ELSEVIER

Image and Vision Computing xx (0000) xxx–xxx

image AND vision COMPUTING

www.elsevier.com/locate/imavis

Hand gesture recognition using a real-time tracking method and hidden Markov models☆

Feng-Sheng Chen, Chih-Ming Fu, Chung-Lin Huang*

Institute of Electrical Engineering, National Tsing Hua University, Hsin Chu 300, Taiwan, ROC

Received 15 January 2001; received in revised form 2 January 2003; accepted 20 March 2003

Abstract

In this paper, we introduce a hand gesture recognition system to recognize continuous gesture before stationary background. The system consists of four modules: a real time hand tracking and extraction, feature extraction, hidden Markov model (HMM) training, and gesture recognition. First, we apply a real-time hand tracking and extraction algorithm to trace the moving hand and extract the hand region, then we use the Fourier descriptor (FD) to characterize spatial features and the motion analysis to characterize the temporal features. We combine the spatial and temporal features of the input image sequence as our feature vector. After having extracted the feature vectors, we apply HMMs to recognize the input gesture. The gesture to be recognized is separately scored against different HMMs. The model with the highest score indicates the corresponding gesture. In the experiments, we have tested our system to recognize 20 different gestures, and the recognizing rate is above 90%.

Keywords: Hand gesture recognition; Hidden Markov model; Hand tracking

1. Introduction

Hand gesture has been one of the most common and natural communication media among human being. Hand gesture recognition research has gained a lot of attentions because of its applications for interactive human-machine interface and virtual environments. Most of the recent works related to hand gesture interface techniques [1] has been categorized as: glove-based method [2,3] and vision-based method. Glove-based gesture interfaces require the user to wear a cumbersome device, and generally carry a load of cables that connect the device to a computer. There are many vision-based techniques, such as model-based [4] and state-based [5] which have been proposed for locating objects and recognizing gesturers. Recently, there have been an increasing number of gesture recognition researches using vision-based methods Table 1.

Huang et al. [6] use 3D neural network method to develop a Taiwanese Sign Language(TSL) recognition system to recognize 15 different gestures. David and Shah [7] propose 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. Darrell and Pentland [8] propose space-time gesture recognition method. Signs are represented by using sets of view models, and then are matched to stored gesture patterns using dynamic time warping.

Starner et al. [9] describe an extensible system which uses one color camera to track hands in real time and interprets American sign language (ASL). They use hidden Markov models (HMMs) to recognize a full sentence and demonstrate the feasibility of recognizing a series of complicated series of gesture. Instead of using instrumented glove, they use vision-based approach to capture the hand shape, orientation and trajectory. The vision-based method selects the 3-D input data as the feature vectors for the HMM input, other HMM-based [10,11] hand gesture recognition systems have also been development. Liang et al. [12] develop a gesture recognition of TSL by using Data-Glove to capture the flexion of 10 finger joints, the roll of palm and other 3-D motion information Table 2.

Cui and Weng [13] develop a non-HMM-based system which can recognize 28 different gestures in front of

☆ This manuscript is submitted to image and vision computing.

* Corresponding author.
E-mail address: cllhuang@ee.nthu.edu.tw (C.-L. Huang).

0262-8856/03/$ - see front matter © 2003 Published by Elsevier Science B.V.
doi:10.1016/S0262-8856(03)00070-2

Table 1
The error rate of the gesture recognition system using only FD

Gesture	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20
Error (%)	6	1	2	12	15	6	12	10	19	5	12	17	1	0	11	3	23	25	0	0

complex backgrounds. The recognition of this system is 93.1% but it relies on a slowly segmentation scheme which takes 58.3 sec for each image. Nishikawa et al. [14] propose a new technique for description and recognition of human gestures. The proposed method is based on the rate of change of gesture motion direction that is estimated using optical flow from monocular motion images.

Nagaya et al. [15] propose a method to recognize gestures using an approximate shape of gesture trajectories in a pattern space defined by the inner-product between patterns on continuous frame images. Heap and Hogg [16] present a method for tracking of a hand using a deformable model, which also works in the presence of complex backgrounds. The deformable model describes one hand posture and certain variations of it and is not aimed at recognizing different postures. Zhu and Yuille [17] develop a statistical framework using principal component analysis and stochastic shape grammars to represent and recognize the shapes of animated objects. It is called flexible object recognition and modeling system (FORMS). Lockton et al. [18] propose a real-time gesture recognition system which can recognize 46 ASL letter spelling alphabet and digits. The gestures that are recognized by [18] are 'static gestures' of which the hand gestures do not move.

Different from [18], this paper introduces a hand gesture recognition system to recognize 'dynamic gesture' of which a gesture in performed singly in complex background. Different from previous HMM-based gesture recognition systems, our system do not use instrumented glove, nor any markers, but use 2D video input. Our system tracks the moving hand and analyzes the hand-shape variation and motion information as the input to the HMM-based recognition system. The system consists of three modules: a real-time hand tracking, feature extraction, HMM training, and HMM-based gesture recognition. First, we introduce a real time hand gesture tracking technique which can track the moving hand and then extract the hand shape from complex background. It is a simple and reliable method developed as a real-time image processing subsystem which consists of five basic complementary image processes: motion detection, skin color extraction, edge detection, movement justification, and background subtraction.

We apply the FD to characterize the spatial information and the optical flow method for motion analysis to characterize the temporal information. We combine FD and motion information of the input image sequence as our feature vector. With these extracted feature vectors, we can train our system using HMM approach which is used to recognize the input gesture. In training phase, we apply HMM to describe the gestures in term of model parameters for each different gesture. The gesture to be recognized in separately scored against different HMMs. The model with the highest score is selected as the recognized gesture. Our system consists of 20 different HMMs which are used to test 20 different hand gestures. The experimental results show that the average recognition rate is above 90%.

