Sobel edge detection [10] is used to detect the edge. For face detection, since there are obvious differences between the background and the region of the skin color in the facial binary image, the edge detection usually employed the first derivative of the image to estimate the regional edge and was used to calculate the size of the gradient for image processing. The Sobel edge detection employs the gray scale difference at the position of the edge and weighted the top and bottom, or left and right, to detect the edge of the object.

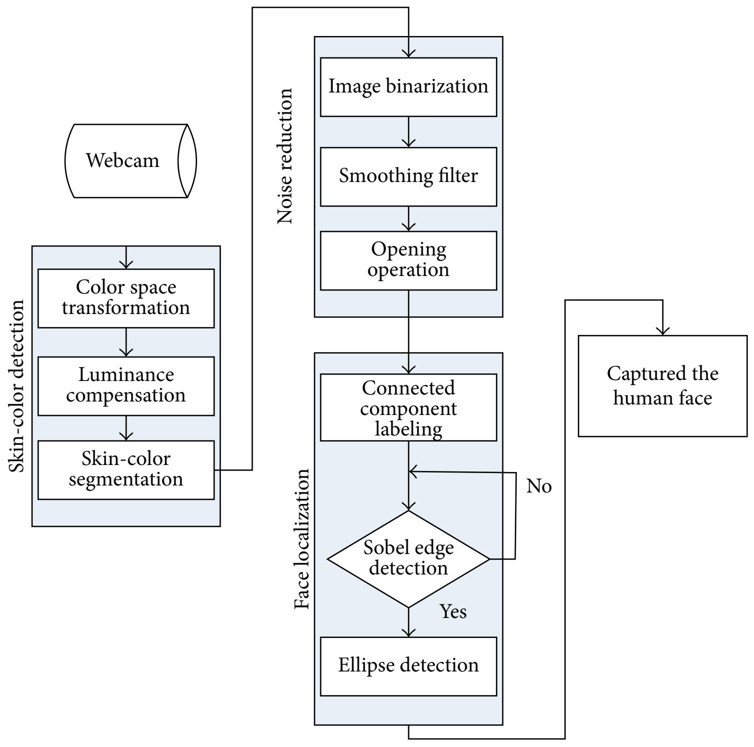


Figure 1: Face detection system flowchart.

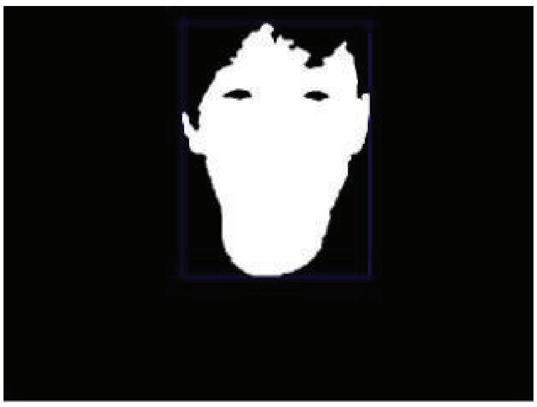


Figure 2: The biggest region for labeling of the binary image.

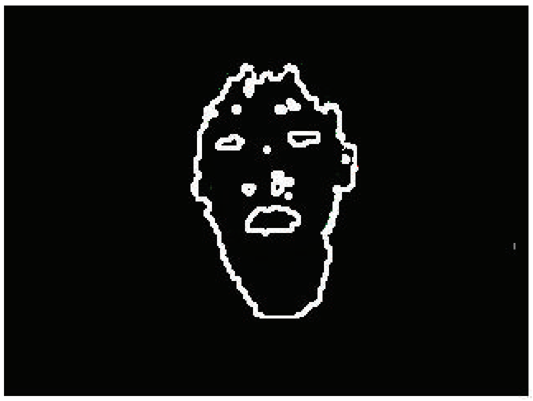


Figure 3: The result of the Sobel edge detection.

Through the Sobel edge detection processing, the region of the face is shown in Figure 3.

Since the size of a human face is similar to the elliptical model with the ratio of the vertical axis and the horizontal axis as approximately 1.2 : 1, thus, an ellipse mask is used to locate the human face, which marks a boundary to extract the edge of the image. After the edge detection, the region that resembles more the shape of an ellipse will provide the position of the face. The center of the ellipse with the use of its circumference and the length of its axis can help to determine its position, shape, and size. In order to meet the size of the facial change, the elliptical model must adjust the ratio of the length of the axis to scan the image in real time. Therefore, we designed an elliptical model that determines the coordinates of the center of the ellipse (x_0, y_0) with x as the radius of the horizontal axis and y for that of the vertical axis.

Face detection is then finished after the ellipse detection processing. The ellipse detection cooperates with the connected component labeling to find the facial region. The locations of the detected faces in the original image and the binary image are shown in Figures 3 and 4, respectively. Next, the unnecessary region outside of the elliptical range was

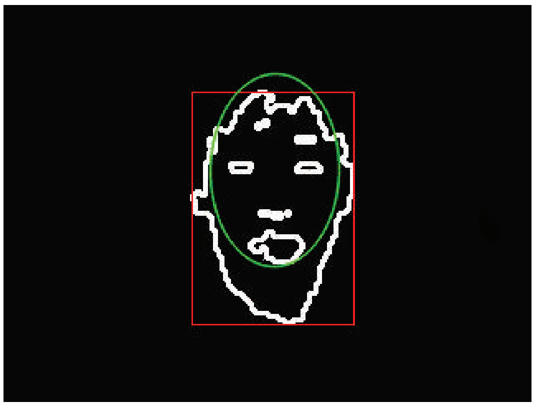


Figure 4: Face location of the binary image.

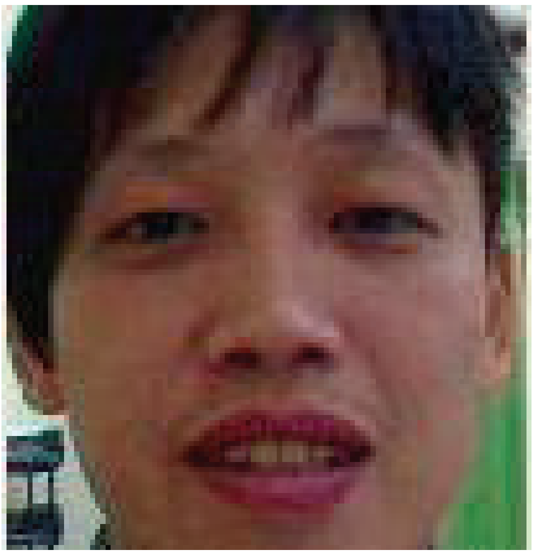


Figure 5: The result of the captured facial image.

removed. This procedure will help in eliminating the image of the neck and locate the facial region. The relative coordinate was used to capture the face for the original image. The result of the captured facial image is shown in Figure 5.

