Expert Systems with Applications 38 (2011) 5406–5415

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

3D reconstruction and face recognition using kernel-based ICA and neural networks

Shye-Chorng Kuo [a,c], Cheng-Jian Lin [b,*], Jan-Ray Liao [a]

[a] Department of Electrical Engineering, National Chung Hsing University, Taichung City 402, Taiwan, ROC
[b] Department of Computer Science and Information Engineering, National Chin-Yi University of Technology, Taichung County 411, Taiwan, ROC
[c] Department of Computer Science and Information Engineering, Nan Kai University of Technology, Nantou County 542, Taiwan, ROC

ARTICLE INFO

Keywords:
Independent component analysis
3D human face reconstruction
3D human face recognition
Back-propagation algorithm
Neural networks

ABSTRACT

Kernel-based nonlinear characteristic extraction and classification algorithms are popular new research directions in machine learning. In this paper, we propose an improved photometric stereo scheme based on improved kernel-independent component analysis method to reconstruct 3D human faces. Next, we fetch the information of 3D faces for facial face recognition. For reconstruction, we obtain the correct normal vector's sequence to form the surface, and use a method for enforcing integrability to reconstruct 3D objects. We test our algorithm on a number of real images captured from the Yale Face Database B, and use three kinds of methods to fetch characteristic values. Those methods are called contour-based, circle-based, and feature-based methods. Then, a three-layer, feed-forward neural network trained by a back-propagation algorithm is used to realize a classifier. All the experimental results were compared to those of the existing human face reconstruction and recognition approaches tested on the same images. The experimental results demonstrate that the proposed improved kernel independent component analysis (IKICA) method is efficient in reconstruction and face recognition applications.

1. Introduction

When we use the camera to capture the images of 3D objects and scenes, we also lose the depth information of 3D objects and only obtain the 2D image information. However, the depth information of 3D objects plays an important role in many applications such as the 3D-object recognition and 3D-object display. In order to show the original information of 3D objects, the problems of 3D object reconstruction from 2D images is 3D surface reconstruction. In this paper, we propose the new techniques to solve the above-mentioned problems.

One of the approaches to computer vision is the photometric stereo approach for surface reconstruction. This approach is able to estimate local surface orientation by using several images of the same surface taken from the same viewpoint but under illuminations from different directions. The main limitation of the classical photometric stereo approach is that the light source positions must be accurately known. This necessitates a fixed, calibrated lighting rig. Hence, an improved photometric stereo method for estimating the surface normal and the surface reflectance of objects without a prior knowledge of the light source direction or the light source intensity was proposed by Hayakawa (1994). Hayakawa's method uses the singular-value decomposition (SVD) method to factorize an image data matrix of three different illuminations into a surface reflectance matrix and a light source matrix based on the Lambertian model. However, Hayakawa still uses one of the two added constraints for finding the linear transformation between the surface reflectance matrix and the light source matrix. McGunnigle (1998) introduced a simple photometric stereo scheme, which only considered a Lambertian reflectance model, where the self and cast shadows, as well as the inter-reflections, were ignored. Three images at a tilt angle of 90° increments are captured. McGunnigle suggested using his method as a first estimate for an iterative procedure.

Lin et al. proposed a novel ICA-based photometric stereo approach based on a non-Lambertian model (Lin, Cheng, & Liang, 2005). The goal of the ICA model is to separate the independent component of a surface normal on each point of an image. However, the ICA model still has the problem of the *x*-axis, *y*-axis and *z*-axis values of the separated normal vector not being arranged in turn. So a constrained independent components analysis (cICA) model (Hyvärinen, Karhunen, & Oja, 2001; Lu & Rajapakse, 2001) was devised. It is a supervised ICA model that makes the outputs of a normal vector's coordinate values arranged in turn. Lee et al. proposed a cICA-based photometric stereo reconstruction method to solve the normal vector disorder problem (Lee, Cheng, & Lin, 2006). However, we still found that the cICA model had another problem. Generally, the input data fed into the cICA model was linear data. However, the input data that we obtained

* Corresponding author.
E-mail address: cjlin@ncut.edu.tw (C.-J. Lin).

0957-4174/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2010.10.015

from 2D images was non-linear. So all the non-linear data must be changed into the linear data first before cICA process is used. However, this transformation would cause some distorted unavoidably. One kind of linear transformation method change that we used was with a kernel algorithm (Muller, Mike, Ratsch, Tsuda, & Scholkopf, 2001; Kocsor & Toth, 2004), which did not require that we knew the necessary change parameter during the course of changing, and enabled us to change smoothly and fast.

In this paper, we propose an improved kernel-independent component analysis (IKICA) method to take a synthetic, sphere-surface object's normal vector as supervised reference. The proposed IKICA extends the traditional kernel-independent component analysis (KICA) model (An & Ruan, 2006; Bach & Jordan, 2003; Cheng, Liu, & Lu, 2004; Martiriggiano, Leo, Spagnolo, & D'Orazio, 2005, Xu & Guo, 2006). It is a non-linear ICA model which can directly transform non-linear data from 2D images. So the introduction data does not need to undergo linear conversion in advance. Thus, it can effectively reduce the normal vector error of 3D objects. Then, the 3D surface model is reconstructed from the surface normal on each pixel of an image, obtained by the IKICA technique, using a method for enforcing integrability (Frankot & Chellappa, 1988). The reason for these proposed methods is the ease of implementation. After using the reconstructive method, we are able to populate many 3D human faces in our 3D database. Then, we are able to fetch the 3D information of 3D face model to make 3D human recognition.

The rest of this paper is organized as follows. The details of the proposed IKICA-based reflectance model and its derivations are presented in Section 2. We present the 3D model reconstruction in Section 3. After face reconstruction, we discuss 3D face recognition in Section 4. Experimental results are given in Section 5. Finally, the last section summarizes the conclusions.

2. The improved KICA model

2.1. The KICA model

ICA is a technique that transforms a multivariate random signal into a signal having components that are mutually independent in the complete statistical sense (Hyvärinen et al., 2001). Let the time-varying observed signal be $\mathbf{x} = (x_1, x_2, \ldots, x_m)^T$, and the desired signal consisting of independent components (ICs) be $\mathbf{s} = (s_1, s_2, \ldots, s_n)^T$. The classical ICA assumes that the signal x is an instantaneous linear mixture of ICs, or independent sources s_i, $i = 1, 2, \ldots, m$. Therefore, $x = As$, where the matrix A of size $n \times m$ represents the linear memoryless mixing channels. The goal of the ICA is to obtain a $m \times n$ demixing matrix W to recover all the ICs of the observed signal. $y = (y_1, y_2, \ldots, y_m)^T$ is given by $y = Wx$. For simplicity, in this section, we address the case of a complete ICA, in which $n = m$.

The main idea of KICA is to map the input data into an implicit feature space F firstly: $\Phi : x \in R^N \to \Phi(x) \in F$. Then KICA is performed in F to produce a set of non-linear features of input data. As ICA algorithm described in the above part, the input data X is whitened in feature space F. The whitening matrix is: $\tilde{W}_\Phi = (\Lambda_\Phi)^{\frac{1}{2}}(V_\Phi)^T$, here Λ_Φ, V_Φ are the eigenvalues matrix and eigenvectors matrix of covariance matrix $\hat{C} = \frac{1}{n}\sum_{i=1}^{n} \Phi(X) = (\Lambda_\Phi)^{-1}\alpha^T K$, respectively. Then we can obtain the whitened data X_Φ^W as

$$X_\Phi^W = (\tilde{W}_\Phi)^T \Phi(X) = (\Lambda_\Phi)^{-1}\alpha^T K, \tag{1}$$

where K is defined by: $K_{ij} := (\Phi(x_i) \cdot \Phi(x_j))$ and α is the eigenvectors matrix of K. After the whitening transformation, the learning algorithm calculated by the following iterative algorithm:

$$\tilde{Y}_\Phi = W_\Phi X_\Phi, \tag{2}$$

$$\Delta W_\Phi = \left[J + \left(J - \frac{2}{1 + e^{-Y_\Phi}}\right)\right](\tilde{Y}_\Phi)^T W_\Phi, \tag{3}$$

$$\tilde{W}_\Phi = W_\Phi + \rho \Delta W_\Phi \to W_\Phi, \tag{4}$$

until W_Φ converged, and ρ is a learning constant. According to the above algorithm, the feature of a test data s can be obtained by:

$$y = W_\Phi (\Lambda_\Phi)^{-1}\alpha^T K(X, s), \tag{5}$$

where $K(X,s) = [k(x_1,s), k(x_2,s), \ldots, k(x_n,s)]^T$, k is a kernel function.