Fig. 1 shows the flow diagram of our hand gesture recognition system consisting of three phases: the feature extraction phase, the training phase, and the recognition

Table 2
The recognition rate of one-hand gesture

Two Methods	Training data (%) (1200 sequences)	Testing data (%) (1200 sequences)
FD only	97	90.5
FD and motion	98.5	93.5

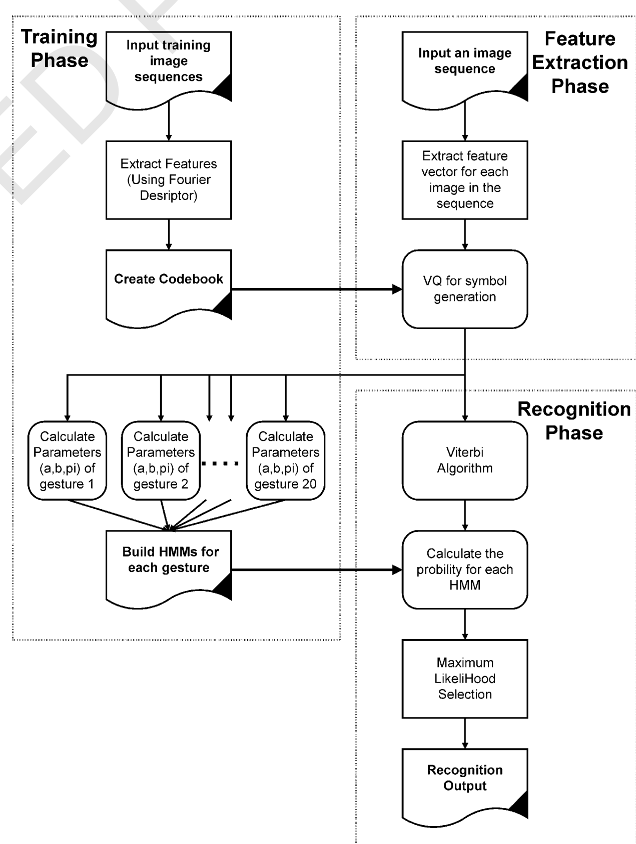


Fig. 1. The flow diagram of hand gesture recognition system.

Fig. 2. (a) The origin frame, (b) apply our threshold, (c) apply Ostu thresholding.

phase. We combine FD and motion features as the feature vector to describe the moving object. Each feature vector is represented by a symbol. Each symbol corresponds to the designated partition generated through the vector quantization of the feature vectors of all possible hand-shapes of the training gestures. For each feature vector, a symbol is assigned. In our system, we represent the input image sequence by a sequence of symbols. In training phase, we need to build a HMM for each gesture. In the recognition phase, a given input gesture is tested by every HMM with different model parameters. The outcome of the HMM with the maximum likelihood function is identified to recognize the gesture.

2. Hand tracking and handshape extraction

Here, we develop a real-time hand tracking method which is robust and reliable in complex background. To track the moving hand and then extract the hand shape fast and accurately, we need to consider the trade-off between the computation complexity and robustness.

2.1. Feature extraction

In our system, the motion of the object provides important and useful information for object localization and extraction. To find the movement information, we assume that the input gesture is non-stationary. When objects move in the spatial-time space (an image sequence), motion detector is able to track the moving objects by examining the local gray-level changes. Let $F_i(x,y)$ be the ith frame of the sequence and $D_i(x,y)$ be the difference image between the ith and the $(i+1)$th frame defined as

$$D_i(x,y) = T_i\{|F_i(x,y) - F_{i+1}(x,y)|\} \tag{1}$$

where T_i is a thresholding function, $F_i(x,y)$ and $D_i(x,y)$ are all 160×120 images, and $D_i(x,y)$ is binary image defined as follows

$$D_i(x,y) = \begin{cases} 1, & |F_i(x,y) - F_{i+1}(x,y)| \geq \text{threshold} \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

(1) *Thresholding*. Having extracted the moving object region, we can apply the thresholding on the frame difference (i.e. Eq. (2)) to extract the possible moving region in complex background. We find that conventional thresholding methods, such as Ostu thresholding [27], are not suitable for the case of detecting motion difference. Instead, we use a simple thresholding technique to extract moving regions. The threshold for motion detection is determined as $t_{\mathrm{M}} = 0.2\mu$, where μ is the average luminous of captured image $F_i(x,y)$. Fig. 2 shows that if there is no significant movement, Ostu thresholding method will generate a lot of noise. We choose the weighting factor 0.2 because we do not need highly precise segmented image. Our thresholding technique is not very sensitive to the speed of the hand movement, so that our method more stable than the Ostu method.

Fig. 3. (a) The origin frame, (b) extracted skin regions satisfying R > G > B, and (c) compare the colors of the extracted skin regions with the sample skin color.

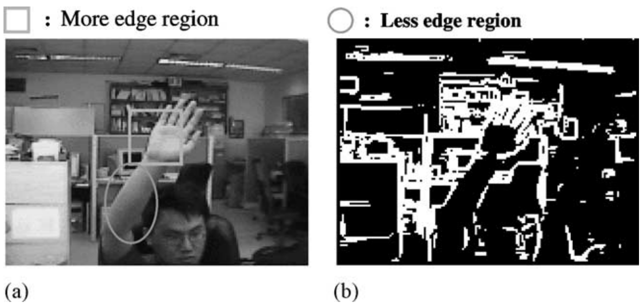


Fig. 4. (a) The origin frame, (b) the edge detection result.

(2) *Skin color detection*. Skin can be easily detected by using the color information. First, we use the constraint, i.e. R > G > B, to find the skin color regions which may include a wide range of colors, such as red, pink, brown, and orange color. Therefore, we will find many regions other than the skin regions. However, those non-skin regions satisfy our constraint will be excluded due to there is no motion information, e.g. a region in orange color will not be misidentified as the hand region. Second, we may obtain some sample colors from the hand region. To find the skin regions, we compare the colors in the regions with the pre-stored sample color. If they are similar, then the region must be skin region. The hand region is obtained by the hand tracking process in the previous frame. Fig. 3 shows our skin detection results. The rectangular region is the hand region in the previous frame. Finally, we may eliminate some skin-similar colors, e.g. the orange color, and denote the skin color image as $S_i(x,y)$.

(3) *Edge detection*. Edge detection is applied to separate the arm region from the hand region. It is easy to find that there are fewer edges on the arm region than on the palm

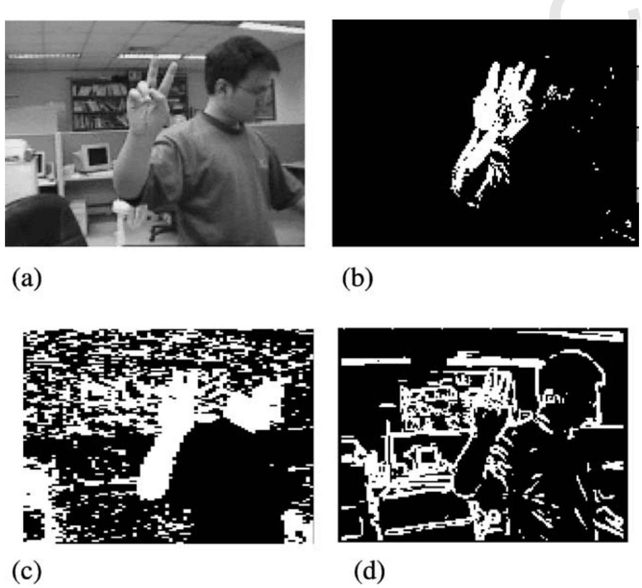


Fig. 5. The hand gesture information. (a) Original image $F_i(x,y)$, (b) motion region $D_iS(x,y)$, (c) skin color region $S_i(x,y)$, (d) edge region $E_i(x,y)$.

