Image and Vision Computing 18 (2000) 865–879

www.elsevier.com/locate/imavis

Gesture recognition using the multi-PDM method and hidden Markov model

Chung-Lin Huang*, Ming-Shan Wu, Sheng-Hung Jeng

Institute of Electrical Engineering, National Tsing-Hua University, Hsin-Chu, Taiwan, ROC

Received 14 March 1997; received in revised form 14 September 1998; accepted 17 September 1999

Abstract

This paper introduces a multi-Principal-Distribution-Model (PDM) method and Hidden Markov Model (HMM) for gesture recognition. To track the hand-shape, it uses the PDM model which is built by learning patterns of variability from a training set of correctly annotated images. However, it can only fit the hand examples that are similar to shapes of the corresponding training set. For gesture recognition, we need to deal with a large variety of hand-shapes. Therefore, we divide all the training hand shapes into a number of similar groups, with each group trained for an individual PDM shape model. Finally, we use the HMM to determine model transition among these PDM shape models. From the model transition sequence, the system can identify the continuous gestures representing one-digit or two-digit numbers. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Gesture recognition; Multi-PDM method; Hidden Markov model

1. Introduction

Humans are experts at using gestures for communication. Hand gestures have been widely used in the deaf community as the major communication media called sign language. Gesture input aims to exploit this natural expertise for human–computer interface. If the machine can understand the human gesture either static or dynamic effectively, then it will greatly benefit the human beings. In the last several years, there has been an increased interest in trying to introduce human–machine interaction through human body motion which coincides with a growing interest in a closely related field—virtual reality.

Huang et al. [1] presented a review of the most recent studies related to hand gesture interface techniques: glove-based technique, vision-based technique, and analysis of drawing gesture. The vision-based technique is the most natural way of constructing a human–computer interface which has many applications [13–15]. However, it has difficulties in: (1) segmentation of the moving hands; (2) tracking and analyzing the hand motion; and (3) recognition.

The vision-based gesture recognition methods avoid using expensive wired “dataglove” equipment [2]. In this paper, we are interested in developing new vision-based methods. Huang et al. [3] have developed a Chinese sign language recognition system to recognize 15 different gestures by using Hausdorff distance measurement and a 3-D neural network. Tamura et al. [4] developed a system which can recognize 20 Japanese sign gestures based on matching simple cheremes. Davis et al. [5] 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. [6] proposed 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 American Sign Language (ASL) symbols.

Rehg et al. [7] have designed a system called *DigitEyes* that uses a 3-D cylindrical kinematics model of human hand with 27 degrees of freedom. Finger tips and links were chosen as the model matching features and were extracted from either single or stereoscopic images. Darrell et al. [8] have proposed another space–time gesture recognition method. 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. [9] have used a Hidden Markov Model (HMM) for visual recognition of complex, structured hand gestures

* Corresponding author. Fax: +886-35715971.
E-mail address: clhuang@ee.nthu.edu.tw (C.-L. Huang).

PII: S0262-8856(99)00042-6

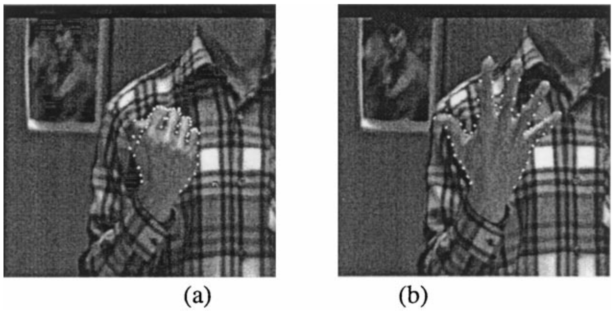

Fig. 1. The positions of the labeled points are shown around the boundary of the hand: (a) the labeled points of the fist that grasps firmly; (b) the five fingers are straight.

such as ASL. They applied HMM to recognize "*continuous*" ASL of a full sentence and demonstrated the feasibility of recognizing complex gestures.

Cui et al. [10] have proposed a learning-based hand gesture recognition framework. It consists of a multi-class multivariant discriminant analysis to automatically 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. [11] explored posture estimation based on the 2-D projective hand silhouettes for vision-based gesture recognition. Wilson et al. [12] presented a state-based technique for the representation and recognition of gesture. States are used to capture both the variability and repeatability evidenced in a training set for a given gesture. They developed a method for recognizing gesture from an unsegmented continuous stream of sensor data. However, most of the previous studies are limited by (1) simple background; (2) simple hand figures with only trajectory analysis; (3) use of special gloves.

This paper presents a multi-PDM-based method for hand tracking and handshape extraction, and then generates an ordered sequence of model transitions by using the hidden Markov Model (HMM). The PDM-based hand shape extraction is resistant to complex background influence, and the model transition is invariant to the non-uniform changes in speed and viewing direction. Our method has the advantage that the gesture recognition depends on how the system makes the PDM model transition instead of how exactly it reaches a certain position in 3-D space. Our goal is to convert the variances of the gesture in the spatio-temporal space into a sequence of PDM model transitions as a gesture symbolical representation.

The gesture recognition technique includes tracking the object of interest and identifying the non-rigid hand-shape. The major assumption for a successful tracking algorithm is that the 2-D shape of the moving hand-shape changes smoothly between two consecutive frames. The system has two stages: (1) multi-PDM-based hand-shape tracking and measurement and (2) HMM-based PDM model transition determination. First, we find that the PDM (or Active Shape Model [16]) method can only fit new hand examples similar to shapes of the corresponding training set. Since there are so many different hand shapes with lots of varieties, we cannot use the PDM shape model to deal with the entire sequence of hand gesture. Therefore, we need to divide all the hand shapes into a number of similar groups, with each group trained for an individual PDM model. Second, for each frame, with the observation of the fitness function, we apply HMM to determine the PDM model transition. The model transition is required when the current flexible model is no longer suitable for a large variation of the hand-shape in the following frames.

2. Hand shape extraction

Here, we modify the Active Shape Model [16] (or Point Distribution Model (PDM)) method to extract the hand shapes. For PDM, the average example is calculated and the deviation of each example from the mean is established. A principal component analysis of the covariance matrix of deviations reveals the main mode of variation. Usually only a small number of model parameters is required to reconstruct the training examples. Lanitis et al. [17] applied the PDM to track human face. Heap et al. [18] extended the works of Ref. [16] by proposing a Cartesian–Polar Hybrid PDM which allows the angular movement to be modeled directly.

We may generate new examples of the shape, which will be similar to those in the training set, by varying the parameters within certain limits. The mean shape model is placed in the image, and is allowed to interact dynamically until it fits to the location of a newly suggested position for each model point based on the matching of the local intensity model. Different from Refs. [16,17] which deform each model point individually, we propose another approach: (1) moving and deforming the entire PDM shape model simultaneously by changing the shape parameters and (2) measuring the model-image fitness by using the overall gray-level fitness measure. Here, we apply the gradient-descent-based shape parameter estimation that minimizes the overall gray-level model fitness measure. By varying the shape parameters that are consistent with the training set, we can find the best shape model fitted with the real face in the image. However, in Refs. [16,17], each model point moves independently and the movements are not consistent with the PDM shape model, therefore, they need to adjust the model points by estimating the PDM shape parameters and then readjusting the movements which are computation-intensive operations.

2.1. Point distribution model

To deal with various facial expressions on different persons, we need to build a model which describes both shape and variability. We manually locate the feature points on the training set images by following some rules to ensure that each point plays an essential role on the boundary of the images. This will ensure the coherence of points on the

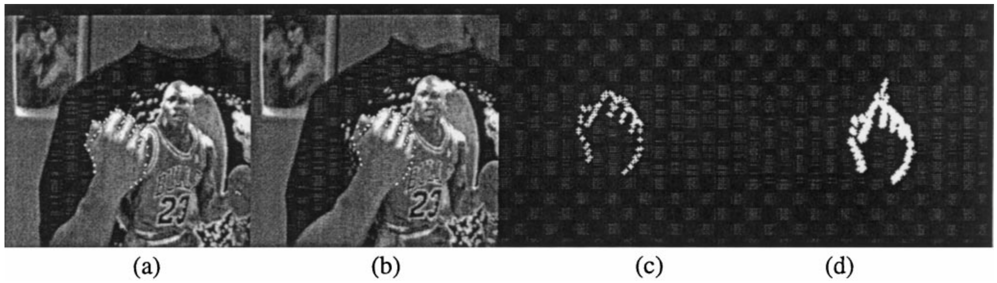

Fig. 2. (a) and (b) illustrate the hand shapes with labeled points. (c) shows the result that (b) is aligned with (a). (d) shows the aligned shape of a training set.

different features. We call these points "landmark points". If the choice of landmark points is improper, the method may fail to capture shape variability reliably. We select the landmark points (see Fig. 1) based on the following rules:

1. The points mark some parts of the object with particular application-dependent significance, such as the center of an eye on the face model or sharp corners of a boundary.
2. The points can be interpolated from the pre-selected points, for instance, the landmark on the boundary at equal distances to the other two neighboring landmarks.