In the above iteration algorithm, the function Φ is an implicit form. The kernel function k can be computed to instead of Φ. This trick is named as kernel trick. Many functions can be chosen for the kernel such as polynomial kernel:

$$k(x, s) = (x \cdot s)^d. \tag{6}$$

Gaussian kernel $k(x,s) = \exp\left(\frac{-\|x-s\|^2}{2\sigma^2}\right)$ and sigmoid kernel $k(x,s) = \tanh(k(x,s) + \theta)$. Liu et al. use a cosine kernel function (Cheng et al., 2004) derived from the polynomial kernel function as shown in Eq. (6), which can give a better performance than the polynomial kernel function for feature extraction:

$$\hat{k}(x, s) = \frac{k(x, s)}{\sqrt{k(x, x)k(s, s)}}, \tag{7}$$

where k is a polynomial kernel. Practically speaking, Kernel-ICA = Kernel-Centering + Kernel-Whitening + ICA. Selecting an appropriate kernel function for a particular application area can be difficult and remains largely an unresolved issue. Any new kernel function derived from the kernel $k(x,s)$ with form $\hat{k}(x,s) = c(x)c(s)k(x,s)$, has been proved to be a valid kernel function when $c(x)$ is a positive real valued function of x, which is always satisfied. So, the cosine kernel is a valid kernel function. We adopt cosine kernel in our experiments.

2.2. The IKICA model

Our previous research used the KICA model to solve the problem of finding the surface normal on each point of an image. But in the KICA model, it is easy to see that the following ambiguities exist in Fig. 1: (1) we cannot determine the variances (energies) of the independent components; and (2) we cannot determine the order of the independent components. We generally discover that finding the surface normal vector involves the two problems. For those reasons, we use a constrained learning adaptation algorithm (IKICA) based on image intensities to handle these ambiguities.

The IKICA algorithm described in Lu and Rajapakse (2001) brings in the use of a constraint which is used to obtain an output that is statistically independent of other sources and is closest to a reference signal $r(t)$. This constraining signal need not be a perfect match but it should be enough to point the algorithm in the direction of a particular IC spanning the measurement space. The closeness constraint can be written as

$$g(w) = \varepsilon(w) - \xi \leqslant 0, \tag{8}$$

where w denotes a single demixing weight vector such that $y = w^T v$; $\varepsilon(w)$ represents the closeness between the estimated output y and the reference r, and ξ represents some closeness threshold. The measure of closeness can take any form, such as mean squared-error (MSE) or correlation, or any other suitable closeness measure. In our implementation of the algorithm, we use correlation as a measure of closeness such that $g(w)$ becomes

$$g(w) = \xi - E\{r(w^T v)\} \leqslant 0, \tag{9}$$

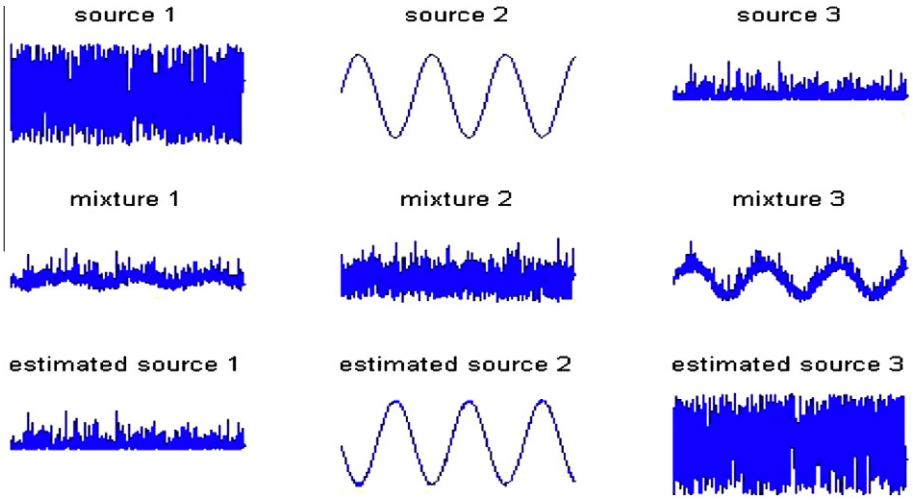

Fig. 1. "Ambiguities of KICA model." The source signals are shown in first row, mixing underlying sources shown in second row, and estimated sources shown in third row.

where ξ now becomes the threshold that defines the lower bound of the optimum correlation.

With the constraint in place, the IKICA problem is modeled as follows:

$$\begin{aligned}&\text{Maximize}: \ f(w)=\rho[E\{G(w^T v)\}-E\{G(v)\}]^2,\\ &\text{Subject to}: \ g(w)\leqslant 0, h(w)=E\{y^2\}-1=0 \quad \text{and} \quad E\{r^2\}-1=0,\end{aligned} \tag{10}$$

where $f(w)$ denotes the one-unit IKICA contrast function; $g(w)$ is the closeness constraint; $h(w)$ constrains the output y to having a unit variance; and the reference signal r is also constrained to having a unit variance. In Lu and Rajapakse (2001), the problem of Eq. (10) is expressed as a constrained optimization problem which is solved through the use of an augmented Lagrangian function, where learning of the weights and the Lagrange parameters is achieved through a Newton-like learning process.

For example, the IKICA algorithm was tested using a synthetic data set of four known sources were seen in Fig. 2(a), which had been used for IKICA work. The sources were linearly mixed by a randomly generated mixing matrix, producing the dataset shown in Fig. 2(b). With this mixture of data, the IKICA algorithm was run 100,000 times, each time with one of the five reference signals shown in Fig. 2(c) as a reference. The first four of these references were obtained from the sign of the four original sources, and these were purposely kept as coarse representations of the true sources. The fifth reference is a sine wave which has a frequency radically different than any of the original sources, allowing study of the algorithm's behavior given a "false" reference. Typical outputs of the algorithm are depicted in Fig. 2(d). Thus, if we want to find the surface normal vector on each point of an image, we can use the IKICA model to find it.

3. 3D model reconstruction

3.1. Determining the surface normal of objects using the IKICA model

Suppose that the recovering of surface shape, denoted by $z(x,y)$, from shaded images depends upon the systematic variation of image brightness with surface orientation, where z is the depth field, and x and y form the 2D grid over the domain D of the image plane. Then, the Lambertian reflectance model used to represent a surface illuminated by a single point light source is written as:

$$R_d(\mathbf{n}(x,y),\alpha(x,y))=\max\{L\alpha(x,y)\mathbf{s}^T\mathbf{n}(x,y),0\}, \quad \forall x,y\in D, \tag{11}$$

where $\boldsymbol{R}(.)$ is reflectance component intensity, $\alpha(x,y)$ is reflectance albedo on position (x,y) of surface, s is a column vector indicating the direction of point light, and L is light strength. The surface normal on position (x,y), denoted by $\mathbf{n}(x,y)$, can be represented as

$$\mathbf{n}(x,y)=\frac{[-p(x,y)-q(x,y)1]^T}{\sqrt{p^2(x,y)+q^2(x,y)+1}}, \tag{12}$$

where $p(x,y)$ and $q(x,y)$ are the x- and y- partial derivatives of $z(x,y)$, respectively. In Eq. (11), $\max\{\cdot\}$ sets all negative components that correspond to the surface points lying in attached shadow to zero, where a surface point (x,y) lies in an attached shadow iff $\mathbf{n}(x,y)\mathbf{s}<0$ (Woodham, 1980).

