Machine Vision and Applications (1998) 10: 292–307

Machine Vision and Applications


# Sign language recognition using model-based tracking and a 3D Hopfield neural network

**Chung-Lin Huang, Wen-Yi Huang**

Electrical Engineering Department, National Tsing-Hua University, Hsin-Chu, Taiwan, ROC; e-mail: clhuang@ee.nthu.edu.tw, Fax: 886-35-715971

Received: 13 September 1996 / Accepted: 28 October 1997

**Abstract.** This paper presents a sign language recognition system which consists of three modules: model-based hand tracking, feature extraction, and gesture recognition using a 3D Hopfield neural network (HNN). The first one uses the Hausdorff distance measure to track shape- variant hand motion, the second one applies the scale and rotation-invariant Fourier descriptor to characterize hand figures, and the last one performs a graph matching between the input gesture model and the stored models by using a 3D modified HNN to recognize the gesture. Our system tests 15 different hand gestures. The experimental results show that our system can achieve above 91% recognition rate, and the recognition process time is about 10 s. The major contribution in this paper is that we propose a 3D modified HNN for gesture recognition which is more reliable than the conventional methods.

**Key words:** Gesture recognition – Model-based tracking – Feature extraction – 3D Hopfield neural network (HNN) – Hausdorff distance

## 1 Introduction

Humans are good at using gesture for human-to-human conversation. Gesture (or sign language) has been widely used in the deaf community. In the foreseeable future, gesture inputs can be widely applied for human- computer interface. Huang et al. [1] presented a review of the most recent works related to hand gesture interface techniques: glove-based technique, vision- based technique, and analysis of drawing gesture. Vision-based technique is the most natural way of constructing a human-computer interface which has many applications [2–4]. However, it has difficulties in (1) segmentation of moving hands; (2) tracking and analysis of hand motion; and (3) recognition.

*Sign language* consists of static hand gesture and dynamic hand gesture. The former is characterized by the hand posture determined by a particular finger-thumb-palm configuration, whereas the latter is depicted by hand movements that start with a slow initial move from the rest position, continue with a stroke (the hand shape may change during the stroke), and end by returning to the rest position. Tamura et al. [5] developed a system which can recognize 20 Japanese gestures based on matching simple cheremes. Davis et al. [6] proposed a model-based approach by using a finite-state machine to model four qualitatively distinct phases of a generic gesture. Hand shapes are described by a list of vectors and then matched with the stored vector models. Charayaphan et al. [7] proposed an image-processing method to detect the direction of hand motion by tracking the hand location, and they also used adaptive clustering of stop location, simple shape of the trajectory, and matching of the hand shape at the stop position to analyze 31 American Sign Language (ASL) signs.

Rehg et al. [8] designed a system called *DigitEyes* by using a 3D cylindrical kinematics model of the human hand with 27 degrees of freedom. Fingertips and links were chosen as the model-matching features and were extracted from either single or stereoscopic images. Darrell et al. [9] proposed another space-time gesture recognition. They represented the gestures by using sets of view models, and then matched the view model with the stored gesture models using dynamic time warping. Starner et al. [10] used a Hidden Markov Model (HMM) for visual recognition of complex, structured hand gestures such as ASL. They applied HMM to recognize "continuous" ASL of a full sentence and demonstrated the feasibility of recognizing complicated series of gesture.

Cui et al. [11] proposed a learning-based hand gesture recognition framework. It consists of a multiclass multivariant discriminant analysis to select the most discriminating feature (MDF), a space partition tree to achieve a logarithmic time complexity for a database, and a general interpolation scheme to do view inference. Hunter et al. [12] explored posture estimation based on the 2D projective hand silhouettes for vision-based gesture recognition. They used Zernike moments and normalization to separate the rough posture estimate from spatial specific (translation, rotation, and scaling). Wilson et al. [13] presented a state-based technique for the representation and recognition of gesture. States are used

*Correspondence to*: C.-L. Huang

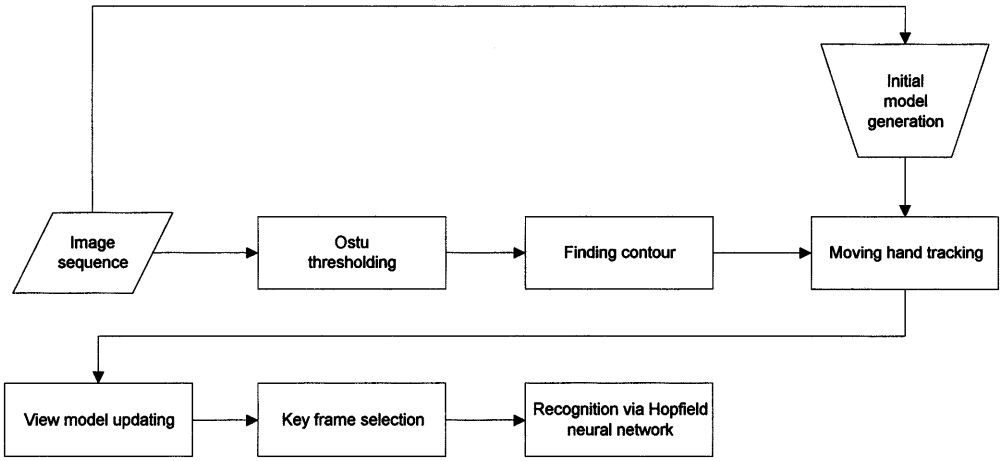



**Fig. 1.** The flow diagram of sign language recognition system

to capture both the variability and repeatability evidenced in a training set of a given gesture.

The major difficulty of the articulated objects (hands) analysis is the large variation of 2D hand-shape appearance, the fingers' and arms' articulartory motion, the viewpoint-sensitive 2D hand motion trajectories, the interference between the meaningful hand gesture and the meaningless moving background, and the occlusion between the moving articulated objects. To interpret sign language, we need to carry out 2D non-rigid shape analysis and the model-based motion tracking. The major assumption for a successful tracking algorithm is that the 2D shape of a moving hand changes slowly between two consecutive frames. In this paper, model-based motion tracking and analysis (using the Hausdorff distance) are developed to analyze the global motion and characterize the moving objects' trajectories. Then, the local motion is described by the shape variation of the moving objects and the orientation of the hands. Finally, a 3D Hopfield neural network (HNN) [19,20] is applied to recognize the gestures. The HNN has been used to solve the graph-matching problems [21–23] by using an appropriate energy function which represents the constraints that the nodes in the two graphs have to satisfy.

Figure 1 shows the flow diagram of our sign language recognition system. Our system has three phases: the feature extraction phase, the training phase, and the recognition phase. In the first phase, the system (1) tracks the moving shape-varying hand figures to identify their motion trajectories, (2) applies the corona effect smoothing and border extraction algorithm to find the border of the hand shape, (3) uses the Fourier descriptor (FD) to describe the hand shapes, (4) finds the symmetric axis of the hand shape and its orientation, (5) applies the shape and motion information to select the key frames. In the training phase, given as many as training samples, for each key frame of different gestures, the system classifies the motion and shape information of the corresponding key frames of different gesture image sequences into clusters. Then, the mean of every cluster is generated for the stored model. In the recognition process, the motion and shape information of the key frames are required by the 3D HNN to identify the best match between the input model and the stored models for gesture identification.

## 2 Model-based tracking

We propose a model-based tracking method which may extract a 2D model from the current input image frame and match to the successive frames in the image sequence. The model of the moving object is generated dynamically from the image sequence, rather than being provided a priori. The main constraint imposed by this method is that the 2D shape of the moving object does not change a lot between two consecutive frames. So, we apply a model-based tracking algorithm to track the moving objects. The model-based matching between the current view model and the next incoming frame is to find the best match in the next frame and then update the view model if the matching score is below a certain threshold. In view-model-based tracking, the shape difference is measured by the Hausdorff distance [14,15].

### *2.1 The Hausdorff distance*

Hausdorff distance measure is a nonlinear operator. It measures the extent to which each point of a "model" set lies near some point of an "image" set and vice versa. Thus, this distance can be used to determine the degree of resemblance between two objects that are superimposed on one another. Hausdorff distance measurement operates on a contour image. Since only edge points are selected, it will significantly reduce the computation burden. For two point sets A and B, the Hausdorff distance is defined as

$$H(A, B) = \max(h(A, B), h(B, A)) \,, \tag{1}$$

where $h(A, B) = \max_{a \in A} \min_{b \in B} ||a - b||$, $A = \{a1, a2, \ldots a_p\}$, $B = \{b_1, b_2, \ldots b_q\}$, and $||\cdot||$ is some underlying norm on the points of $A$ and $B$ . The function $h(A, B)$ is called the directed Hausdorff distance from $A$ to $B$.

The Hausdorff distance measures the mismatch between two sets that are at fixed positions with respect to one another. However, in real application to object matching, we denote $M$ as the model. The objects similar to $M$ may be located in any position in an image $I$. Thus, we are primarily interested in measuring the mismatch between all possible relative positions of two sets $I$ and $M$, as given by the Hausdorff distance as a function of relative position, that is, we need to modify Eq. 1 as

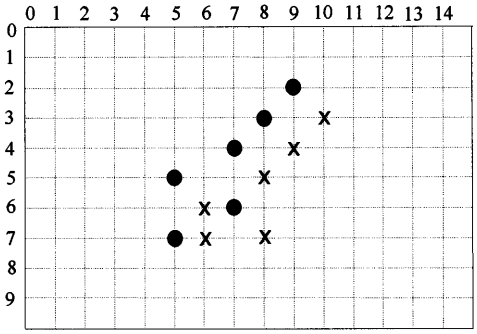


**Fig. 2.** Two point sets: $H(I, M) = 2^{1/2}$, $H_T(I, M) = 1$, $I = \{\bullet\}$, $M = \{\times\}$

$$H_T(I, M) = \min_t H(I, M \oplus t) \,, \tag{2}$$

where $M \oplus t = \{m + t | m \in M\}$.