2.1.1. Aligning the training set

The PDM-based method analyzes the statistics of the coordinates of the labeled points over the training set. To have a concise shape model, we must label (using landmark points) different features on the images in 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. 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 of the N shapes; and then align every shape to the mean shape. The detailed algorithm of the aligned shapes of the training set (see Fig. 2) can be found in Ref. [16].

2.1.2. Statistical analysis of the aligned shapes

Having generated the 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

$$d\mathbf{x}_i = \mathbf{x}_i - \bar{\mathbf{x}}. \quad (1)$$

Then, we can obtain the $2n \times 2n$ covariance matrix $\mathbf{S}$ as

$$\mathbf{S} = \frac{1}{N}\sum_{i=1}^{N} d\mathbf{x}_i \, d\mathbf{x}_i^{\mathrm{T}} \quad (2)$$

Applying the principal component analysis, 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 $\mathbf{S}$ (i.e. $p_1, \ldots, p_{2n}$) such that

$$\mathbf{S}\mathbf{p}_k = \lambda_k \mathbf{p}_k \text{ with } \mathbf{p}_k^{\mathrm{T}}\mathbf{p}_k = 1 \quad (3)$$

where λ_k is the kth eigenvalue of $\mathbf{S}$, with $\lambda_k \geq \lambda_{k+1}$. According to the principal component analysis, 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.

To determine how many terms is enough for shape variation description, we define λ_{T} as

$$\lambda_T = \sum_{k=1}^{2n} \lambda_k \text{ and } \lambda_t = \sum_{k=1}^{t} \lambda_k. \quad (4)$$

Then, based on the experimental results, $\lambda_t/\lambda_T = 0.8$ is sufficient. We use 51 landmark points ($n = 51$) and 4 eigenvectors ($t = 4$) 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 $\mathbf{b} = (b_1, \ldots b_t)^{\mathrm{T}}$ 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 illustrated in Fig. 3.

Fig. 3. Illustration of the effects of varying the parameters b1, b2, b3, and b4 of hand model from first row in order.

2.2. The gray-level model

Since the facial contours do not indicate the existence of strong edges, whereas, some face feature points are so close to one another that the edge information on one point may interfere with the edge of the other point. To resolve these drawbacks, Cootes et al. [16] introduced the gray-level model. Since every point on the face is on a particular

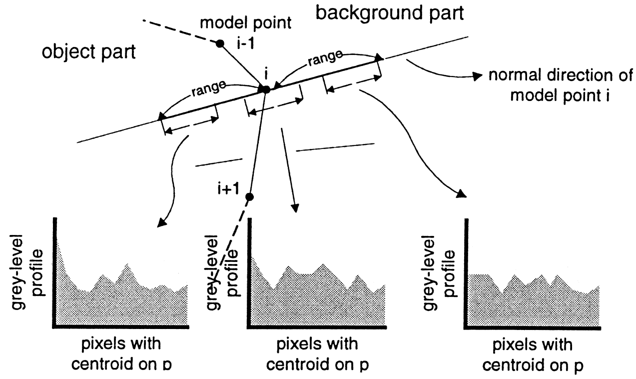

Fig. 4. Illustration of moving the center point of the 15-pixels kernel within the specific range, and calculate their $b_{g(\mathrm{new})}$ from the gray-level distribution.

position, its gray-level appearance for every face in the training set will be similar. There are several ways to describe the gray-level appearance. We may use a rectangular window with the centroid located on the feature point and find the 1-D profile which is normal to the curve passing through the feature point to record the gray-level appearance. To reduce the error caused by the background luminance variation, we sample the difference of the gray-level along the profile and then normalize it.

For every feature point in the training set, we can extract a profile, $\mathbf{g}_j$ $(j = 1,\dots,n)$, of length $n_p + 1$ pixels, centered at the point j. If the profile's samples starts at $\mathbf{x}_{\mathrm{start}}$ and ends at $\mathbf{x}_{\mathrm{end}}$ with length $n_p + 1$ pixels (see Fig. 4), the intensity of the kth element of the profile is

$$g_{jk} = I_j(\mathbf{y}_k) \tag{5}$$

where $\mathbf{y}_k$ is the location of the point along the profile,

$$\mathbf{y}_k = \mathbf{x}_{\mathrm{start}} + \frac{k-1}{n_p}(\mathbf{x}_{\mathrm{end}} - \mathbf{x}_{\mathrm{start}}) \tag{6}$$

and $\mathbf{I}_j(\mathbf{y}_k)$ is the gray-level at the position $\mathbf{y}_k$. Then, we calculate the normalized difference of $\mathbf{g}_j$ by using the following equation:

$$\mathbf{g}'_j = \mathbf{g}''_j / \sum_{k=1}^{n_p} |g''_{jk}| \tag{7}$$

where $\mathbf{g}''_j = [g''_{j1}, g''_{j2}, \dots g''_{j(n_p+1)}]$, $g''_{jk} = g_{jk} - g_{j(k-1)}$, $k = 1 \dots n_p + 1$, and g_{jk} is the kth pixel for the jth feature point's gray-level profile on the current frame. For convenience, we will simply substitute $\mathbf{g}_j$ for $\mathbf{g}'_j$. Here, we use principal component analysis to describe the statistical property of the gray-level. For each feature point, we calculate a mean profile $\bar{\mathbf{g}}$, then get a $n_p \times n_p$ covariance matrix $\mathbf{S_g}$, an eigenmatrix $\mathbf{P_g}$ and a set of eigenvalue λ_k $(k = 1, \dots n_p)$. For an arbitrary sampled profile $\mathbf{g}$, we apply the following function to evaluate how well it can be fitted to a particular landmark point j (with position $\mathbf{x}_j$) as

$$F(\mathbf{x}_j) = \sum_{j=1}^{n_p} \frac{b_{gj}^2}{\lambda_j} \tag{8}$$

where $\mathbf{b}_g = \mathbf{P}_g^{\mathrm{T}}(\mathbf{g} - \bar{\mathbf{g}})$ and $\mathbf{b}_g = (b_{g1}, b_{g2}, \dots, b_{gn_p})$. In the fitting process (see Fig. 6), we measure the F value to determine the displacement of a particular point from the initial position to the best fit position. Along the normal direction of each model point, we find the smallest F value that indicates the best match between the gray-level profile of the current position of the test model point and the mean profile of the corresponding feature point. Suppose the displacement is d_{best}, then the adjusted displacement $|d\mathbf{X}| = 0.5d_{\mathrm{best}}$ if $d_{\mathrm{best}} < d_{\max}$ otherwise $|\mathrm{d}\mathbf{X}\| = 0.5d_{\max}$. We set the $d_{\max}$ value adaptively to reduce the calculation time, it decreases as the number of iterations increases.

Here, we assume (1) the background does not change much during the gray-level model generation phase, and (2) the illumination variation is linear. We may neglect the influence of the background on the gray-level generation by applying the differentiation and normalization on gray-level profile (i.e. Eq. (7)) to reduce the error caused by the illumination changes.

2.3. Shape model and feature points interaction

This section describes how to use the PDM and the gray-level model to extract the hand-shape. Suppose the current shape position is $\mathbf{X}$ (with centroid $\mathbf{X}_{\mathrm{c}}$) and we need to adjust the global shape variation (including the translation $\mathrm{d}\mathbf{X}_{\mathrm{c}} = (\mathrm{d}\mathbf{X}_{\mathrm{c}}, \mathrm{d}\mathbf{Y}_{\mathrm{c}})$, rotation $\mathrm{d}\theta$, the scale $\mathrm{d}s$) and the local shape variation $\mathrm{d}\mathbf{b}$ to find the next fitting position $\mathbf{X} + \mathrm{d}\mathbf{X}$,

$$\mathbf{X} + \mathrm{d}\mathbf{X} = (\mathbf{X}_{\mathrm{c}} + \mathrm{d}\mathbf{X}_{\mathrm{c}}) + \mathbf{M}((s + \mathrm{d}s), (\theta + \mathrm{d}\theta))\cdot[\bar{\mathbf{x}} + \mathbf{P}\cdot(\mathbf{b} + \mathrm{d}\mathbf{b})] \tag{9}$$

where $\mathbf{M}(s, \theta)$ is a 2×2 rotation matrix. By finding gray-level profiles of every point j on $\mathbf{X} + \mathrm{d}\mathbf{X}$ ($\mathbf{x}_j \in \mathbf{X} + \mathrm{d}\mathbf{X}$) as $\mathbf{g}_j$, we calculate the gray-level profile fitness value $F(\mathbf{x}_j)$ and find the overall F values (i.e., $\sum_j F(\mathbf{x}_j)$ for $\mathbf{x}_j \in \mathbf{X} + \mathrm{d}\mathbf{X}$) of all landmark points. If the $\sum_j F(\mathbf{x}_j)$ is minimized then the position $\mathbf{X} + \mathrm{d}\mathbf{X}$ indicates the best fitted shape. In the following, we illustrate a modified PDM-based fitting process.