3. The Face Recognition System

3.1. Description of the Face Recognition System. The face recognition processing will be executed after finishing face detection. However, since the dimensions of the facial images are not the same, the normalization process for the facial image becomes important. The processed image to be analyzed contains the same information of the environment to make the dimensions of the facial regions the same for each face. After the image normalization processing, the captured image serves as the input data for the face recognition system. And then, principal component analysis is utilized to calculate the feature. This method could reduce the dimension of the image and save computation time. It could surmount the problems of the changed expression or

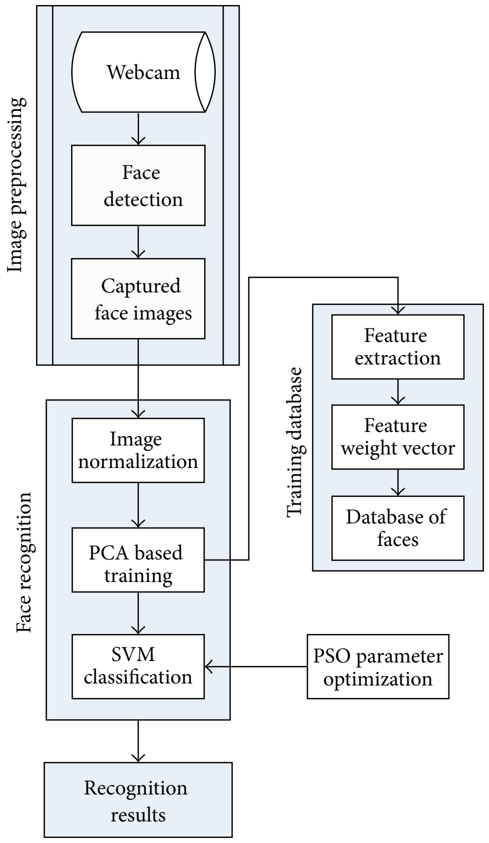


Figure 6: The scheme of the face recognition system.

presence of glasses on the face, because it treats the whole face as a feature.

The weighting vectors of the processed facial image will be calculated by using PCA. All the weighting vectors are collected to build the database for the face recognition system. The user identity could be determined by the support vector machine that compares the current image with the image in the database. This way is a fast and accurate classifier that can be applied to classification and comparison applications. It shows that the velocity and accuracy of the face recognition system can be increased through the support vector machine processed. At the same time, the particle swarm optimization (PSO) is used to design the parameter of the support vector machine. This method makes the face recognition system complete and quick. The face recognition system flowchart proposed in this paper is as shown in Figure 6.

3.2. Image Normalization. Achieving high recognition rate for the face recognition system does not only need a good recognition algorithm but also needs a robust face image for the preprocessing of the image. It could reduce the difference for each input image and changed each image to the same dimension for the database. Therefore the bilinear interpolation is used to amend the image that detects the face with the size of the pixel at 80 $\times$ 100. Through the bilinear interpolation processing, feature extraction and recognition are executed for the facial image.

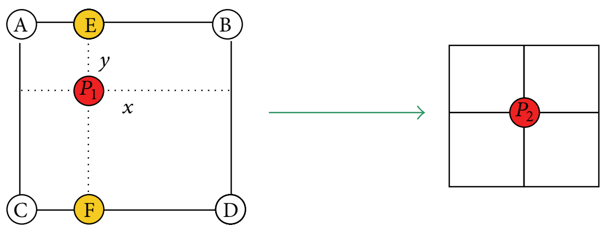


FIGURE 7: The diagram of the bilinear interpolation.

The relation of the pixel coordinate between the original image and final image is not the same as the general image processing when the image is zoomed. A pixel of the original image could project into many pixels for the final image, when the images are zoomed in. Similarly, many pixels of the original image could project onto a pixel for the final image, when the images are zoomed out. Therefore, the final image must search for a pixel to substitute the original image. And with this in mind, the interpolation process is needed to calculate the pixel of the final image; otherwise the final image would produce tremendous distortion. Actually the interpolation used discrete samplings to calculate the continuous function which passes through the coordinates of these samples and then employ this function to find the value of nonsamplings. The image zooming could define the signal resampling because the image was a signal that is composed of the two-dimensional discrete sampling. Then the bilinear interpolation adopted the four neighboring pixels to calculate the new pixel. The diagram of the bilinear interpolation is shown in Figure 7.

In order to obtain the pixel P_2 as a projection of the original image, P_2 is supposed to project onto P_1; thus the four neighboring pixels (A, B, C, D) were used to calculate the distances between P_1 and the coordinates of four pixels. If the four pixels are closer to P_1 then the contribution is large for P_2. Conversely the influence is smaller if the distance is farther. Therefore the effect is inversely proportional to the distance. In fact, the bilinear interpolation calculated the linear interpolation continuously for three times. The first interpolation was calculated to be the influence between two points (A, B) and P_2 in order to obtain pixel E. The equation of the first interpolation is expressed in

$$\mathrm{E} = (1-x)\,\mathrm{A} + x\mathrm{B}. \tag{2}$$

And then the second interpolation was calculated using the influence between two points (C, D) and P_2 to obtain pixel F. The equation of the second interpolation is expressed in

$$\mathrm{F} = (1-x)\,\mathrm{C} + x\mathrm{D}. \tag{3}$$

Finally, the third interpolation was used to calculate the pixel of P_2 for two points (E, F). The equation of the third interpolation is expressed in

$$P_2 = (1-y)\,\mathrm{E} + y\mathrm{F}, \tag{4}$$

where x represents the relative horizontal distance of P_1 in relation to the four neighboring pixels; y is the relative vertical distance of P_1 corresponding to the four neighboring pixels. If the distance from pixel to pixel is one unit as assumed in this paper, it can be represented by $0 < x$ and $y < 1$, then the adjusted facial dimension is shown in Figure 8.