Equation 2 is the minimum Hausdorff distance under translating $t$. We focus primarily on the case where the relative positions of model with respect to the image is the group of translations. Without loss of generality, we fix the set $I$ and allow only $M$ to translate. Figure 2 shows two sets of points, where the set $I$ is denoted by dots and set $M$ is depicted by crosses. $H(I, M)$ is large because there are points of $I$ that are not near any point of $M$, and vice versa. $H_T(I, M)$ is small, however, because there is a translation of $M$. That makes each point of $I$ nearly coincide with some points of $M$, and vice versa.

### *2.2 Comparing portions of shapes*

The Hausdorff distance measure is very sensitive to the presence of any outlying points. Besides, it is important to be able to identify the instances of a model that are only partly visible (either due to occlusion or to failure of the sensing device to detect the entire object). Thus, we use a more general rank order 'distance', which replaces the maximization operation in Eq. 2 with a rank operation (i.e., selection of the median value). A median is a more robust measure than the maximum, as is commonly known in statistics. This '*partial distance*' is defined as

$$h_K(M, I) = K^{th}_{m \in M} \min_{i \in I} ||m - i|| \,, \tag{3}$$

where $K^{th}_{m \in M} f(m)$ denotes the $K$-th ranked value of $f(m)$ over the set $M$. That is, if we consider the points in $M$ to be in sequence ordered by their values $f(m_1) \leq \ldots \leq (f(m_q)$, the $K$-th element in this sequence, $f(m_K)$, is the $K$-th ranked value. For example, the $q$-th ranked value is the maximum, and the $q/2$-th ranked value is the median value. We assume that the model $M$ has $q$ points.

The partial distance, $h_K(M, I)$, identifies the subset of $M$ of size $K$ which has the smallest directed Hausdorff distance to the set $I$. Intuitively, $h_K(M, I) = d$, when there is some subset of $M$ of size at least $K$ such that each point in this subset is within distance $d$ of some point in $I$. Thus, this measurement allows for a portion of $M$ which does not correspond to anything in $I$ (i.e., occlusion). To compute the partial directed distance $h_K(M, I)$, we specify some fraction $0 \leq f_1 \leq 1$ of the points of $M$ that are to be considered. For instance, if we compute $h_K(M, I)$ with $K = \lfloor f_1 q \rfloor$ where $f_1 = 0.7$ (i.e., using the 70-th percentile distance), then up to 30% of the points of $M$ need not be near any points of $I$. Here, we choose $f_1 = 0.7$ for moving-hands tracking.

### *2.3 Tracking using the Hausdorff distance*

The tracking process performs two functions: (i) finding the view model $M_t$ in the next frame $I_{t+1}$ by comparing the current view model $M_t$ obtained from the current frame to the next frame image $I_{t+1}$ and (ii) developing a new view model $M_{t+1}$ from the subset of $I_{t+1}$ for the next tracking step. Figure 3 illustrates the initial and updated models by demonstrating a model, an image, and the translation of the model that minimizes the partial directed Hausdorff distance for $K = \lfloor 0.7q \rfloor$.

### *2.4 Initial model and view model updating*

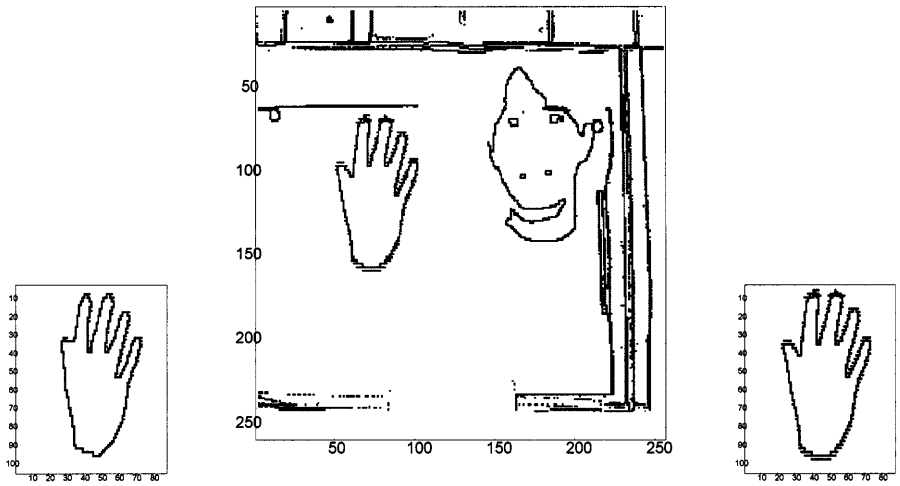
In this paper, we assume that in the early beginning of the input image sequence, there is no motion and the frame difference detector (FDD) [25] identifies nothing from the difference of two consecutive frames. Once the sign language speaker starts making gestures, the FDD will identify the hand shape as the initial model $M_0$. Here, we have the speaker wearing dark clothes with long sleeves, so that the motion of the arm will not be detected. Our algorithm tolerates the 2D shape change between the current view model and the consecutive image $I_{t+1}$ . However, we are interested in finding some subsets of the image $I_{t+1}$ that match well with the transformed view model $g^*(M_t)$. The subset is defined as

$$M_{t+1} = \left\{ i \in I_{t+1} | \min_{m \in M_t} ||g^*(m) - i|| \leq \delta \right\} \,, \tag{4}$$

which consists of image points that are within distance $\delta$ of some points of $g^*(M_t)$. The $\delta$ value controls the flexibility of the tracker which may tolerate the shape variation of the moving objects. The larger the value of $\delta$, the larger scale the view model is updated, it indicates that we may track the moving objects with larger shape variations. For example, if $\delta = 0$, then only those pixels of $I_{t+1}$ that are directly superimposed on $g^*(M_t)$ will be included in $M_{t+1}$, and thus the method will track only the objects moving without shape changes.

## 3 Local motion analysis

The local motion information consists of rotation, shape deformation and scaling. The fast-moving hand creates a blurred boundary and a zigzag object border called "corona effect". The "jaggedness" of the image is caused by interlacing the CCD camera which can be removed by a corona effect smoothing process (i.e., Fig. 4). Here, we propose a process to smooth the zigzag border, and then analyze the shape and orientation of the moving-hand figures.



a b c

**Fig. 3. a** Initial view model; **b** test image; **c** updated view model

### *3.1 Corona effect smoothing and border extraction*

Assume the hand shape is described by external boundary points: $V_0, V_1, \ldots V_m$, then we may use the corona effect smoothing and boundary extraction to extract fewer boundary points: $V_0, V_1, \ldots V_n$ with $n < m$. The corona effect smoothing and border extraction process is mentioned in Appendix A. In Fig. 4a, we show an example of the border extraction algorithm on a digital figure. Figure 4c and e illustrate the operation of this algorithm on a binary hand-shape image. Notice that applying this algorithm can eliminate small and thin branches of the hand figures. From Fig. 4a, we can make a brief description of the border extraction algorithm that if it is a region pixel, turn left and take a step; else turn right and take a step. Pixels (10, 6), (7, 7), (4, 7), (4, 6) are encountered twice. Hence, they are eliminated.

### *3.2 Orientation finding*

The orientation of the hand shape indicates to which direction the hand is pointing. Here, we assume that the elongated object $b(x, y)$ has an axis of elongation which can be used to define the orientation of the object. For an elongated object, we choose the axis of the least second moment ( the axis of the least inertia) as its orientation. Once the orientation $\theta$ is found, we have to find one of its two directions (180° difference) indicating to which direction the axis points. The directional symmetry axis starts from the centroid and ends at the fingertips. The 2D image coordinate $(i, j)$ is pointing downward, with the original (0, 0) located at the top-left point of the screen. By extending the directional symmetric axis upwards to intercept the $x$ axis (or $i$ axis), we find the containing angle $\alpha$ as it real orientation (see Fig. 5).

### *3.3 Fourier descriptor*

The Fourier descriptor (FD) will be invariant to the small hand-shape and trajectory variations and it is also tolerant to gesture-speed variation. Let the boundary points be defined as $\{(x(m),\ y(m))|\ m = 1, 2, \ldots N\}$, which can be described by Fourier series representation with coefficients $a(n)$ and $b(n)$, respectively [16–18]. Then, the boundary representation of a closed contour can be depicted by a FD vector. The elements of the FD vector are derived as

$$S(n) = r(n)/r(1)\ , \tag{5}$$

where $r(n) = \sqrt{|^2 + |b(n)|^2}$. Shridhar et al. [18] suggested that using FD vectors of dimension 10 for handwritten digit recognition is sufficient. Here, we assume that the local variation of hand shape is smooth, so that the high order terms of its FD are not necessary. In this approach, using 25 harmonics of the FDs is enough to describe the macroscopic information of the hand figures. They also mentioned that their FD vectors are invariant to rotation, translation, and scaling. These properties are proved in Lemma 1 mentioned in Appendix B.

Figure 6 shows the original hand shape image and after being rotated. Figure 6d shows the FD coefficients before and after rotating with angles of 45° and 90°, respectively. Figure 6e shows the difference of the FD vectors between the original and the rotated one. Notice that we do not show the first term of the FD vector, because its scale is large compared with the other 24 terms ($s(1) = 1$). Figure 7 shows the dilation-invariant properties of the FD vector. Figure 8 shows the difference between two distinct hand shapes. The distance between two FD vectors is defined as

$$Dis\,(\mathbf{S}_a, \mathbf{S}_b) = \sqrt{\sum_{i=2}^{25} |s_a(i) - s_b(i)|^2}\ , \tag{6}$$

where $\mathbf{S}_a = [s_a(1), s_a(2), \ldots s_a(25)]$ and $\mathbf{S}_b = [s_b(1), s_b(2), \ldots s_b(25)]$ are two FD vectors.