1. *Initial Hand Model Position Estimation.* In the hand-shape extraction process, we may encounter the problem that if the positions of some fitting points are too far away from the actual positions, then the adjustment may require a lot of iterations to pull the landmarks points to the proper place. Therefore, we apply frame difference operation to find the moving regions one of which is supposed to be the moving hand. From these extracted regions, we can roughly estimate the position of the hand to place the initial PDM shape model.
2. *Shape Adjustment Process.* Here, we apply the two-step

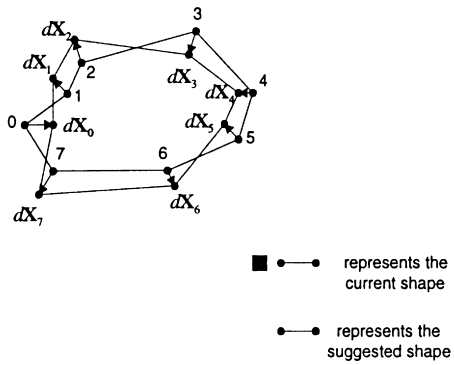

Fig. 5. The movement of the model points from the original shape (solid lines) to the suggested shape (dash lines) by measuring the similarity between the current extracted contour and the ones stored in the database.

estimations for the global shape variation parameters (i.e. the translation $\mathrm{d}\mathbf{X}_c$, the rotation $\mathrm{d}\theta$, the scale ds) and the local shape variation parameter (i.e., d$\mathbf{b}$). First, we assume that the current global shape is $\mathbf{X}$, then we can do the global shape variation for the new global shape as $\mathbf{X} + \mathrm{d}\mathbf{X} = \mathbf{M}(s + \mathrm{d}s, \theta + \mathrm{d}\theta)\cdot[\mathbf{x}] + (\mathbf{X}_c + \mathrm{d}\mathbf{X}_c)$, where $\mathbf{M}$ is a 2×2 rotation matrix, $\mathbf{x}$ represents the aligned shape, and $\mathbf{X}_c$ represents the central point of current shape. Second, we may also deform the current local shape $\mathbf{x}$, by changing local shape parameter d$\mathbf{b}$ to generate the new local shape as $\mathbf{x} + \mathrm{d}\mathbf{x} = \bar{\mathbf{x}} + \mathbf{P}(\mathbf{b} + d\mathbf{b})$.

3. *Gradient-Descent-Based Shape Parameters Estimation*. To find the best fitted shape, we propose a gradient-descent-based shape parameters estimation method. The global and local shape parameters estimation for the *i*th iteration is illustrated in the following steps:
   1. Find the next shape $\mathbf{X} + \mathrm{d}\mathbf{X}$ by using the new global shape parameters $((\mathbf{X}_c + \mathrm{d}\mathbf{X}_c), s + \mathrm{d}s, \theta + \mathrm{d}\theta)$.
   2. Find the gray-level profile $(\mathbf{g}_j)$ of each landmark point j on $\mathbf{X} + \mathrm{d}\mathbf{X}$ $(\mathbf{x}_j \in \mathbf{X} + \mathrm{d}\mathbf{X})$ and calculate the corresponding fitness value $F(\mathbf{x}_j)$.
   3. Add the F values for all landmark points on $\mathbf{X} + \mathrm{d}\mathbf{X}$ to see if $\sum_j F(x_j)$ exceeds the pre-selected threshold F_m. If $\sum_j F(\mathbf{x}_j) > F_m$ then it indicates that the shape model does not fit to the real face on the image at all. Choose another initial value of $\mathbf{X}_c$ by adding a larger variation $\mathrm{d}\mathbf{X}_c$. Determine the $\mathrm{d}\mathbf{X}_c$ by selecting the median one of all the best d$\mathbf{X}$ of the landmark points (see Fig. 5). If $\sum_j F(\mathbf{x}_j) > F_m$ go to step 1, otherwise continue (it indicates a rough shape fitness).
   4. Determine the decrement or increment of the global shape parameters (i.e., $\pm$ds and $\pm$dθ) by examining $\sum_j F(\mathbf{x}_j)$ (i.e. $\{[\sum_j F(\mathbf{x}_j)]_i - [\sum_j F(\mathbf{x}_j)]_{i+1}\} > 0$ or <0).
   5. If $\sum_j F(\mathbf{x}_j)$ does not decrease (i.e. $\{[\sum_j F(\mathbf{x}_j)]_i - [\sum_j F(\mathbf{x}_j)]_{i+1}\} > 0$) for all small variations ds and dθ then continue else go to step 4.
   6. Examine the final $\sum_j F(\mathbf{x}_j)$. If $\sum_j F(\mathbf{x}_j) > F_n$ (another pre-select threshold) then go back to step 3 (to avoid being trapped in the local minimum), otherwise continue.
   7. Change the local shape parameters d$\mathbf{b}$ for the new local shape $\mathbf{x} + \mathrm{d}\mathbf{x}$ and then find the minimum $\sum_j F(\mathbf{x}_j)$, which indicates the best fitness of the PDM shape model. The decrement or increment of the local shape parameters d$\mathbf{b}$ is determined by the value of overall gray-level profile fitness (i.e. $\{[\sum_j F(\mathbf{x}_j)]_i - [\sum_j F(\mathbf{x}_j)]_{i+1}\} > 0$ or <0).
   8. Stop if $\{[\sum_j F(\mathbf{x}_j)]_i - [\sum_j F(\mathbf{x}_j)]_{i+1}\} > 0$ for all variations of d$\mathbf{b}$, otherwise go to step 7.

3. Multi-PDM model transition using hidden Markov model

The allowable shape domain cannot be enormously large for a single PDM shape model. If the hand shapes undergo enormous shape changes in the image sequence (the variance of the cloud of each corresponding model point of aligned shapes is very large), then we need to divide the training set of all the possible hand shapes into several similar shape groups. The variance of each cloud of aligned shapes in each group has to be small for tracking the variable hand shapes. Then each group is treated as an individual training set and trained as a different PDM shape model.

If the hand shape extraction by using current PDM shape model is no longer effective, the specific HMM can be found to determine when to replace it by another PDM model that is called PDM model transition (see Fig. 6). In the feature extraction process, we stop changing the PDM parameters,

(a) (b) (c) (d)

Fig. 6. Illustration of the process of model transition: (a) shows the fitting of *i*th image frame using the model gesture-0; (b) when the flexible model meets the $(i+1)$th frame, the current model can not fit the hand shape exactly; (c) given an initial hand-shape, the model transition occurs; (d) $(i+1)$th frame is fitted exactly using the newly suggested flexible model.

d**b**, once we find $\{[\sum_j F(\mathbf{x}_j)]_i - [\sum_j F(\mathbf{x}_j)]_{i+1}\} > 0$. Then, we examine the $\{F(\mathbf{x}_j)\}$ to determine whether the current PDM mode is appropriate or not. If not, then which PDM can be chosen for the next feature extraction. The measurements $\{F(\mathbf{x}_i)\}$ for certain landmark points are used as an observation sequence for the system to determine which HMM has the highest model probability that indicates the most appropriate PDM model transition. The measurement $\{F_i\}$ at the landmark points is a very important information (observations) for the system to calculate the model probability of the probable HMMs, and the highest one normally indicates the most appropriate PDM model transition.

3.1. Hidden Markov model

A hidden Markov Model (HMM) is Markov chain whose states cannot be observed directly, but can be observed through a sequence of observations. There are three key problems in HMM: evaluation, estimation, and decoding. The evaluation problem is that given an observation sequence **O** and a model, what is the probability that the observed sequence is generated by the model, $P(\mathbf{O}|\lambda)$. The estimation problem concerns how to adjust the model λ to maximize $P(\mathbf{O}|\lambda)$ given an observation sequence **O**. In decoding, the goal is to recover the state sequence given an observation sequence.

Let T be the length of observation sequence, N is the number of the state in the model, $\mathbf{O} = (O_1, \ldots O_N)$ is the observation sequence. In this paper, we consider the each observation O_t as a fitness vector $(F(\mathbf{x}_j), \ldots F(\mathbf{x}_k))$ for certain key features points $\mathbf{x}_j, \ldots \mathbf{x}_k$ defined by the PDM model. A HMM is characterized by the initial state probabilities, $\pi_i, i = 1, \ldots, N$, the state transition $a_{ij}, i, j = 1, \ldots N$, and the observation probability density $b_j(O_t)$, $j = 1, \ldots N$, $t = 1, \ldots T$. Let $B = \{b_j(O_t)| j = 1, \ldots N\}$, $N \times N$ transition probability matrix $A = [a_{ij}]$, and the initial state probability vector $\pi = [\pi_1 \ldots \pi_N]$, we may define the triple $\lambda = (\pi, A, B)$ as a HMM. Let $\alpha_1(i) = \pi_i b_i(O_1)$, we may calculate $\alpha_t(j)$ for $t = 2, \ldots T$ and all j as $\alpha_t(j) = [\sum_i \alpha_{t-1}(i) a_{ij}] b_j(O_t)$ and finally find the $P(\mathbf{O}|\lambda) = \sum_{i \in S_F} \alpha_T(i)$.