3.3. Principal Component Analysis. After normalization, if the face recognition is directly implemented it would cost a lot of computing time. This is due to the fact that all the information of the original image is spread in each pixel; hence, there is the need to reduce the dimensions of the image. And then, the suitable features are captured to express a lot of information in lower dimensions. It could reduce many variations for data with the use of PCA and applying some mutually independent linear combinations to substitute for the original data. Through the linear combination computing, the difference of the variation was the large influence of the data. This analysis made data to display the biggest individual differences. Below is the process for implementing PCA.

If there were N facial images as the training samples the original feature parameters were $\{X_1, X_2, \ldots, X_N\}$. The objective of PCA was order to find the linear transformation matrix P with a size of $n \times m$. It extracts feature parameter X_k of the original n dimension to transform the more representative of the feature parameter Z_k that the dimension was m $(m \le n)$. The equation of the transformation expressed in

$$Z_k = P^T X_k, \quad k = 1, 2, \ldots, N. \tag{5}$$

Before the transformation computing, the mean vector was $\overline{X}$. After the transformation, the mean vector was $\overline{Z}$ expressed in

$$\overline{Z} = \frac{1}{N}\sum_{k=1}^{N} Z_k = \frac{1}{N}\sum_{k=1}^{N} P^T X_k = P^T\left(\frac{1}{N}\sum_{k=1}^{N} X_k\right) = P^T\overline{X}. \tag{6}$$

Then the total scatter matrix was used to indicate the dispersion of all feature parameters that were opposite to the mean vector $\overline{X}$. Before the transformation computation, the total scatter matrix was S_{t_x} which has a size of $n \times n$. The equation of the total scatter matrix S_{t_x} is expressed in

$$S_{t_x} = \sum_{k=1}^{N}\left(X_k - \overline{X}\right)\left(X_k - \overline{X}\right)^T. \tag{7}$$

Through (18) to (20), the total scatter matrix S_{t_z} can be obtained having a size of $m \times m$ after the transformation. The equation of the total scatter matrix S_{t_z} is expressed in

$$\begin{aligned} S_{t_z} &= \sum_{k=1}^{N}\left(Z_k - \overline{Z}\right)\left(Z_k - \overline{Z}\right)^T \\ &= \sum_{k=1}^{N}\left(P^T X_k - P^T\overline{X}\right)\left(P^T X_k - P^T\overline{X}\right)^T \\ &= P^T\left(\sum_{k=1}^{N}\left(X_k - \overline{X}\right)\left(X_k - \overline{X}\right)^T\right)P = P^T S_{t_z} P. \end{aligned} \tag{8}$$

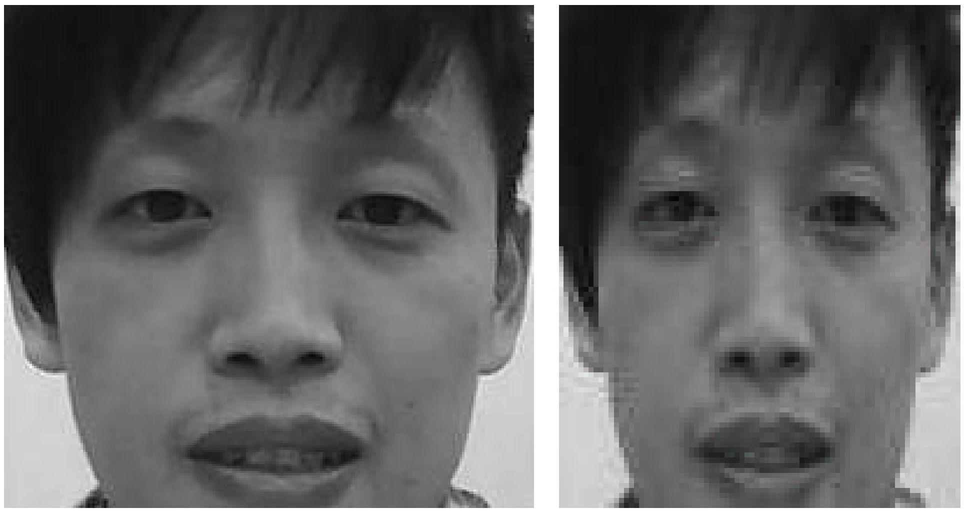

(a) The original image (b) The image after facial dimension adjusting (80 × 100)

Figure 8: Facial dimension adjusting.

In order to increase the dispersion between the mean value and the feature parameters of the transformation, the transformation matrix P_{opt} should be calculated that could make the maximization of S_{t_z}. The equation of the transformation matrix P_{opt} is expressed in

$$P_{\text{opt}} = \arg\max_{P}\left(P^T S_{t_z} P\right). \tag{9}$$

According to the theory of linear algebra, the trace or the determinant can be used to express the element's distribution. Therefore, (9) could be rewritten as

$$P_{\text{opt}} = \arg\max_{P}\ \operatorname{tr}\left(P^T S_{t_z} P\right). \tag{10}$$

In (10), in order to limit the value of tr $(P^T S_{t_z} P)$ and avoid the occurrence of infinity, a limit condition as $P^T P = I$ was added for the transformation matrix P that has a size of $n\times m$:

$$F(P) = P^T S_{t_z} P - \lambda\left(P^T P - 1\right). \tag{11}$$

As follows, the value of $F(P)$ is maximum while the first derivative for P is zero:

$$\frac{\partial F(P)}{\partial P} = 2S_{t_z} P - 2\lambda P = 0. \tag{12}$$

Then, (12) becomes (13) through simplification using transposition:

$$\left(S_{t_z} - \lambda I\right) P = 0. \tag{13}$$

In (13), it needs to compute the eigenvectors of matrix S_{t_z} by composing the matrix for P. The eigenvalues and the eigenvectors are the special form in matrix algebra and its elements could be restructured in the matrix. In addition, the important information would be concentrated in the larger eigenvalue that would correspond to the eigenvectors.

The advantages of PCA in face recognition can be divided into three advantages. First, it could quickly and easily calculate the result. Second, PCA could retain the largest information of the projection data in the linear projection. Third, PCA used the whole face to do feature extraction that could overcome the presence of glasses and the changes of the facial expression. Below is the operational procedure of PCA.