## 4 Key frame selection

After global/local motion analysis, the hand gesture image sequence is analyzed for either the stored model generation

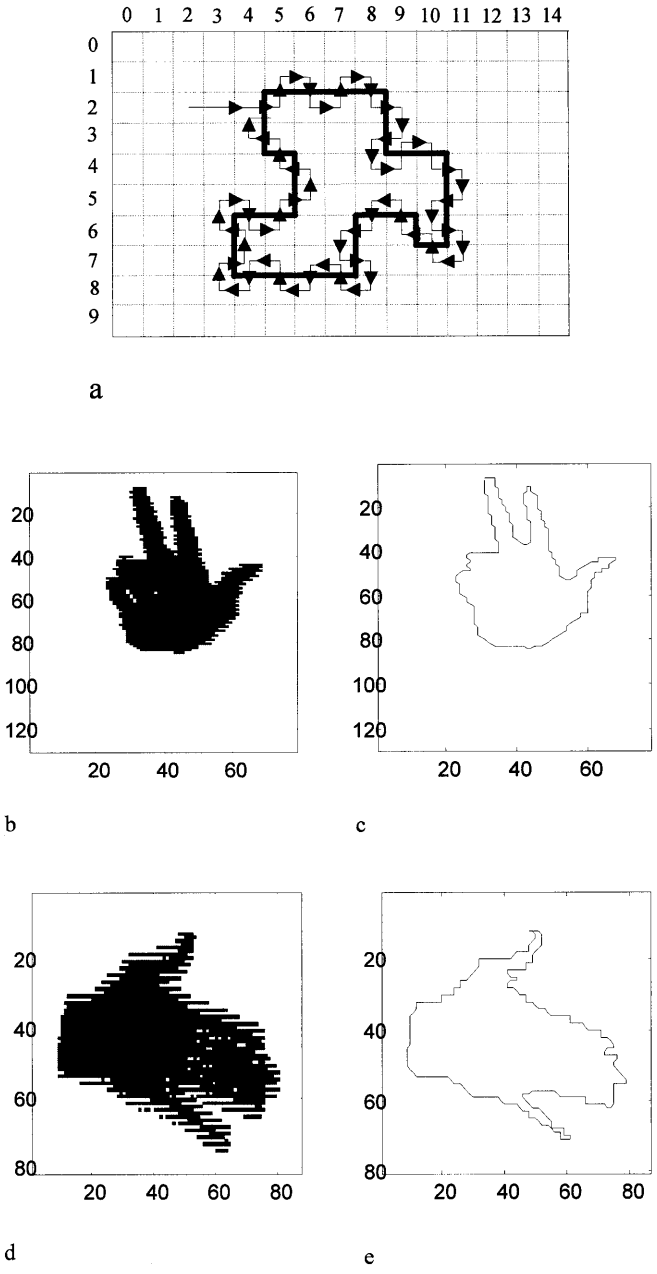



**Fig. 4a–e.** An example of border extraction algorithm on a digital figure. Two hand shapes image and their contours obtained from the border extraction algorithm

(in the training phase) or the input model generation (in the recognition phase). Not every frame in the image sequence is selected for the training or the recognition. Since the hand shapes between two consecutive view models are very similar to each other, we only need select some key frames for the stored model generation and the input model generation. For each unknown hand gesture image sequence, we generate an input model which is basically a graph consisting of key frames as the nodes. Each node has a local feature property (shape and local motion) as well as relational properties (global motion) with the other nodes. The input model and stored model are described by graphs which consist of nodes and links. The nodes describe the local information of the key frames, whereas the links depict the global motion properties. In the recognition phase, we use the HNN approach to solve the graph-matching problem between the input model and the stored model. The activity of each neuron represents matching function between the nodes and the links of any two models.

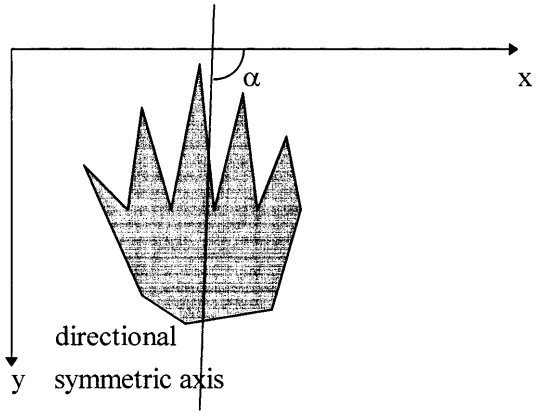



**Fig. 5.** The directional symmetric axis

The closed boundary of hand shape can be described by an FD vector with the first 25 coefficients. Due to the properties of rotation, translation, dilation invariant, we may reduce the database space of the stored models. For example, if a gesture consists of constant hand shape moving with rotational and translational motion from one position to another, we select the first frame and the last frame as our key frames. If the distance between two FD vectors of two different figures exceeds some threshold, then they must have significant shape difference. However, it is possible that there may exist two distinct figures and the distance of their FD vectors is below $\theta_1$. Therefore, we need another criterion for key frame selection which is based on the variation between the motion trajectories of the moving object. If the motion trajectory of the object in current frame is not smooth, it can be selected as another key frame. Here, we define the difference between two motion vectors as the inner product of two unit vectors. A threshold value $\theta_2$ is predetermined for identifying the smoothness of motion. Assume that the current frame and its succeeding and preceding key frames are considered and these three consecutive frames are used to determine whether the current frame is to be selected as a key frame. Let the centroid of the hand shape in the three frames $i-1$, $i$, $i+1$ be denoted as $C_{i-1}$, $C_i$, $C_{i+1}$, then the motion coherence ($MC$) measure is defined as

$$MC = 1 - \frac{\overline{C_{i-1}C_i} \cdot \overline{C_i,\, C_{i+1}}}{|\overline{C_{i-1}C_i}||\overline{C_iC_{i+1}}|} \,. \tag{7}$$

The smoother the motion $C_{i-1} \to C_i \to C_{i+1}$ is, the smaller value $MC$ may be generated with $0 \le MC \le 1$. The key frame selection algorithm can be summarized as follows:

*(1) Compute the FD vector of the moving object in the current frame.*
*(2) Find the unit motion vector with respect to the preceding and following frames and then calculate the MC between their unit motion vectors.*
*(3) Calculate the shape variance (SV) and orientation variance (OV) of the moving object between the current*

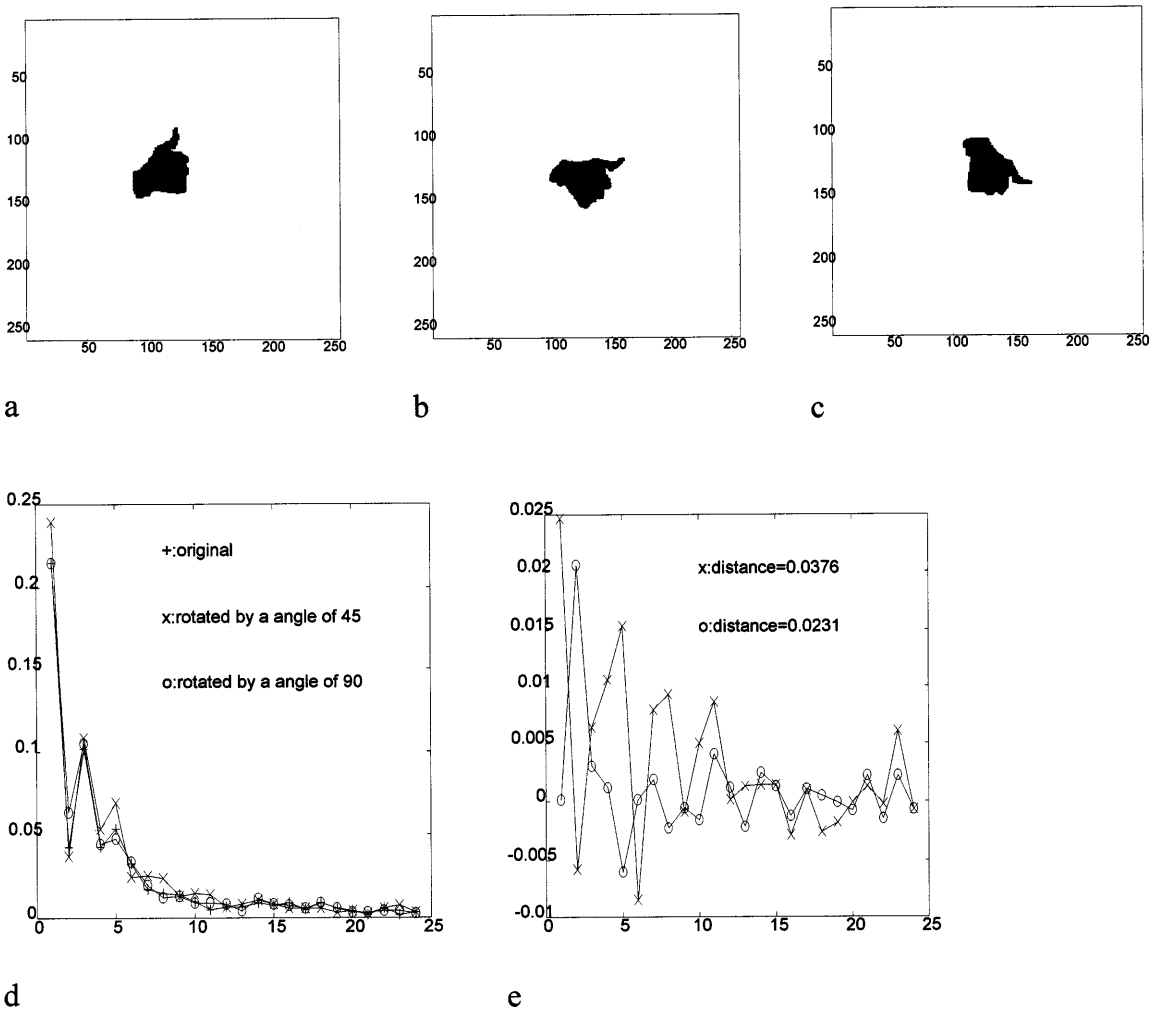



**Fig. 6a–e.** Illustrations of rotation-invariant: **a** original hand shape, **b** rotated by an angle of 45°, **c** rotated by an angle of 90°, **d** Fourier descriptor vectors without the first term; **e** 'x' indicates the difference of Fourier descriptor vectors between **a** and **b**. 'o' indicates the difference between **a** and **c**

*frame and those of the preceding and following key frames. Assume the two hand shapes in frames* $i-1$ *and* $i$ *are described by two Fourier vectors* $S_{i-1}$ *and* $S_i$ *with orientations* $\alpha_{i-1}$ *and* $\alpha_i$*, we may have* $SV = Dis\left(S_{i-1}, S_i\right)$ *(i.e., Eq. 6) and* $OV = |\alpha_{i-1} - \alpha_i|$.