Here, we create one HMM for each possible PDM transition between two consecutive frames. We use the observations, $\mathbf{O} = \{F_i\}$, from current frame, to estimate the optimum parameters for each HMM, i.e. we obtain the model parameter λ_p, for the pth HMM. Given the measurement $\mathbf{O} = \{F_i\}$ of current frame and a HMM, which may indicate certain unknown model transition, we calculate $P(\mathbf{O}|\lambda)$. The $P(\mathbf{O}|\lambda)$ can be calculated by summing the probability over all the possible state sequence $S = (s_0, s_1, \ldots s_T)$, where $s_t \in \{1, 2, \ldots N\} = Z_N$, in a HMM model for the observation sequence:

$$P(\mathbf{O}|\lambda) = \sum_{\text{all } S} \pi_{s_0} \prod_{t=1}^{T} a_{s_{t-1}s_t} b_{s_t}(O_t) \tag{10}$$

The objective in maximum likelihood estimation is to maximize $P(\mathbf{O}\|\lambda)$ over all parameters λ for a given observation. The above maximum likelihood estimation can be effectively solved by Baum–Welch algorithm [21]. Here we consider different optimization criterion for estimating the parameters of HMM. Instead of using the likelihood function (10), we apply the following function as the optimization objective (it is called the state-optimized likelihood):

$$\max_s P(\mathbf{O}, \mathbf{S}|\lambda) = \max_s \pi_{s_0} \prod_{t=1}^{T} a_{s_{t-1}s_t} b_{s_t}(O_t) \tag{11}$$

Then we may apply the segmental K-means algorithm [20] for estimating the parameters of the HMMs which involves two fundamental steps: segmentation and optimization. Starting from an initial model λ, the segmentation step uses the sequential decoding procedure to generate a state sequence (with $\max_s P(\mathbf{O}, \mathbf{S}/\lambda)$ which can be optimally performed via a generalized Viterbi algorithm [22]). Given the state sequence **S** and the observation **O**, the optimization step finds a new set of model parameter λ_p so as to maximize the above state-optimized likelihood, as

$$\lambda_p = \underset{\lambda}{\operatorname{argmax}}\, P(\mathbf{O}, S|\lambda) \tag{12}$$

We replace the original model by new model and iterate the above steps until the state-optimized likelihood converges within a prescribed threshold. $P(\mathbf{O}, \mathbf{S}^*|\lambda)$ is called optimal likelihood function, and $\mathbf{S}^*$ is the optimal state sequence.

We choose the best HMM u^* (indicating the appropriate PDM model transition) by finding the highest model probability, i.e.

$$u^* = \operatorname{argmax}_{1 \le u \le U}[P_u] \tag{13}$$

where $P_u = P_u(\mathbf{O}, \mathbf{S}^*|\lambda_p)$, and λ_p makes $\max_\lambda P(\mathbf{O}, \mathbf{S}^*/\lambda)$. For a given λ, an efficient method to find $\max_s P(\mathbf{O}, \mathbf{S}/\lambda)$ is the well-known Viterbi algorithm. Viterbi algorithm can be viewed as a special form of 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 steps of the Viterbi algorithm are

1. *Initialization.* For all states i, $\alpha_1(i) = \pi_i b_i(O_1)$; $\psi_i(i) = 0$.
2. *Recursion.* From $t = 2$ to T for all state j, $\alpha_t(j) = \max[\alpha_{t-1}(i) a_{ij}] b_j(O_t)$; $\psi_t(j) = \operatorname{argmax}_i [\alpha_{t-1}(i) a_{ij}]$.
3. *Termination.* $\Pr = \max_{s \in S_F}[\alpha_T(s)]$; $s = \operatorname{argmax}_{s \in S_F}[\alpha_T(s)]$.
4. *Recovering the state sequence.* From $t = T - 1$ to 1, $s_t = \psi_{t+1}(s_{t+1})$.

3.2. HMM training

Since our decision rule is based on the state-optimized likelihood function, the estimated parameter λ' should be such that $\Pr(\mathbf{O}|\lambda')$ is maximized over the training set. The training problem is the crucial one for most applications of HMMs. It allows us to optimally adapt model parameters to the observed training data, and then create the best models

Table 1
The possible HMMs related to the current selected HMM

Current PDM model	Possible related and tested HMMs									
m_0	HMM_0	HMM_{01}	HMM_{02}	HMM_{03}	HMM_{04}	HMM_{05}	HMM_{06}	HMM_{07}	HMM_{08}	HMM_{09}
m_1	HMM_1	HMM_{10}	HMM_{12}	HMM_{13}	HMM_{14}	HMM_{15}	HMM_{16}	HMM_{17}	HMM_{18}	HMM_{19}
m_2	HMM_2	HMM_{20}	HMM_{21}	HMM_{23}	HMM_{24}	HMM_{25}	HMM_{26}	HMM_{27}	HMM_{28}	HMM_{29}
m_3	HMM_3	HMM_{30}	HMM_{31}	HMM_{32}	HMM_{34}	HMM_{35}	HMM_{36}	HMM_{37}	HMM_{38}	HMM_{39}
m_4	HMM_4	HMM_{40}	HMM_{41}	HMM_{42}	HMM_{43}	HMM_{45}	HMM_{46}	HMM_{47}	HMM_{48}	HMM_{49}
m_5	HMM_5	HMM_{50}	HMM_{51}	HMM_{52}	HMM_{53}	HMM_{54}	HMM_{56}	HMM_{57}	HMM_{58}	HMM_{59}
m_6	HMM_6	HMM_{60}	HMM_{61}	HMM_{62}	HMM_{63}	HMM_{64}	HMM_{65}	HMM_{67}	HMM_{68}	HMM_{69}
m_7	HMM_7	HMM_{70}	HMM_{71}	HMM_{72}	HMM_{73}	HMM_{74}	HMM_{75}	HMM_{76}	HMM_{78}	HMM_{89}
m_8	HMM_8	HMM_{80}	HMM_{81}	HMM_{82}	HMM_{83}	HMM_{84}	HMM_{85}	HMM_{86}	HMM_{87}	HMM_{89}
m_9	HMM_9	HMM_{90}	HMM_{91}	HMM_{92}	HMM_{93}	HMM_{94}	HMM_{95}	HMM_{96}	HMM_{97}	HMM_{98}

for real phenomena. In this paper, we define the observation sequence in terms of spatial order (for each input frame) as $\mathbf{O} = (O_1, O_2, O_3, O_4, O_5)$ where $O_1 = (F(x_5), F(x_6), F(x_7), F(x_8), F(x_9), F(x_{10}), F(x_{11}))$, $O_2 = (F(x_{14}), F(x_{15}), F(x_{16}), F(x_{17}), F(x_{18}), F(x_{19}), F(x_{20}))$, $O_3 = (F(x_{22}), F(x_{23}), F(x_{24}), F(x_{25}), F(x_{26}), F(x_{27}), F(x_{28}))$, $O_4 = (F(x_{30}), F(x_{31}), F(x_{32}), F(x_{33}), F(x_{34}), F(x_{35}), F(x_{36}))$, $O_5 = (F(x_{38}), F(x_{39}), F(x_{40}), F(x_{41}), F(x_{42}), F(x_{43}), F(x_{44}))$. The central feature points x_8, x_{17}, x_{25}, x_{33}, x_{41} are located on the finger-tip of the thumb, the index finger, the middle finger, the ring finger, and the little finger, respectively. Each observation vector O_t may be assigned to one of the three different states: bending (S_b), half-bending (S_h), and straight (S_s) indicating the status of each finger.

We start with a training sequence consisting of a number of repetitions of the gesture frames (made by many gesture-makers). For each HMM model, we first adjust the model parameters λ so that $\Pr(\mathbf{O}|\lambda)$ is maximized. Then we use Viterbi algorithm to find the optimal state sequence associated with the given observation sequence. The results are used to re-estimate the model parameter λ'. The initial model defines a critical point of the likelihood function, in which $\lambda' = \lambda$. Baum–Welch algorithm [21] has been proposed to re-estimate a new model λ' which is more likely in a sense that $\Pr(\mathbf{O}|\lambda') > \Pr(\mathbf{O}|\lambda)$. The model λ' indicates that the observation sequence is more likely to be produced. Instead of finding the λ_p that minimizes $P(\mathbf{O}|\lambda)$ (i.e. $\max_{\lambda_p} P(\mathbf{O}|\lambda)$), which requires summing all possible state sequences (see Eq. (10)), we focus on the most likely state sequence (see Eq. (11)), and apply the segmental K-means algorithm [20] which had been proved to have faster convergence and higher flexibility.

It is shown that the segmental K-means algorithm [20] converges to the maximized state-optimized likelihood function for the Gaussian density. We use K-means algorithm [19] to cluster all the training vectors into N clusters, each cluster is chosen as a state and numbered from 1 to N. The tth vector O_t of a training sequence $\mathbf{O}$ is assigned to state i, denoted as $O_t \in i$, if its distance to the state i is smaller than its distance to any other state j, $j \neq i$. This is the initial step for the complete procedure.