Fig. 6. The combined region $C_i(x,y)$.

region. Here, we use a simple edge detection technique (e.g. Kirsch edge operator) to obtain different direction edges, and then choose the absolute maximum value of each pixel to form the edge image of *i*th frame as $E_i(x,y)$. Fig. 4 shows that the edges on the arm region are less than those on the palm region. We combine edge, motion, skin color region information to allocate the hand region.

(4) *Combination of motion, skin color, and edge*. The hand gestures information consists of movement, skin color and edge feature. We use the logic 'AND' to combine these three types of information, that is

$$C_i(x,y) = D_i(x,y) \wedge S_i(x,y) \wedge E_i(x,y) \tag{3}$$

where $D_i(x,y)$, $S_i(x,y)$ and $E_i(x,y)$ indicate the movement, skin color and edge images. The combined image $C_i(x,y)$ as many features that can be extracted. Because the different image processing methods have extracted different kind of information. Each image consists of different characteristic regions such as motion regions, skin color regions and edge regions as shown in Fig. 5. Fig. 6 shows the combined region $C_i(x,y)$. The combined image consists of a large region in the palm area and some small regions in the arm area. We may separate these two regions to allocate the hand region.

(5) *Region identification*. A simple method for region identification is to label each region with a unique integer number which is called the labeling process. After labeling, the largest integer label indicates the number of regions in the image. After the labeling process, the small regions can be treated as noise and then be removed. Fig. 7(a) shows that the labeling results and Fig. 7(b) shows the center position $p_c(i)$ of the hand region. We use $L_i(x,y)$ to indicate the largest labeled region in Frame *i*.

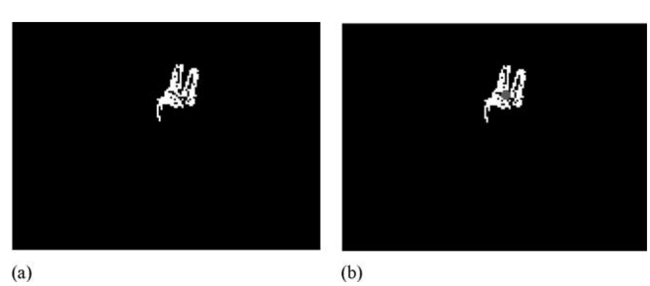


Fig. 7. (a) The labeling results $L_i(x,y)$, (b) the correct center position.

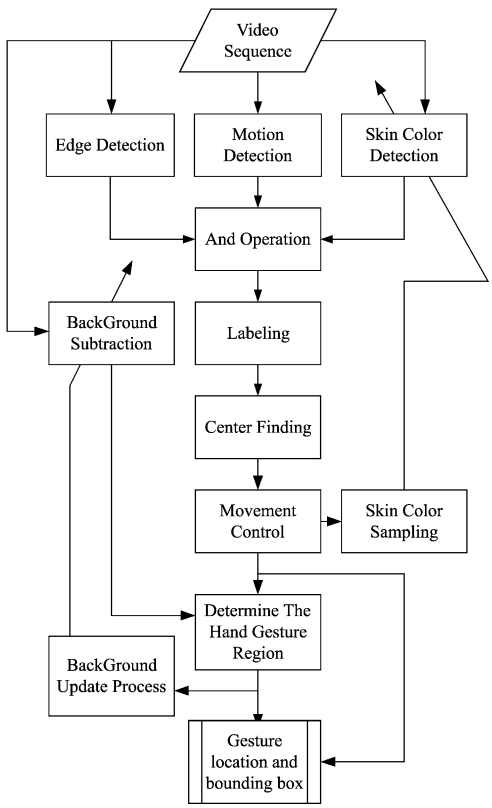


Fig. 8. The flow diagram of hand gesture tracking system.

2.2. Robustness and low complexity

Using motion and color information is not sufficient, and hand-shape is not always the largest labeled region. If there are other skin-color objects moving rapidly, the tracking process may fail. We need to take advantage of the motion smoothness constraint for trajectory justification, then use background subtraction to find the foreground object, and finally identify the hand region.

2.2.1. Hand gesture trajectory justification

Based on the assumption that the hand object move smoothly between two connected frames, we develop a trajectory justification algorithm. We assume that the movement of the hand is in a constant speed. For current frame F_i, we get the center point $p_C(i)$ of the extracted hand region. We assume smooth trajectory so that the variation of $p_C(i)$ is constrained in a certain range. If the variation of $p_C(i)$ is out of a certain range (i.e. $|p_C(i) - p_R(i-1)| > \delta$), we increase the wrong (or *bumpy*) position counter, i.e. $\mathrm{WC} = \mathrm{WC} + 1$, else we set $p_R(i) = p_C(i)$. To avoid the trajectory of $p_C(i)$ being bumpy for while, we check if $\mathrm{WC} > 3$. If it is not, then the hand gesture is suppose to be at a right position, and we may set $p_R(i) = p_R(i-1)$. If $\mathrm{WC} > 3$, then the hand gesture may be identified at a wrong position, therefore, we change the right position $p_R(i) = p_C(i)$, reset $\mathrm{WC} = 0$, and go to next frame F_{i+1}.

2.2.2. Processing ROI

Fig. 8 shows the flow diagram of our hand gesture tracking system. In previous section, we have mentioned how to generate five image frames: $D_i(x,y)$, $S_i(x,y)$, $E_i(x,y)$, $C_i(x,y)$ and $L_i(x,y)$. The three function blocks indicate motion detection, edge detection, skin color detection, which can operate in parallel. To reduce the computation complexity, we do not process the entire image frame but concentrate on the region of interest (ROI). For instance, one ROI is a part of $F_i(x,y)$, where the corresponding $D_i(x,y) \neq 0$. We deal with the first ROI to obtain $S_i(x,y)$. The other ROI is also part of the $F_i(x,y)$, where the corresponding $S_i(x,y) \neq 0$. Similarly, we process the second ROI to obtain $E_i(x,y)$. Fig. 9 shows the step-by-step processing of motion detection, skin color detection, and edge detection. We can dramatically reduce the computation complexity of our system.