In this section, we describe the method of applying the IKICA model to estimate the normal vector $\mathbf{n}(x,y)$ on the object surface corresponding to each pixel in an image. Since the $\mathbf{n}(x,y)$ vector is a 3×1 column vector, we need at least three images under illumination from lights coming from different directions for the normal vector $\mathbf{n}(x,y)$ estimation. Hence, to reconstruct the 3D surface of an object using its images, we have to take three gray-value images under three different illuminants. Assuming an image contains T pixels in total, we can rearrange all the gray values of the three images into a $3\times T$ matrix, with each row representing an image, and each column representing the gray values of a single pixel under three different illuminants. When this matrix is put into Eq. (11) is compared with $\mathbf{x}=\mathbf{As}$, we find that $\boldsymbol{s}$ is the $\mathbf{n}(x,y)$ vector that we are looking for.

Using the IKICA decomposition, we rewrite equation Eq. (11) in matrix form as

$$\mathbf{x}(i)=\mathbf{As}(i)=\alpha(i)\hat{\mathbf{A}}\hat{\mathbf{n}}(i), \tag{13}$$

where $\hat{\mathbf{A}}=[\mathbf{a}_1,\mathbf{a}_2,\mathbf{a}_3]^T=\hat{W}^{-1}$ is the matrix depending on the lighting and viewing directions and has unit length; $\hat{\mathbf{n}}(t)$ is the estimated normal vector corresponding to the tth pixel, $i=1,2,\ldots,T$; and $\alpha(i)$ is the albedo of the ith pixel. However, the decomposition in Eq. (13) is not unique. If there is an invertible matrix G, which satisfies

$$\mathbf{A}=\hat{\mathbf{A}}\mathbf{G} \quad \text{and} \quad \mathbf{n}(i)=\mathbf{G}^{-1}\hat{\mathbf{n}}(i), \tag{14}$$

where A is the true matrix depending on the lighting and viewing directions of the images, and $\mathbf{n}(i)$ is the normal vector of the ith pixel in the standard XYZ coordinates, then the linear ambiguity belongs to the subset of GBR (Belhumeur, Kriegman, & Yuille, 1997; Georghiades, Belhumeur, & Kriegman, 2001; Georghiades, 2003). On the one hand, according to Georghiades (2003) studies, if the surface of an object is seen under variable light directions, but with

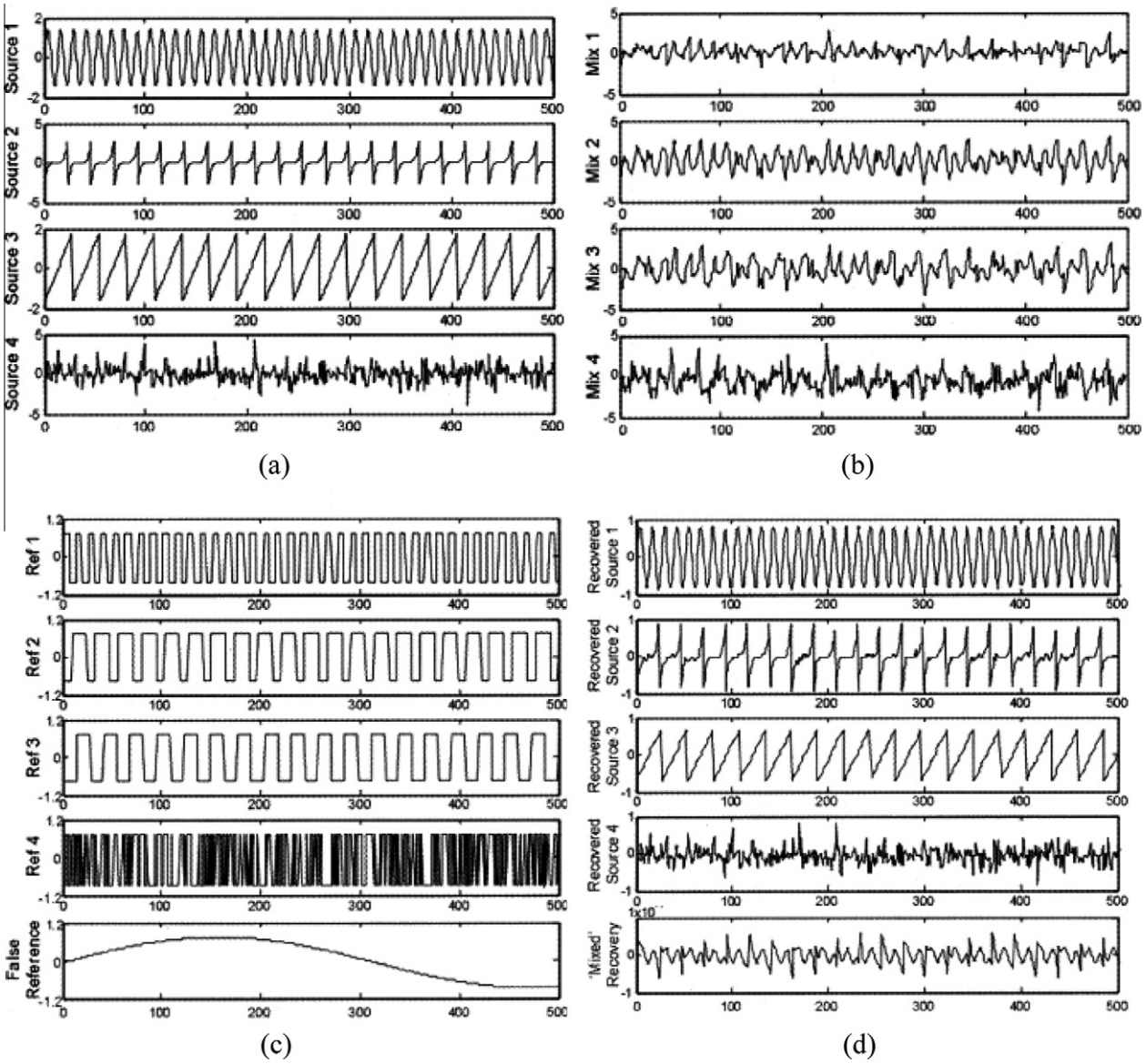

Fig. 2. (a) The four underlying sources of the synthetic dataset. (b) The linearly mixing underlying sources shown in (a). (c) The different references used for executions of IKICA on 4 channels of data in (b). The first four references are derived from the signs of the four underlying source, the fifth reference is a "false" reference. (d) Examples of each recovered source using only the references given in (c), the fifth recovered source shows a "mixture" of two underlying sources.

a fixed viewpoint, then the linear ambiguity can be reduced to three GBR parameters. As far as the surface normal vectors are concerned, we can only recover $\mathbf{n} \cong \mathbf{G}^{-1}\hat{\mathbf{n}}$, and

$$\mathbf{G}^{-1} = \frac{1}{g_3}\begin{bmatrix} g_3 & 0 & 0 \\ 0 & g_3 & 0 \\ -g_1 & -g_2 & 1 \end{bmatrix}, \tag{15}$$

where g_i are the three GBR parameters. On the other hand, the three light sources corresponding to the three images do not lie in the same plane (non-coplanar); therefore, the columns of matrix A are linearly independent. In addition, using the IKICA decomposition in Eq. (13), we can obtain an independent basis matrix $\hat{\mathbf{A}}$; thus the ambiguity can further be denoted by a diagonal matrix, i.e., $g_1 = 0$ and $g_2 = 0$. The relation, then, between the normal vectors in the standard *XYZ* coordinates and those in the independent coordinates system differs only by the g_3 factor. For the performance evaluation of 3D image reconstruction, both estimated surfaces and synthetic surfaces are normalized within the interval [0,1]. Therefore, the influence of the g_3 factor on the estimated 3D surface can be removed.