After normalization, M number of facial images was trained using PCA. The size of each sample was $N\times N$ matrix. Next, each sample is rearranged as the augmented vector Γ which has the size $N^2 \times 1$ as shown in (14). $\Gamma_1, \Gamma_2, \ldots, \Gamma_M$ represent the M facial images processed.

Each facial sample corresponded to Γ, and the mean vector Ψ was calculated by the M amount of Γ as expressed in

$$\Psi = \frac{1}{M}\sum_{k=1}^{M}\Gamma_k. \tag{14}$$

The mean vector Ψ is the mean face which indicates the mutual parts of all face. And then the mutual parts for M facial images were deleted to highlight the different parts between them. Therefore the different image vector of each image was obtained as shown in

$$\varphi_k = \Gamma_k - \Psi \quad k = 1, 2, 3, \ldots, M, \tag{15}$$

wherein matrix A equals $[\varphi_1, \varphi_2, \varphi_3, \ldots, \varphi_M]$ that had the size $N^2 \times M$ and the covariance matrix C of all faces was defined as

$$C = \frac{1}{M}\sum_{k=1}^{M}\phi_k\phi_k^T = AA^T. \tag{16}$$

The eigenvalue λ_k and the eigenvector u_k of the matrix C are expressed in

$$Cu_k = \lambda_k u_k \quad k = 1, 2, 3, \ldots, N^2, \tag{17}$$

where $\lambda_k = (1/M)\sum_{j=1}^{M}(u_k^T\varphi_j)^2$ and $\varphi = \{\varphi_1, \varphi_2, \ldots, \varphi_M\}$.

Since the size of matrix A was $N^2 \times M$, it makes the size of matrix C be $N^2 \times N^2$. For such a large matrix calculating the eigenvalues and eigenvectors is time consuming. Thus, if the dimensions of the matrix could be reduced, it could effectively save calculation time. Therefore, the matrix A^TA must be calculated first and the dimensions of the matrix must be reduced as $M \times M$ to obtain the eigenvector v_k which expressed in

$$A^TAv_k = \mu_k v_k \quad k = 1, 2, 3, \ldots, N^2. \tag{18}$$

Equation (18) multiplies by the matrix A to obtain

$$AA^TAv_k = \mu_k Av_k, \tag{19}$$

in which AA^T has the same eigenvalue and eigenvector with A^TA, because the matrix C equals AA^T. By comparing (17) and (19), (20) can be obtained as follows:

$$u_k = Av_k \qquad \lambda_k = \mu_k. \tag{20}$$

By using (18), the matrix of A^TA is used to calculate the eigenvector v_k which determines the eigenvalue u_k. It is considered as an eigenface as expressed in

$$u_k = \sum_{j=1}^{M}\varphi_j v_{kj}, \quad i = 1, 2, \ldots, M. \tag{21}$$

The vector $\Gamma_1, \Gamma_2, \ldots, \Gamma_M$ of the individual facial image combined with the corresponding eigenvector to build the feature space. And we calculate the weight vector V from the feature space as expressed in

$$\begin{aligned}\Omega_k = u_k^T(\Gamma - \Psi) = u_k^T\varphi \qquad & V = [\varphi_1, \varphi_2, \varphi_3, \ldots, \varphi_M] \\ & k = 1, 2, 3, \ldots, M.\end{aligned} \tag{22}$$

Finally, each training sample Γ_i of the face is inputted to substitute the Γ of (22) and calculate the eigenvector V_i in the feature space. Through the computation, the matrix V will be taken as the database of the facial images after the training.

3.4. SVM Based Classification for Face Recognition. Euclidean distance based methods [11] aim to calculate the difference value of the distance measurement, which are usually used in pattern recognition system. The resemblance computation directly calculates the difference between two vectors. The smaller value means that the two vectors are closer. It also indicates that the features of two images are also closer, and there is the presence of similarity in the images. The equation of the Euclidean distance is expressed in

$$\begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ \vdots \\ d_M \end{bmatrix} = \begin{bmatrix} \sqrt{\sum_{i=1}^{k}(a_{i1} - b_{i1})^2} \\ \sqrt{\sum_{i=1}^{k}(a_{i2} - b_{i2})^2} \\ \vdots \\ \sqrt{\sum_{i=1}^{k}(a_{iM} - b_{iM})^2} \end{bmatrix}, \tag{23}$$

in which $d_1, d_2, d_3, \ldots, d_M$ are the Euclidean distances between the eigenvector of each image and the eigenvector of the target image; a_i is the ith element of the input eigenvector; b_i is the ith element of the eigenvector saved in the database; k is the dimension of the eigenvector; and M is the Mth image saved in the database.

In general, if the Euclidean distance method is directly used in face recognition system, it would require a lot of computation time, because the Euclidean distance applies bubble sort for comparison. For example, if there are one thousand data in the database, it will require the process to be performed one thousand times and the larger the database the longer the comparison time. Therefore, support vector machine is used to assist such face recognition problems. The calculated Euclidean distances are inputted the feature space of the support vector machine to perform the comparison and classification.

The most important goal of face recognition system is how to raise the accuracy and shorten the computing time of the system. In fact, principal component analysis could indeed raise the accuracy as shown in previous experiments. However, the comparison of the Euclidean distance would require a lot of computing time. Therefore, the design of the support vector machine classification shortens the program's computing time for the face recognition system. The SVM is performed through the following process. First, the hyperplane is designed to classify the Euclidean distance as shown in Figure 9.

