Machine Vision and Applications (2001) 12: 243–258

Machine Vision and Applications
© Springer-Verlag 2001

A model-based hand gesture recognition system

Chung-Lin Huang, Sheng-Hung Jeng

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

Received: 19 May 1999 / Accepted: 4 September 2000

Abstract. This paper introduces a model-based hand gesture recognition system, which consists of three phases: feature extraction, training, and recognition. In the feature extraction phase, a hybrid technique combines the spatial (edge) and the temporal (motion) information of each frame to extract the feature images. Then, in the training phase, we use the principal component analysis (PCA) to characterize spatial shape variations and the hidden Markov models (HMM) to describe the temporal shape variations. A modified Hausdorff distance measurement is also applied to measure the similarity between the feature images and the pre-stored PCA models. The similarity measures are referred to as the possible observations for each frame. Finally, in recognition phase, with the pre-trained PCA models and HMM, we can generate the observation patterns from the input sequences, and then apply the Viterbi algorithm to identify the gesture. In the experiments, we prove that our method can recognize 18 different continuous gestures effectively.

Key words: Hand gesture recognition – Principal component analysis (PCA) – Hidden Markov model (HMM) – Hausdorff distance measurement – Viterbi algorithm

1 Introduction

Hand gesture is normally used in our daily life to communicate with one another. Children know how to make gesture communication before they can talk. Clearly, gesture recognition has become one of the most interesting research topics in human-computer interface. Most of the recent works [1] related to hand gesture interface techniques have been categorized as: glove-based methods and vision-based methods. The vision-based methods, based on the computer vision techniques, have been proposed for locating objects and recognizing gestures. The gloved-based gesture recognition methods require expensive wired "Dataglove" equipment [2]. Gesture recognition research has many applications such as window-based user interface [3] and video coding [4].

The model-based static gesture recognition approach proposed by Davis and Shah [5], uses 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. A dynamic gesture recognition system for American sign language (ASL) interpretation has been developed by Charayaphan et al. [6]. They propose a method to detect the direction of hand motion by tracking the hand location, and use adaptive clustering of stop location, simple shape of the trajectory, and matching of the hand shape at the stop position to analyze 31 ASL signs.

A more reliable method called the space-time gesture recognition method developed by Darrell et al. [7] represents gestures by using sets of view models. It recognizes the gestures by matching the view models to stored gesture patterns using dynamic time warping. Cui and Weng [8] propose a learning-based hand sign recognition framework by using the multiclass, multivariate discriminant analysis system to select the most discriminating feature (MDF), and then applying a space partition tree to reduce time complexity. Hunter et al. [9] explore posture estimation based on the 2D projective hand silhouettes for vision-based gesture recognition. They use Zernike moments and normalization to separate the rough posture estimate from specific translation, rotation, and scaling.

The most difficult part of gesture identification is to classify the posture against complex backgrounds. Triesch et al. [10] employ elastic graph matching for the classification of hand postures in gray-level images. Heap et al. [11] construct a 3D deformable point distribution model of the human hand. Then, they use this model to track an unmarked human model with six degrees of freedom. Another simplified method (by Lee and Kunii [12]) assumes that the positions of fingertips in the human hand, relative to the palm, is almost always sufficient to differentiate the gestures. They propose the skeleton-based model consisting of 27 bones and 19 links, each link has different degrees of freedom.

However, gesture recognition is more generally treated as a time variation problem, therefore, more and more com-

Correspondence to: C.-L. Huang

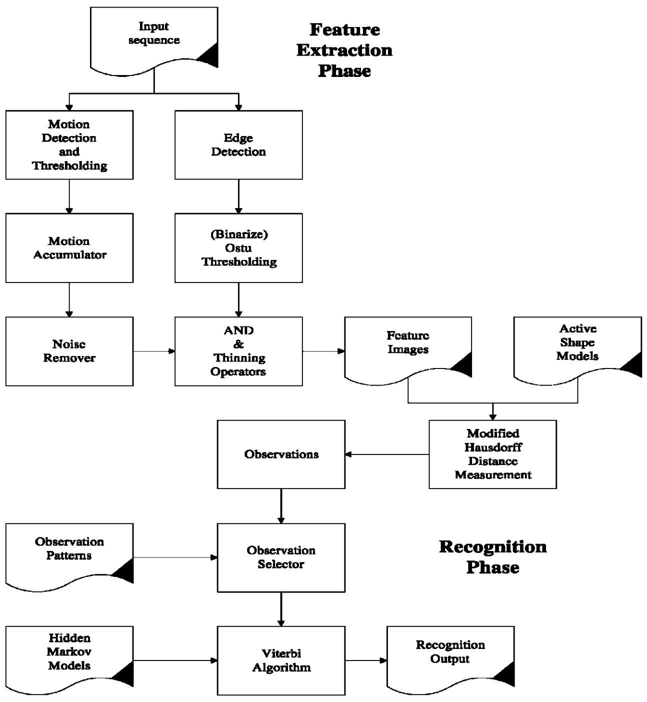


Fig. 1. The flow diagram of hand gesture recognition system

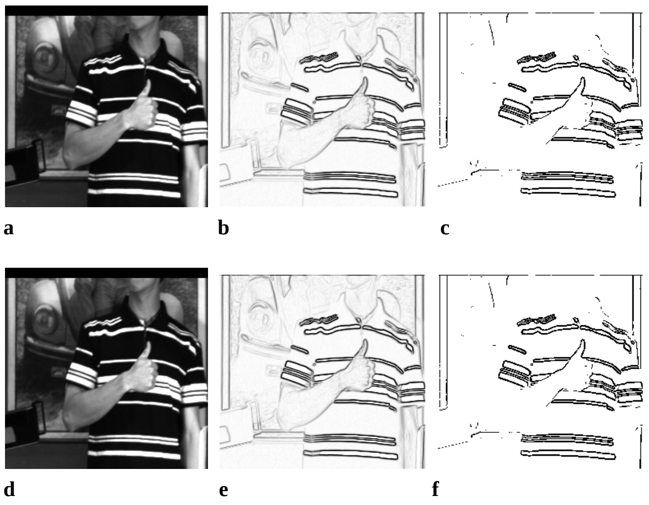


Fig. 2a–f. Edge information. **a** and **d** are the original images. **b** and **e** are the gradient strength images. **c** and **f** are results after the Ostu thresholding

puter vision researchers have become aware of using hidden Markov models (HMMs) to model the image sequence of gestures. Starner et al. [13] use HMM to recognize a full sentence and demonstrate the feasibility of recognizing a series of complicated series of gesture. Bobick et al. [14] present a state-based method for representation and recognition of gesture from a continuous stream of sensor data. The variability and repeatability evidence in a training set of a given gesture is classified by states.

The major difficulties of the complex articulated-objects analysis are the appearance of large variation of 2D hand shapes, the view point sensitive for 2D hand shapes and motion trajectories, the transition between the meaningful gestures, and the interference of complex background. The gesture image sequence is basically composed of spatial and temporal variation signals, so we need to apply the principal component analysis (PCA) and HMM to model the spatial and the temporal shape variation of the gestures. Figure 1 shows the flow diagram of our model-based hand tracking and recognition. We use the Hausdorff distance measurement to measure the differences of the input gestures and pre-stored PCA shape models. The differences are then converted to observations for HMM training (in the training phase) or for state sequence evaluation (in the recognition phase) by using the Viterbi algorithm based on the pre-trained HMM. The most likely state transition sequence is associated with the gesture to be recognized.

The hand gesture recognition system can be described in three phases: the training phase, the feature extraction phase, and the recognition phase. We represent each gesture by a sequence of states. The state transition indicates the spatial temporal variability of the gesture, which is invariant to the speed of motion. In the feature extraction phase, we develop a new method, which combines the edge and motion information to extract the shape of the moving object. Then, in training phase, we develop a simple training algorithm for the PCA models and HMM which characterize the spatial and the temporal variation of gestures. Finally, in the recognition phase, we apply pre-stored PCA models to observe the input gesture, and use the Hausdorff distance measure to find the similarity between the extracted features and the pre-stored PCA models. In the experiments, we show that our gesture recognition system is insensitive to motion speed and trajectory direction, and it can precisely recognize 18 different gestures in complex background.

2 Feature extraction phase

Here, we assume that the moving objects in complex background are somehow identifiable by their edge boundaries. Usually, the edge information is too noisy to be applicable for computer vision system, and most of the edge information is redundant. Here, we assume that the background is complex but stationary, the moving hand is the only moving object in the scene. Using the frame difference, we can partially capture the motion information. By accumulating the motion information of the moving objects in several consecutive frames, we may localize the moving pixels more accurately.

First, we apply the Sobel operators [17] and Ostu thresholding method [18] to extract the edges in the scene (see Fig. 2). Second, we find the motion information by using the motion accumulator and the noise remover. Finally, we apply the AND operation on the edges and the accumulated motion pixels to acquire the real moving edges.