(4) *Select current view model as the key frame if the* $SV > \theta_1$, $MC > \theta_2$, $OV > \theta_3$.

Figure 9 shows gesture 'conscience' from frame 0 to frame 22. Figure 10 shows the images after using the Ostu thresholding method [24]. Figure 11a illustrates the key frames obtained from the above algorithm. The selected key frames are the 0-th, 16-th, 19-th frame, respectively, in this sequence. Figure 11d shows the difference of FD vectors with respect to previous key frame.

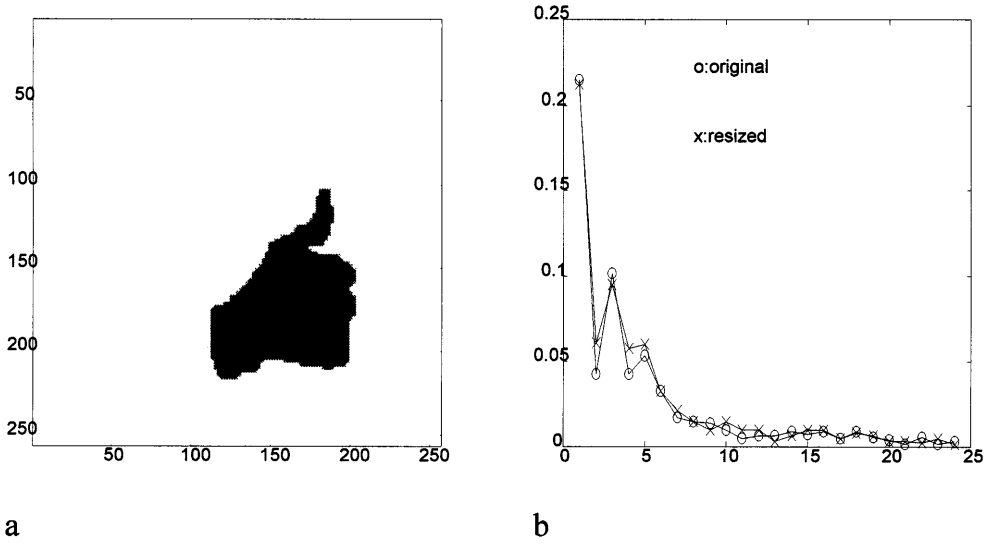
## 5 Gesture recognition using the 3D HNN model

The Hopfield neural network (HNN) is characterized by a network of neuron-like units with symmetric connections between units, continuous output values, sigmoidal input-output transfer function, and the appropriate global energy function. Hopfield [19,20] had shown that the equation of motion for a fully connected network with symmetry connection always leads to convergence of stable state with global minimum energy. The net input to any neuron is the sum of the current flowing through the set of resistors. The input depends on the unit similarity of each neuron and relationship similarity between the neuron and its neighboring neurons.

A 3D binary HNN with fully interconnected neurons is constructed as shown in Fig. 12. Each input model is matched with the stored gesture models for the best match. A global match is achieved between two models when the network settles down to a stable state, usually, at its minimum energy. The number of key frames in the input model is $F$, and there are $S$ stored gesture models with different number of key frames given as $N_1, \ldots N_S$. The state of each neuron $V_{iks}$ represents the matches between the two key frames (nodes) $i$ and $k$ from the input model and the $s$-th stored model. $C_{iks,jls}$ is a compatibility measure (see Eq. 24) between two neurons with $V_{iks}$ and $V_{jls}$. The compatibility measure will influence the matches of the two models which can be described by the following energy function

$$E = AE_1 + BE_2 \,, \tag{8}$$

where



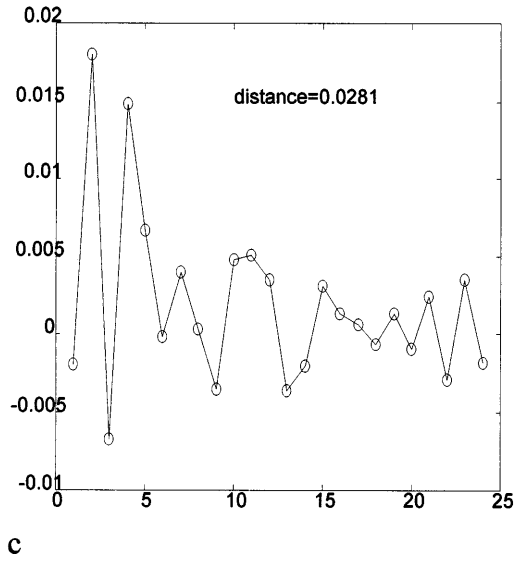




**Fig. 7a–c.** Illustration of dilation-invariant: **a** resized hand shape from Fig. 6a; **b** FD vectors without the first term; **c** the difference of FD vectors between Figs. 6a and 7a is the distance = 0.0281

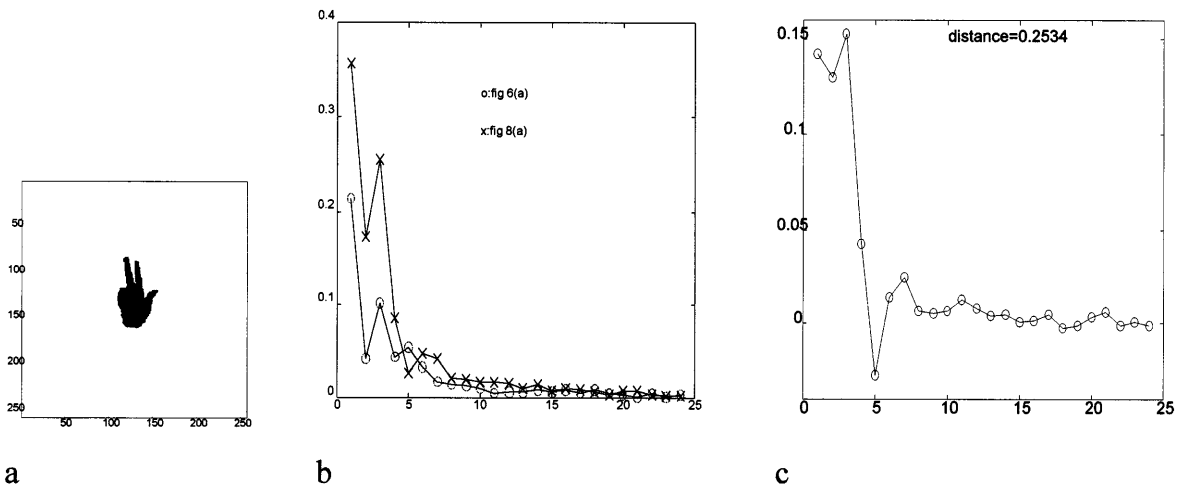



**Fig. 8a–c.** The difference between Figs. 6a and 8a: **a** a hand shape image, **b** FD vectors without the first term, **c** the distance is 0.2534