In the figure, d_i denotes the Euclidean distance between the target image and ith image of the database. There are M images in the database with $i = 1, 2, 3, \ldots, M$. Besides, d_{mean} indicates the mean value of the Euclidean distance between the target image and all images in the database. The equation of d_{mean} is expressed in

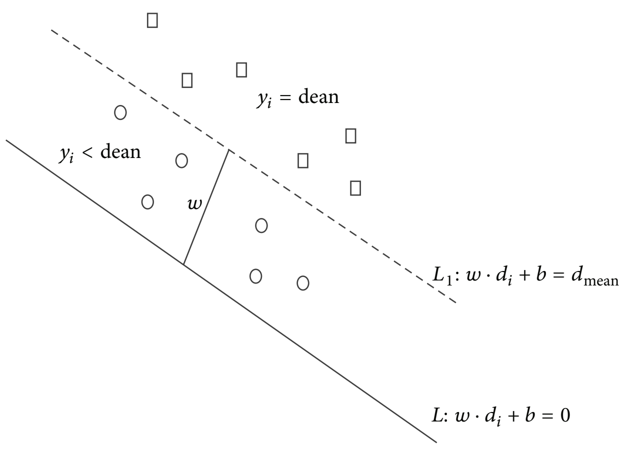
$$d_{\text{mean}} = \frac{\sum_{i=1}^{M} d_i}{M}. \tag{24}$$

In this method, the training data is shown:

$$\begin{aligned}(d_1, y_1), \ldots, (d_M, y_M), \quad & d_1 \in R^n \\ i = 1, 2, \ldots, M \quad & y_i \in d_{\text{mean}}.\end{aligned} \tag{25}$$

Then, consider

$$(w \cdot d_i) + b \geq d_{\text{mean}} \longrightarrow y_i = d_{\text{mean}}, \quad i = 1, 2, \ldots, M \tag{26}$$



y_i = dean
y_i < dean
w
L_1: $w \cdot d_i + b = d_{\text{mean}}$
L: $w \cdot d_i + b = 0$

Figure 9: Hyperplane of the Euclidean distance.

in which w is the normal vector of the hyperplane and b is the deviation value. In order to find the division of the hyperplane the question of quadratic optimization needs to be resolved. The constraint is expressed in

$$y_i (w \cdot d_i - b) \geq d_{\text{mean}}, \quad i = 1, 2, \ldots, M. \tag{27}$$

The minimum value of $d(w) = (1/2)\|w\|^2$ must be determined because the equation above is quadratic with a linear constraint. This is a typical quadratic optimization problem. So, the Lagrange multiplier is resolved to the question of quadratic optimization with linear constraint to obtain

$$L(w, b, \alpha) = \frac{1}{2} w^2 - \sum_{i=1}^{M} \alpha_i \left[y_i (w \cdot d_i + b) - d_{\text{mean}} \right] \quad \alpha_i \geq 0. \tag{28}$$

However, the support vector machine still does not produce the optimal solution. The method in which the problem is dealt with was to address the dual question. The dual question is expressed in

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{M} \alpha_i y_i d_i = 0 \longrightarrow w = \sum_{i=1}^{M} \alpha_i y_i d_i.$$
$$\frac{\partial L}{\partial b} = \sum_{i=1}^{M} \alpha_i y_i = 0. \tag{29}$$

And a new equation was left after performing the substitution as expressed in

$$L_D = \sum \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j d_i d_j. \tag{30}$$

After having determined the optimal solution to the dual question, each Lagrange modulus α_i is mapped onto each trained data. If $\alpha_i \geq 0$, this means that the data is the support vector of this question and it is located in the margin separating the hyperplane. The final function is expressed in

$$f(d) = \operatorname{sgn} \left[\sum_{i=1}^{M} \alpha_i y_i (d_i d) + b \right]. \tag{31}$$

By the method of support vector machine, d_i found to be located between zero and d_{mean} in the hyperplane are retained. These data are used to recalculate the d_{mean} and reexecute the classification by the same way. This procedure would be repeated until data are all plotted before terminating the program. This data represents the Euclidean distance closest to the target image. Finally, this data shows the image which is the recognition results needed.

3.5. PSO-SVM Classifier. The particle swarm optimization (PSO) was proposed by [12–15]. This method is a concept of swarm intelligence which belongs to the territory of the evolutionary search. This algorithm is an evolutionary optimization implementation similar to the genetic algorithm (GA). First, they could produce the initial solution and apply evolution to find the optimal solution. The difference is that PSO does not have the procedures of crossover and mutation. It belongs to the signal-channel messaging, and the process of searching update is changed according to the current optimal solution. Therefore, in the general optimal questions, the PSO converges to the optimal solution more quickly than the GA.

The origin of the PSO is from the concept of the predation on bird populations. Kennedy used this concept to research the solution of the optimal question, and this question is just like a bird which flies in space, called particle. There is a fitness function of the objective function mapping for all particles that moved in the space. In addition, each particle has the velocity to determine the direction and the distance of the movement. The particles fligh in the solution space by the individual successful experience and the trajectory of the best particle in the current population. In addition, each particle could search independently in the PSO space. When the individual particle found the optimization of the function, the best search variable will be recorded in the individual memory. Thus, each particle owns the best memory of the search variable for itself. It would change the next search direction by the individual best memory of the search variable, and this procedure is called the cognition-only model. Every search would compare the optimization extent between the individual best search variable and the best search variable of the population. This procedure would adjust the variable memory of the best function for the population. At the same time, each particle could change the search velocity of the particle for next time according to the best variable memory of the population, and this process was called the social-only model. Through the evolutionary computation, the PSO would calculate the optimal solution according to the best fitness value of the particles [16]. The flowchart of the PSO is shown in Figure 10.

In the space of the SVM, it requires the design of an important parameter w. Therefore, PSO is applied to optimize this parameter. The particle's position in the PSO space was used to substitute the parameter w of the SVM space. Through the evolutionary computation, each particle would update the position and the parameter w would be updated continually too. By this procedure, we could find the optimization value of the parameter w.