2.2.3. Background subtraction

For gesture recognition process, we need more hand gesture information. We use a simple background subtraction technique to obtain the hand gesture shape. We create the background model BG_i by using the first frame $F_1(x,y)$. Fig. 10 shows the foreground region, and Fig. 11 shows the procedure to obtain the foreground.

To update our background model, we adapt our background model by using current frame F_i and foreground region FG_i. We have generated two different types of foreground regions, one is $FG1_i = FG_i$, which is used to obtain the hand gesture region; and the other is $FG2_i$, ($FG2_i$ is obtained by dilating $FG1_i$), which is applied for background updating process. $FG1_i$ has a compact shape, so that it can be used to obtain the hand region. Because there are small errors on the boundary of foreground and background, we do not use $FG1_i$ to update the background.

(a) $D_i(x, y)$. (b) $S_i(x, y)$. (c) $E_i(x, y)$.

Fig. 9. The three function blocks: (a) motion detection, (b) skin color detection, (c) edge detection.

(a) original image (b) foreground image

Fig. 10. The result of background subtraction.

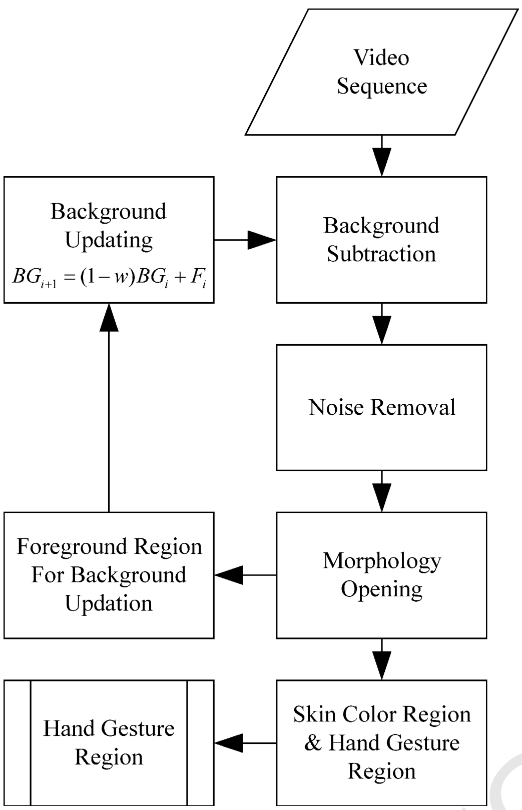


Fig. 11. Background subtraction process.

We generate $FG2_i$ for background updating. We only update the background region where $FG2_i \neq 0$. Fig. 12 shows the difference of these foreground regions. The background update equation is

$$BG_{i+1} = (1-w)BG_i + wF_i \tag{4}$$

We update background gradually, and the weighting factor w is 0.1. The updating process is more reliable for a smaller w. Finally, we have the foreground region which does not really indicate the human hand. We need to apply the skin color analysis and the hand region position tracking to correctly extract the hand region. Fig. 13 shows the results of hand gesture region extraction process.

2.2.4. Local tracking of the hand gesture region

To find a more precise hand region, we use the foreground region information. The hand position has been found by using motion, skin color and edge information. Sometime, the identified locations will not at the center of the real hand region. This is because the extracted information are located on the boundary of the moving object. Therefore, the local refinement is necessary. The overall system for hand region tracking has two stages: the first stage is focus on the motion information, whereas the second stage is focus on the foreground information. The local tracking processing is mentioned as follows: (a) select the foreground and skin color region near the inaccurate center; (b) select the boundary points in the foreground region; and (c) find the center of the boundary points as a new center. We may formulate the process as

$$p_{C2}(i) = T_C\{T_R(p_C(i), FG_i \wedge E_i \wedge S_i)\} \tag{5}$$

Where $p_{C2}(i)$ is the new center at the second stage. $T_C\{\bullet\}$ is a center finding operator, and $T_R(A,B)$ is an operator to find a region in B that is near the point A. Fig. 14 shows the difference between those two stages.

After refining the hand gesture center point, we may find the bounding box of hand region. We find the bounding box by using the foreground, the skin color information, and the center point located in the second stage. We search the boundary of hand region from center to top, bottom, left, and right. We use four parameters to describe the width and the height of the extracted hand region, e.g. LW,RW,TH, and BH shown in Fig. 15(a).

Since the arm region is not the target region, we develop a simple criterion to obtain a more precise hand region. The bounding box is determined by the following criteria: (1) If RW $>$ LW then RW $=$ 1.1LW else LW $=$ 1.1RW; and (2) If TH $>$ BH then TH $=$ 1.1BH else BH $=$ 1.1TH. In the Fig. 15(a), the length TH is shorter than BH, we let BH $=$ 1.1TH, and similarly RW $=$ 1.1LW. Fig. 15(b)

(a) (b) (c)

Fig. 12. Different type of foreground: (a) original image, (b) foreground FG_1 for gesture tracking, (c) foreground FG_2 for updating the background.

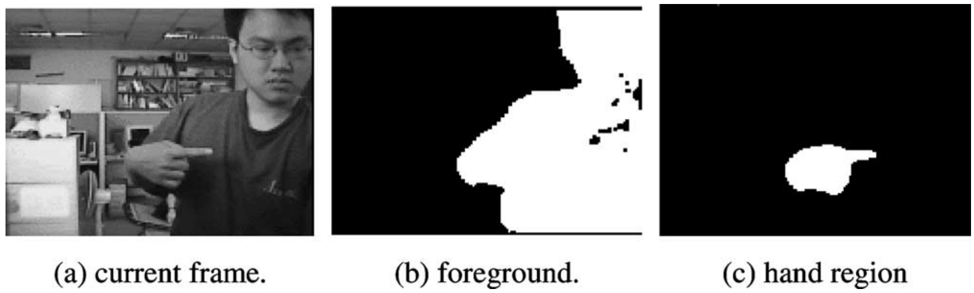


Fig. 13. Foreground region combining skin color and hand gesture position.

shows the updated bounding box. The new bounding box does not include the arm region. The method is effective for the following gesture recognition process.