3.2. 3D surface reconstruction using the method for enforcing integrability

In this section, we discuss using the method for enforcing integrability to obtain detailed information for reconstructing the surface of an object using its normal vectors. This approach was proposed by Frankot and Chellappa (1988).

Suppose that we represent the surface $z(x,y)$ by the functions $\phi(x,y,\boldsymbol{\omega})$ so that

$$z(x,y) = \sum_{\boldsymbol{\omega}\in\Omega} c(\boldsymbol{\omega})\phi(x,y,\boldsymbol{\omega}), \tag{16}$$

where $\boldsymbol{\omega} = (u,v)$ is a two-dimensional index, Ω is a finite set of indexes, and the members of $\{\phi(x,y,\boldsymbol{\omega})\}$ are not necessarily mutually orthogonal. We choose the discrete cosine basis so that $\{c(\boldsymbol{\omega})\}$ is exactly the full set of discrete cosine transform (DCT) coefficients

of $z(x,y)$. Since the partial derivatives of the basis functions $\phi_x(x,y,\omega)$ and $\phi_x(x,y,\omega)$ are integrable, the partial derivatives of $z(x,y)$ are guaranteed to be integrable as well; that is, $z_{xy}(x,y) = z_{yx}(x,y)$. Note that the partial derivatives of $z(x,y)$ can also be expressed in terms of this expansion, giving

$$z_x(x,y) = \sum_{\omega\in\Omega} c(\omega)\phi_x(x,y,\omega). \tag{17}$$

$$z_y(x,y) = \sum_{\omega\in\Omega} c(\omega)\phi_y(x,y,\omega), \tag{18}$$

where $\phi_x(x,y,\omega) = \partial\phi(\cdot)/\partial x$ and $\phi_y(x,y,\omega) = \partial\phi(\cdot)/\partial y$.

Suppose we now have the possibly non-integrable estimate $\mathbf{n}(x,y)$ from which we can easily deduce from Eq. (12) the possibly non-integrable partial derivatives $\hat{z}_x(x,y)$ and $\hat{z}_y(x,y)$. These partial derivatives can also be expressed as a series, giving

$$\hat{z}_x(x,y) = \sum_{\omega\in\Omega} \hat{c}_1(\omega)\phi_x(x,y,\omega), \tag{19}$$

$$\hat{z}_y(x,y) = \sum_{\omega\in\Omega} \hat{c}_2(\omega)\phi_y(x,y,\omega). \tag{20}$$

This method can find the expansion coefficients $c(\omega)$ given a possibly non-integrable estimate of surface slopes $\hat{z}_x(x,y)$ and $\hat{z}_y(x,y)$:

$$c(\omega) = \frac{p_x(\omega)\hat{c}_1(\omega) + p_y(\omega)\hat{c}_2(\omega)}{p_x(\omega) + p_y(\omega)}, \quad \text{for } \omega = (u,v) \in \Omega, \tag{21}$$

where

$$p_x(\omega) = \int\int |\phi_x(x,y,\omega)|^2 dxdy, \tag{22}$$

$$p_y(\omega) = \int\int |\phi_y(x,y,\omega)|^2 dxdy. \tag{23}$$

In the end, we can reconstruct an object's surface by implementing the inverse 2D DCT on the coefficient $c(\omega)$.

4. 3D human face recognition

We success in reconstructing 3D human faces which are used to be 3D face database for human face recognition. There are the three methods which are proposed by us for characteristic values fetching. The neural network is used as classifier to discriminate the 3D face characteristic value of 3D human face database. In order to adjust the parameter of the neural network efficiently, we used back-propagation as a learning algorithm. The detail of the neural network and back-propagation is described in follows.

4.1. Method of characteristic fetch

In our system, in order to extract the characteristic value of 3D human faces, three type of fetching methods are defined. Then, we classified the characteristics by back-propagation learning network to complete 3D human face recognition. We would state the three kinds of characteristic value fetching methods separately in following section.

4.1.1. Contour-based fetching method

We can get the coordinate value of the human image every pixel in 3D picture, we fetch z coordinate from it that is the depth of faces of people. In the depth of 3D frontal face, we can be found a supreme point in person's 3D face model which is the so-called nose tip. We utilize the deep relation of 3D faces of people, accord with its deep size, and draw up contour map such as it. We suppose all numbers of value in every contour is n, every deep value is X, and each contour have j pieces of X. The method that we get ith characteristic value is

$$contour_i = \sum_{j=1}^{n} X_j. \tag{24}$$

4.1.2. Circle-based fetching method

First, we fetch z coordinate from 3D human face that is the depth of faces of people. In the depth of 3D frontal face, we can be found a supreme point in person's 3D face model which is the nose tip. Regard nose tip as the center of a circle, make the radius of the proportion of 3D human face, and draw a round of k in order. There is r a number of deep values in a round of k separately. The q order point depth is Y in each circle of deep value. We adds these deep values which written as

$$Sum_k = \sum_{q=1}^{r} Y_q. \tag{25}$$

Then, we fetch necessary k a characteristic value on average r pieces of Sum value

$$Circle_k = \frac{Sum_k}{r}. \tag{26}$$

4.1.3. Feature-based fetching method

Many researchers have investigated facial feature extraction from a frontal view. A number of manual feature extraction algorithms have been proposed, from frontal views, mostly for facial animation (Koch, 1991; Liu, Zhang, Jacobs, & Cohen, 2000; Nagel, Wingbermühle, Weik, & Liedtke, 1998; Zhang, 2001). Zhang (2001) manually picks five feature points from two images to deform a generic mesh model. Liu et al. (2000) relie on manual feature extraction for synthesizing realistic facial expression from video. Nagel et al. (1998) use stereo frontal images of the face to compute the depth of manually picked feature points. Koch (1991) extracts the centers of the eyes and mouth based on head motion in video frames and knowledge about the facial geometry. Zhang (1996) elaborates on the work of Koch (1991) and extracts the corners of the eyes and mouth using template matching for each corner. Gordon (1995) utilizes two views of the face and only extracts automatically the pupils' centers from the frontal view image based on eye template and pupil detector. Yin, Yin, and Yourst (2003) use similar algorithm as Gordon (1995) but estimates the 3D points of each feature from two 2D points from each view. The tip of the nose, chin, and upper and lower lip feature points are determined by tracking local maximum curvature at the profile view.

In this section, we automatically extract 15 corresponding facial feature points from the frontal view. These features are landmark points chosen based on their importance in representing a face. Fig. 3(c) shows the facial features considered in this section for the frontal view. We measure the nose tip point to the distance of eyes (six points), the nose (three points), the mouth (six points) regards as characteristic value for 3D human face recognition system.

4.2. The structure of multi-layer neural networks

Multi-neural network is the science of investigating and analyzing the algorithms of the human brain, and using the similar algorithm to build up a powerful computational system to do the tasks like pattern recognition, identification, controlling of dynamical system, system modeling, and nonlinear prediction of time series. The multi-neural network owns the capability, to organize its structural constituents, the same as the human brain. So the most attractive character of multi-neural network is that it can be taught to achieve the complex tasks.