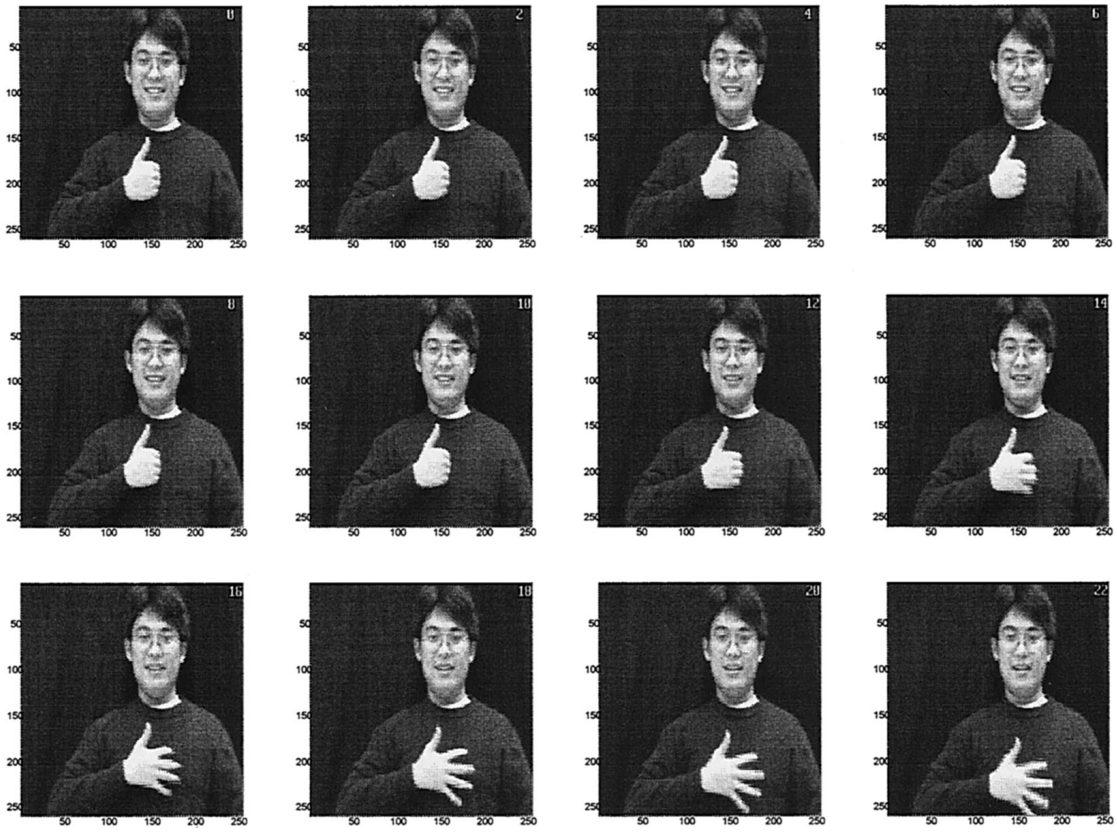


**Fig. 9.** Original input sequence

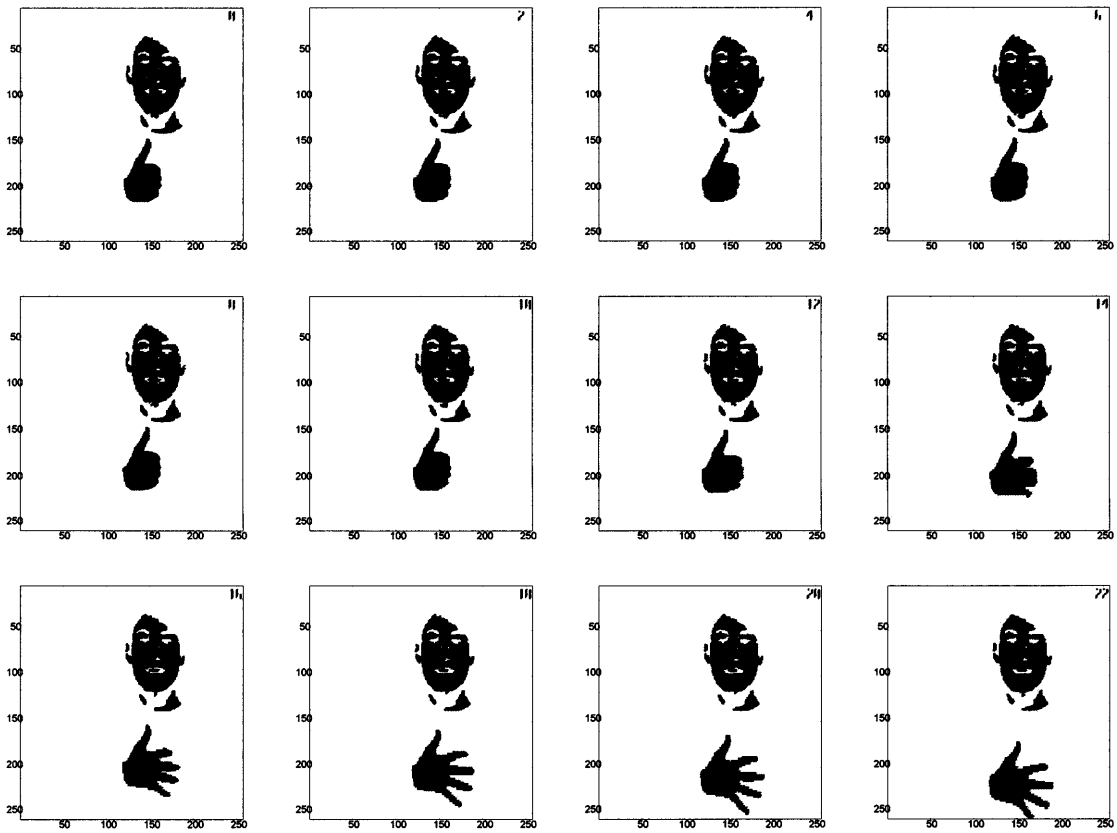


**Fig. 10.** Binary images produced by Ostu thresholding

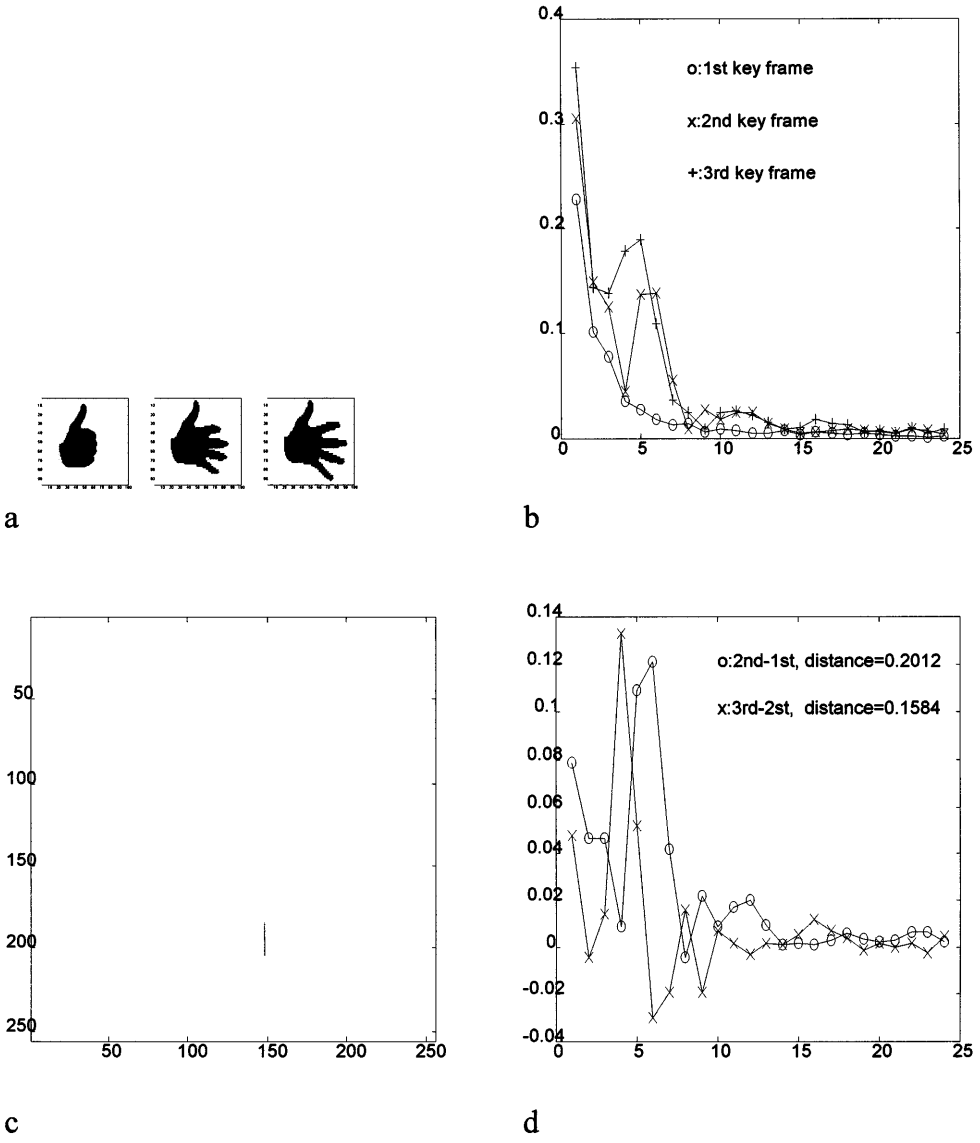



**Fig. 11a–d.** The key frame selection: **a** key frames obtained from Fig. 10, **b** FD vectors without the first term, **c** motion trajectory, **d** difference of FD vectors between key frames

$$E_1 = -\sum_{p=1}^{S}\sum_{q=1}^{S}\sum_{i=1}^{F}\sum_{k=1}^{N_p}\sum_{j=1}^{F}\sum_{l=1}^{N_q} C_{ikp,jlq} V_{ikp} V_{jlq} \ ,$$

$$E_2 = \sum_{p=1}^{S}\left(\sum_{i=1}^{F}\left(1-\sum_{k=1}^{N_p} V_{ikp}\right)^2 + \sum_{k=1}^{N_p}\left(1-\sum_{i=1}^{F} V_{ikp}\right)^2\right) + \sum_{i=1}^{F}\sum_{k=1}^{N_p}\left(1-\sum_{p=1}^{S} V_{ikp}\right)^2 \ .$$

The sub-energy $E_1$ describes the excitatory interaction between the nodes of the same model, the sub-energy $E_2$ depicts inhibitory interaction of the nodes on the same row or the same column, and the nodes of different models. To simplify the computation, we only consider the relation between neuron $V_{iks}$ and its $5 \times 5 \times 5$ neighboring neurons. The compatibility measure $C_{iks,jls}$ is only considered when the two nodes $k$ and $l$ are in the same model. If they are not in the same model, it becomes absolutely inhibitive.

The equation is equivalent to minimizing the Hopfield-type energy function [19] as

$$E = -\frac{1}{2}\sum_{p=1}^{S}\sum_{q=1}^{S}\sum_{i=1}^{F}\sum_{k=1}^{N_p}\sum_{j=1}^{M}\sum_{l=1}^{N_q} T_{ikp,jlq} V_{ikp} V_{jlq} - \sum_{p=1}^{S}\sum_{i=1}^{F}\sum_{k=1}^{N_p} V_{ikp} I_{ikp} \ , \tag{9}$$

where $T_{ikp,jlq}$ denotes the connection between a neuron $V_{ikp}$ and another neuron $V_{jlq}$. The energy change $\Delta E$ due to the a change $\Delta V_{ikp}$ in the state of neuron $ikp$ is given as

$$\Delta E = -\Delta V_{ikp}\left(\sum_{q=1}^{S}\sum_{j=1}^{F}\sum_{l=1}^{N_q} T_{ikp,jlq} V_{jlq} + I_{ikp}\right) \ . \tag{10}$$