2.3. Illustrations and discussion

Here, we illustrate some experimental results of the hand tracking process. The tracked hand-shape includes different types of gestures and the gestures made by different persons. We assume no camera panning nor zooming, and there is only one hand needs to be tracked. We allow the other moving objects in the background, but there is only one moving hand in the foreground. For each frame of the video sequence, the bounding box tracked automatically is compared to a bounding box selected manually to measure the error in width w and height h. There are two performance measurements; one is the gesture location missed percentage

$$e_{\mathrm{m}} = \frac{\mathrm{numberoftimesthecentersarenotlocated}}{\mathrm{totalframenumber}} \tag{6}$$

The other is the normalize error in bounding box size defined as below

$$e_{\mathrm{w}}(i) = 1 - \frac{w(i)}{w_a(i)} \quad \text{and} \quad e_h(i) = 1 - \frac{h(i)}{h_a(i)} \tag{7}$$

where $w(i)$ and $h(i)$ are the correct dimensions of the bounding box of ith frame selected manually, and $w_a(i) = \mathrm{LW}(i) + \mathrm{RW}(i), h_a(i) = \mathrm{TH}(i) + \mathrm{BH}(i)$ are the dimensions of the bounding box of ith frame selected by our system. The image size is 160×120. Fig. 16 shows some input image sequences and the extracted hand shapes. In the experiments, we have tested about 200 video sequences, and the accuracy percentage is measured in terms of e_m, e_w and e_h which are shown in Fig. 17.

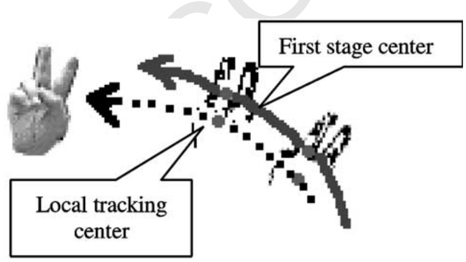


Fig. 14. Difference between the first stage center and the local tracking center. The solid line is the trajectory of the first stage center, and dotted line is the trajectory of the second stage center.

3. Feature selection for object description

Features are obtained from the input image sequence of hand gestures, they are further converted to symbols which are the basic elements of the HMM. Effective and accurate feature vectors play a crucial role in model generation of the HMM. For selecting good features, the following criteria are considered useful: (1) Features should be preferably independent on rotation, translation and scaling. (2) Features should be easily computable. (3) Features should be chosen so that they do not replicate each other. This criterion ensures efficient utilization of information content of the feature vector. The features obtainable from the image

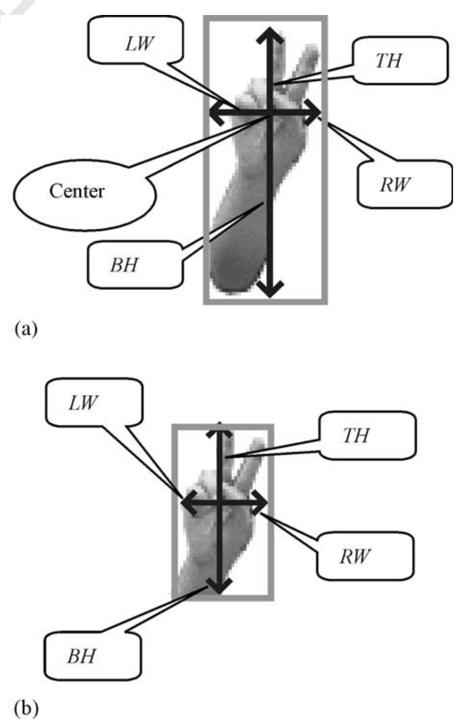


Fig. 15. (a) Four parameter of hand gesture bounding box, (b) new hand gesture bounding box.

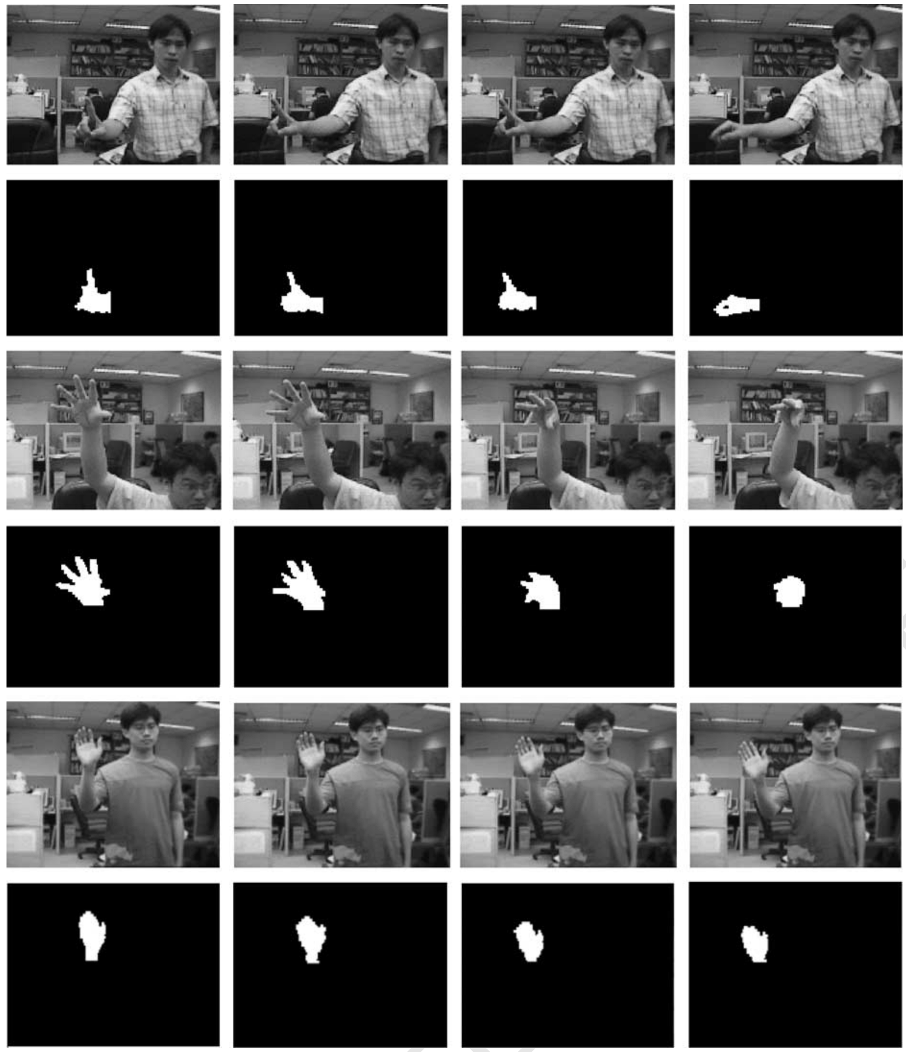


Fig. 16. Some image sequence in our database and processing result.

sequence of hand gesture are spatial and temporal features. To extract the shape features, we choose the FD to describe the hand shape, and to extract the temporal features, we use motion analysis to obtain the non-rigid motion characteristics of the gesture. These features should be invariant to the small hand shape and trajectory variations and it is also tolerant to small different gesture-speed.