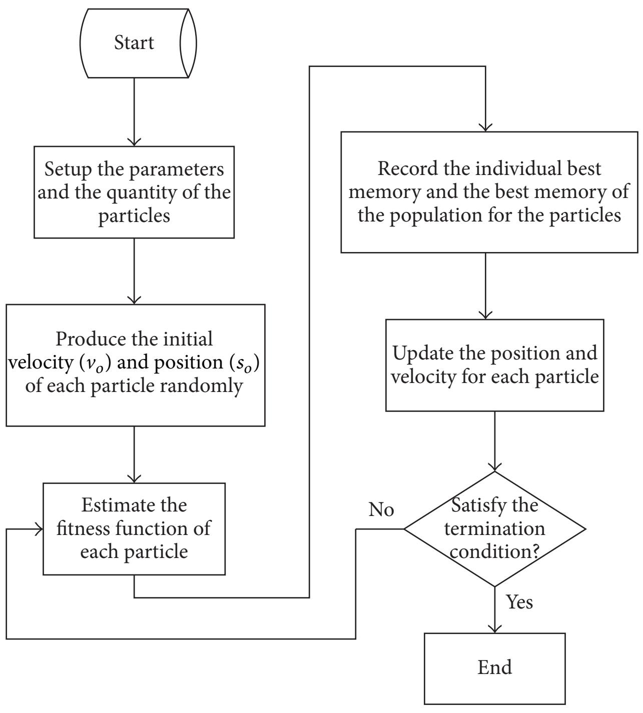

Start
Setup the parameters and the quantity of the particles
Produce the initial velocity (v_o) and position (s_o) of each particle randomly
Estimate the fitness function of each particle
Record the individual best memory and the best memory of the population for the particles
Update the position and velocity for each particle
Satisfy the termination condition?
No
Yes
End

Figure 10: The flowchart of the PSO.

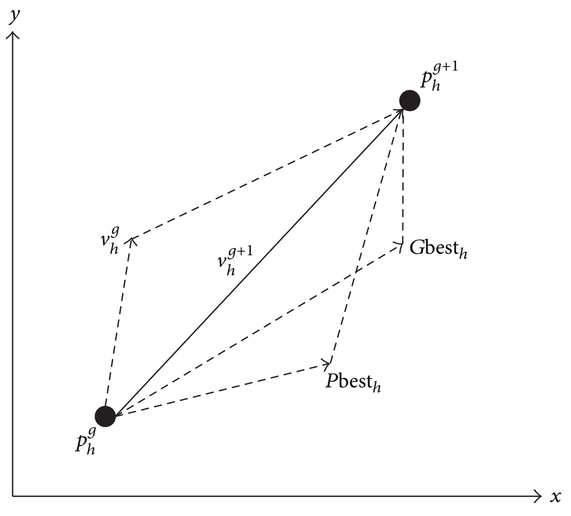


Figure 11: The PSO search in the space.

The PSO produces the particles of the initial population randomly and through the evolutionary computation to find the optimal solution for the function. In each evolution, the particle would change the individual search direction by two search memories. The first search is the optimal individual variable memory *P*best and the other is the optimal variable memory of the population *G*best. After the computation, the PSO would calculate the optimal solution according to the optimal variable memory. Figure 11 shows the PSO search in a particular space.

Having a range of $x \in [0,20]$ and $y \in [0,20]$, it is supposed that the coordinate of particle's position was (p_{hx}^g, p_{hy}^g); therefore the parameter w_h^g of the SVM could be calculated by

$$w_h^g = \frac{\sqrt{\left(p_{hx}^g\right)^2 + \left(p_{hy}^g\right)^2}}{28.29}. \tag{32}$$

If the set p^g with N particles is called the population in the gth generation, it can be expressed in

$$p^g = \left(p_1^g, p_2^g, \ldots, p_h^g, \ldots, p_N^g\right). \tag{33}$$

The velocity vector and position vector of the hth particle ($h \in [1,2,3,\ldots,N]$) in the gth generation ($g \in [1,2,3,\ldots,G]$) are expressed in (34) and (35), respectively, as follows:

$$v_h^g = \left(v_1^g, v_2^g, \ldots, v_h^g, \ldots, v_N^g\right), \tag{34}$$

$$p_h^g = \left(p_1^g, p_2^g, \ldots, p_h^g, \ldots, p_N^g\right), \tag{35}$$

in which the position of the hth particle in the gth generation is p_h^g. Also, the processes of the PSO could be explained as follows.

Step 1. The initialization of the PSO was set to $g = 1$, $F_1^{p\text{best}} = F_2^{p\text{best}} = \cdots = F_N^{p\text{best}} = 0$. The number of the particles (N), the number of the generation (G), and four parameters of C_1, C_2, $\gamma_{\max}$, and $\gamma_{\min}$ are given.

Step 2. The initial velocity $v_h^1 = (v_1^1, v_2^1, \ldots, v_h^1, \ldots, v_N^1)$ and the initial position $p_h^1 = (p_1^1, p_2^1, \ldots, p_h^1, \ldots, p_N^1)$ of N particles are created.

Step 3. The fitness value of each particle in the gth generation is calculated by using (36). fit(·) is the fitness function which is expressed by the reciprocal computing time of the recognition system:

$$F\left(p_h^g\right) = \operatorname{fit}\left(p_h^g\right), \quad h = 1,2,\ldots,N. \tag{36}$$

Step 4. The $F_h^{P\text{best}}$ and $p_h^{P\text{best}}$ for each particle were determined and the equation of $F_h^{P\text{best}}$ is expressed in (37), and the equation of $p_h^{P\text{best}}$ is expressed in (38) as follows:

$$F_h^{P\text{best}} = \begin{cases} F_h^g, & \text{if } F_h^{P\text{best}} \le F_h^g \\ F_h^{P\text{best}}, & \text{otherwise,} \end{cases} \quad h \in \{1,2,\ldots,N\}, \tag{37}$$

$$p_h^{P\text{best}} = \begin{cases} p_h^g, & \text{if } F_h^{P\text{best}} \le F_h^g \\ p_h^{P\text{best}}, & \text{otherwise,} \end{cases} \quad h \in \{1,2,\ldots,N\}, \tag{38}$$

where, $p_h^{P\text{best}}$was the individual optimal fitness value $F_h^{P\text{best}}$ form the starting to the current generation.

Step 5. The index q of the particle with the highest fitness function is designed by

$$q = \arg \max_{h \in \{1,2,\ldots,N\}} F_h^{P\text{best}}. \tag{39}$$

And then, F^{Gbest} and p^{Gbest} are determined by

$$F^{\text{Gbest}} = F_q^{P\text{best}} = \max_{h\in\{1,2,\dots,N\}} F_h^{P\text{best}}, \qquad p^{\text{Gbest}} = p_q^{P\text{best}}, \quad (40)$$

in which p^{Gbest} is the position vector of the particle with the global optimal fitness value F^{Gbest} from the starting to the current generation.

Step 6. If $g = G$, and then go to Step 10, otherwise go to Step 7.