Equation 8 can be rewritten as follows

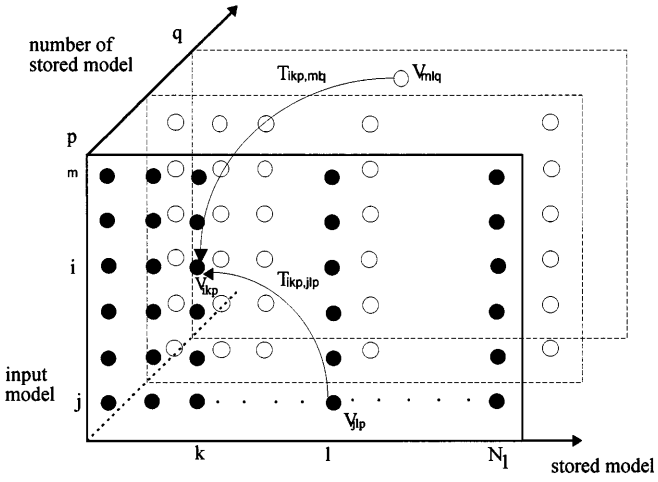



**Fig. 12.** 3D Hopfield neural network

$$E = -\sum_{p=1}^{S}\sum_{i=1}^{F}\sum_{k=1}^{N_q} 6BV_{ikp} + \sum_{p=1}^{S}\sum_{q=1}^{S}\sum_{i=1}^{F}\sum_{k=1}^{N_p}\sum_{j=1}^{F}\sum_{l=1}^{N_q} \left(-AC_{ikp,jlq} + B\delta_{ij}\left(2\delta_{pq} + \delta_{kl}\right)\right) V_{ikp}V_{jlq} \quad (11)$$

where there are $S$ stored models, each stored model $p$ has a number of $N_p$ neurons, and the input model is specified by $F$ neurons. $A$ and $B$ are two constants which are determined experimentally as $A$ = 5, $B$ = 1. Comparing the above two energy functions, it can be shown that

$$T_{ikp,jlq} = 2AC_{ikp,jlq} - 2B\delta_{ij}\left(2\delta_{pq} + \delta_{kl}\right) \quad \text{and} \quad I_{ikp} = 6B\ , \quad (12)$$

where $\delta_{ij}$ = 1 if $i = j$, and $\delta_{ij}$ = 0 otherwise. The input to the neuron $ikp$ is the sum of the current from its neighboring neurons $V_{jlq}$ which is described as

$$U_{ikp} = \sum_{q=1}^{S}\sum_{j=i-2}^{i+2}\sum_{l=k-2}^{k+2} T_{ikp,jlq}V_{jlq} + I_{ikp}\ . \quad (13)$$

The compatibility measure, $C_{ikp,jlq}$, is determined by the unit similarity and relationship similarity among the neurons. The unit similarity is defined in terms of the shape similarity and trajectory similarity of the nodes (key frames) of the input model and the stored model. Let the input hand shape of node $i$ with orientation $\alpha_i$ be described by an FD vector as $A_i$ = [$s_i$(1), $s_i$(2), ... $s_i$(25)], and the stored hand shape of node $k$ with orientation $\alpha_k$ be depicted by another FD vector $A_k$ = [$s_k$(1), $s_k$(2), ... $s_k$(25)]. Then the shape similarity is measured in terms of the distance between the two feature vectors and orientations, i.e., $||A_i - A_k||$ and $|\alpha_i - \alpha_k|$.

Now let $G_i$ be a unit motion vector of the moving hand in the input gesture. The initial states of all neurons are set to 1 or 0, depending on the degree of matching between the spatial similarity and temporal similarity between two feature vectors $A_i$ and $A_k$ as follows:

$$\begin{cases} V_{ikp} = 1 \text{ if } ||A_i - A_k|| < \theta_1, G_i \cdot G_k > \theta_2, \\ \qquad \text{and } |\alpha_i - \alpha_k| < \theta_3 \\ V_{ikp} = 0 \quad \text{otherwise} \end{cases}, \quad (14)$$

where $||A_i - A_k||$ defines the distance between two feature vectors, and $G_i \cdot G_k$ is the inner product between two vectors. The thresholds $\theta_1$, $\theta_2$ and $\theta_3$ are given as 0.15, 0.707, and 0.33, respectively, in our experiments. Both $G_i$ and $G_k$ are unit vectors. Hence, the range of $G_i \cdot G_k$ is between -1 and +1. It corresponds to the cosine of the angle between two motion vectors. For every two neurons $ikp$ and $jlq$, the unit similarity is defined as

$$\begin{aligned} &R(A_i, A_k) = ||A_i - A_k||\ , \\ &R(G_i, G_k) = G_i \cdot G_k\ , \\ &R(\alpha_i, \alpha_k) = |\alpha_i - \alpha_k|\ . \end{aligned} \quad (15)$$

The relationship similarity is determined as

$$\begin{aligned} &R\left(Ar_{ij}, Ar_{kl}\right) = |\,||A_i - A_j|| - ||A_k - A_l||\,| \\ &R\left(Gr_{ij}, Gr_{kl}\right) = |G_i \cdot G_j - G_k \cdot G_l|\ , \\ &R\left(\alpha r_{ij}, \alpha r_{kl}\right) = ||\alpha_i - \alpha_j| - |\alpha_k - \alpha_l||\ . \end{aligned} \quad (16)$$

Therefore, the compatibility measure is formulated as follows:

$$C_{ikp,jlq} = \begin{cases} w_1 f_1\left(R\left(A_i, A_k\right)\right) + w_1 f_1\left(R\left(A_j, A_l\right)\right) \\ +w_2 f_2\left(R\left(G_i, G_k\right)\right) + w_2 f_2\left(R\left(G_j, G_l\right)\right) \\ +w_3 f_3\left(R\left(\alpha_i, \alpha_k\right)\right) + w_3 f_3\left(R\left(\alpha_j, \alpha_l\right)\right) \\ +w_4 f_4\left(R\left(Ar_{ij}, Ar_{kl}\right)\right) \\ +w_5 f_5\left(R\left(Gr_{ij}, Gr_{kl}\right)\right) \\ +w_6 f_6\left(R\left(\alpha r_{ij}, \alpha r_{kl}\right)\right) \text{ if } p = q \\ -1 \quad \text{otherwise} \end{cases} \quad (17)$$

where $\sum_{i=1,...6} w_i$ = 1. $C_{ikp,jlq}$ is close to 1 for a pair of corresponding neurons, while for non-corresponding pairs, it is near $-1$. In our experiments, the functions $f_i$ are depicted as follows

$$\text{if } R(\bullet) \le \theta_i, \text{ then } f_i(R(\bullet)) = \beta_i, \text{ else } f_i(R(\bullet)) = \gamma_i\ , \quad (18)$$

where $i = 1, 2, \ldots, 6$. The weighting factors are assigned as: $w_1$ = 0.25, $w_2$ = 0.1, $w_3$ = 0.05, $w_4$ = 0.25, $w_5$ = 0.25, $w_6$ = 0.1, respectively. The $\beta_i$ = 0.5, $\gamma_i = -0.5$ for $i$ = 1, 2, 3 and $\beta_i$ = 1, $\gamma_i = -1$ for $i$ = 4, 5, 6. The thresholds are defined as: $\theta_1$ = 0.15, $\theta_2$ = 0.96, $\theta_3$ = 0.5, and $\theta_i$ = 0.02 for $i$ = 4, 5, 6. We apply the deterministic relaxation to update the state of each neuron synchronously. The algorithm is summarized in the following steps.

(1) Initialize the state of neurons by using Eq. (14).
(2) Randomly pick up a node $ikp$.
(3) Calculate its input by using Eq. (13).
(4) Decide the new state of each neuron according to the following rules:

$$\begin{aligned} &V_{ikp} \to 1 \text{ if } U_{ikp} > 0.5 \\ &V_{ikp} \to 0 \text{ if } U_{ikp} < -0.5 \\ &V_{ikp} \to \text{ no change if } -0.5 \le U_{ikp} \le 0.5 \end{aligned} \quad (19)$$

(5) Count the changes of the states. If there is no change after a number of iterations, stop and go to step 6, otherwise repeat the process from step 2.
(6) Output the final states of neurons $V_{ikp}$, which will be the final matches between the stored gesture models and the input gesture.

The above algorithm is based on noiseless dynamics of the HNN model – noiseless in the sense that there is no noise present in the synaptic transmission of signals. However, actually, the synaptic transmission in a nervous system is a

noisy process brought on by random fluctuations from the release of neurotransmitters, and other probabilistic causes. According to the original HNN model, to account for the effects of synaptic noise in a neural network, we may apply a probabilistic mechanism in the firing of neurons, i.e., to represent the effects of synaptic noise by thermal fluctuations. Specifically, a neuron $ikp$ decides to fire according to the value of the net potential input $U_{ikp}$ acting on it with probability of firing being defined by $P\left(U_{ikp}\right)$. The state $V_{ikp}$ of neuron $ikp$ is defined by the probabilistic rule

$$\begin{aligned} V_{ikp} &= +1 \text{ with probability } P\left(U_{ikp}\right) \ , \\ V_{ikp} &= 0 \quad \text{with probability } 1 - P\left(U_{ikp}\right) \ . \end{aligned} \tag{20}$$

A standard choice for $P\left(U_{ikp}\right)$ is the sigmoid-shaped function as

$$P\left(U_{ikp}\right) = \frac{1}{1 + \exp\left(-2U_{ikp}/T\right)} \ , \tag{21}$$

where $T$ is a pseudo-temperature that is used to control the noise level. When the input potential $U_{ikp} = 0$, then $V_{ikp} = \pm 1$, each with probability 1/2. When $T \to 0$, which corresponds to the noiseless limit, the width of the threshold region shrinks to zero, the probabilistic firing rule is reduced to deterministic rule.

We may foresee that the proposed approach may find a single match between the input model and stored model with all the neurons which have come to their active stable states are in the same sub-net (all neurons $\{ikp\}$ with $V_{ikp} = 1$ are in the same sub-net, $p$=constant). The single match is the best result that we can expect; however, it may happen that the finally active stable neurons may not be in the same sub-net ($p \neq$constant). In this case of multiple matches, we may apply the simulated annealing again or randomize the neurons ignition sequence to ensure the total energy has reached the real global minimum. The finally stable active neurons may change to a single match or another multiple matches. If these active neurons indicate the same multiple matches, then, we may select the sub-net with the largest number of active stable neurons as the best matched stored model.

## 6 Experimental results and discussion

In the experiments, we asked the persons (subjects) to wear the dark clothes with long sleeves and stand before a dark curtain under normal lighting. Although the proposed model-based tracking system can track and identify the moving objects in front of a complicated background, however, the identified objects may include partial background information. Therefore, by having the persons wear the dark clothes and stand before a dark curtain, we can avoid the background information being extracted and treated as the moving objects. The limitations are necessary for the recognition process to be more reliable, after all, the gesture-makers are not required to wear white gloves as usual.