3.1. Fourier descriptor

We may describe the objects by their features in the frequency domain, rather than those in the spatial domain. The local feature property of the node is represented by its Fourier Descriptors (FD) [19,20]. Assume the hand-shape is described by external boundary points, $\{x(m), y(m)\}$, then we may use the FD representation for boundary description. To extract the external boundary points of a hand shape, we may use the contour following algorithm. To represent the boundary points, we may find the Fourier series of $x(m)$ and $y(m)$, which are defined as $a(n)$ and $b(n)$. For a closed boundary, this representation is called FD. The elements of the vector are derived as $S(n) = r(n)/r(1)$ where, $r(n) = [(a(n))^2 + (b(n))^2]^{1/2}, n = 1, 2, \ldots$ Using of FD vectors of dimension 10 for hand written digit recognition is sufficient [20]. Here we assume that the local variation of hand-shape is smooth so that the high order terms of its FD are not necessary, so using 22 harmonics of the FD's is enough to describe the macroscopic information of the hand figures.

The advantage of using the FD is due to its size-invariant properties. For different scaled objects, only the magnitudes of their FD coefficients are changed by the same factor. Furthermore, from Fig. 18, we may find that rotating the object only causes a phase change. The magnitude $S(n)$ is independent of the phase, and it is unaffected by rotation. If the magnitude of the FD coefficients is normalized, the FD representation is invariant to object size. Finally, we consider the effect of noise and quantization errors on the boundary. This will cause local variation of high frequency,

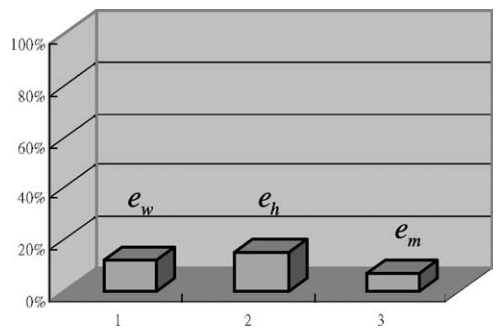


Fig. 17. The percentage error in bounding box size and location loss rate.

and it will not change to low frequencies. Hence, if the high frequency components of the spectrum are ignored, the rest of the spectrum is unaffected by noise.

3.2. Motion analysis

In the image sequence of hand gesture, there are local motion and global motion. The global motion is the translation of the hand and the local motion is non-rigid motion of the fingers or rotation of the hands. Therefore, we need to estimate the entire motion field of the two consecutive image frames. The motion estimation is based on the space-temporal image intensity gradients called the optical flow Equation [21]. The optical flow equation is developed in conjunction with an appropriate space-temporal smoothness constraint, which requires that the neighboring displacement vectors very smooth. The magnitude and phase of the motion vector field indicates the speed and moving direction of the moving object. The histogram distribution of the magnitude and phase of the motion vector field are extracted as the motion features.

Here, we partition the magnitude distribution of the motion vectors into ten intervals. Let $P_e(i)$ denote the number of motion vectors belonging to magnitude interval i, then we have $f_e(i) = P_e(i)/$ (total pixels of a frame), where $1 \le i \le 10$ and $f_e(i)$ denotes the features extracted from the magnitude of the motion vector. We also partition the phase distribution of the motion vector field into 8 intervals. Let $P_p(i)$ denote the number motion vectors belong direction interval I, we have $f_p(i) = P_p(i)/$(total pixels of a frame), where $1 \le i \le 8$ and $f_p(i)$ denote the features extracted from the direction of the motion vector.

From motion analyzing of two consecutive image frames, we find that the motion vectors are pointing in different directions. The motion of different fingers creates the motion vectors in various directions. From the phase histogram, we find that the peaks of some intervals represent the major directions of different local motions. From the magnitude histogram, we can also find the only peak of the first interval that indicates the global motion (see Fig. 19).

Besides the above analysis of the motion direction and magnitude distribution, we can find other features related to the distribution of the motion vector field. We can assume that the motion vector field is an intensity of motion in 2D space $X = \{(x, y)\}$. We apply the vector field $v(x, y, t)$ to characterize the motion distribution. A suitable feature for the characterization of the motion distribution at time instance t is the center of gravity $\vec{m}(t)^T = [mx(t), my(t)]$ as

$$mx(t) = \frac{\sum_{x,y} x.v(x,y,t)}{\sum_{x,y} v(x,y,t)} \qquad my(t) = \frac{\sum_{x,y} y.v(x,y,t)}{\sum_{x,y} v(x,y,t)} \tag{8}$$

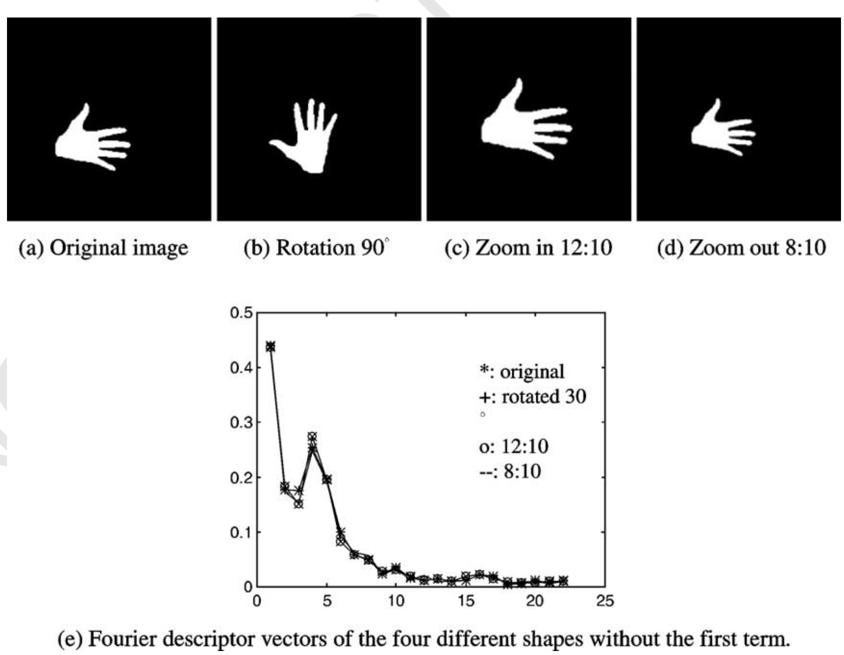


Fig. 18. Illustration the invariant properties of Fourier descriptor.