Step 7. The velocity vector is updated for each particle by

$$\begin{aligned} v_h^{g+1} &= \gamma \cdot v_h^g + c_1 \cdot \text{rand1}\,() \cdot \left(p_h^{P\text{best}} - p_h^g\right) \\ &\quad + c_2 \cdot \text{rand2}\,() \cdot \left(p^{\text{Gbest}} - p_h^g\right), \end{aligned} \quad (41)$$

where v_h^g is the current velocity vector of the hth particle in the gth generation. v_h^{g+1} is the next velocity vector of the hth particle in the $(g+1)$th generation. rand1() and rand2() are two uniformly distributed random numbers in $[0,1]$. C_1 and C_2 are the constant values that are set to 2. γ was the weight value which is defined by

$$\gamma = \gamma_{\max} - \frac{\gamma_{\max} - \gamma_{\min}}{G} \cdot g, \quad (42)$$

where $\gamma_{\min}$ and $\gamma_{\max}$ are, respectively, the minimum value and the maximum value of γ, and the $\gamma_{\min}$ is set to 0.4; the $\gamma_{\max}$ is set to 0.9.

Step 8. Then the position vector is updated for each particle by

$$p_h^{g+1} = p_h^g + v_h^{g+1}, \quad (43)$$

where p_h^g is the current position vector of the hth particle in the gth generation. p_h^{g+1} is the next position vector of the hth particle in the $(g+1)$th generation.

Step 9. Let $g = g + 1$ and go to Step 3.

Step 10. The optimal position vector of the particle p^{Gbest} with the optimal fitness value F^{Gbest} is determined.

After the above procedures, the particle moves in the generation that would create the new parameter w for the SVM. The computing time of the face recognition system is compared for each parameter w, and the best w in current generation is searched for. It projects the space of the PSO to be the optimal global solution for the next generation. Through this method, the best parameter w of the SVM could be found to reduce the computing time of the face recognition system.

4. Experimental Results

4.1. Actual Experiments. In the experiments of the face recognition system, the facial database used contained ten people as training samples. Each person has ten facial images used as the input samples, and then there are one hundred test samples in the real-time face recognition system. Through the real-time detection, the current facial image is captured to be the test face. The PCA-SVM-PSO algorithm is used to execute the face recognition system. The Euclidean distance between the test face and the samples of the facial database would be classified, and then it would find the sample in the face database in which the Euclidean distance is closer to the test face to be the result. The experiments of the real-time face recognition system are shown in Figure 12.

Table 1: The comparison with the experiments with fifty samples.

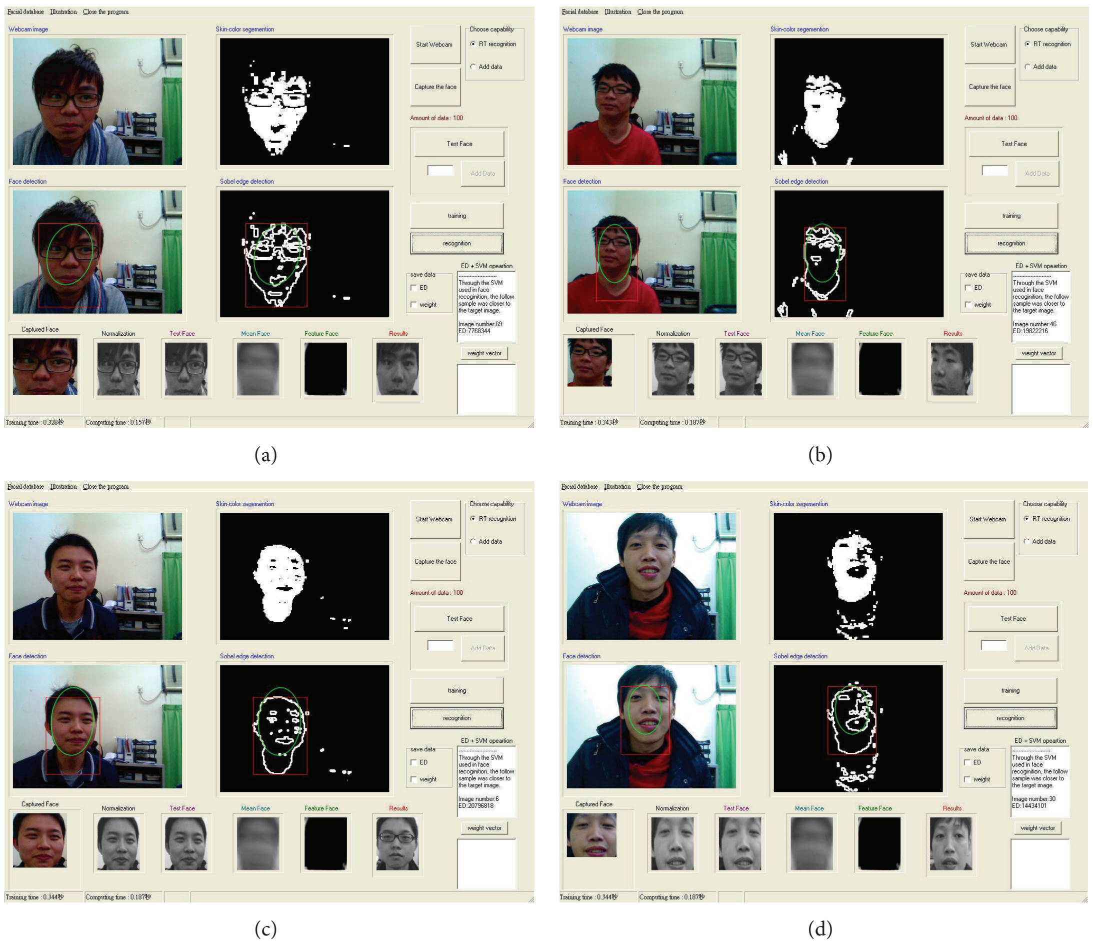
	PCA + ED	PCA + SVM	PCA + SVM + PSO
Sample numbers	50	50	50
Test times	80	80	80
Successful numbers	74	76	76
Success rate	93%	95%	95%
Average training time	0.179 s	0.177 s	0.177 s
Average computing time	0.149 s	0.137 s	0.129 s

Table 2: The comparison with the experiments with one hundred samples.