2.1 Motion accumulator

To find the edge information from the object movement, we assume that the gesture in the sequence is non-stationary. In the spatial-temporal space, the motion detector may capture all the possible moving objects by examining the local gray-level changes. Let F_i be the ith frame of the sequence and

D_i be the difference image of ith and $(i+1)$th frame defined as

$$D_i = T_t \{|F_i - F_{i+1}|\}, \tag{1}$$

where T_t is a thresholding operation with threshold t:

$$T_t(f(x,y)) = \begin{cases} 1, \text{ if } f(x,y) \geq t, \\ 0, \text{ if } f(x,y) < t, \end{cases}$$

where $D_i = \{f(x,y) | 0 \leq x \leq 255, 0 \leq y \leq 255\}$. Here, we let the image size of F_i and D_i be 256 $\times$ 256, and t = 20 which is experimentally determined. Pixels with value 1 in the difference image are treated as the motion pixels. Since we are interested in the motion information, however, the difference image between two continuous frames provides very limited motion information. Therefore, we develop a motion accumulator to collect more motion information. A motion accumulator used to collect the motion pixels from D_i to D_{i+n} and from D_i to D_{i-n} is defined as

$$\overset{n}{\underset{j=0}{\Omega}} A_{i+j} = D_{i-j} \text{ OR} \ldots \text{OR } D_{i-1} \text{ OR } D_i \text{ OR } D_{i+1} \text{OR} \ldots$$
$$\ldots \text{OR } D_{i+j}\,. \tag{2}$$

Using the difference images, we accumulate the motion pixels forward one frame and backward one frame, and then put them in the motion accumulator. The accumulation operation continues until the total number of motion pixels is larger than a certain threshold that will be mentioned in the next section. For the ith frame, the motion accumulator collects the motion pixels from difference images, e.g., D_i, D_{i+1}, D_{i-1}, ..., D_{i-n}, D_{i+n}.

2.2 Search region finding

In the first image frame, it is difficult to locate the object's position (inside search region); however, in the succeeding frames, it is easier to track the moving object by referring to the rectangle (enclosing the search region) of the first frame. The search region of the first frame is determined by the following four steps.

1. Determine the minimum frame duration number f_s^* as

$$f_s^* = \arg\min_{f_s} \sum_y \sum_x \left\{ \overset{f_s}{\underset{n=-f_s}{\Omega}} A_{i+n}(x,y) \right\} \geq t_s\,,$$
$$1 \leq f_s < T\,, \tag{3}$$

where t_s = 1800 is the search region threshold, T is the length of image sequence, and Ω is defined in Eq. 2.

2. Use f_s^* to find the image A_s:

$$A_s = \overset{f_s^*}{\underset{n=-f_s}{\Omega}} A_{i+n}\,. \tag{4}$$

Figures 3a and 3c illustrate the A_s of these two examples.

3. Since the image A_s is noisy, we apply a 3 $\times$ 3 median filter and the noise remover to reduce the noise. The noise remover will be introduced in the next section.
4. Find a rectangle to circumscribe the search region based on the density of the motion pixels inside A_s, and the distribution of the motion pixels.

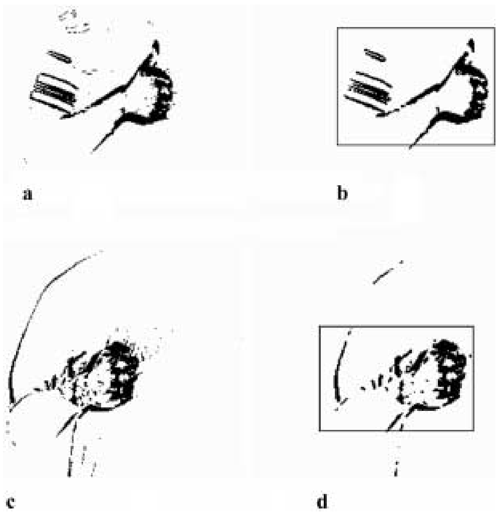


Fig. 3a–d. Determine the search region for the first frame. **a** and **c** are the accumulated images which are noisy. **b** and **d** are noise-filtered images which are used to determine the search region

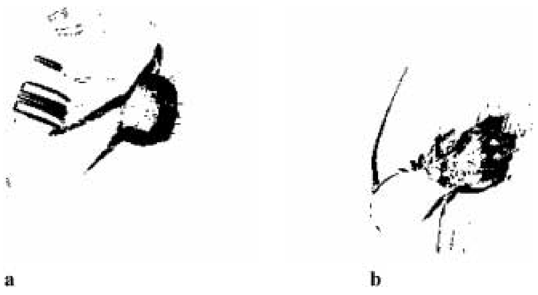


Fig. 4a,b. Illustrates the results of different thresholds

The operation of the last step is straightforward, it tries different-sized rectangle blocks, at various locations of the image A_s, to enclose a certain amount of motion pixels. The rectangle block at a certain location that encloses the motion pixels with the largest ratio of the number of motion pixels to the entire rectangle area will be selected as the search region. In this way, we can exclude the motion information of the arm from the search region since the motion pixels in the arm area are loosely distributed. Figures 3b and 3d illustrate the results of noise reduction and the search regions.

2.3 Feature image generation

Once we find the search region, we apply the same operation (Eqs. 3 and 4) again with a higher threshold (ts = 2400) to obtain the images (see Figs. 4a and 4b) with more accumulated motion pixels. Comparing Fig. 4 with Fig. 3, we find that the former provides more motion information. Since the image A_s is noisy, we need to apply a 3 $\times$ 3 median filter and then use the *noise remover* to reduce noise. The *noise remover* (see Fig. 6) consists of three operations illustrated as follows.

1. Search and mark all the 8 $\times$ 8 overlapped regions on an 256 $\times$ 256 image $A_s(x,y)$, these regions, which enclose at least 20 motion pixels, are defined as

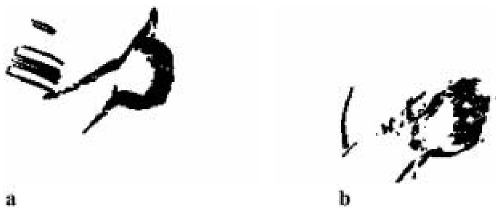


Fig. 5a,b. The extracted active motion pixels: **a** and **b** show the effect of noise remover applied on with Fig. 4a and 4b, respectively

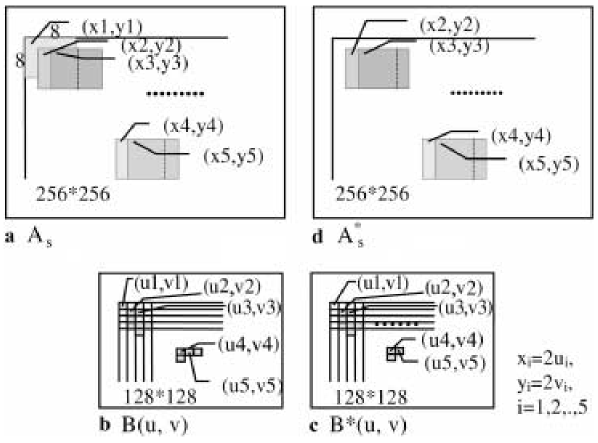


Fig. 6a–d. The operation of noise remover. Between **a** and **b** is a mapping from 8 × 8 overlapped regions to points defined in Eq. 5. Between **b** and **c** is continuity checking defined in Eq. 6. We see that $(u1, v1)$ in $B^*(u, v)$ is eliminated. Finally, we restore all motion pixels as active motion pixels by referring **c** for the blocks moved from **a** to **d** as defined by Eq. 7

$$B(u,v) = \begin{cases} 1, \sum_{y=y_1}^{y_1+7} \sum_{x=x_1}^{x_1+7} A_s(x,y) \geq 20 \\ 0, \text{otherwise} \end{cases},$$

$$u, v \in I, \quad x_1 = 2u, \quad y_1 = 2v, \tag{5}$$

where $I = \{1, \ldots 128\}$, and $B(u,v)$ is a 128 × 128 image. These regions are selected because the motion pixels belonging to the same object are close together.

2. Check the local continuity of these regions. The local continuity at $B(u,v) = 1$ is satisfied if and only if at least two of its eight-connected neighbors have value 1.

$$B^*(u,v) = \begin{cases} 1, \sum_{i=-1}^{1} \sum_{j=-1}^{1} B(u+i, v+j) \geq 3, \\ 0, \text{otherwise}. \end{cases} \tag{6}$$

3. Retrieve the active motion pixels by using

$$\{A_s^*(x+i, y+j) \mid i, j = 0, 1, \ldots, 7\}$$

$$= \begin{cases} \{A_s(x+i, y+j) | i, j = 0, 1, ..., 7)\}, \\ \qquad \text{if } B^*(u,v) = 1, \\ 0, \qquad \text{otherwise}, \end{cases} \tag{7}$$

where $x = 2u$, $y = 2v$. Figures 5a and 5b illustrate the results of applying the noise remover on Figs. 4a and 4b.

After having extracted the edge (spatial) and the motion (temporal) information, we can combine these two kinds of information to locate the object by using "AND" and "thinning" operations [24]. The "AND" operator keeps all the pixels in both the binarized edge image and the motion detected image, whereas the "thinning" operator removes the redundant contour pixels. Figure 7 illustrates the results of "AND" and "thinning" operations. The feature image is generated after applying "AND" and "thinning" operations on the motion pixel mages. In the experiments, we assume that the background is stationary, however, a small movement of the body during the gesture making is acceptable.

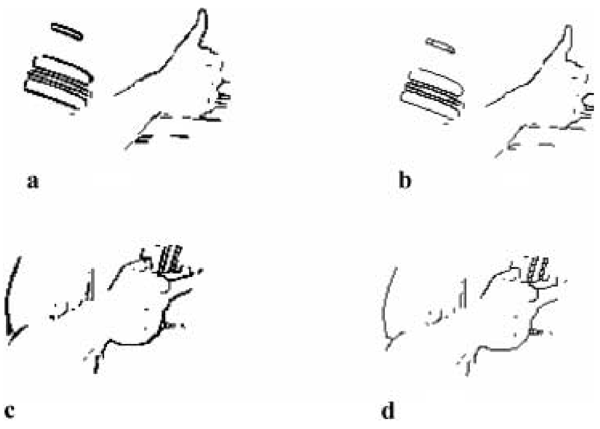


Fig. 7a–d. Results of "AND" and "Thinning" operations. **a** is obtained from the "AND" operation on Fig. 2c and Fig. 5a. **c** is obtained from the "AND" operation on Fig. 2f and Fig. 5b. **b** and **d** are the thinned images of **a** and **c** respectively

3 Model generation phase

Model-based vision is a robust approach for locating and recognizing the object motion in the real scene with a lot of spatial-temporal varieties. Here, we use PCA as the spatial description model and HMM as the temporal description model for the gestures.