Given a state sequence $\mathbf{S}$ and the observation $\mathbf{O}$, the optimization step finds a new model parameters λ' so as to maximize the above state-optimized likelihood (see Eq. (12)). Note that the maximization of the state-optimized likelihood in Eq. (12) may not be straightforward. For each state i, the generalized iteration algorithm may have to be employed, depending on the choice of the observation densities which need to be T-converge [20,22]. We then replace the original model λ by the new λ' and iterate the above two steps (the segmentation and optimization steps) until the state-optimized likelihood converges within a predefined threshold.

4. System implementation

In this paper we develop a system to interpret the gestures made only for decimal numbers. Here, we define some criteria for gesture making so that the gestures can be identified by our system.

4.1. Gesture making (the segmentation and optimization steps)

To make a single-digit number gesture, we start the gesture-making operation from holding our fist, then raise certain fingers to indicate the specific number (see Fig. 9), and finally bend those fingers to return to fist-holding state. If one want to make gesture indicating two-digit number, then he may repeat the above operation. However, if we want to make a gesture indicating a single-digit ‘0’, then we may differentiate the beginning/ending fist-holding gesture from the gesture indicating digit ‘0’. Therefore, we use the forward translation motion between the beginning fist-holding gesture and the gesture indicating digit ‘0’ and then use the reverse translation motion between the gesture indicating digit ‘0’ and the ending fist-holding gesture. The translation motion is also applicable to the gesture of the other nine digits so that the system can differentiate the beginning/ending fist-holding gesture from the gesture-digit ‘0’.

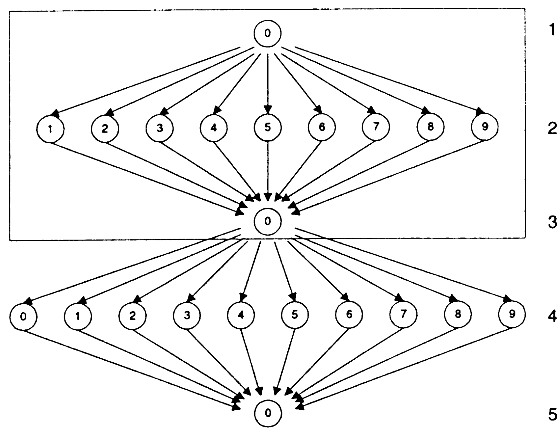

Fig. 7. Illustration of the gesture recognition with the model transition having a global motion. The level 1 represents the initial model, the levels 2 and 4 represent the active model, the level 3 is the intermediate model, and the level 5 represents the final model.

4.2. PDM transition sequence generation for gesture identification

For each frame, we can track the hand gesture by using the most appropriate PDM models (applied to the previous frame) to calculate the $\{F(x_i)\}$ as an observation sequence. Using the observations of current frame, we apply all possible related HMMs (see Table 1) and find the best HMM with the highest state-optimized likelihood that indicates the most appropriate PDM model for the current frame. In our system, we have trained two different categories of HMMs. The first one has 10 HMMs (HMM_i, $i = 0, 1, \ldots 9$) indicating no PDM model transition. The second one consists of 45 HMMs (HMM_{ij}) corresponding to a PDM model transition, from current PDM model m_i to the other PDM model m_j. We assume that the measurement statistics $\{F(x_i)\}$ corresponding to HMM_{ij} representing the transition from PDM model m_i to PDM model m_j and the other HMM_{ji} indicating the transition from PDM model m_j to PDM model m_i are trained as the same HMM. Given an observation sequence, we need to find the optimal HMM which indicates whether there is an PDM model transition or not. If there is a PDM model transition, then what kind of PDM model transition may occur. During the training process, given as many known input frames as possible, we train 55 different HMMs individually for our system. The best trained HMM is the one indicating no PDM model transition. Since the measurement statistics $\{F(x_i)\}$ of most of the frames in the image sequence favor the first category HMM.

To recognize the hand gesture, we need to convert an image sequence to a sequence of PDM model transitions. Our system can identify the gesture by interpreting the ordered sequence of PDM model transitions. In this study, we let the PDM model m_0 play two different roles in the transition sequence as: (1) a conjunctive PDM model representing the initial, intermediate, or final PDM models and (2) a sign PDM model representing the digit '0'. Each gesture can be described by a PDM model transition sequence that starts from the initial PDM m_0, and ends with the final PDM model m_0.

Here, we assume that the PDM model transition can also be determined if the hand movement is tracked by measuring the displacement of the centroid of the extracted hand shapes in two consecutive frames. Therefore, to make a gesture indicating digit '0' is made, we apply a hand translation motion to indicate the PDM model transition from the initial conjunctive model m_0 to the sign model m_0. A input image sequence of a gesture indicating a single-digit number '*n*', will be processed and described by three consecutive PDM models m_0, m_n, and m_0. Hence, the PDM model m_0 plays two different roles: (1) m_0 is a conjunctive PDM model, if some sort of translation motion is detected and the hand has moved away from the original position. (2) m_0 is a sign PDM model, if no translation motion is found for a small time interval and then the hand has returned to the original position.

To give a more specific illustration of how to interpret the

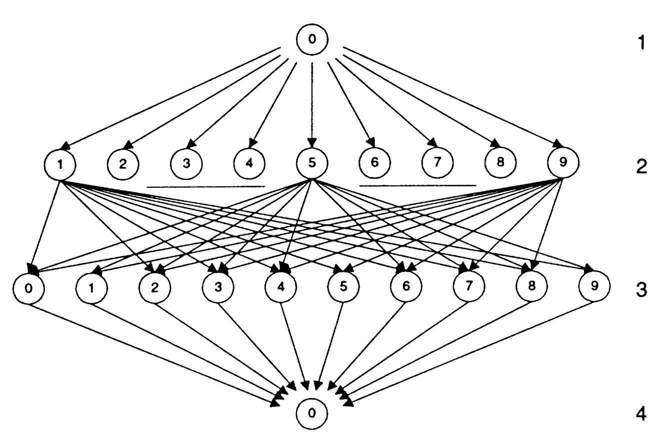

Fig. 8. Illustration the gesture recognition without intermediate state of continuous gesture model transition. The level 1 represents the initial model, the level 2 and 3 represent the active model, and the level 4 represents the final model.

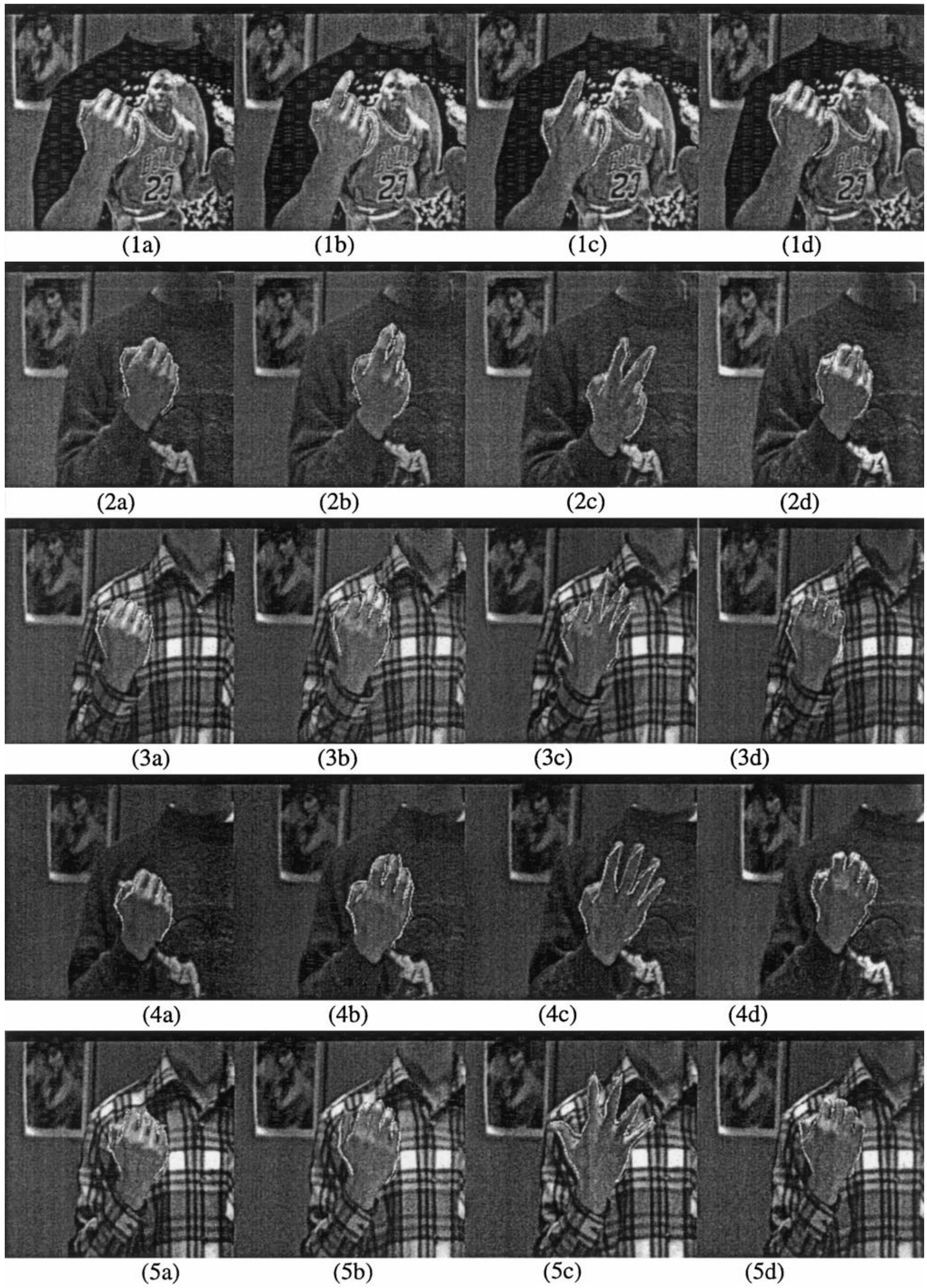

Fig. 9. The image sequence tracking of the single-digit gestures from "1" to "9", the PDM model transition starts from m_0 to m_i, and finally returns m_0.