	PCA + ED	PCA + SVM	PCA + SVM + PSO
Sample numbers	100	100	100
Test times	80	80	80
Successful numbers	71	73	74
Success rate	89%	91%	93%
Average training time	0.344 s	0.345 s	0.344 s
Average computing time	0.296 s	0.203 s	0.180 s

4.2. Experimental Comparison. In the part of the comparison with the experiments, the main comparisons made are on the training time, computing time, and recognition rate for three ways. The first way is using PCA and Euclidean distance with the bubble sort to the face recognition system. The second is to apply the PCA and SVM based classification to the face recognition system. The third is to adopt PSO as a parameter adjusted scheme for SVM based classification. There are fifty samples and one hundred samples used in the analyses of experimental results. Table 1 expresses the comparison with the experiments with fifty samples. Table 2 summarizes the comparison with the experiments with one hundred samples, in which ED denotes the Euclidean distance with the bubble sort.

From the results in Tables 1 and 2, one can see that the proposed algorithm has really raised the recognition



(a)

Amount of data : 100

ED + SVM opeartion

Through the SVM used in face recoginition, the follow sample was closer to the target image.

Image number:69
ED:7768344

Training time : 0.328秒 Computing time : 0.157秒

(b)

Amount of data : 100

ED + SVM opeartion

Through the SVM used in face recoginition, the follow sample was closer to the target image.

Image number:46
ED:19822216

Training time : 0.343秒 Computing time : 0.187秒

(c)

Amount of data : 100

ED + SVM opeartion

Through the SVM used in face recoginition, the follow sample was closer to the target image.

Image number:6
ED:20796818

Training time : 0.344秒 Computing time : 0.187秒

(d)

Amount of data : 100

ED + SVM opeartion

Through the SVM used in face recoginition, the follow sample was closer to the target image.

Image number:30
ED:14434101

Training time : 0.344秒 Computing time : 0.187秒

FIGURE 12: The experimental results of the real-time face recognition system.

rate and reduced the computing time for the real-time face recognition system. For the method of combining PCA and Euclidean distance with bubble sort, when the number of samples is doubled from 50 to 100, the average computing time also increases almost linearly from 0.149 s to 0.296 s. It is noted that the time needed for the way of the PCA combined SVM-PSO only increases by 40% correspondingly. Based on such view, one can say that the proposed algorithm is clearly superior for large-sample-size cases. It also concludes that the proposed method is faster and more efficient than other common methods for face recognition.

5. Conclusions

A real-time face recognition system is designed by using a combination of PCA and hybrid biology algorithm face recognition system application and this method has effectively reduced the computing time. There is a time savings of 60% after doubling samples from 50 to 100 samples as compared to other methods. Furthermore, the SVM-PSO scheme is designed to speed up the recognition and also enhances the performance of the face recognition. In the future, the result of the face recognition system can be further developed in a chip of an embedded system.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

This work is supported by the National Science Council, Taiwan, under Grant nos. NSC 102-2221-E-218-017 and NSC 100-2632-E-218-001-MY3.

References

[1] M. M. Dehshibi and A. Bastanfard, "A new algorithm for age recognition from facial images," *Signal Processing*, vol. 90, no. 8, pp. 2431–2444, 2010.

[2] W.-J. Zeng, X.-L. Li, X.-D. Zhang, and E. Cheng, "Kernel-based nonlinear discriminant analysis using minimum squared errors criterion for multiclass and undersampled problems," *Signal Processing*, vol. 90, no. 8, pp. 2333–2343, 2010.

[3] J. Zhang and L. Ye, "Local aggregation function learning based on support vector machines," *Signal Processing*, vol. 89, no. 11, pp. 2291–2295, 2009.

[4] Y. E. Shao, C.-J. Lu, and Y.-C. Wang, “A hybrid ICA-SVM approach for determining the quality variables at fault in a multivariate process,” *Mathematical Problems in Engineering*, vol. 2012, Article ID 284910, 12 pages, 2012.

[5] J.-S. Chiou and K.-Y. Wang, “Application of a hybrid controller to a mobile robot,” *Simulation Modelling Practice and Theory*, vol. 16, no. 7, pp. 783–795, 2008.

[6] C.-C. Wong, H.-Y. Wang, and S.-A. Li, “PSO-based motion fuzzy controller design for mobile robots,” *International Journal of Fuzzy Systems*, vol. 10, no. 1, pp. 284–292, 2008.

[7] Y. Liu and B. Niu, “A novel PSO model based on simulating human social communication behavior,” *Discrete Dynamics in Nature and Society*, vol. 2012, Article ID 791373, 21 pages, 2012.

[8] W. Zou, Y. Zhu, H. Chen, and X. Sui, “A clustering approach using cooperative artificial bee colony algorithm,” *Discrete Dynamics in Nature and Society*, vol. 2010, Article ID 459796, 16 pages, 2010.

[9] P. Umapathy, C. Venkataseshaiah, and M. S. Arumugam, “Particle swarm optimization with various inertia weight variants for optimal power flow solution,” *Discrete Dynamics in Nature and Society*, vol. 2010, Article ID 462145, 15 pages, 2010.

[10] N. Kazakova, M. Margala, and N. G. Durdle, “Sobel edge detection processor for a real-time volume rendering system,” in *Proceedings of the IEEE International Symposium on Cirquits and Systems*, vol. 2, pp. II913–II916, May 2004.

[11] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, John Wiley & Sons, 2001.

[12] C.-Y. Lee, J.-J. Leou, and H.-H. Hsiao, “Saliency-directed color image segmentation using modified particle swarm optimization,” *Signal Processing*, vol. 92, no. 1, pp. 1–18, 2012.

[13] H.-Y. Chen and J.-J. Leou, “Saliency-directed image interpolation using particle swarm optimization,” *Signal Processing*, vol. 90, no. 5, pp. 1676–1692, 2010.

[14] R. Eberhart and J. Kennedy, “New optimizer using particle swarm theory,” in *Proceedings of the 6th International Symposium on Micro Machine and Human Science*, pp. 39–43, October 1995.

[15] J. Kennedy and R. Eberhart, “Particle swarm optimization,” in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, December 1995.

[16] D. Srinivasan, W. H. Loo, and R. L. Cheu, “Traffic incident detection using oarticle swarm optimization,” in *Proceedings of IEEE Swarm Intelligence Symposium*, pp. 144–151, 2003.