We use a simple three-layer multi-layer neural network as shown in Fig. 4 for 3D face recognition. In Fig. 4, we have m PEs

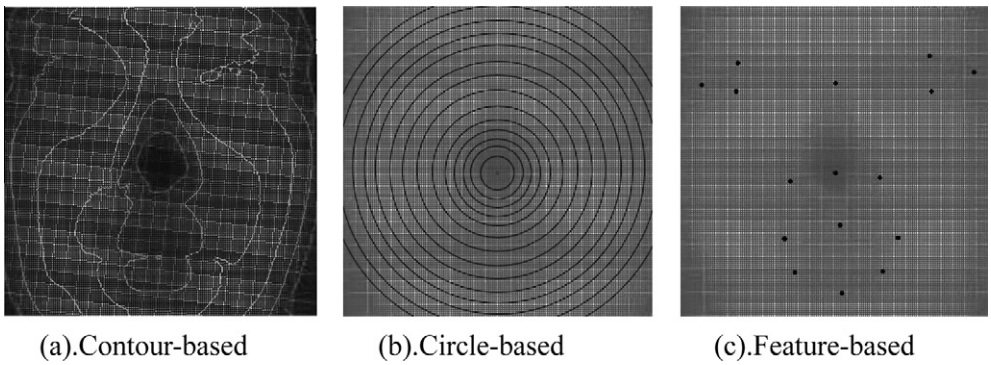

Fig. 3. The three characteristic value fetching methods of 3D human face.

in the input layer, I PEs in the hidden layer, and n PEs in the output layer; the solid lines show the forward propagation of signals, and the dashed lines show the backward propagation of errors.

Let us consider an input–output training pair (x,d), where the superscript k is omitted for notation simplification. Given an input pattern x, a PE q in the hidden layer receives a net input of

$$net_q = \sum_{j=1}^{m} v_{qj} x_j, \tag{27}$$

and produces an output of

$$z_q = a(net_q) = a\left(\sum_{j=1}^{m} v_{qj} x_j\right). \tag{28}$$

The net input for a PE i in the output layer is then

$$net_i = \sum_{q=1}^{l} w_{iq} z_q = \sum_{q=1}^{l} w_{iq} a\left(\sum_{j=1}^{m} v_{qj} x_j\right), \tag{29}$$

and it produces an output of

$$y_i = a(net_i) = a\left(\sum_{q=1}^{l} w_{iq} z_q\right) = a\left(\sum_{q=1}^{l} w_{iq} a\left(\sum_{j=1}^{m} v_{qj} x_j\right)\right). \tag{30}$$

Transfer function is to do an input weighting value of input of the neuron the summation and is transferred to a kind of mapping rule that is outputted in the function of neural network, influence a kind of design that is channeled into the network of the non-linear one too. The common non-linear transfer functions have:

Two-value function.
Sigmoid function.
Hyperbolic tangent function.

The back propagation neural network most frequently used non-linear transfer function is sigmoid function (Fig. 5). So we use sigmoid function Eq. (31) as transfer function

$$f(x) = \frac{1}{1+e^{-x}}. \tag{31}$$

4.3. Back-propagation learning method

The back-propagation learning algorithm (Lacher, Hruska, & Kuncicky, 1992) is one of the most important historical developments in neural networks. It has reawakened the scientific and engineering community to the modeling and processing of many quantitative phenomena using neural networks. This learning algorithm is applied to multilayer feed forward networks consisting of processing elements with continuous differentiable activation functions. Such networks associated with the back-propagation learning algorithm are also called back-propagation networks. Given a training set of input–output pairs $\{(x(k),d(k))\}$, $k = 1,2,\ldots,p$, the algorithm provides a procedure for changing the weights in a back-propagation network to classify the given input patterns correctly. The basis for this weight update algorithm is simply the gradient-descent method as used for simple perceptrons with differentiable units. Back-propagation algorithm has been widely adopted as a successful learning rule to find the appropriate values of the weights for NNs. Generally speaking, because we use the back-propagation learning algorithm, so the error

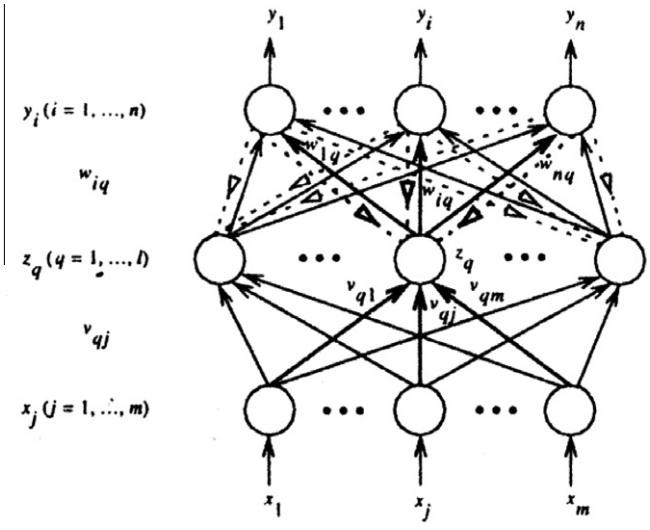

Fig. 4. Three-layer multi-layer neural network.

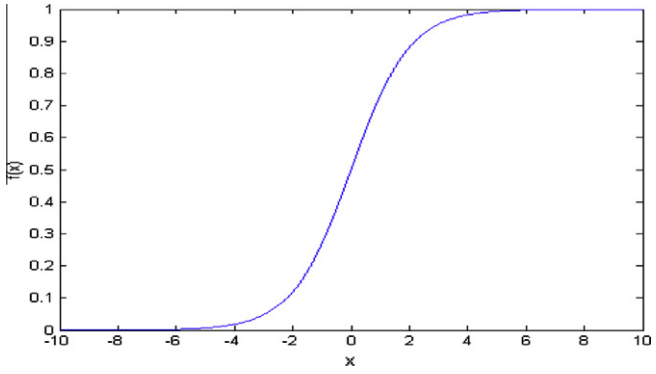

Fig. 5. Sigmoid function.

quantity must propagate through time from a stable state to initial state.

For a given input–output pair $(x(k),d(k))$, the back-propagation algorithm performs two phases of data flow. First, the input pattern $x(k)$ is propagated from the input layer to the output layer and, as a result of this forward flow of data, it produces an actual output $y(k)$. Then the error signals resulting from the difference between $d(k)$ and $y(k)$ are back-propagated from the output layer to the previous layers for them to update their weights. The above section's equations indicate the forward propagation of input signals through the layers of neurons. Next, we consider the error signals and their back propagation. We first define a cost function:

$$E(w) = \frac{1}{2}\sum_{i=1}^{n}(d_i - y_i)^2 = \frac{1}{2}\sum_{i=1}^{n}[d_i - a(net_i)]^2 = \frac{1}{2}\sum_{i=1}^{n}\left[d_i - a\left(\sum_{q=1}^{l} w_{iq}z_q\right)\right]^2. \quad (32)$$

Then according to the gradient-descent method, the weights in the hidden-to-output connections are updated by

$$\Delta w_{iq} = -\eta\frac{\partial E}{\partial w_{iq}}. \quad (33)$$

Using Eqs. (29), (30), (32), and the chain rule for $\partial E/\partial w_{iq}$, we have

$$\Delta w_{iq} = -\eta\left[\frac{\partial E}{\partial y_i}\right]\left[\frac{\partial y_i}{\partial net_i}\right]\left[\frac{\partial net_i}{\partial w_{iq}}\right] = \eta[d_i - y_i][a'(net_i)][z_q] \triangleq \eta\delta_{oi}z_q, \quad (34)$$

where δ_{oi} is the error signal and its double subscript indicates the ith node in the output layer. The error signal is defined by

$$\delta_{oi} \triangleq -\frac{\partial E}{\partial net_i} = -\eta\left[\frac{\partial E}{\partial y_i}\right]\left[\frac{\partial y_i}{\partial net_i}\right] = [d_i - y_i][a'(net_i)], \quad (35)$$

where net_i is the net input to PE i of the output layer and $a'(net_i) = \partial a(net_i)/\partial a(net_i)$. The result thus far is identical to the delta learning rule obtained from a single-layer perceptron whose input is now the output z_q of the hidden layer.

For the weight update on the input-to-hidden connections, we use the chain rule with the gradient-descent method and obtain the weight update on the link weight connecting PE j in the input layer to PE q in the hidden layer,

$$\Delta v_{qj} = -\eta\left[\frac{\partial E}{\partial v_{qj}}\right] = -\eta\left[\frac{\partial E}{\partial net_q}\right]\left[\frac{\partial net_q}{\partial v_{qj}}\right] = -\eta\left[\frac{\partial E}{\partial z_q}\right]\left[\frac{\partial z_q}{\partial net_q}\right]\left[\frac{\partial net_q}{\partial v_{qj}}\right]. \quad (36)$$

From Eq. (32), it is clear that each error term $[d_i - y_i]$, $i = 1,2,\ldots,n$, is a function of zq.