3.1 Spatial description models

The PCA is the kernel concept of the so-called active shape models (ASMs) [15]. This method models the natural variability within a class of shapes. Each instance of an object's shapes can be represented by an ordered set of characteristic points located on the boundary. We manually locate the feature points on the training set images (i.e., Figs. 8a and 8b) by ensuring that each point plays an essential role on the boundary of the images. These points characterizing the shape feature are called "landmark points". The PCA-based method analyzes the statistics of the coordinates of these points over the training set. These landmark points on different images have minimal difference, so that we can align them with different scale, rotation, and translation before training (see Figs. 8c and 8d). By minimizing a weighted sum of squares of distances between corresponding points on different shapes, we align every shape to the first shape, calculate the mean shape, and then align every shape to the mean shape. The details of the alignment processing of the training set can be found in [15, 16]. Having generated the

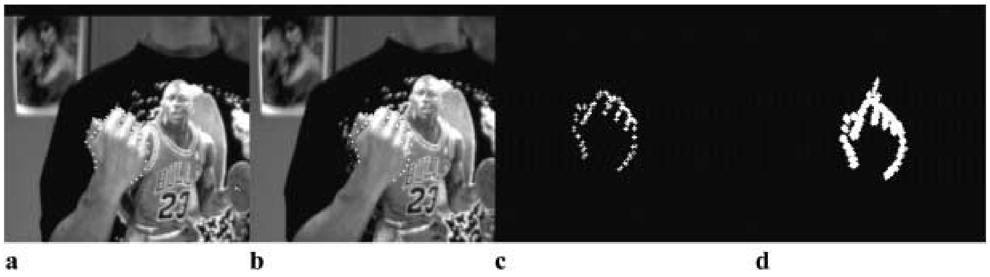


Fig. 8a–d. The positions of the labeled points are shown around the boundary of the hand. **a** and **b** illustrate the hand shapes with labeled points. **c** shows the result that **b** is aligned with **a**. **d** shows the aligned shape of a training set

N aligned shapes and the mean shape $\bar{\mathbf{x}}$, we may calculate the deviation of the aligned shapes from the mean shape, $d\mathbf{x}_i$ as

$$d\mathbf{x}_i = \mathbf{x}_i - \bar{\mathbf{x}}\,. \tag{8}$$

Then, we can obtain the $2n \times 2n$ covariance matrix, **S**, as

$$\mathbf{S} = \frac{1}{N}\sum_{i=1}^{N} d\mathbf{x}_i d\boldsymbol{x}_i^T\,. \tag{9}$$

Applying the PCA, we can project the original $2n$ dimension shape points vector to another axis to reduce the dimension. We first calculate the eigenvectors of the covariance matrix **S** (i.e., $p_1, \ldots, p_{2n}$) such that

$$\boldsymbol{Sp}_k = \lambda_k \mathbf{p}_k \quad \text{with} \quad \mathbf{p}_k^T \mathbf{p}_k = 1\,, \tag{10}$$

where λ_k is the kth eigenvalue of **S**, with $\lambda_k \geq \lambda_{k+1}$. According to the PCA, it is sufficient to use the first t eigenvectors to describe the shape variation. Another advantage of this method is that the models represent the global variation rather than the local variation of the shape.

Now we determine how many terms are enough for us to describe the shape variation. If we define λ_T as

$$\lambda_T = \sum_{k=1}^{2n} \lambda_k \quad \text{and} \quad \lambda_t = \sum_{k=1}^{t} \lambda_k\,, \tag{11}$$

then, based on the experimental results, $\lambda_t/\lambda_T = 0.8$ is sufficient. We use 90 landmark points ($n = 90$) and 10 eigenvectors ($t = 10$) which suffice the constraint. Given an arbitrary shape, we can use $\mathbf{x} = \bar{\mathbf{x}} + \mathbf{P}{\cdot}\mathbf{b}$ to approximate it, where $\mathbf{P} = (\mathbf{p}_1, \ldots \mathbf{p}_t)$ is the matrix of the first t eigenvectors, and $\mathbf{b} = (b_1, \ldots b_t)^T$ is a vector of weights which are determined by the eigenvalues $(\lambda_1, \ldots, \lambda_t)$. The shape variations can be described by the first four principal components ($t = 4$) illustrated in Fig. 9.

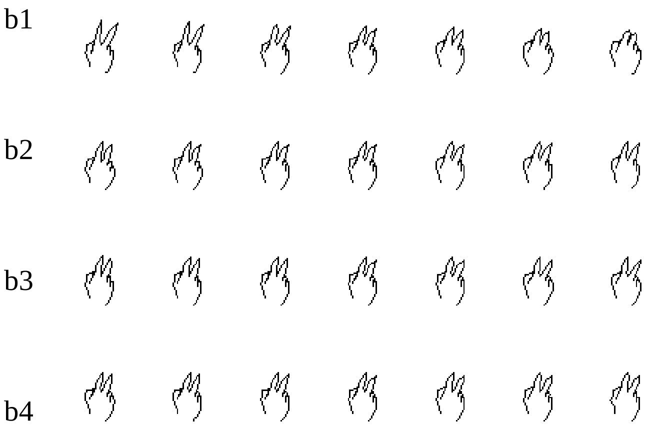


Fig. 9. Illustration of the effects of varying the parameters, b1, b2, b3, and b4 of hand model

3.2 Temporal description models

HMMs have been used to model speech signal pattern for speech recognition [19] and have been applied in the visual communication dealing with the problems in which the time variation is significant. The earliest application by Yameto et al. [20] uses HMM to recognize tennis swings. Recently, HMMs have been applied to recognize gestures [13, 14, 21–23]. However, the previous HMM-based methods either avoid the problem of identifying the hand shape against complex background or require stereo camera to obtain 3D data for input. Most of the HMMs are trained by using a complicated Baum-Welch training algorithm [19].

An HMM is a doubly stochastic process with an unobservable hidden stochastic process, but it can only be observed through another set of stochastic processes that produce the observed sequence. The hidden states “drive” the model dynamic. At each time instance, the model is in one of its hidden states. Transitions among the hidden states are generated by probabilistic rules. The observable states produce outcome during hidden state transition or while the model is in one of its hidden states. Such outcome is measurable by an outside observer and is also governed by a set of probabilistic rules. Here, we limit our HMM as the forward-chaining HMM which is a special case of HMMs that have zero probability of returning to an earlier state.

If there are N states and M possible observations, the HMM is described by $(Q, \pi, \boldsymbol{A}, \boldsymbol{B}, V)$, where $Q(= \{q_1, q_2, \ldots, q_N\})$ denote a set of states, $V(\{v_1, v_2, \ldots, v_M\})$ is the set of output observations (symbols).

$\boldsymbol{A}(= \{a_{ij}\}$, with $a_{ij} = Pr(q_j$ at $t+1 \,|\, q_i$ at $t))$ denotes the *state transition probability distribution* (STPD).

$\boldsymbol{B}(= \{b_j(k)\}$, with $b_j(k) = Pr(v_k$ at $t \,|\, q_j$ at $t))$ is the *observation probability distribution* (OPD).

$\pi(= \{\pi_\iota\}$, with $\pi_l = Pr(q_l$ at $t = 1))$ is the *initial state probability distribution* (SPD).

At some time step t, a new state is reached or stays in the same state, based on the STPD. The system outputs a symbol at each time step and the symbol is stochastically chosen from among a discrete set.

In speech recognition, one HMM is associated with each different phoneme or word. In our gesture recognition, one HMM is trained for 18 different gestures and

Table 1. Initial state probability distribution (SPD)

State	1	2	3	4	5	6	7	8	9	10	11	12
Pr	1/16	1/16	1/16	1/16	1/16	1/16	1/16	1/16	1/16	0	0	1/16
State	13	14	15	16	17	18	19	20	21	22	23	24
Pr	1/16	1/16		1/16	1/16	1/16	1/16	0	0	0	0	0

the observation is the similarity measurement between the pre-defined PCA models and the feature image. The training process for the HMM with different gestures is required so that the parameters of HMM can truly describe the spatio-temporal dynamics of the desired gesture action. The training is to optimize the maximum likelihood measure, $\log(Pr(observation \mid model))$, over a set of training examples for the particular gesture associated with the model. Such optimization involves the use of computationally costly expectation-maximization procedure – Baum-Welch algorithm [19]. To simplify our implementation of HMM, we assume that the HMM is a first-order stochastic process which is stationary. Therefore, instead of using the Baum-Welch algorithm, we may implement the following training process for our HMM.

3.3 Training procedures for HMM

Here, we are developing a system to recognize 18 different gestures. The hand shape of these 18 gestures can be classified into 24 different groups, so we define 24 states (N = 24) to represent the possible instant appearance of the designated gestures. Some of them are intermediary states that can never appear in the beginning frame of the gestures. The rest of them are assumed to have the same initial probability of appearance. Here, we assume that each unknown input gesture is either a *simple gesture* or is a combination of two or more simple gestures called a *hybrid continuous gesture*. For a hybrid gesture, we may find that there exists an*interconnecting gesture* between every two simple gestures, which does not indicate any semantic meaning. These interconnecting gestures will be observed as one of the so-called *intermediary states*. However, we may also find that two simple gestures may be connected smoothly to make a hybrid gesture without any interconnecting gesture. Table 1 shows the SPD. The states with probability zero means that these states belong to the intermediary states of gestures and they will never appear in the beginning of gestures. Here, we assign the zero probability for the unreachable states and an equal probability for the others.