There are fifteen sign gestures illustrated in Fig. 13. Each consists of a sequence of image frames (30 frames) capturing a single hand moving in different directions with constant or time-varying hand shape. Each image sequence is taken at 30 frames per second, and each gesture takes about 1 s. The input image sequence is taken at three different time intervals: in the first (begin) period, the sign language speaker remains silent (no gesture), then in the second (action) period, the speaker starts making one simple hand gesture, and finally, in the last (end) period, the speaker remains silent again. Normally, the second interval in completed in about 0.5 s. The FDD may easily identify the beginning and the end of the gesture in the image sequence.

In the experiments, four gestures have constant hand shape, whereas eleven gestures have time-varying hand shape. They may have similar or different moving trajectories. They are very simple hand gestures, so that they can be completed in less than 1 s. We used an Oculus F/64 frame grabber with 8M Video RAM and a SONY XC-7500 CCD camera with electronic shutter to capture a fast-moving hand gesture without causing too much blur effects on the images. The main bottleneck of the system is the data transfer of the entire image sequence between the frame gabber and the host computer. The host computer was equipped with a Pentium CPU and 16 MB main memory running at 133 MHz.

For each input frame, the tracking process needs to select the best threshold value for the input image binarization. Then the contour of the hand shape is extracted for the tracking process to do the model-based hand tracking. In the experiments, the image preprocessing and feature extraction processes take about 5 s, which includes thresholding, model-updating, hand tracking and FD analysis. In total, the recognition system requires less than 10 s from image sequence capturing to gesture recognizing. In the training stage, for each gesture, we have taken many different hand gesture image sequences from many different subjects, and for each gesture, we have 15 different training image sequences to generate the corresponding stored model. The gestures made by different subjects may have different speeds, which are captured by the image sequences with different numbers of frames in the second (action) period; however, only the key frames are needed for the stored model and input model. Normally, the number of selected key frame numbers from different image sequences is the same.

In our experiments, we found that, if given an unknown gesture not included in the pre-trained 15 gestures, the network converges to the stable state, indicating the match of the most similar stored model. For instance, if the gesture ‘seven’ is excluded from the stored model, and is assigned as an input gesture, then we may find that the most similar gesture ‘a coin’ will be identified. The key frame number is different for different gestures. The gesture with only two selected key frames indicating the constant hand shape motion. We need not normalize the numbers of key frames from different gestures. In the 3D HNN, we need 60 neurons for recognizing 15 different gestures. The initial state of the HNN is determined by comparing the key frame of the input model with all the key frames of the stored model. The initial state of each neuron is determined by Eq. 14. Then, the state of each neuron will be influenced by its $5 \times 5 \times 5$ neighboring neurons before reaching a stable state. On the average, each neuron iterates 3–4 times and the entire neural network takes 4 s to reach a table state. Since there are not many neurons in the neural network, we need not use the probabilistic mechanism (Eqs. 20 and 21) to update the states

**Fig. 13.** Five selected frames of every input sequence of 15 different gestures from Taiwanness Sign Language: (1)a coin; (2)cigarette, (3)flower, (4)reluctantly, (5)row, (6)take, (7)immediately, (8)understand, (9)hate, (10)left, (11)seven, (12)moon, (13)eight, (14)walk, (15)conscience


**Fig. 13.** (continued)

**Table 1.** The hand gesture recognition rate of input image sequences from different subjects.

| Methods | Tested sequence | Correct Unique match | Multiple matches | Recognition rate |
|---|---|---|---|---|
| PRM | 759 | 645 | 61 | 85% |
| HNN | 759 | 690 | 0 | 91% |

of the neurons. Once the number of stored image models is increased, the neuron number needs to be increased and the operation time for the HNN to reach a stable state will be extended.

In the recognition phase, if we use the image sequences from the same subjects as the ones participating in the training, most of the input gestures are accurately identified, and the recognition rate is above 96%. For each gesture, we tried at least 50 different unknown input hand gesture image sequences from different subjects (different from the training subjects). On the average, the system achieves a 91% recognition rate (see Table 1). The system may not identify the input gesture because the shape distance between the key frames is small and their trajectories are also very similar. To avoid the misrecognition, the outcome of the HNN will provide a list of possible matches with the corresponding matching scores. The 3D HNN may reach a stable state with the active neurons in the different subsets, which indicates multiple matches. The matching score is calculated by the number of the active neurons for the match between the input model and the designated stored model.

In the recognition phase, we may use the shape pattern and trajectory pattern to describe each model and then use conventional pattern matching [26] to find the resemblance between the input model and the stored model. However, there may be more than one match (i.e., matching score above a certain threshold). Sometimes, the best match is not necessarily the correct match. This situation may occur when the key frames of the input model A are similar to part of the key frames of the stored model B, and their trajectories are also very similar. In such a case, the conventional pattern matching method may provide more than one best match. A probabilistic relaxation method (PRM) [27] was also implemented and tested on the input image sequences for comparison with the HNN. The PRM is implemented as follows:

1. Let $P_{iks}$ represent the degree of match between the $i$-th key frame of an input model and the $k$-th key frame of the $s$-th stored model. The initial value of $P_{iks}$ is determined by Eq. 14.
2. Each $P_{iks}$ is iteratively updated according to the following rule: $P_{iks}^{r+1} = \frac{1}{F}\sum_{j=1}^{F} \max_{1\le l\le N_s} C_{iks,jls} P_{jls}^{r}$, where $C_{iks,jls}$ is the compatibility measure between two matched pairs ($i$, $k$) and ($j$, $l$) defined in Eq. 17.
3. After several iterations, $P_{iks} \to 1$, then we may find the best matched pairs.
4. If the input model is matched with the $s$-th stored model, then for each $i$-th key frame, there is only one maximum $P_{iks} \to 1$ in each row and each column, whereas the others $P_{ijs} \to 0$ for $j \neq k$ or $P_{jks} \to 0$ for $j \neq i$.

The PRM has shown comparable results with that of the HNN results only for recognizing simple distinctive gestures. For some ambiguous gestures, we may have multiple matches by using PRM. The experimental results of using the PRM are given in Table 1.

To avoid multiple matches, we apply the HNN with embedded inhibitory connections among different models. The advantage of HNN over PRM is that the matching constraints can easily be incorporated explicitly in the energy function. Therefore, by using the HNN approach, we find that the inhibitive force between the neurons of different models will make neurons of the network stabilize on the same subnet which indicates only one best correct match. Since the extracted features in the input image sequence cannot be reliable, we need to apply the HNN model, which imposes global constraints on the solution by allowing only a one-to-one match between the input model and stored models. Another advantage of the HNN architecture is that it can be implemented in hardware and operated in parallel during the gesture recognition process. With the parallelism processing mechanism, the HNN model can provide fast and accurate matching.

## 7 Conclusion

Although we have demonstrated a successful simulation system to interpret simple hand gestures, there are still some problems for further studies to (1) increase tracking process speed; (2) enlarge the library of stored models; (3) normalize the input image sequence for slow and fast gesture; (4) extract the moving hands from complex backgrounds; (5) implement a real-time recognition system, (6) minimize the number of thresholding parameters which are experimentally determined. Besides the above problems, the toughest problem in sign language recognition is how to translate the "continuous" sign language sentence. Since each sign language sentence consists of several cheremes. Each chereme whose meaning can be a verb, a noun, or an adjective is encoded in a sequence of image frames. How to separate the entire image sequence representing a sign language sentence into several sub-sequences representing several cheremes is a very difficult problem. Currently, we only consider a chereme-based image sequence as the basic input to our system for recognition.

## References


1. Huang TS, Pavlovic VI (1995) Hand Gesture Modeling, Analysis and Synthesis. In: Int. Workshop on Automatic Face and Gesture Recognition, 26–28 June 1995, Zurich, Switzerland, pp 73–79
2. Maggioni C (1995) Gesture Computer – New Ways of Operating a Computer. In: Int. Workshop on Automatic Face and Gesture Recognition, 26–28 June 1995, Zurich, Switzerland, pp 166–171
3. Freeman W, Weissman C (1995) Television control by hand gesture. In: Int. Workshop on Automatic Face and Gesture Recognition, 26–28 June 1995, Zurich, Switzerland, pp 179–183
4. Kjeldsen R, Kender J (1995) Visual Hand Gesture Recognition for Window System Control. In: Int. Workshop on Automatic Face and Gesture Recognition, 26–28 June 1995, Zurich, Switzerland, pp 184–188
5. Tamura S, Kawasaki S (1988) Recognition of Sign Language Motion Images. Pattern Recognition 21(4): 343–353