Evaluating the chain rule, we have

$$\Delta v_{qj} = \eta\sum_{i=1}^{n}[(d_i - y_i)a'(net_i)w_{iq}]a'(net_q)x_j. \quad (37)$$

Using Eq. (35), we can rewrite Eq. (37) as

$$\Delta v_{qj} = \eta\sum_{i=1}^{n}[\delta_{oi}w_{iq}]a'(net_q)x_j = \eta\delta_{hq}x_j, \quad (38)$$

where δ_{hq} is the error signal of PE q in the hidden layer and is defined as

$$\delta_{hq} \triangleq -\frac{\partial E}{\partial net_q} = -\eta\left[\frac{\partial E}{\partial z_q}\right]\left[\frac{\partial z_q}{\partial net_q}\right] = a'(net_q)\sum_{i=1}^{n}\delta_{oi}w_{iq}, \quad (39)$$

where $netq$ is the net input to the hidden PE q Eq. (27). The error signal of a PE in a hidden layer is different from the error signal of a PE in the output layer, as seen in Eqs. (35) and (39). Because of this difference, the above weight update procedure is called the generalized delta learning rule. We observe from Eq. (39) that the error signal δ_{hq} of a hidden PE q can be determined in terms of the error signals δ_{oi} of the PEs, y_i, that it feeds. The coefficients are just the weights used for the forward propagation, but here they are propagating error signals (δ_{oi}) backward instead of propagating signals forward. This is shown by the dashed lines in Fig. 4. This also demonstrates one important feature of the back-propagation algorithm-the update rule is local; that is, to compute the weight change for a given connection, we need only quantities available at both ends of that connection.

Note that both Eqs. (34) and (38) are in the same form of the general weight learning rule in common. The above derivation can be easily extended to the network with more than one hidden layer by using the chain rule continuously. In general, with an arbitrary number of layers, the back-propagation update rule is in the form

$$\Delta w_{ij} = \eta\delta_i x_j = \eta\delta_{\text{output-}i}\cdot x_{\text{input-}j}, \quad (40)$$

where "output-i" and "input-j" refer to the two ends of the connection from PE j to PE i, x_j is the proper input-end activation from a hidden PE or an external input, and δ_i is the learning signal which is defined by Eq. (35) for the last (or output) layer of connection weights and defined by Eq. (39) for all the other layers. When the bipolar sigmoid function is used as the activation function, then using Eqs. (35) and (39), respectively, become

$$\delta_{oi} = \frac{1}{2}(1 - y_i^2)[d_i - y_i], \quad (41)$$

$$\delta_{hq} = \frac{1}{2}(1 - z_q^2)\sum_{i=1}^{n}\delta_{oi}w_{iq}. \quad (42)$$

In order to make the neural network works properly, we need to find the weights and biases by the learning algorithms. In this study, the supervise learning is used. The network is given a set of training data which contains a number of input pattern and corresponding target output. The learning objective is to reach the high accuracy of classification by minimize the error between target output and network output in the training set. Furthermore, the trained classifier also should to provide a good performance in the untrained data (testing set).

5. Experimental results

We implemented each method in Matlab 7.0 software on a 1.8 GHz K8-based PC with 1024 MB RAM According to the results. For 3D reconstruction, we tested the algorithm on a number of real images from The Yale Face Database B showing variability due to illumination. There are varying albedos in each point of the surface of the human faces. First, we arbitrarily took from these test images the images of the same person who was photographed under three different light sources, as shown in Fig. 8. We fed the normalized images into our algorithm. For the face surface reconstruction problem, the normal vectors of a sphere's surface were used as the reference values for the IKICA model due to their similar structures. The true depth map of the synthetic sphere object is generated mathematically as

$$z(x,y) = \begin{cases} \left|\sqrt{r^2 - x^2 - y^2}\right|, & \text{if } x^2 + y^2 \leqslant r^2, \\ 0, & \text{otherwise}, \end{cases} \quad (43)$$

where $r = 48$, $0 < x$, $y \leqslant 100$, and the center is located at $(x,y) = (51,51)$. The sphere object is shown in Fig. 6. Fig. 7 shows the normal vectors of a sphere's surface.

After updating the parameters by several iterations, we obtained the normal vector of the surfaces of the human faces corre-

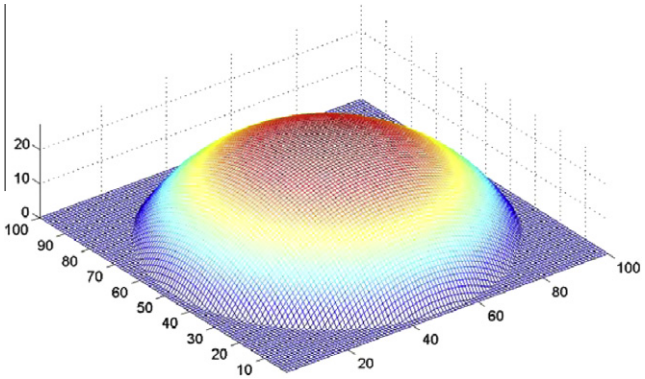

Fig. 6. Synthetic sphere surface object.

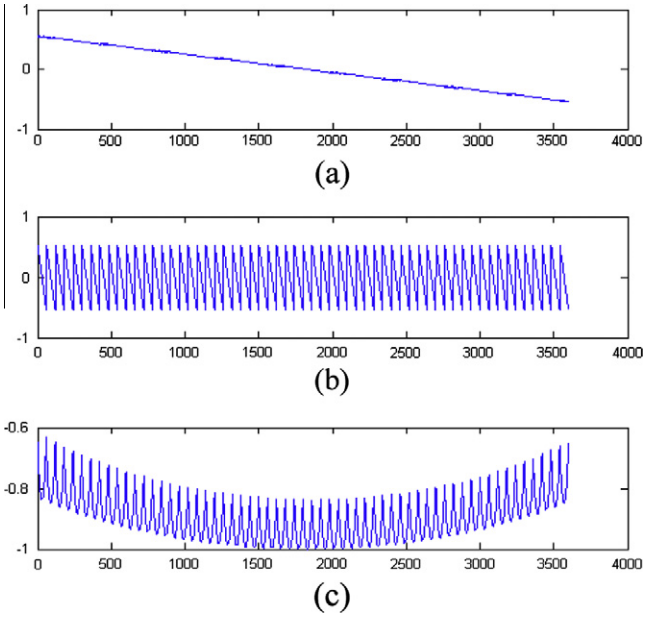

Fig. 7. The normal vectors of a sphere's surface (a) the *X*-component, (b) the *Y*-component, and (c) the *Z*-component of the normal vectors.

sponding to each pixel in the image in the output nodes. The results are shown in the second row in Fig. 8, which give the *X*-component, the *Y*-component, and the *Z*-component of the surface normal vector in order. Fig. 9(a) shows the surface albedo of the human face shown in Fig. 8. Fig. 9(b) shows the result from using our proposed reconstructive algorithm.

Third, the data set in The Yale Face Database B is also used in our experiments for objective comparison. The database consists of 3D face coordinate data and their corresponding 2D front view. Fig. 10 shows 6 individuals in the database with $160 * 160$ image size. The results of 3D model reconstruction by our proposed algorithm are showed in second row.

For face recognition, we propose three effective algorithms for 3D human face recognition by back-propagation neural networks. We use The Yale Face Database B for reconstruction that provide the geometric properties of 3D face database. The Yale database B contains 10 people faces and each person has 50 2D images, per image is taken under different directions of source light. Therefore 1 person make 10 3D images, 100 3D images regard our 3D database as altogether. We take 50 3D images regard as training data, 50 3D images regard as test data in our 3D database for back-propagation learning network.