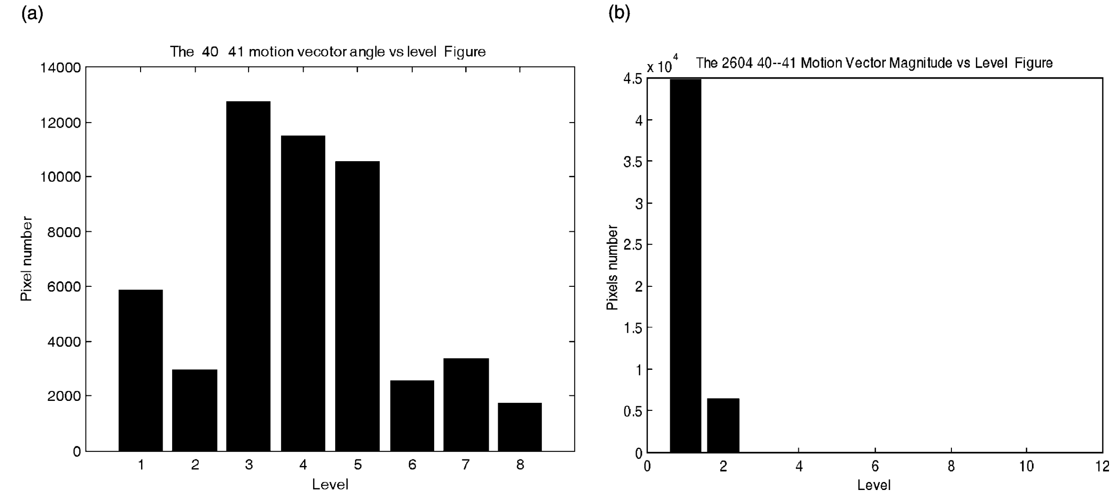


Fig. 19. (a) Phase distribution and (b) magnitude distribution of the motion vectors.

The vector $\vec{m}(t)$ can also be interpreted as the 'center of motion' of the image. $v(x,y,t)$ denotes the magnitude of the motion vector at position (x,y) at time t. To increase the modeling capacity of the HMMs for the movements of the center, we include the delta features $\Delta\vec{m}(t)$ of $\vec{m}(t)$ for the 'center of motion' into the feature vector. The delta features are defined as $\Delta mx(t) = mx(t) - mx(t-1)$ and $\Delta my(t) = my(t) - my(t-1)$. Another useful feature is the average absolute deviation of the motion in all points of the images from the center of motion $\sigma x(t)^T = [\sigma x(t), \sigma y(t)]$, which is defined as

$$\sigma x(t) = \frac{\sum_{x,y} v(x,y,t)|x - mx(t)|}{\sum_{x,y} v(x,y,t)} \quad \sigma y(t) = \frac{\sum_{x,y} v(x,y,t)|y - m_y(t)|}{\sum_{x,y} v(x,y,t)} \tag{9}$$

This feature is very similar to the second translation invariant moment of the distribution, but it is more robust against noise in the image sequence. It can also be considered as 'wideness of the movement'. In motion analysis, we have created 24 features, they are 10 motion magnitude features, 8 motion direction features, $m_x, m_y, \Delta m_x, \Delta m_y, \sigma_x$, and σ_y.

4. Gesture recognition using HMMs

HMMs have been widely and successfully used in speech recognition and handwriting recognition [22]. Consequently, they seem to be effective for visual recognition of complex, structured hand gestures such as sign language recognition [23,24]. A HMM can be employed to represent the statistical behavior of an observable symbol sequence in terms of a network of states. For each observable symbol, it can be modeled as one of the states of the HMM, and then the HMM either stays in the same state or moves to another state based on a set of state transition probability associated with the state. The variety of the observable symbols for which the HMM uses a particular state is described in terms of the distribution of probability that each observable symbol will occur from that state. Thus, an HMM is a doubly (observable and hidden) stochastic model where the observable symbol probability distribution for each state captures the intra-state variability of the observable symbols, and the state transition probability describe the underling dynamic structure of the observable symbols.

We use HMMs to recognize different gestures because of their simplicity and reliability. The HMM uses only three parameters: the initial state probability vector, the state-transition probability matrix, and the observable symbol probability matrix. Analysis of dynamic images naturally will yield more accurate recognition than that of a single static image. Gestures are recognized in the context of entire image sequences of non-constant lengths. Using an HMM for gesture recognition is advantageous because it is analogous to human performance which is a doubly stochastic process, involving a hidden immeasurable human mental state and a measurable, observable human action.

4.1. Vector quantization for symbol generation

To model various gesture expressions, we train different HMMs to model different hand gestures. First, we must convert multi-dimensional vector sequences to one-dimensional symbol sequences. The preprocessing algorithm is the vector quantization (VQ) [25,26]. In an HMM-based approach, we need to quantize each multi-dimensional feature vector sequence into a finite symbol sequence for HMMs. The purpose of designing an M-level VQ (called a codebook with size M) is to partition all k-dimensional training feature vectors into M clusters, whose centroid is the k-dimensional vector c^i, with a quantized value named codeword (symbol) o^i. VQ will cause a quantization error between each training feature vector x and c^i. As the size of the codebook increases, the quantization error decreases, however, the required storage for the codebook entries increases. There is a trade-off to define the size of the codebook.

To have a good recognition performance in using HMMs, we design a codebook for vector quantizing each k-dimensional training feature vector x into a symbol o^i with minimum quantization error. According to our experimental result, the recognition system has high performance when the size $M = 64$ of the codebook. This VQ algorithm uses iterative method, splits the training vectors from assuming whole data to be one cluster to $2, 4, 8, \ldots, M(M = 2^n)$ clusters, and determines the centroid for each cluster. The centroid of each cluster is refined iteratively by k-means clustering. Once the final codebook is obtained, it is used to quantize each training and testing feature vector into a symbol. A symbol is assigned to each partition of the k-dimensional VQ space. The symbol generation process is illustrated in Fig. 20.