6. Davis J, Shah M (1994) Visual Gesture Recognition. IEE Proc Vis Image Signal Process, 141(2): 101–106
7. Charayaphan C, Marble AE (1992) Image Processing System for Interpreting Motion in American Sign Language. J. Biomed Eng 15: 419–425
8. Rehg JM, Kanade T (1994) DigitEyes: Vision-based hand tracking for human-computer interaction. In: Workshop on Motion of Non-rigid and Articulated Objects, 11–12 November 1994, Austin, Tx., pp 16–22
9. Darrell T, Pentland A (1993) Space-Time Gestures. In: IEEE Conf. CVPR-93, 15–17 June, New York, pp 335–340
10. Starner TE, Penland A (1995) Visual Recognition of American Sign Language using Hidden Markov Models. In: Int. Workshop on Automatic Face and Gesture Recognition, June 26–28 1995, Zurich, Switzerland, pp 189–194
11. Cui Y, Weng J (1995) Learning-based Hand Sign Recognition. In: Int. Workshop on Automatic Face and Gesture Recognition, 26–28 June 1995, Zurich, Switzerland, pp 201–206
12. Hunter E, Schlenzig J, Jain R (1995) Posture Estimation in Reduced-Model Gesture Input System. In: Int. Workshop on Automatic Face and Gesture Recognition, 26–28 June 1995, Zurich, Switzerland, pp 290–295
13. Wilson AD, Bobick AF (1995) Configuration States for the Representation and Recognition of Gesture. In: Int. Workshop on Automatic Face and Gesture Recognition, 26–28 June 1995, Zurich, Switzerland, pp 129–134
14. Huttenlocher DP, Noh J, Rucklidge WJ (1993) Tracking Non-rigid Objects in Complex Scene. In: Proc. 4th ICCV, pp 93–101
15. Huttenlocher DP, Klanderman GA, Rucklidge WJ (1992) Comparing Images using the Hausdroff distance. IEEE Trans PAMI, pp 437–452
16. Zahn CT, Roskies RZ (1972) Fourier Descriptor for Plane Closed Curves. IEEE Trans Comput 21(3): 269–281
17. Persoon E, Fu KS (1977) Shape Discrimination Using Fourier Descriptor. IEEE Trans SMC 7(3): 170–179
18. Shridhar D, Badreldin A (1984) High-Accuracy Character Recognition Algorithm using Fourier and Topology Descriptors. Pattern Recogn 17: 515–524
19. Hopfield JJ (1984) Neurons with graded response have collective computational properties like those two-state neuron. Proc Natl Acad Sci USA 81: 3088–3092
20. Hopfield JJ, Tank DW (1985) Neural Computing of decisions in optimization problems. Biol Cybernetics 52: 141–152
21. Nasser N, Li W (1991) Object Recognition by a Hopfield Neural Network. IEEE Trans SMC 21( 6): 1523–1535
22. Mjolsness E, Gindi G, Anandan P (1989) Optimization in model matching and perceptual organization. Neural Comput 1: 218–229
23. Kuner P, Ueberreoter B (1988) Pattern Recognition by graph matching combinatorial versus continuous optimization. Int J Pattern Recogn Artif Intell 2(3): 527–542
24. Ostu N (1979) A Threshold Selection Method from Gray-Level Histograms. IEEE Trans Syst Man Cybernetics SMC-9: 62–66
25. Huang CL, Chao TT (1996) Motion-Compensated Interpolation for Scan Rate Up-Conversion. Opt Eng 35 (1): 166–176
26. Ballard DH, Brown CM (1982) Computer Vision. Prentice Hall, Englewood Cliffs, N.J.
27. Rosenfeld A, Kak AC (1982) Digital Image Processing, Vols. I and II. Academic Press, New York


## Appendix A

*Corona effect smoothing and border extraction algorithm*

(1) Set $N$ to 0 ( boundary points counter); and pointer $x = 0$, $y = 0$;
(2) Scan the image, using pointer $x$ and $y$, from top to bottom ($y = y + 1$), left to right ($x = x + 1$) until a region pixel $p(x, y) \neq 0$ is encountered;
(3) Push the location $(x, y)$ of starting point (i.e., $p(x, y) = 1$, the first boundary point encountered) into top of stack (TOS=$(x, y)$) and save $(x, y)$ as $Px = x$, $Py = y$, then decrease pointer $y$ by $y = y - 1$;
(4) Increase counter $N$ by $N = N + 1$;
(5) do

if $y - Py = -1$ and $x - Px = 0$, then
  if $p(x, y) = 1$, then $Px = x$; $Py = y$;
    insert_boundary_point$(y, x, \&N)$;
    $x = x - 1$;
  else
    $Px = x$; $Py = y$;
    $x = x + 1$;
else if $y - Py = 1$ and $x - Px = 0$, then
  if $p(x, y) = 1$, then $Px = x$; $Py = y$;
    insert_boundary_point$(y, x, \&N)$;
    $x = x + 1$;
  else
    $Px = x$; $Py = y$;
    $x = x - 1$;
else if $y - Py = 0$ and $x - Px = 1$, then
  if $p(x, y) = 1$, then $Px = x$; $Py = y$;
    insert_boundary_point$(y, x, \&N)$;
    $y = y - 1$;
  else
    $Px = x$; $Py = y$;
    $y = y + 1$;
else if $y - Py = 0$ and $x - Px = 1$, then
  if $p(x, y) = 1$, then $Px = x$; $Py = y$;
    insert_boundary_point$(y, x, \&N)$;
    $y = y + 1$;
  else
    $Px = x$; $Py = y$;
    $y = y - 1$;

} while($(x, y)$ is not starting point or $N$ is equal to 0);

(6) Procedure: insert_boundary_point$(y, x, \&N)$

{ if $(x, y)$ is not equal to TOS
    push $(x, y)$ into TOS;
    $N = N + 1$;
 else
    pop TOS;
    $N = N - 1$;
}

## Appendix B

**Lemma 1.** *Let* $\{(x(m), y(m))|m = 1, 2, \ldots N\}$ *describe the boundary of a closed shape and the Fourier coefficients be described as* $\{a(n), b(n)\}$. *Now, if the boundary points are rotated by an angle* $\theta$, *then the* $r^2(n) = |a(n)|^2 + |b(n)|^2$ *is invariant to this rotation.*

*Proof.* Let the boundary of contour points be rotated by the following

$$x_\theta(m) = \cos(\theta)x(m) - \sin(\theta)y(m) ,$$
$$y_\theta(m) = \sin(\theta)x(m) + \cos(\theta)y(m) \qquad \text{(B.1)}$$

$a_\theta(n)$, and $b_\theta(n)$ are the Fourier coefficients after rotation

$$a_\theta(n) = \sum_{m=1}^{N} x_\theta(m)e^{-j2\pi mn/N}$$

$$b_\theta(n) = \sum_{m=1}^{N} y_\theta(m) e^{-j2\pi mn/N} \tag{B.2}$$

By substituting (B.1) into (B.2), we have

$$a_\theta(n) = \sum_{m=1}^{N} (\cos(\theta)x(m) - \sin(\theta)y(m)) e^{-j2\pi mn/N} ,$$
$$b_\theta(n) = \sum_{m=1}^{N} (\sin(\theta)x(m) + \cos(\theta)y(m)) e^{-j2\pi mn/N} , \tag{B.3}$$

which can be rewritten as

$$\begin{aligned} a_\theta(n) &= \sum_{m=1}^{N} (\cos(\theta)x(m)) e^{-j2\pi nmn/N} \\ &\quad - \sum_{m=1}^{N} (\sin(\theta)y(m)) e^{-j2\pi nmn/N} \\ b_\theta(n) &= \sum_{m=1}^{N} (\sin(\theta)x(m)) e^{-j2\pi nmn/N} \\ &\quad + \sum_{m=1}^{N} (\cos(\theta)y(m)) e^{-j2\pi nmn/N} \end{aligned} \tag{B.4}$$

The rotated coefficients $a_\theta(n)$ and $b_\theta(n)$ can also be assumed to be rotated version of the coefficients $a(n)$ and $b(n)$ as

$$\begin{aligned} a_\theta(n) &= \cos(\theta)a(n) - \sin(\theta)b(n) , \\ b_\theta(n) &= \sin(\theta)a(n) + \cos(\theta)b(n) \end{aligned} \tag{B.5}$$

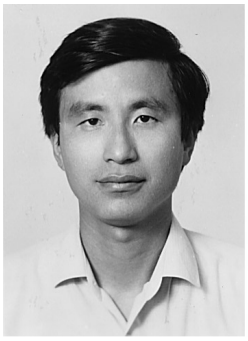
Now we take the square of magnitude of the two coefficients $a_\theta(n)$ and $b_\theta(n)$

$$\begin{aligned} |a_\theta(m)|^2 &= [\cos(\theta)Re(a(n)) - \sin(\theta)Re(b(n))]^2 \\ &\quad + [\cos(\theta)Im(a(n)) - \sin(\theta)Im(b(n))]^2 \\ |b_\theta(m)|^2 &= [\sin(\theta)Re(a(n)) + \cos(\theta)Re(b(n))]^2 \\ &\quad + [\sin(\theta)Im(a(n)) + \cos(\theta)Im(b(n))]^2 \end{aligned}$$

The $r^2(n)$ (Eq. 12) can be proved to be rotation invariant as

$$\begin{aligned} |a_\theta(n)|^2 + |b_\theta(n)|^2 &= \cos^2(\theta)Re^2(a(n)) + \sin^2(\theta)Re^2(b(n)) \\ &\quad + \cos^2(\theta)Im^2(a(n)) + \sin^2(\theta)Im^2(b(n)) \\ &\quad + \sin^2(\theta)Re^2(a(n)) + \cos^2(\theta)Re^2(b(n)) \\ &\quad + \sin^2(\theta)Im^2(a(n)) + \cos^2(\theta)Im^2(b(n)) \\ &= Re^2(a(n))(\cos^2(\theta) + \sin^2(\theta)) \\ &\quad + Re^2(b(n))(\cos^2(\theta) + \sin^2(\theta)) \\ &\quad + Im^2(a(n))(\cos^2(\theta) + \sin^2(\theta)) \\ &\quad + Im^2(b(n))(\cos^2(\theta) + \sin^2(\theta)) \\ &= Re^2(a(n)) + Re^2(b(n)) + Im^2(a(n)) \\ &\quad + Im^2(b(n)) \\ &= |a(n)|^2 + |b(n)|^2 \end{aligned}$$

Therefore, the boundary representation using Fourier Descriptor vectors is invariant to rotation. Q.E.D.



**Chung-Lin Huang** was born in Tai-Chung, Taiwan, in 1955. He received his B.S. degree in Nuclear Engineering from the National Tsing-Hua University, Hsin-Chu, Taiwan, ROC, in 1977, and M.S. degree in Electrical Engineering from National Taiwan University, Taipei, Taiwan, ROC, in 1979, respectively. He obtained his Ph.D. degree in Electrical Engineering from the University of Florida, Gainesville, FL, USA, in 1987. From 1981 to 1983, he was an associate engineer in ERSO, ITRI, Hsin-Chu, Taiwan, ROC. From 1987 to 1988, he worked for the Unisys Co., Orange County, Calif., USA. as a project engineer. Since August 1988 he has been with the Electrical Engineering Department, National Tsing-Hua University, Hsin-Chu, Taiwan, ROC. Currently, he is a professor in the same department. His research interests are in the area of image processing, computer vision, and visual communication. Dr. Huang is a member of IEEE and SPIE.

**Wen-Yi Huang** was born in Tainan, Taiwan, in 1970. He received his B.S. degree from the National Taiwan Institute of Technology in 1993 and M.S. degree from the Electrical Engineering Department, National Tsing-Hua University, Hsin-Chu, Taiwan, in 1995, respectively. His research interests are signal processing and image processing. Currently, he is working for Tridan Co., Taiwan.