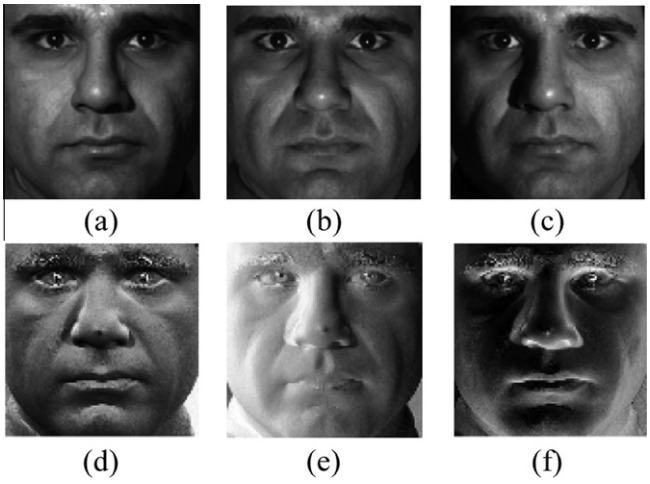

Fig. 8. (a)–(c) Three training images with different light source positions from Yale Face Database B (The Yale Face Database B, xxxx) in frontal. (d)–(f) Surface normal corresponding to the three source images.

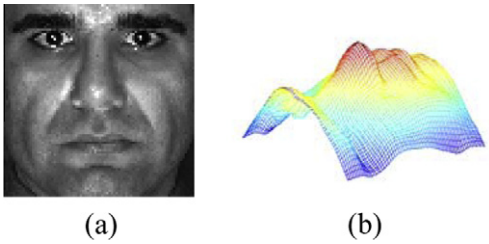

Fig. 9. (a) The surface albedo of human face in Fig. 8. The results of 3D model reconstruction by (b) our proposed algorithm.

The number of the input nodes is defined by the number of characteristic values. Since the 3D face recognition is classified to 10 persons, we set the 10 output nodes $y_1 - y_{10}$ corresponding to each person, respectively. If an input pattern is given, we expect that the value of the output node which is corresponding to the mental category of input pattern is near to 1, otherwise is near to 0. When an unknown data inputs to the network, we can determine which mental task the data belongs to by find the index of maximum value in the 10 output nodes. In order to make the output value between 0 and 1, the sigmoid function Eq. (31) is used as the activate function.

Pick the method of fetching in three characteristics that we propose, extract 15 pieces of characteristic value as inputs separately for training. Back-propagation trains each 3D faces independently, it is adequate to recognition applications in which a model base is frequently updated. For the evaluation of a neural network, the root-mean-square-error (RMSE) is used to compute the average output error

$$\mathrm{RMSE} = \sqrt{\frac{1}{nTr}\sum_{i=1}^{nTr}\sum_{j=1}^{nOut}(t_{i,j} - y_{i,j})^2}, \tag{44}$$

where the *nTr* is the number of training data, *nOut* is the number of network output, and $t_{i,j}$ and $y_{i,j}$ is the *j*th target output and real network output of *i*th training data, respectively. The back-propagation algorithm's learning curves by three kind of characteristic value fetching methods are show by Fig. 11.

The performance of the back-propagation network is affected by many factors. If the performance is not good then the recogni-

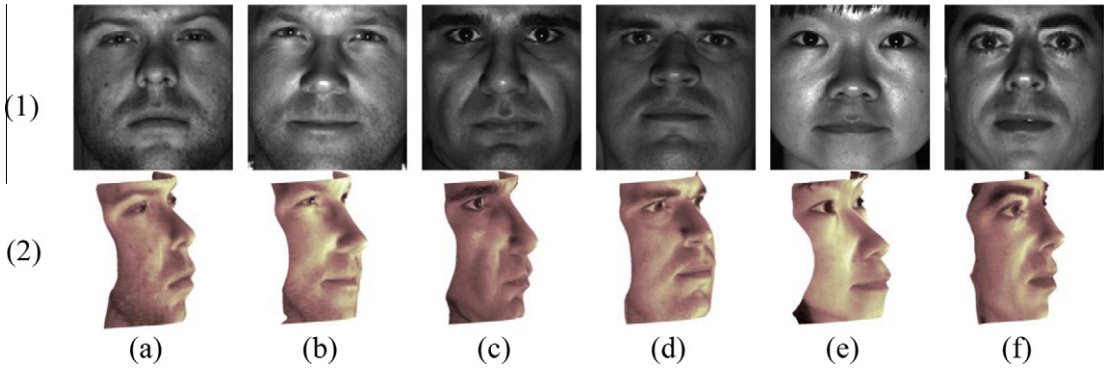

Fig. 10. (1) Six individuals in The Yale Face Database B used to test our algorithm (these images include both males and females.). The results of 3D model reconstruction by (2) our proposed algorithm (second row).

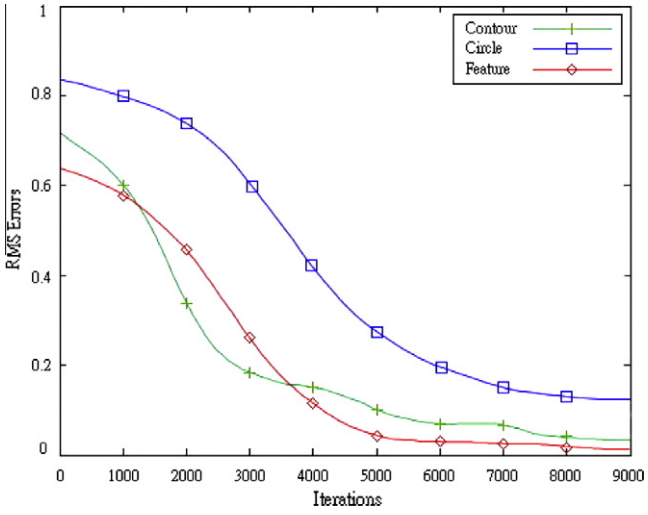

Fig. 11. Back-propagation algorithm's learning curves by three kind of characteristic value fetching methods.

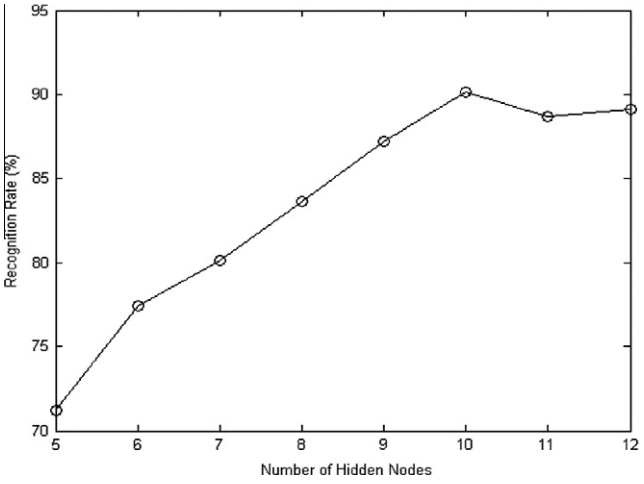

Fig. 12. The back-propagation learning network structure's hidden node number each and counts impact on recognition rate.

tion rate of our method is also bad. One of the effect factors is hidden node numbers. In Fig. 12, we test the back-propagation learning network structure's hidden node number each and counts impact on recognition rate. Feature-based fetching algorithm has the minimum RMSE of our methods, and we want to know the recognition rate with hidden node number's effect by feature-based fetching method. So we make the network's hidden node numbers of feature-based fetching method, for 10–15, probe into node number impact on recognition rate respectively from this experiment. In there shows the best recognition rate in network with 10 hidden nodes. We find its recognition rate is up to 90%, so our network structure uses 10 hidden nodes to hidden layer.