gesture through the PDM model transition sequence, we illustrate the following examples:

- *Example one:* As illustrated in Fig. 7, to make a gesture indicating two-digit number '*jk*', we can use a so-called *the gesture with translation motion*. This gesture can be described successfully by four PDM model transitions as: $m_0 \rightarrow m_j \rightarrow m_0 \rightarrow m_k \rightarrow m_0$.
- *Example two:* As shown in Fig. 8, to make another gesture indicate the same two-digit number '*jk*', we can use a so-called *the gesture without translation motion*. This gesture can also be depicted by another PDM model transition sequence as: $m_0 \rightarrow m_j \rightarrow m_k \rightarrow m_0$. Here, the hand translation motion is unnecessary to imply the PDM model transition from m_j to m_k.
- *Example three:* If we want to recognize a gesture of a double-digit number '*nn*', then we may find the intermediate conjunctive PDM model m_0 between two sign PDM models m_n. The corresponding PDM model transition sequence is represented as $m_0 \rightarrow m_n \rightarrow m_0 \rightarrow$

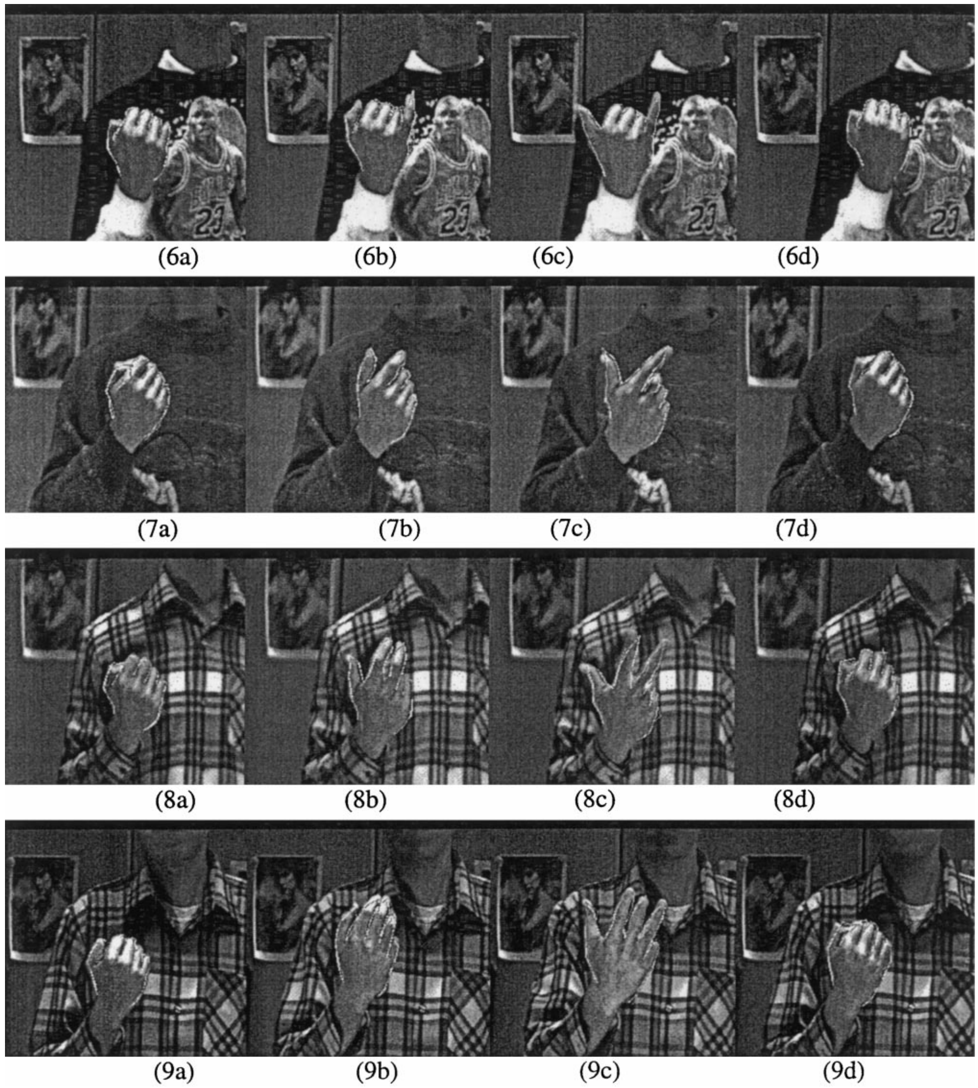

Fig. 9. (*continued*)

$m_n \rightarrow m_0$. There is only one kind of gesture, "*the gesture with translation motion*", that can be used to indicate a double-digit number.

- *Example four:* However, we can only use one type of gesture (*the gesture with translation motion*) to represent the same number '*n*0'. We may find the intermediate model m_0 between two sign models m_n and m_0, since there is noticeable hand movement between the sign model m_0 and the intermediate (or end) model m_0. This example can be represented by the PDM model transition sequence as $m_0 \rightarrow m_n \rightarrow m_0 \rightarrow m_0 \rightarrow m_0$, in which the second PDM model m_0 acts as an intermediate model.

From the above examples, we may find that we can use two kinds of gestures (with/without motion) to indicate the one-digit or two-digit numbers. However, for the double-digit number '*nn*' or the number with digit '0', we can only apply *the gestures with translation motion* to avoid the misunderstanding between the sign model m_0 and the conjunctive model m_0. The rules can also be applied to other gestures indicating multi-digit numbers.

5. Experimental results

We have developed a system to recognize a gesture representing any one-digit or two-digit number. First, we take 30 typical frames for training each HMM which indicates a specific PDM transition. There are five vectors ($T = 5$) in each observation sequence indicating current information of the five fingers and three different states ($N = 3$) for each model indicating the bending, half-bending and straight status of each finger. In the training process, we take average of all training sequences of each class to get an average sequence for each class. To train the model, we use the K-means algorithm [19] to cluster all the observation vectors into N cluster.

In the experiments, we present each gesture with an

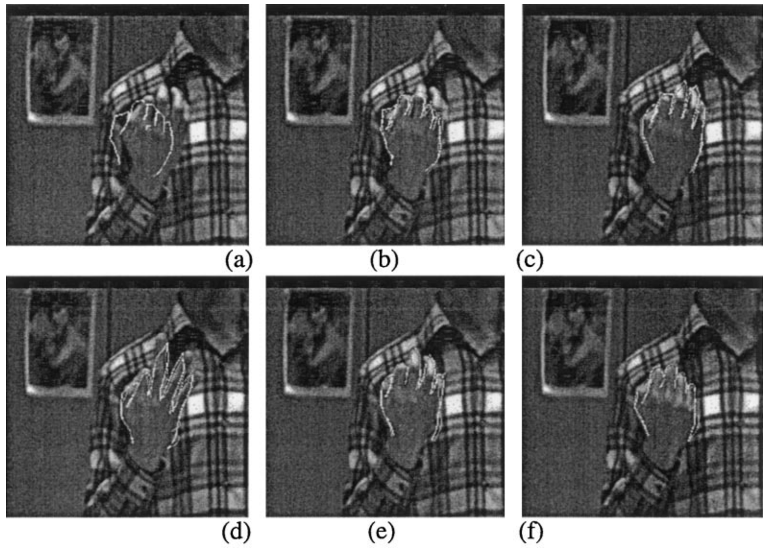

Fig. 10. The image sequence tracking single-digit gesture "3", but its first hand shape is not well described by model m_0: (a) shows the initial hand shape located near the real hand in the first frame; (b)–(f) present the model transition from m_0 to model m_3, and finally return to model m_0.

ordered model sequence ended always with model m_0. From the identified PDM model transition, we can do the gesture recognition effectively. In the experiments, we take the gesture sequences from the 12 volunteers, each one demonstrates different hand gestures. We take 10 image sequences for every volunteer, and overall, we take 120 image sequences. There are 15 pictures in an image sequence, and the size of the picture is 256×256. The camera used in our experiment is a SONY XC7500. For each gesture, an image sequence of 30 frames is taken at 30 frames/s and stored in DRAM on an Oculus-F/64 frame grabber which is transferred to the host computer (a PC with Pentium CPU) for further processing. Here, we present several experimental results of hand gesture tracking. The PDM-based hand-shape tracking of image sequence of the single (or two) digit gesture has to deal with the following problems: (1) The initial hand shape is not the standard shape as described by PDM model m_0. (2) The hand shape is occluded by face, neck, or upper arm.