In the training phase, given all possible gestures, we investigate the STPD of all the possible transitions for each state. Each state may have 5, 4, 3, or 2 transition choices. Since we have a sequence of all possible shape varieties that belong to the same state, we may divide the duration of state into *steps*. In each step, we have the smallest pre-defined identifiable and allowable shape varieties of the corresponding gesture. Here, we categorize these 24 states into three classes based on (1) different duration (or number of steps), (2) intermediate state or meaningful state, (3) large or small shape varieties. The first class consists of state-2, 12, 13, 14, 16, 17, and state-18 and their average duration is 40 steps. The second class covers state-1, 3, 4, 5, 6, 7, 8, 9, 10, 11,

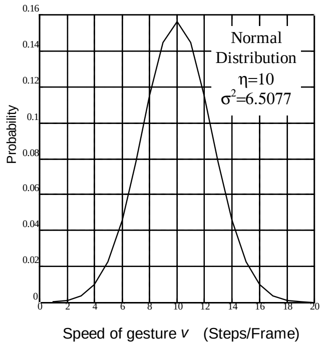


Fig. 10. Normal distribution with mean = 10 steps/frame and variance = 6.5077

15, 20, 21, 22, 23, and state-24, and their average duration is 28 steps. The last class is state-19 with average duration 20 steps.

Furthermore, we subdivide each state transition into several step transitions. Without any *a priori* information, we assume that the transition probability of each step is equal. We model the speed of gesture (or the number of steps in each state) as a normal distribution shown in Fig. 10. The mean value is 10 steps/state and the variance is 6.5077. The distribution depends on the experiments of the gestures captured. Next, we calculate the interstate transition probability. The *s* is the step number of the state, and *v* is the speed of gesture (*v* steps/frame). The interstate transition probability *p* is

$$p = \sum_i Pr\left(v \geq i \mid s = i\right) Pr\left(s = i\right), \quad 1 \leq i \leq \text{ duration of the state}, \tag{12}$$

where i is the current step, and we sum up all possible steps. The results are stored in the STPD (see Table 2).

In the training phase, to obtain the OPD, the system extracts all possible features in the input frame and then identifies the moving object in this frame. The outputs of measurement are the possible observations that are weighted by the degree of similarity to the corresponding PCA models of the designated state. For each input frame, which may be assigned to state *l*, we select the best five observations by comparing the corresponding edge feature image with a set of PCA models. The similarity weighting between the edge feature image (A_l) and the PCA models (B_j) is defined as

$$W_{l,j} = \sum_{i=1}^{5} H^*\left(A_l, B_i\right) \Big/ H^*\left(A_l, B_j\right), \quad j = 1, 2, \cdots, 5, \tag{13}$$

where j indicates the specific one of the five selected observations for each input frame, the numerator $\Sigma_i H^*(A_l, B_i)$ is the normalization factor, and H^* is the modified Hausdorff distance measure, which will be defined in Sect. 4.1. A smaller distance $H^*(A_l, B_j)$ will generate a larger weight $W_{l,j}$.

For each observed gesture (an image sequence), we find that in any time instance it may be assigned to several possible states with different state probabilities $p(q_j)$. With the state initial probability π_i and the state transition probability

Table 2. State Transition Probability Distribution (STPD). Note that ε → 0.

S\TS	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	0.6428	0.0893	ε	ε	ε	ε	ε	ε	0.0893	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.0893	0.0893	ε	ε
2	0.2500	0.7500	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
3	ε	ε	0.6428	0.0893	ε	ε	ε	ε	ε	ε	0.0893	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.0893	ε	0.0893
4	ε	ε	0.1191	0.6428	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.1191	ε	ε	ε	ε	ε	ε	ε	0.1191	ε
5	ε	ε	ε	ε	0.6428	ε	ε	ε	ε	0.0893	0.0893	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.0893	ε	0.0893	ε
6	ε	ε	ε	ε	ε	0.6428	ε	ε	ε	ε	ε	ε	ε	0.1786	ε	ε	ε	ε	ε	0.1786	ε	ε	ε	ε
7	ε	ε	ε	ε	ε	ε	0.6428	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.3572	ε	ε	ε	ε
8	ε	ε	ε	ε	ε	ε	ε	0.6428	0.1786	ε	ε	ε	ε	ε	0.1786	ε	ε	ε	ε	ε	ε	ε	ε	ε
9	0.0893	ε	ε	ε	ε	ε	ε	0.0893	0.6428	0.0893	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.0893
10	ε	ε	ε	ε	0.1786	ε	ε	ε	0.1786	0.6428	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
11	ε	ε	0.1786	ε	0.1786	ε	ε	ε	ε	ε	0.6428	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
12	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.7500	0.2500	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
13	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.2500	0.7500	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
14	ε	ε	ε	ε	ε	0.2500	ε	ε	ε	ε	ε	ε	ε	0.7500	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
15	ε	ε	ε	0.1786	ε	ε	ε	0.1786	ε	ε	ε	ε	ε	ε	0.6428	ε	ε	ε	ε	ε	ε	ε	ε	ε
16	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.7500	0.2500	ε	ε	ε	ε	ε	ε	ε
17	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.2500	0.7500	ε	ε	ε	ε	ε	ε	ε
18	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.7500	0.2500	ε	ε	ε	ε	ε
19	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.5000	0.5000	ε	ε	ε	ε	ε
20	ε	ε	ε	ε	ε	0.1786	0.1786	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.6428	ε	ε	ε	ε
21	0.1786	ε	ε	ε	0.1786	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.6428	ε	ε	ε
22	0.1786	ε	0.1786	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.6428	ε	ε
23	ε	ε	ε	0.1786	0.1786	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.6428	ε
24	ε	ε	0.1786	ε	ε	ε	ε	ε	0.1786	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.6428

a_{ij}, we can calculate the $p(q_j)$. For instance, we find that, when the gesture-1 is an input gesture, at a certain time instance, we have the probabilities of state-1, state-2, state-21 and state-22 as 0.6949, 0.0142, 0.0181, and 0.2729, respectively. Similarly, for each frame in the image sequence, we may have different observations with different observation probabilities (i.e., $p(\nu_k)$). It is computed based on the modified Hausdorff distance measurement between the input hand shape and the possible corresponding observations described by the PCA pre-stored models. In this training phase, given as many images as possible, we calculate the $p(q_j)$ and $p(\nu_k)$. For each state q_j, we may calculate $p(q_j|\nu_k)$ by using Eq. 13 to generate $W_{l,j}$ and accumulating $W_{l,j}$ for observation ν_k as

$$p\left(q_j\,|\nu_k\right) = \frac{\sum_{N_T} W_{j,k}}{\sum_{N_T}\sum_{l=1}^{N} W_{l,k}}\,, \tag{14}$$

where the N is the number of states and the N_T is the number of training image frames. To obtain $p(\nu_k|q_j)$ and then generate the OPD (as Table 3), we can use the following relationship

$$p(\nu_k|q_j) = p(q_j|\nu_k)p(\nu_k)/p(q_j)\,. \tag{15}$$

4 Recognition phase

To recognize a continuous gesture, we use the Hausdorff distance measurement to observe the input frame, and then refer to the OPD and STPD to generate the observation patterns and all the possible observation sequences. With these observation sequences, we may use the Viterbi algorithm to generate the best match state sequence that has the maximum probability to indicate the correct gesture.

4.1 Hausdorff distance measurement

Here, we observe the input frame by applying the Hausdorff distance measurement to measure the similarity between the pre-defined PCA models and the feature image. Let A be the feature image of the desired object and B be the instance of PCA model (by adjusting the shape parameters, we can generate different shapes). Sets A and B are two finite point sets, i.e., $A = \{a_1,\ldots a_p\}$ and $B = \{b_1,\ldots b_q\}$, the forward and the reverse Hausdorff distance are defined as

$$h(B,A) = \max_{b\in B}\min_{a\in A}\|b-a\| \quad \text{and}$$
$$h(A,B) = \max_{a\in A}\min_{b\in B}\|a-b\|\,, \tag{16}$$

where $\|\cdot\|$ is the Euclidean norm. The Hausdorff distance is defined as

$$H(A,B) = \max(h(A,B), h(B,A))\,. \tag{17}$$

In real cases, we only need to compare some portions of sets A and B. In complex background with occluded object contour, we are not sure whether the designated object is inside the feature image or not. The partial Hausdorff distance is defined as

$$H^{f_F f_R}(A,B) = \max(h^{f_F}(A,B),\ h^{f_R}(B,A))\,, \tag{18}$$

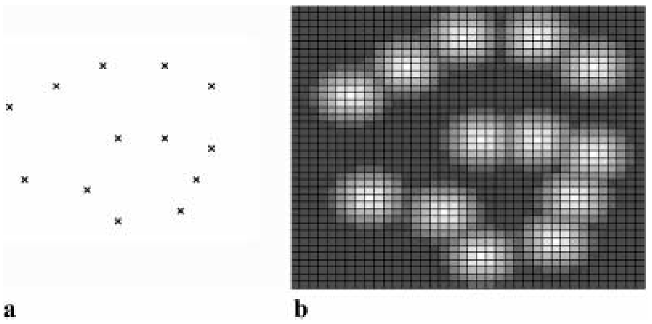


Fig. 11a. is the set of points and **b** is the corresponding Voronoi surface (or distance transform). The *brighter cells* indicate the shorter distance than the *darker cells*

where the partial forward Hausdorff distance, and the partial reverse Hausdorff distance are defined as

$$h^{f_R}(B,A) = f^{th}_{b\in B}\min_{a\in A}\|\,b-a\,\| \quad \text{and}$$
$$h^{f_F}(A,B) = f^{th}_{a\in A}\min_{b\in B}\|\,a-b\,\|\,. \tag{19}$$

The f_F and f_R are the forward fraction and the reverse fraction, $f^{th}_{x\in X}\,q(x)$ denotes the f-th quantized value of $q(x)$ over the set X [24, 25].