Next, we try to find out the three methods of fetching characteristic value, which can be used to reach the better recognition performance. At the same time, we also make other reflectance model's 3D face database by diffuse (Georghiades, Belhumeur, & Kriegman, 2001), specular (Cho & Chow, 2000), and cICA (Lee et al., 2006) reflectance model. We take 50 3D images regard as training data, 50 3D images regard as test data in our 3D database for back-propagation learning algorithm in the three layer feed-forward neural network. For efficiency testing, we take the number 15 characteristic values as training data. The inputs data is classified by the neural network with 10 hidden nodes. The result of recognition rate is showed in Table 1. The best recognition rate of our reflectance model is 90.15%. Furthermore, the recognition rate with used feature-based fetching method is close to the best recognition rate. Among other reflectance model 3D face databases, three recognition rates to fetch method, group different to have different high or low prices each in accordance with reflectance model, but method of us no matter in which characteristic fetching method which recognition rate is most high. It means that the feature-based method is an effective method.

Table 1
Each reflectance model pick the recognition rate followed the example of three fetching methods on characteristic value.

Fetching method	Reflectance model			
	The diffuse reflectance model (Georghiades et al., 2001)	The specular reflectance model (Cho & Chow, 2000)	The cICA reflectance model (Lee et al., 2006)	The proposed reflectance model
Rate of recognition (%)				
Contour method	62.31	31.36	85.12	88.32
Circle method	59.61	45.02	79.52	82.75
Feature method	61.23	44.21	87.16	90.15

6. Conclusions

In this paper, we proposed a new reflectance model for 3D surface reconstruction. IKICA as applied in this paper with temporal constraints results in a useful technique for the fast and efficient extraction of surface normal vectors from three surface reflection images. An important result derived from using the IKICA model for solving photometric stereo problems is that desired output values and smoothing conditions are not needed. This allows for easier convergence and makes the system stable.

For 3D surface reconstruction, several conclusions are listed below. (a) When we estimate the surface shape, the success of the reflectance model depends on two major components, including the diffusion and specular components. (b) In our methods, we do not know the locations of light sources for solving the photometric stereo problems. (c) The proposed IKICA network does not need any special parameter setting and the smoothing conditions.

For 3D face recognition, in order to make the neural network works properly, we need to find the weights and biases by the learning algorithms. In this study, the supervise learning is used. The network is given a set of training data which contains a number of input pattern and corresponding target output. The learning objective is to reach the high accuracy of classification by minimize the error between target output and network output in the training set. Furthermore, the trained classifier also should to provide a good performance in the untrained data.

In the future, we will study other efficient detection algorithms and integrate global characteristic value to further improve the recognition performance of this system. We hope for combining 2D and 3D information of human face in our recognition system effectively to achieve higher recognition rate. Finally, we expect that one day the proposed method could be used to realize a 3D human face recognition system.

References

An, G. Y., & Ruan, Q. (2006). KICA for face recognition based on kernel generalized variance and multiresolution analysis. In *IEEE proceedings of the first international conference on innovative computing, information and control* (pp. 3721–3733).

Bach, F. R., & Jordan, M. I. (2003). Kernel independent component analysis. *IEEE ICASSP*, 876–879.

Belhumeur, P. N., Kriegman, D. J., & Yuille, A. L. (1997). The bas-belief ambiguity. *CVPR*, 1060–1066.

Cheng, J., Liu, Q., & Lu, H. (2004). Texture classification using kernel independent component analysis. In *IEEE proceedings of the 17th international conference on pattern recognition* (pp. 567–578).

Cho, S. Y., & Chow, T. W. S. (2000). Learning parametric specular reflectance model by radial basis function network. *IEEE Transactions on Neural Networks, 11*(6), 1498–1503.

Frankot, R. T., & Chellappa, R. (1988). A method for enforcing integrability in shape from shading algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 10*(4), 439–451.

Georghiades, S. (2003). Incorporating the torrance and sparrow model of reflectance in uncalibrated photometric stereo. In *Proceedings of the ninth IEEE international conference on computer vision* (pp. 352–361).

Georghiades, S., Belhumeur, P. N., & Kriegman, D. J. (2001). From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 23*(6), 643–660.

Georghiades, S., Belhumeur, P. N., & Kriegman, D. J. (2001). From few to many: Illumination cone models for face recognition under variable lighting and pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 23*(6), 643–660.

Gordon, G. (1995). Face recognition from frontal and profile views. In *Proceedings of IWAFGR, Zurich* (pp. 47–52).

Hayakawa, K. (1994). Photometric stereo under a light source with arbitrary motion. *Journal of the Optical Society of America: A, 11*(11), 621–639.

Hyvärinen, A., Karhunen, J., & Oja, E. (2001). *Independent component analysis*. John Wiley & Sons, Inc.

Koch, R. (1991). Adaptation of a 3D facial mask to human faces in videophone sequences using model based image analysis. In *Picture coding symposium, Tokyo, Japan* (pp. 285–288).

Kocsor, A., & Toth, L. (2004). Kernel-based feature extraction with a speech technology application. *IEEE Transactions on Signal Processing, 52*(8), 2250–2263.

Lacher, R. C., Hruska, S. I., & Kuncicky, D. C. (1992). Back-propagation learning in expert networks. *IEEE Transactions on Neural Networks, 3*(1), 62–72.

Lee, W. S., Cheng, W. C., & Lin, C. J. (2006). A cICA-based photometric stereo for 3D human face reconstruction. In *19th IPPR conference on computer vision, graphics and image processing* (pp. 67–73).

Lin, C. T., Cheng, W. C., & Liang, S. F. (2005). A 3-D surface reconstruction approach based on postnonlinear ICA model. *IEEE Transactions on Neural Networks, 16*(6), 1638–1650.

Liu, Z., Zhang, Z., Jacobs, C., & Cohen, M. (2000). *Rapid modeling of animated faces from video*. Technical Report, Microsoft Corporation.

Lu, W., & Rajapakse, J. C. (2001). ICA with reference. In *Proceedings of the 3rd independent component analysis and blind signal separation: ICA2001* (pp. 120–125).

Martiriggiano, T., Leo, M., Spagnolo, P., & D'Orazio, T. (2005). Facial feature extraction by kernel independent component analysis. In *IEEE conference on advanced video and signal based surveillance* (pp. 270–275).

McGunnigle, G. (1998). *The classification of textured surfaces under varying illuminant direction*. Ph.D. thesis. Department of Computing and Electrical Engineering, Heriot-Watt University, Edinburgh.

Muller, K. R., Mike, S., Ratsch, G., Tsuda, K., & Scholkopf, B. (2001). An introduction to kernel-based learning algorithm. *IEEE Transactions on Neural Network, 12*(2), 181–201.

Nagel, B., Wingbermühle, J., Weik, S., & Liedtke, C. E. (1998). Automated modeling of real human faces for 3D animation. In *Proceedings of ICPR, Brisbane, Australia* (pp. 657–676).

The Yale Face Database B. Available from: http://cvc.yale.edu/projects/yalefacesB/yalefacesB.html.

Woodham, R. J. (1980). Photometric method for determining surface orientation from multiple images. *Journal of Optical Engineering, 19*(1), 323–367.

Xu, X. J., & Guo, P. (2006). KICA feature extraction in application to FNN based image registration. In *International joint conference on neural networks, Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada* (pp. 3602–3608).

Yin, L., Yin, L., & Yourst, M. (2003). 3D face recognition based on high-resolution 3D face modeling from frontal and profile view. *Proceedings of ACM SIGMM workshop on biometrics methods and applications, Berkley, California* (pp. 1–8).

Zhang, L. (1996). Estimation of eye and mouth corner point positions in a knowledge-based coding system. *SPIE, digital compression technologies and systems for video communications Berlin* (Vol. 2952, pp. 21–28).

Zhang, Z. (2001). Image-based modeling of objects and human faces. In *Proceedings of SPIE, video metrics and optical methods for 3D shape measurement, San Jose, USA* (Vol. 4309, pp. 1–15).