In our experiments, the size of each image frame is 256×256, its frame rate is 30 frames/s, and the number of frames of each gesture is less than 30. Fig. 9 shows the hand tracking of single-digit-number gestures before three different complex backgrounds. These gestures represent numbers 1, 2, 3, 4, 5, 6, 7, 8 and 9, respectively. Fig. 10 demonstrates the tracking process of the hand shape in the first image frame which is not similar to the standard initial shape.

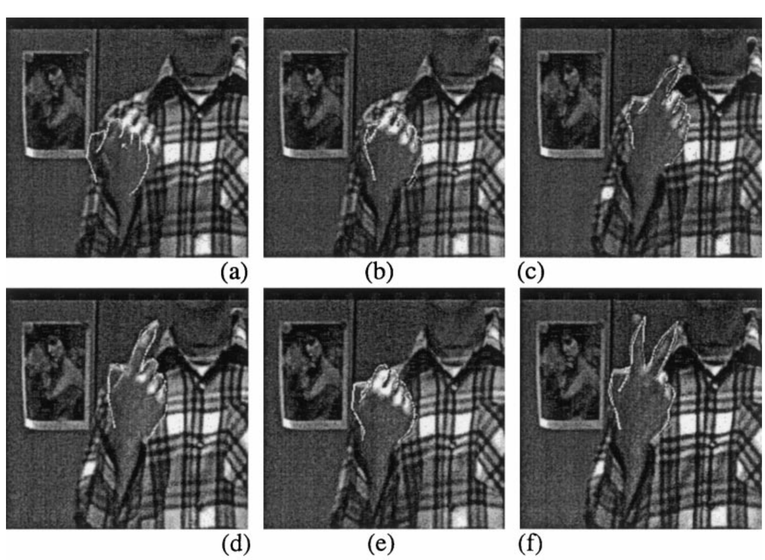

Fig. 11. The image sequence tracking of two-digit gesture "12": (a) shows the initial hand-shape located near the real hand in the first frame; (b)–(f) present the PDM model transition from model m_0 to m_1, then return to model m_0, finally transition to model m_2.

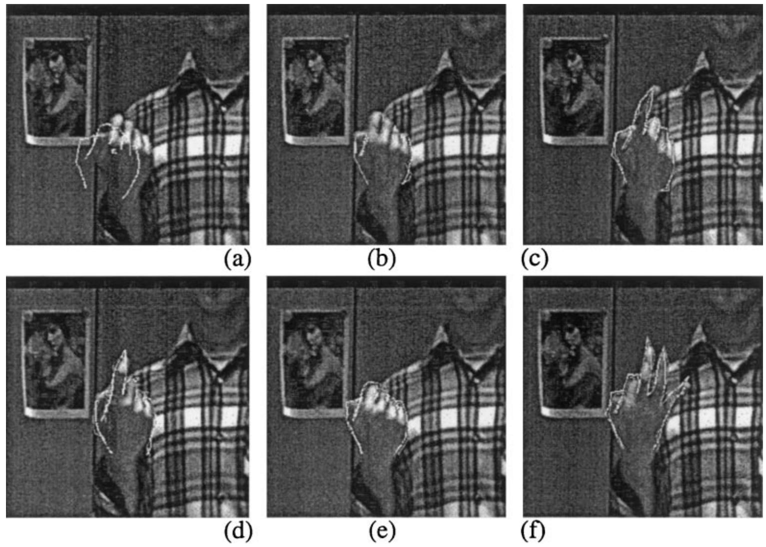

Fig. 12. The image sequence tracking of two-digit gesture “13”: (a) shows the initial hand shape located near the real hand in the first frame,; (b)–(f) present the training set transition from m_0 to m_1, then return to m_0, finally transition to model m_3.

Figs. 11–14 show the continuous hand tracking of two-digit gestures having model m_0 as an intermediate conjunctive model. These gestures represent numbers ‘12’, ‘13’, ‘27’, and ‘38’, respectively. Figs. 15 and 16 show the continuous hand tracking of two-digit number gestures without having PDM model m_0 as an intermediate conjunctive model (i.e. model transition without referring to the hand translation motion). These gestures represent numbers ‘12’, and ‘24’, respectively.

In the above sequence, most of the model transitions detected by HMM are accurate. The incorrect PDM model transitions are identified when (1) the observation vector (provided by the PDM-based hand-shape extraction process) is not accurate, (2) the movements of the raising or bending fingers are not coherent. For instance, the gesture of number ‘2’, normally, requires both the index finger and the middle finger raised up-right almost at the same time. If the middle finger is raised faster by one frame or two, then the selected HMM may not indicate the correct PDM model transition. The error will influence the selection of all possible HMMs tested for the succeeding frames. If the current selected HMM is not correct, then the correct HMM for the next frame is normally not in the set of possible HMMs. The recognition rate of using HMM in the experiments to test the 120 image sequences (30 frames/sequence) is illustrated in Table 2.

We have tested four image sequences for each gesture. Most of the input gesture can be identified accurately. We

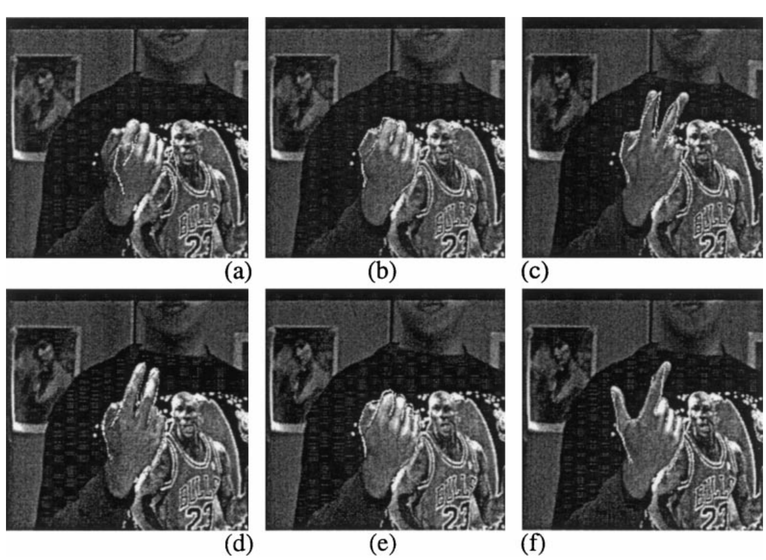

Fig. 13. The entire image sequence tracking of two-digit gesture “27”: (a) shows the initial hand shape located near the real hand in the first frame; (b)–(f) present the training set transition from model m_0 to model m_2, then return to model m_0, finally transition to model m_7.

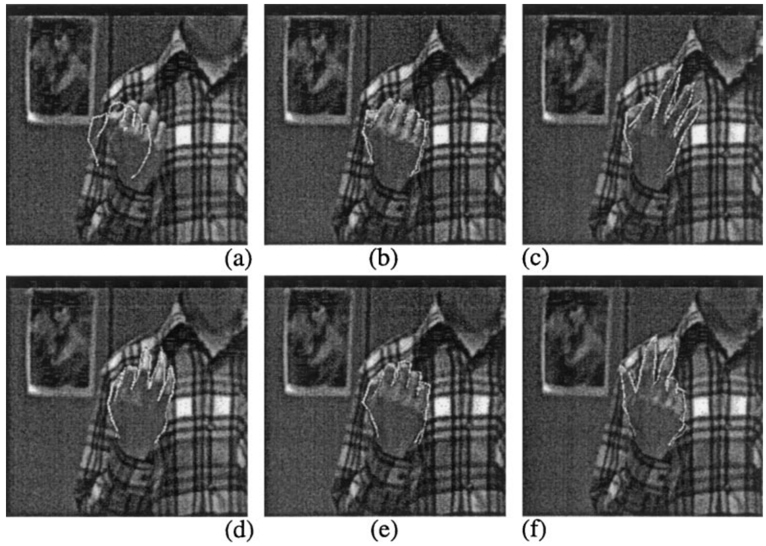

Fig. 14. The image sequence tracking of the two-digit gesture “38”: (a) shows the initial hand shape located near the real hand in the first frame; (b)–(f) present the PDM model transition from model m_0 to model m_3, then return to model m_0, finally transition to model m_8.

have made the gestures, including the single-digit gestures, two-digit gestures with/without hand translation motion. These gestures are made in front of three different complex backgrounds (i.e. Fig. 9). The feature extraction results for the gestures of single-digit number (see Fig. 9) are very accurate that makes the corresponding recognition rate the highest. Since there are fewer model transitions in the transition sequence, the selected HMMs have better chance to indicate the correct PDM model transitions, and the new PDM models can be used to extract the features more precisely.