Dubuisson et al. [26] propose a modified Hausdorff distance which has better performance for object matching than the original Hausdorff distance. This is due to the fact that the Hausdorff distance value indicates the maximum distance between these two point sets. Generally, for less noisy images, the modified Hausdorff distance has better measuring performance. The modified Hausdorff distance is defined as

$$H^*(d(A,B), d(B,A)) = \frac{N_a d(A,B) + N_b d(B,A)}{N_a + N_b}\,, \tag{20}$$

with the forward and reverse Hausdorff distances defined as

$$d(B,A) = \frac{1}{N_b}\sum_{b\in B} d(b,A)\,,$$
$$d(A,B) = \frac{1}{N_a}\sum_{a\in A} d(a,B)\,, \tag{21}$$

where $d(b,A) = \min_{a\in A}\|b-a\|$ and N_a and N_b are the number of points of two point sets A and B, respectively.

Usually, the number of points in the feature image A is much larger than the number of points of the shape model B. From the equation of forward Hausdorff distance, we need to find a minimum distance from every point in set B to all the points in set A. The computation is so time consuming that we need to develop a fast method by using the Voronoi surface, which has also been referred to as a distance transform to reduce the redundant computation of distance. Let set A be the point set shown in Fig. 11a, (we use 13 points to show it clearly) and the Voronoi surface $d(p)$ of the set A is

$$d(p) = \min_{a\in A}\|\,p-a\,\|\,, \tag{22}$$

where p is any point in the same space. Fig. 11b illustrates the Voronoi surface of set A. Once the Voronoi surface having been generated, we can easily get the minimum distance from each point of set B to set A without computing the distances between all points of A.

Table 3. The Observation Probability Distribution (OPD). Note that ε → 0.

O\S	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	0.6949	0.0142	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.0180	0.2729	ε	ε
2	0.2814	0.6848	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.0338	ε	ε	ε
3	0.0048	0.0480	0.1504	0.0982	0.0742	0.0255	0.0037	0.0582	0.0899	0.0755	0.0727	ε	0.0152	ε	0.0325	ε	ε	ε	ε	0.0278	0.0480	0.0230	0.0799	0.0725
4	0.0556	0.0934	0.0707	0.1892	0.0080	0.0578	ε	0.1003	0.0352	0.0663	0.0484	ε	0.0636	0.0171	0.0580	ε	ε	ε	ε	ε	ε	0.0140	0.0405	0.0819
5	0.0338	0.0075	0.0456	0.0135	0.1543	0.0213	0.0747	0.0598	0.0324	0.0763	0.0738	0.0450	0.0064	ε	ε	0.0708	0.0748	0.0010	0.0130	0.0216	0.0694	0.0087	0.0426	0.0537
6	ε	ε	ε	ε	ε	0.5792	ε	0.0287	0.0494	ε	ε	ε	ε	0.3427	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
7	0.0073	0.0306	0.0660	0.0178	0.0806	0.0648	0.1600	▯	0.0764	ε	ε	0.0926	0.0352	0.0259	ε	0.0778	0.0746	0.0710	0.0705	0.0488	ε	ε	ε	ε
8	0.2292	ε	ε	0.0786	ε	ε	ε	0.5441	0.0085	ε	ε	ε	ε	ε	0.1395	ε	ε	ε	ε	ε	ε	ε	ε	ε
9	0.0144	0.1102	ε	0.1444	0.0026	0.0454	ε	0.1126	0.2397	0.0137	ε	0.0979	0.0797	0.0290	0.0742	ε	ε	ε	ε	0.0086	ε	ε	0.0220	0.0054
10	ε	0.0180	ε	ε	0.0136	ε	ε	ε	0.0066	0.2888	0.1422	ε	ε	ε	0.1388	ε	ε	ε	ε	ε	0.0336	0.0767	0.1445	0.1373
11	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.1598	0.3875	ε	ε	ε	0.0935	ε	ε	ε	ε	ε	0.1645	0.1057	0.0636	0.0254
12	ε	ε	0.1009	ε	ε	0.0774	ε	ε	0.0299	ε	ε	0.2833	0.1174	0.0221	ε	ε	ε	0.1298	0.1297	0.1095	ε	ε	ε	ε
13	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	1.0000	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
14	ε	ε	ε	ε	ε	0.0624	ε	ε	ε	ε	ε	ε	ε	0.9376	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε
15	0.0407	0.0317	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.7769	ε	ε	ε	ε	ε	ε	0.1506	ε	ε
16	ε	ε	0.0098	ε	0.1675	ε	0.0717	ε	ε	ε	ε	ε	0.0060	ε	ε	0.5780	0.1024	0.0326	0.0320	ε	ε	ε	ε	ε
17	ε	ε	ε	ε	0.0863	ε	0.2460	ε	ε	ε	ε	ε	ε	ε	ε	0.0889	0.5494	0.0171	ε	0.0122	ε	ε	ε	ε
18	ε	ε	0.0443	ε	0.0394	0.0109	0.1100	ε	ε	ε	ε	ε	0.0107	0.0589	ε	0.1089	0.1121	0.2620	0.1259	0.1170	ε	ε	ε	ε
19	ε	ε	ε	ε	ε	0.0088	0.1183	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.0961	0.2016	0.4193	0.1560	ε	ε	ε	ε
20	ε	0.0132	0.0550	0.0374	0.0019	0.1190	ε	ε	0.0457	ε	ε	0.1051	0.1091	0.1185	ε	0.0528	ε	0.0592	0.0496	0.2334	ε	ε	ε	ε
21	0.1493	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.0329	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.5672	0.2505	ε	ε
22	0.1042	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.8958	ε	ε
23	0.0130	0.0456	ε	ε	0.0640	ε	ε	ε	ε	ε	0.0402	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.2561	0.0652	0.5158	ε
24	ε	0.0138	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	ε	0.0595	0.1069	ε	ε	ε	ε	ε	ε	0.1997	0.0712	0.5489

4.2 The observations of the input frame

By investigating the results of the shape similarity measurement, we cannot expect that the observation is always accurate, in other words, the measurement between the feature image and the pre-stored PCA models does not imply that the shortest distance always indicates the accurate model. In recognition phase, for each input frame, we may determine the possible observations by comparing the corresponding feature image with all the possible pre-stored models. In a complex and non-stationary scene, the feature image may be noisy, and the comparison is based on the modified Hausdorff distance measurement.

Given an image sequence, we compare all the frames of this sequence with the PCA models and determine the possible observations for each frame. For the first frame, we only select the PCA models which correspond to the states with non-zero initial state probability. We compare each pre-stored PCA model with the feature image by using the modified Hausdorff distance measurement. Here, we choose the first five best observations. From these observations, we refer to the OPD and find all the possible states to which this input frame belongs. Then, we use the current state to generate the next probable states for the next frame observation by referring to the STPD. By investigating all the observations in the training set, we are certain that selecting five best match observations is enough, since the right observation should be one of these five selections. For the other image frames in the input gesture sequence, we do the same observation operations.

4.3 Observation patterns generation for simplifying the observation set

Now, we consider that if we have an average of 30 frames per input image sequence, then we will have 5^{30} (i.e., about 10^{21}) possible observation sequences, of which each contains 30 observations. Therefore, it is computationally unfeasible for the Viterbi algorithm to determine the best matched state path (sequence). To solve this problem, we need another training procedure that simplifies the possible observation sequences for the Viterbi algorithm to operate effectively. Here, we take advantage of the relationships among these observations. For each input frame, there is an observation set that indicates the possible states to which the designated input frame belongs. By investigating all the observation sets of the input frames, we find that some observations have much higher possibility of appearance. The modified Hausdorff distances of these selected observations have been used in the training procedure to generate the observation patterns for each observation in the observation set.

In this training stage, given an image identified as observation j, we use the similarity measure to find the best five observations as $S_j = \{o_{jk} | k = 1, \ldots 5\}$ in which the best matched observation is not necessarily observation j. For each observation in S_j, if it is observation i, i.e., $o_{jk} = O_i$, then the corresponding ith accumulator for O_i is increased by one, i.e., $ACC_i = ACC_i+1$. We test as many input frames as possible for each specific observation j. Having collected and accumulated the selected observations for every observation, we calculate the so-called relative observation appearing probabilities (ROAP) by normalization (by dividing the accumulators by the number of input image frames), and then eliminate the observations with ROAP < 0.15. The results are stored in the so-called *observation patterns table* (OPT) illustrated in Table 4. Figure 12 illustrates the ROAP of each observation in the observation pattern. With the OPT, we may reduce the number of all the possible observation sequences. Finally, the gesture recognition process may refer to the OPT and determine which observations should be selected.

Table 4. OPT: The observation patterns with strength > 0.15

Observation #	Observation Pattern									
1	1	2	4	5	8	9	21	22	X	X
2	2	3	4	7	8	9	10	20	23	X
3	3	4	5	7	10	11	12	18	24	X
4	3	4	6	7	8	9	20	24	X	X
5	3	5	7	11	16	17	21	23	X	X
6	3	4	6	7	9	12	20	X	X	X
7	3	7	12	17	18	19	X	X	X	X
8	3	4	5	8	9	10	15	24	X	X
9	3	5	6	7	9	12	20	X	X	X
10	3	5	8	10	11	24	X	X	X	X
11	3	5	8	10	11	21	23	X	X	X
12	5	7	9	12	13	20	X	X	X	X
13	3	4	9	12	13	18	20	X	X	X
14	6	7	13	14	18	20	24	X	X	X
15	3	4	8	9	10	11	15	23	24	X
16	5	7	16	17	18	20	X	X	X	X
17	5	7	16	17	18	19	X	X	X	X
18	7	12	16	18	19	20	X	X	X	X
19	7	12	16	17	18	19	X	X	X	X
20	5	7	9	12	18	19	20	X	X	X
21	3	4	5	10	11	21	22	23	X	X
22	1	3	10	11	15	21	22	23	24	X
23	3	4	5	10	11	21	23	24	X	X
24	3	4	5	10	11	15	24	X	X	X

Let the selected observation set for frame i be $S_i = \{o_{ij}\}$; we want to find a corresponding reduced observation set S_i'. For each observation o_{ij}, we have referred an observation pattern (referring to Table 4) and form a subset $S_i'(\subseteq S_i)$ in which all $o_{ij}'s$ have at least four common observations in their observation patterns. We illustrate an example in the following. Given an observation set of an input frame as $\{3, 4, 6, 7, 9\}$, we want to reduce the observation set. If the matching condition is severe (i.e., the observation pattern should contain all the same five observations), then the observation set will be reduced to $\{4, 6\}$. If we relax the matching condition, i.e., the observation pattern will contain the same four observations, then the observation set will be reduced to $\{4, 6, 9\}$.

4.4 Gesture recognition using the Viterbi algorithm

Given a sequence of image representing a certain meaning gesture, we apply (1) the feature extraction process to extract the meaningful hand shapes, (2) the Hausdorff distance measure to generate the possible observations, and (3) the

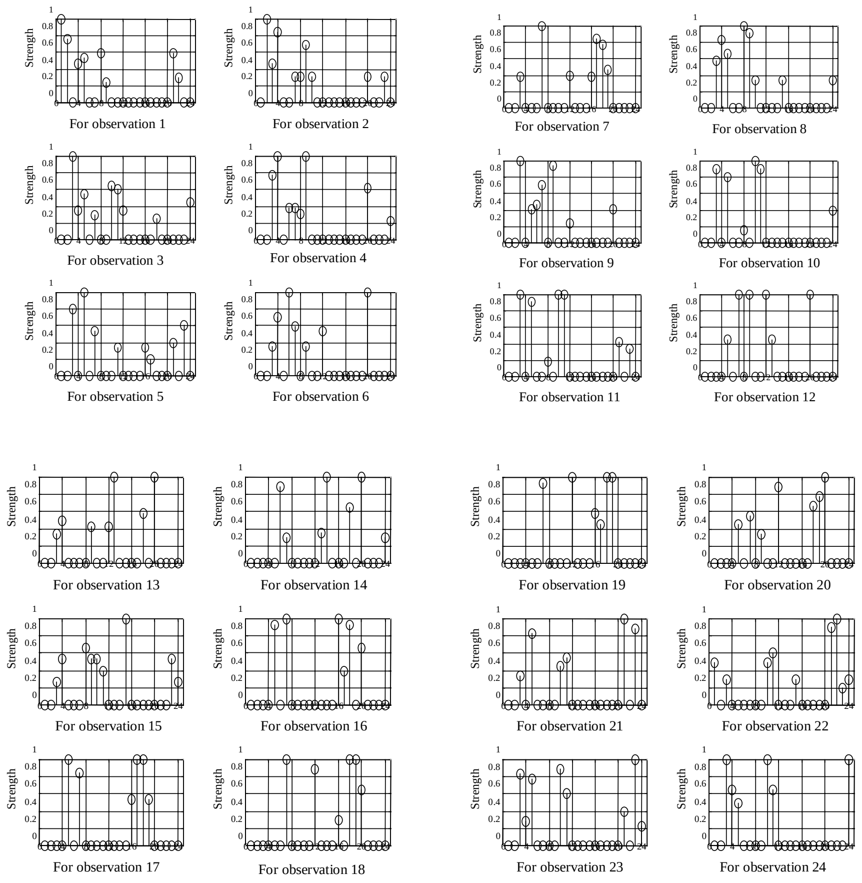


Fig. 12. The ROAP of the each observation will be used to generate the observation pattern table (OPT)

Viterbi algorithm [19] to find the corresponding most probable state transition sequence. The hand gesture recognition system can be illustrated in the following steps.

1. For each input frame, we change the parameters of pre-stored PCA models to find the most similar shape that matches with the feature image and then use the Hausdorff distance measure to make observation by measuring the distance between them.
2. Use different pre-stored PCA models to observe the ith input feature image frame and generate an observation set $S_i = \{O_{ij}\}$.
3. Use the pre-trained OPT (Table 4) to simplify the observation set.
4. Use the Viterbi algorithm to calculate the log-probability of the state transition sequences, and pick the maximum.
5. If the maximum is still below a certain threshold, then reject, else accept.

The Viterbi algorithm can be viewed as a special form of a forward and backward algorithm, where only the maximum path at each step is taken instead of all paths. This optimization reduces the computational load of finding the most likely state sequence. The Viterbi algorithm consists of the following steps.

1. *Initialization.* For all states i, $\alpha_1(i) = \pi_i b_i(O_1)$; $\psi_i(i) = 0$, $i = 1, 2, \ldots N$, where $\alpha_1(i)$ is the probability of which observation O_1 occurs at time $t = 1$ and at state i, and ψ stores the optimal states.
2. *Recursion.* From $t = 2$ to T for all state j, $\alpha_t(j) = \text{Max}_{1 \leqq i \leqq N}\ [\alpha_{t-1}(i)a_{ij}]b_j(O_t)$; $\psi_t(j) = \text{argmax}_{1 \leqq i \leqq N}\ [\alpha_{t-1}(i)a_{ij}]$.
3. *Termination.* $Pr = \text{Max}_{1 \leqq i \leqq N}\ [\alpha_T(i)]$; $i^* = \text{argmax}_{1 \leqq i \leqq N}\ [\alpha_T(i)]$.
4. *Backtracking.* From $t = T - 1$ to $1, i_t^* = \psi_{t+1}(i_{t+1}^*)$.

In Fig. 13a, we show six frames of an image sequence, and in Fig. 13b we illustrate all the possible observations of one of every two frames of the image sequence of the gesture. Then we refer to the OPT (Table 4) to generate the reduced observation set of each frame shown in Fig. 13c. In Fig. 13d, we illustrate the table of the 14 input observation sequences, which have the best corresponding accumulated probabilities and the best matched state sequences. We can see that these fourteen best match state sequences correspond to the same gesture.

5 Experimental results and discussion

Here, we illustrate the experimental results of hand gesture recognition system. The input hand gestures include simple gestures and hybrid gestures (combined of two or more simple gestures). We have tested 10 simple gestures and 8 hybrid gestures (see Fig. 14). In our system, these 10 simple gestures may contain one, two or three states, 5 hybrid gestures may consist of two or more smoothly connected simple gestures without any intermediary gesture, and the other 3 hybrid gestures require one intermediary gesture to connect

a

4	4	4	4	4	4	5	10	3	3	3	3	3	1	1	1	1	1	1	1	1
5	5	5	5	7	7	9		4	4	4	6	6	9	22						22
16	16	16	16	16	16	10		6	6	6	9	9								
								7	7	7										
								9	9	9										

b

5	5	5	5	5	5	5	10	4	4	4	3	6	1	1	1	1	1	1	1	1
16	16	16	16	7	7	10		6	6	6	9	9	9	22						22
								9	9	9										

c

Observation Sequence	Probability	State Sequence
16-16-16-16-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	3.269323e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
16-5-16-16-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	3.011681e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
16-16-16-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	3.011681e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
5-16-16-16-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	3.011681e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
16-16-5-16-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	3.011681e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
5-16-5-16-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	2.774344e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
5-5-16-16-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	2.774344e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
5-16-16-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	2.774344e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
16-5-5-16-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	2.774344e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
16-5-16-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	2.774344e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
16-16-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	2.774344e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
5-5-5-16-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	2.555708e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
5-5-16-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	2.555708e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
5-16-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	2.555708e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
16-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	2.555708e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1	2.354303e-018	5-5-5-5-5-5-10-10-9-9-9-9-9-1-1-1-1-1-1-1-1
16-16-16-16-5-5-5-10-9-9-9-9-9-1-1-1-1-1-1-1-1	1.746734e-018	5-5-5-5-5-5-5-10-9-9-9-9-9-1-1-1-1-1-1-1-1

d

Fig. 13a–d. The best state sequences are determined by applying the Viterbi algorithm. **a** The input image sequence. **b** Possible observations for each frame. **c** Reduced Observations for each frame. **d** The best state sequences

every two simple gestures. For the hybrid gestures, we need to build a corresponding active shape model to extract the new gesture.

In our experiments, the image frame size is 256×256, its frame rate is 30 frames per second. The camera that we use in the experiment is a SONY XC7500. The captured image frames are stored in the DRAM of an Oculus-F/64 frame grabber, and then transferred to the host computer (Pentium PC 300 MHz) for further processing. For each gesture, the frame number of image sequence is above 30. The input image sequence is taken at three different time intervals: no-gesture period, the action (gesture-making) period, and the silent period. Here, we assume that the complex background is stationary and we can use the frame difference detector (FDD) to identify the beginning and the end of the gesturing period in the image sequence.