The results for the gestures of two-digit number without translation (see Figs. 11–14), and the two-digit number with translation (see Figs. 15 and 16) are not as good as the single-digit ones (see Table 2). However, they are acceptable. On the average, the identification rate of our gesture recognition system is about 85%. The translation information provides the system a very important additional information of determining the correct PDM model transition. Therefore, the recognition rate of the one-digit (or two-digit) gestures without translation is lower than the one-digit (or two-digit) gestures with translation. The reasons for mis-identification are: (1) the pre-trained gray-level profiles stored in the database are not sufficient for coping with

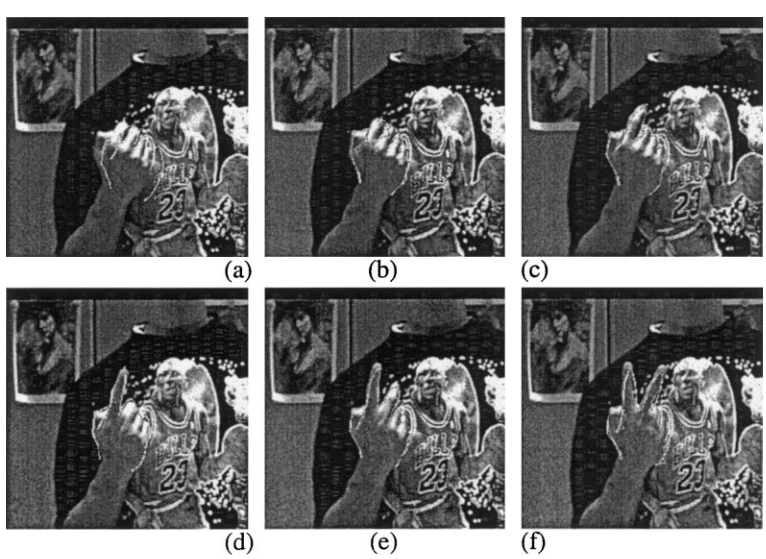

Fig. 15. The image sequence tracking of two-digit gesture “12”. It is different from Fig. 11 that the model transition does not be return to m_0 and the middle finger is straightened directly which can be described by m_2: (a) shows the initial hand shape located near the real hand in the first frame; (b)–(f) present the state transition from model m_0 to model m_1, then transition to model m_2.

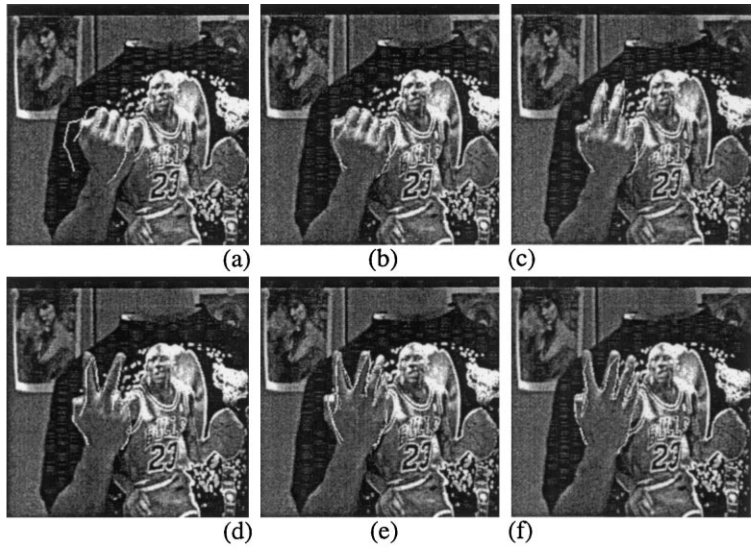

Fig. 16. The image sequence tracking of gesture-24, but it is different from Fig. 11 in that the model transition does not change from m_2 to m_0, and the middle finger is straightened directly which can be described by m_4: (a) shows the initial hand shape located near the real hand in the first frame; (b)–(f) present the training set transition from m_0 to m_2, then to m_4.

every new input gesture; (2) the number of principal components taken from the gray-level profile are not sufficient for all the unknown input gestures.

6. Conclusions

We have developed a recognition system to extract the shape feature and recognizes the gestures. Since the variation of the hands is usually large, it is necessary to have a transition between training sets for effective hand tracking and shape extraction. In the experiments, we have proved that our method is more reliable than the previous methods when dealing with the problems of recognizing gestures before non-stationary backgrounds, complex backgrounds, and similar-intensity occlusion. We may easily extend our system to recognize the gestures indicating more-than-two-digit numbers.

6. Uncited References

Author, these references are not cited in the text. Please add or delete from reference list. [16].

Table 2
The overall gesture recognition rate

Gesture types	Number of test sequences	Recognition rate (%)
Single-digit gestures with translation	120	93
Single-digit gestures without translation	120	91
Double-digit gestures with translation	120	84
Double-digit gestures without translation	120	81

References

[1] V.I. Pavlovic, R. Sharma, T.S. Huang, Visual interpretation of hand gestures for human–computer interaction: a review, IEEE Trans. PAMI 19 (7) (1997) 677–695.

[2] T. Takahashi, F. Kishino, A hand gesture recognition method and its application, Systems Computers Jpn 23 (3) (1992) 38–48.

[3] C.L. Huang, W.Y. Huang, Sign language recognition using model-based tracking and 3-D Hopfield network, Machine Vision Appl 10 (1998) 292–307.

[4] S. Tamura, S. Kawasaki, Recognition of sign language motion images, Pattern Recognition 21 (4) (1988) 343–353.

[5] J. Davis, M. Shah, Visual gesture recognition, IEE Proc.-Vis. Image Signal Process. 141 (2) (1994).

[6] C. Charayaphan, A.E. Marble, Image processing system for interpreting motion in American sign language, J. Biomed. Engng 15 (1992) 419–425.

[7] J.M. Rehg, T. Kanade, DigitEyes: vision-based hand tracking for human–computer interaction, Proc. of Int. Workshop on Automatic Face- and Gesture Recognition, Zurich, Switzerland, June 26–28, 1995.

[8] T. Darrell, A. Pentland, Space–time gestures, Proc. IEEE Conf. CVPR-93.

[9] Starner, A. Penland, Visual recognition of American sign language using hidden Markov models, Proc. of Int. Workshop on Automatic Face- and Gesture Recognition, Zurich, Switzerland, June 26–28, 1995.

[10] Y. Cui, J. Weng, Learning-based hand sign recognition, Proc. of Int. Workshop on Automatic Face- and Gesture Recognition, Zurich, Switzerland, June 26–28, 1995.

[11] E. Hunter, J. Schlenzig, R. Jain, Posture estimation in reduced-model gesture input system, Proc. of Int. Workshop on Automatic Face- and Gesture Recognition, Zurich, Switzerland, June 26–28, 1995.

[12] A.D. Wilson, A.F. Bobick, Configuration States for the Representation and Recognition of Gesture, Proc. of Int. Workshop on Automatic Face- and Gesture Recognition, Zurich, Switzerland, June 26–28, 1995.

[13] C. Maggioni, Gesture computer—new ways of operating a computer, Proc. of Int. Workshop on Automatic Face- and Gesture Recognition, Zurich, Switzerland, June 26–28, 1995.

[14] W. Freeman, C. Weissman, Television control by hand gesture, Proc. of Int. Workshop on Automatic Face- and Gesture Recognition, Zurich, Switzerland, June 26–28, 1995.
[15] R. Kjeldsen, J. Kender, Visual Hand Gesture Recognition for Window System Control, Proc. of Int. Workshop on Automatic Face- and Gesture Recognition, Zurich, Switzerland, June 26–28, 1995.
[16] T.F. Cootes, C.J. Taylor, D.H. Cooper, J. Graham, Active shape models—their training and application, Computer Vision and Image Understanding 61 (1) (1995) 38–59.
[17] A. Lantis, C.J. Taylor, T.F. Cootes, Automatic tracking, coding and reconstruction of human faces, using flexible appearance models, Elect. Lett. 30 (19) (1994) 1587–1588.
[18] T. Heap, D. Hogg, Extending the point distribution model using polar coordinate, Image Vision Comput. 14 (1996) 589–599.
[19] J.T. Tou, R.C. Gonzalez, Pattern Recognition Principal, Addison-Wesley, Reading, MA, 1981.
[20] B.H. Juang, L.R. Labiner, The segmentation K-mean algorithm for estimation parameters of Hidden Markov Model, IEEE Trans. ASSP ASSP-38 (9) (1990) 1639–1641.
[21] L.R. Rabiner, B.H. Juang, An introduction to hidden Markov models, IEEE ASSP Mag. 3 (1) (1986) 4–16.
[22] M.J. Sabin, Global convergence an empirical consistency of the generalized Llyod algorithm, PhD dissertation, Stanford University, Stanford, CA, May 1984.