The feature extraction algorithm is applied in both of the training and recognizing phases. Our method has been tested in indoor scenes with different backgrounds by a variety of gesture sequences made by different users. The results are good enough to provide both reliable tracking and accurate hand shape description. It is easily trained for different users and lighting conditions. The only limitation is that our method is based on the frame difference operation, if the hands are stationary or moving too slowly, then the feature image cannot be extracted accurately.

The shape variability in duration of gesture is accounted for in the recognition or model evaluation process. In recognition process, a sequence of gesture images is tested over the trained HMM in order to decide which gesture it represents. The probabilities of state sequences are evaluated using Viterbi algorithm. Different from [21], who consider the velocity of the hands as training and test feature, we use the hand shape variability to associate with the state of the HMM. The dynamic time-warping ability of the Viterbi algorithm will still hold in our system. However, if the gesture is made too fast, the observation accuracy will be lower, so that the Viterbi algorithm will pass through the fringes of the density function and receive a lower log-probability score.

In the training stage, we may have pre-determined different state sequences for different gestures (see Table 5). During the recognition stage, the outputs of the Viterbi algorithm are the best probable state sequences, which are compared with the pre-determined state sequence of each gesture for identification. There are ten training sequence samples for each gesture. We find that most of the input gestures can be identified accurately. The best match state sequence corresponding to the most probable state sequence that identifies the gesture made in the image sequence. However, in very few cases, we may find that the most probable state sequence does not indicate the same gesture as the other state sequences. Therefore, we can discard it and examine the next best one. The feature extraction process of our system is insensitive to the illumination changes and the complexity of the background. The most time-consuming stage of the recognition system is the observation process,

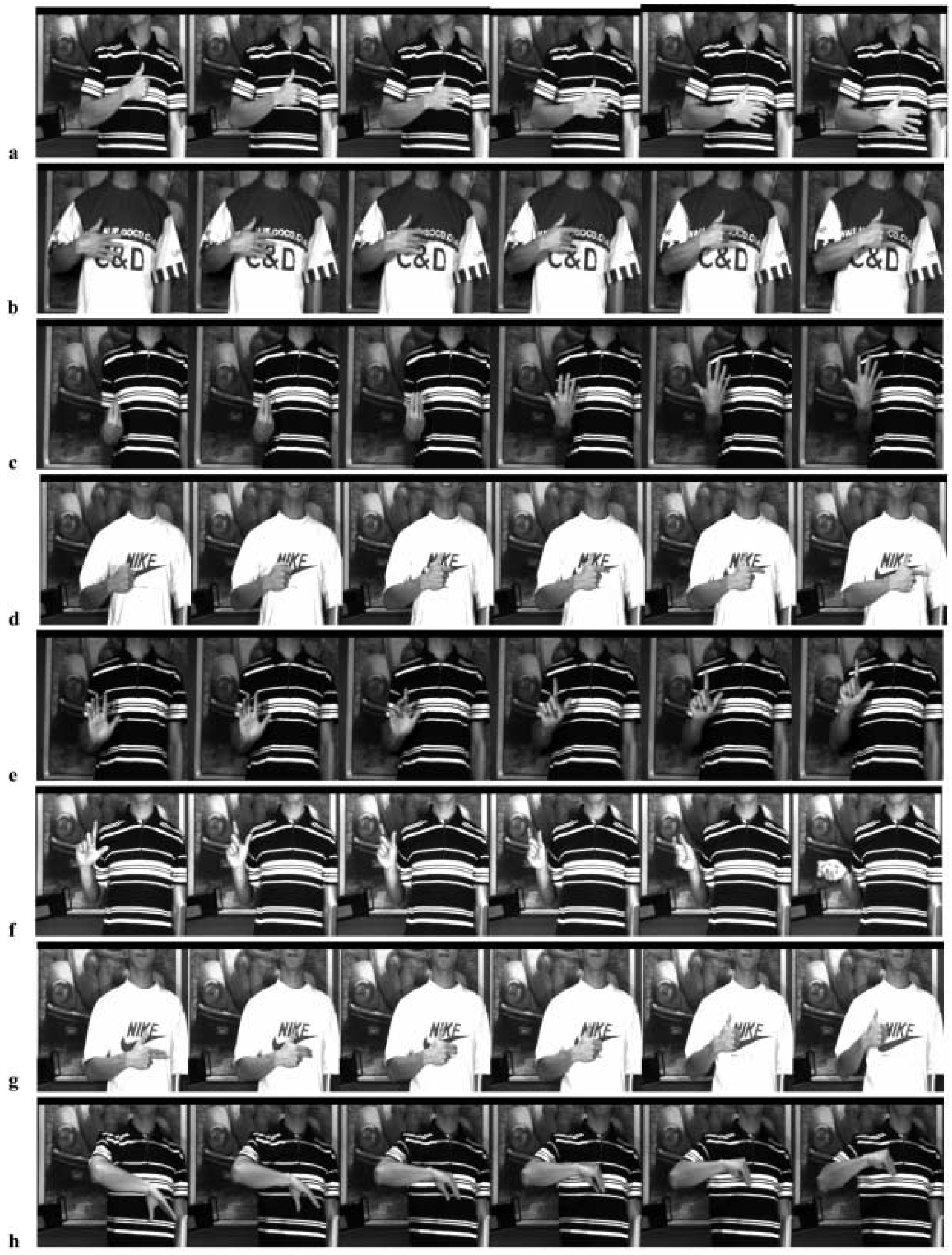


Fig. 14. Continued on next page

which selects the best match shape for each PCA model and applies a one-by-one Haudorff distance measure between the pre-stored shape models and the feature image. It takes about 60 s to generate the feature images, 70 s to observe each image frame, and 25 s to identify the gesture by Viterbi algorithm.

In the experiments, we have asked 20 people to make 18 different gestures. Each individual makes every gesture five times. The motion speed and the motion trajectory of the same gesture made by different persons are somehow quite different. Sometimes, the individual's clothes create a complicated background of the image sequence during gesture making. It may make the system generate wrong observations. On the average, the correct identification rate of our hand gesture recognition system for simple gesture is above 92% and the correct recognition rate of the hybrid gesture is about 87%. The breakdown recognition results for the 18 gestures are given in Table 6. From Table 6, we find that some input gesture image sequences are unidentified instead of misidentified, because if the most probable output state sequences do not match with any pre-determined state sequences (Table 5) completely, then it is unidentified. However, if we allow *partial matching* for gesture recognition, then the recognition rate will be increased (i.e., the recognition rate will be above 95%), however, it also increases the misidentification rate. For instance, it may misidentify gesture a as gesture p; gesture e as gesture m; gesture q as gesture r; or gesture l as gesture *o*, and vice versa. The misidentification has the following two reasons. (1) Feature image contains too little information of the moving object. It

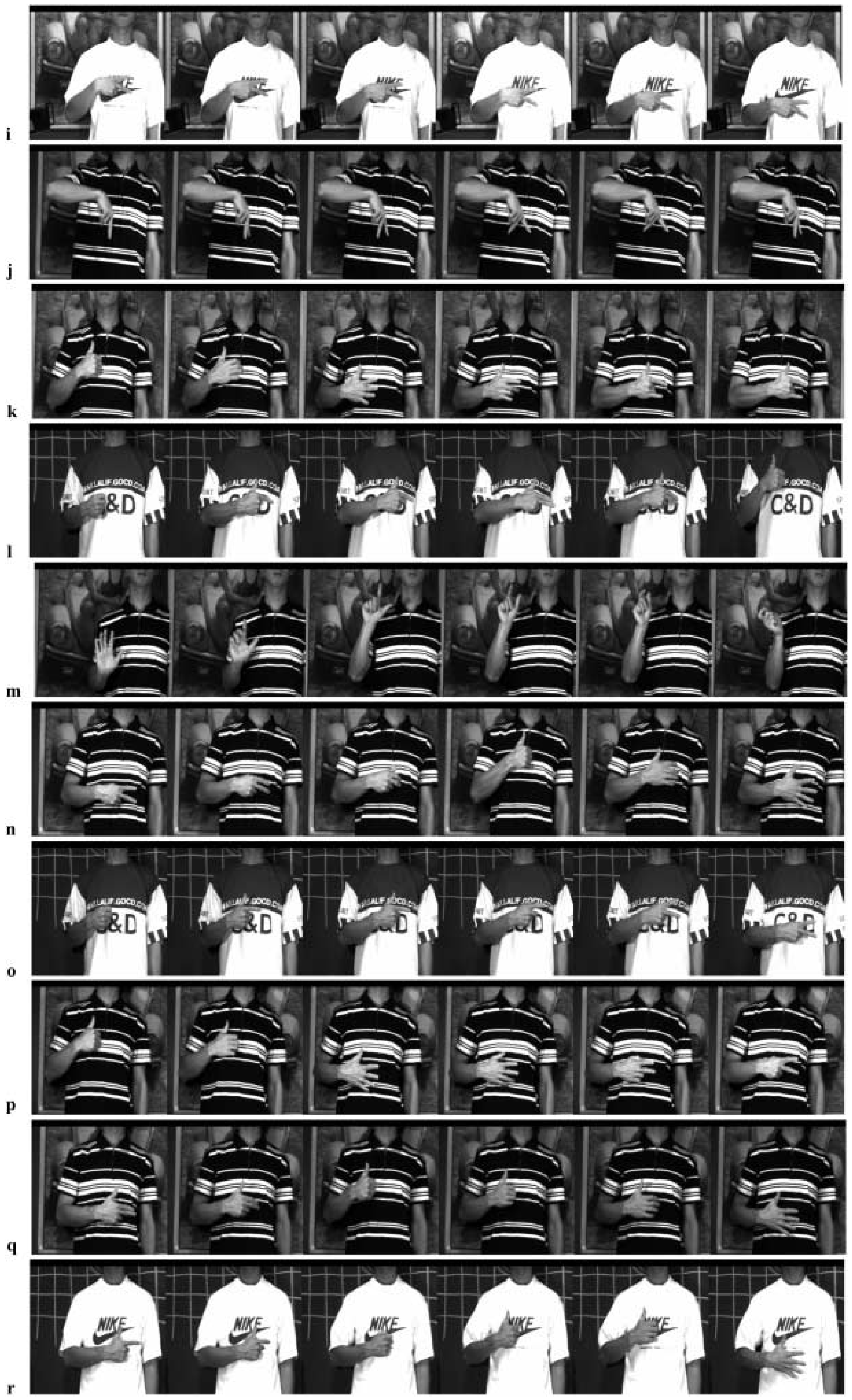


Fig. 14a. shows a gesture consisting of state-9 and state-1. **b** shows a gesture consisting of state-1 and state-2. **c** shows a gesture consisting of state-12 and state-13. **d** shows a gesture consisting of state-3 and state-4. **e** shows a gesture consisting of state-14 and state-6. **f** shows a gesture consisting of state-6, state-20, and state-7. **g** shows a gesture consisting of state-8 and state-9. **h** shows a gesture consisting of state-16 and state-17. **i** shows a gesture consisting of state-5 only. **j** shows a gesture consisting of state-19 and state-18. **k** shows a gesture consisting of state-9, state-1, and state-2. **l** shows a gesture consisting of state-3, state-4, state-15, state-8 and state-2. **m** shows a gesture consisting of state-14, state-6, state-20 and state-7. **n** shows a gesture consisting of state-5, state-10, state-9 and state-1. **o** shows a gesture consisting of state-3, state-4, state-23 and state-5. **p** shows a gesture consisting of state-9, state-1, state-21 and state-5. **q** shows a gesture consisting of state-8, state-9 and state-1. **r** shows a gesture consisting of state-4, state-3 and state-24, state-9 and state-1

Table 5. The state sequence of the 18 gestures in the experiment

Gesture	Corresponding state transition sequence
a	9. . . 1. . .
b	1. . . 2. . .
c	12. . . 13. . .
d	3. . . 4. . .
e	14. . . 6. . .
f	6. . . 20. . . 7. . .
g	8. . . 9. . .
h	16. . . 17. . .
i	5. . .
j	19. . . 18. . .
k	9. . . 1. . . 2. . .
l	3. . . 4. . . 15. . . 8. . . 9. . .
m	14. . . 6. . . 20. . . 7. . .
n	5. . . 10. . . 9. . . 1. . .
o	3. . . 4. . . 23. . . 5. . .
p	9. . . 1. . . 21. . . 5. . .
q	8. . . 9. . . 1. . .
r	4. . . 3. . . 24. . . 9. . . 1. . .

Table 6. The correctly identified sequence numbers (each gesture has 100 test image sequences)

Gesture	a	b	c	d	e	f	g	h	I	j	k	l	m	n	o	p	q	r
Identified number	90	93	89	92	98	94	96	83	94	88	86	79	88	90	93	87	92	81

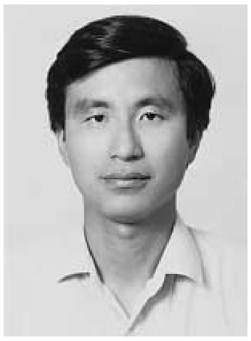
occurs because the desired object is stationary or moving too slow, and the system cannot extract the feature image accurately. (2) The shape variety between two continuous frames is enormous. It happens when the motion of the gesture is too fast.

6 Conclusion and further work

The gesture recognition system consists of four major components: feature extraction, PCA, Hausdorff distance, and HMM. Our method has proved to be very effective to extract the moving handshape in the complex stationary background. The PCA model is also a very reliable method to describe the moving hand shape. The proposed simplified training process for HMM has the drawback that, once we want to add a new gesture into the system, we need to retrain HMM. Therefore, the flexibility of our simplified HMM is the major issue in our future research. We need to eliminate the concern that the more gestures there are to be recognized, the more complicated and time-consuming processes may be expected.

References

1. Pavlovic VI, Sharma R, Huang TS (1997) Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review. IEEE Trans Pattern Anal Mach Intell 19(7): 677–695
2. Takahashi T, Kishino F (1992) A Hand Gesture Recognition Method and Its Application. Systems and Computers in Japan 23(3): 38–48
3. Kjeldsen R, Kender J (1997) Interaction with On-Screen Objects using Visual Gesture Recognition. Proc. of ICPR'97, 17–19 June
4. Wilson A, Bobick A, Cassell J (1997) Temporal Classification of Natural Gesture and Application to Video Coding. Proc. of ICPR'97, 17–19 June
5. Davis J, Shah M (1994) Visual Gesture Recognition. IEE Proc Vision Image Signal Process 141(2): 101–106
6. C. Charayaphan, Marble AE (1992) Image Processing System for Interpreting Motion in American Sign Language. J. Biomed. Eng. 15: 419–425
7. Darrell T, Essa I, Pentland A (1996) Task-Specific Gesture Analysis in Real-Time using Interpolated View. IEEE Trans Pattern Anal Mach Intell 18(12): 1236–1242
8. Cui Y, Weng J (1995) Learning-Based Hand Sign Recognition Using SHOSLIF-M. Proc. Fifth International Conference on Computer Vision, pp. 631–636
9. Hunter E, Schlenzig J, Jain R (1995) Posture Estimation in Reduced-Model Gesture Input System. Proc. of Int. Workshop on Automatic Face- and Gesture-Recognition, Zurich, Switzerland
10. Triesch J, von der Malsburg C (1996) Robost Classification of Hand Posture againt Complex Background. Proc. of Int. Workshop on Automatic Face- and Gesture-Recognition, Vermont, Oct. 14–16, pp. 170–175
11. Heap T, Hogg D (1996) Toward 3D Hand Tracking using a Deformable Model. Proc. of Int. Workshop on Automatic Face- and Gesture-Recognition, Vermont, Oct. 14–16, pp. 140–145
12. Lee L, Kunii T (1995) Model-Based Analysis of Hand Posture" IEEE Comput Graphics Appl June: 77–86
13. Starner T, Pentland A (1995) Visual Recognition of American Sign Language using Hidden Markov Models. Proc. of Int. Workshop on Automatic Face- and Gesture-Recognition, Zurich, Switzerland
14. Bobick AF, Wilson AD (1995) A State-based Technique for the Summarization and Recognition of Gesture. Proc. Fifth International Conference on Computer Vision, pp. 382–388
15. Cootes TF, Taylor CJ, Cooper DH, Graham J (1995) Active Shape Models – Their Training and Application. Comput Vision Image Understanding 61(1): 38–59
16. Cootes TF, Hill A, Taylor CJ, Haslam J (1994) Use of active shape models for locating structures in medical images. Image Vision Comput 12(6): 355–365
17. Gonzalez RC, Woods RE (1992) Digital Image Processing. Addison-Wesley Publishing Company
18. Ostu N (1979) A Threshold Selection Method from Gray-Level Histograms. IEEE Trans Syst Man Cybern 9: 62–66
19. Rabiner LR (1989) A Tutorial on Hidden Markov Model and Selected Application in Speech Recognition. Proc. IEEE 77: 257–286
20. Yamato J, Ohya J, Ishii K (1992) Recognizing Human Action in Time-Sequential Images Using Hidden Markov Model. Proc. IEEE Conference on Computer Vision and Pattern Recognition, pp. 379–385
21. Schlenzig J, Hunter E, Jain R (1994) Recursive Identification of Gesture Inputs Using Hidden Markov Models. Proc. 2nd. Ann. Conf. on Applications of Computer Vision, pp. 187–194
22. Campell L, Becker D, Azarbayejani A, Bobick A, Pentland A (1996) Invariant Features for 3D Gesture Recognition. Proc. of Int. Workshop on Automatic Face- and Gesture-Recognition, Vermont
23. Kurita T, Hayamizu S (1998) Gesture Recognition using HLAC Features of PARCOR Images and HMM based Recognizer. Proc. of Int. Workshop on Automatic Face- and Gesture-Recognition, Japan, APR. 14–16
24. Huttenlocher DP, Klanderman GA, Rucklidge WJ (1993) Comparing Images Using the Hausdorff Distance. IEEE Trans Pattern Anal Mach Intell 15(9): 850–863
25. Rucklidge WJ (1995) Locating Objects Using the Hausdorff Distance. Proc. of 5th ICCV, pp. 457–464
26. Dubuisson M-P, Jain AK (1994) A Modified Hausdorff Distance for Object Matching. Proc. 12th IAPR International Conference on Pattern Recognition, pp. 566–568



Chung-Lin Huang was born in Tai-Chung, Taiwan, in 1995. 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. He obtained his Ph.D. degree in Electrical Engineering from the University of Florida, Gainesville, Fla., USA, in 1987. From 1981 to 1983, he was an associate enginees 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.

Sheng-Hung Jeng was born in KaoShung, Taiwan, in 1973. He received his B.S. degree and M.S. degree from Electrical Engineering Department, National Tsing-Hua University, Hsin-Chua, Taiwan, in 1993 and 1995, respectively. His research interests are signal processing and image processing.