4.2. Hidden Markov models

In the Markov model, the state sequence is observable. The output observable event in any given state is deterministic, not random. This will be too constraining when we use it to model the stochastic nature of the human performance, which is related to doubly stochastic processes, namely human mental states (hidden) and human actions (observable). It is necessary that the observable event is a probabilistic function of the state. HMM is a representation of a Markov process and is a doubly embedded stochastic process with an underlying stochastic process that cannot be directly observed, but can only be observed through another set of stochastic processes that produce the sequence of observable symbols.

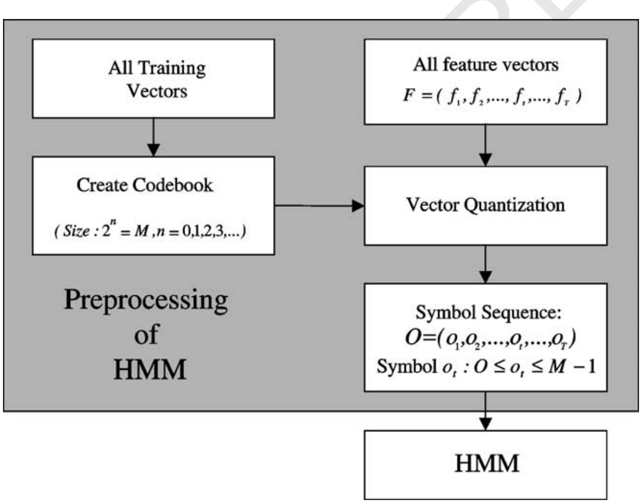


Fig. 20. Preprocessing of the hand gesture recognition system.

We define the elements of an HMM as follows. N is the number of states in the model. The state of the model at time t is q_t, $1 \leqq q_t \leqq N$ and $1 \leqq_t \leqq T$ where T is the length of the output observable symbol sequence. M is the size of the codebook or the number of distinct observable symbols per state. Assume o_t is one of all possible observable symbols for each state at time t, then $0 \leqq o_t \leqq M - 1$ π_N is an N-element vector indicates the initial state probability. $\pi_N = \{\pi_i\}$, where $\pi_i = P(q_t = i), 1 \leqq i \leqq N$. $A_{N\times N}$ is an $N \times N$ matrix specifying the state-transition probability that the state will transit from state i to state j. $A_{N\times N} = \{a_{ij}\}$ where $\times$ $a_{ij} = P(q_t = j|q_{t-1} = i), 1 \leqq i, j \leqq N$ and $a \geqq 0, \sum_{j=1}^{N} \times a_{ij} = 1$. $B_{M\times N}$ is an $M \times N$ matrix specifying that the system will generate the observable symbol o_t at state j and at time t. $B_{M\times N} = \{b_j(o_t)\}$ where $b_j(o_t) = P(O_t = o_t|q_t = j), 1 \leqq i \leqq N, 0 \leqq o_t \leqq M - 1, b_j(o_i) \geqq 0$, and $\sum_{o_t=0}^{M-1} b_j(o_t) = 1$.

The complete parameter set λ of the discrete HMM is represented by one vector π and two matrices A and B. To accurately describe a real-world process such as gesture with an HMM, we need to appropriately select the HMM parameters. The parameter selection process is called the HMM 'training.' This parameter set λ can be used to evaluate the probability $P(O|\lambda)$, that is to measure the maximum likelihood performance of an output observable symbol sequence O, where T is the number of frames for each image sequence. For evaluating each $P(O|\lambda)$, we need to select the number of states N, the number of observable symbols M (the size of codebook), and then compute the results of probability density vector π and matrices A and B by training each HMM from a set of corresponding training data after VQ.

There are three basic problems in HMM design: (1) Probability evaluation: How do we efficiently evaluate $P(O|\lambda)$, the probability (or likelihood) of an output observable symbol sequence O given an HMM parameter set λ. (2) Optimal state sequence. How do we determine an optimal state sequence $q = \{q_1, q_2, \ldots q_T\}$, which is associated with the given output observable symbol sequence O, by given an HMM parameter set λ. (3) Parameter Estimation. How do we regulate an HMM parameter set λ to maximize the output probability $P(O|\lambda)$ of generating the output observable symbol sequence.

(1) *Probability evaluation using the forward-backward procedure*. We compute the output probability $P(O|\lambda)$ with which the HMM will generate an output observable symbol sequence $O = \{o_1, o_2, \ldots o_T\}$ given the parameter set $\lambda = (\pi, A, B)$. The most straightforward way to compute this is by enumerating every possible state sequence of length T, so there will be N^T possible combinations of state sequence where N is the total number of states. Suppose there is one state sequence $q = \{q_1, q_2, \ldots q_T\}$. Fortunately, we can use a more efficient procedure called the Forward-Backward procedure [29] to overcome this limitation.

(2) *Optimal state sequence using the viterbi algorithm*. We use a dynamic programming method called the Viterbi algorithm [28] to find the single best state sequence

$q = (q1, q2, \ldots qT)$ (or the most likely path) given the observable symbol sequence $O = (o_1, o_2, \ldots o_T)$ and the HMM parameter set λ in order to maximize $P(q|O, \lambda)$. Since

$$P(q|O,\lambda) = \frac{P(q,O|\lambda)}{P(O|\lambda)} \tag{10}$$

Maximizing $P(q|O, \lambda)$ is equivalent to maximizing $P(q, O|\lambda)$ using the Viterbi algorithm.

(3) *Parameter estimation using the baum-welch method.* We can use a set of training observable symbol sequences to adjust the model parameters in order to build a signal model that can be used to identify or recognize other sequences of observable symbols. There is, however, no efficient way to optimize the model parameter set that globally maximizes the probability of the symbol sequence. Therefore, the Baum-Welch method [29] is used to choose the maximum likelihood model parameter set $\lambda = (\pi, A, B)$ such that its likelihood function $P(O|\lambda)$ is locally maximized using an iterative procedure.

5. Experimental results

In the experiments, the subject, who uses a single hand to make hand gesture, is standing before any stationary background with normal lighting. The proposed real-time tracking system can track and identify the moving objects in front of a stationary background. We may allow some small objects moving in the background which will not be extracted and mistreated as a moving hand. We have tested twenty different hand gestures selected from TSL. Each hand gesture consists of a sequence of image frames capturing a single hand moving in different directions with constant or time-varying hand shape.

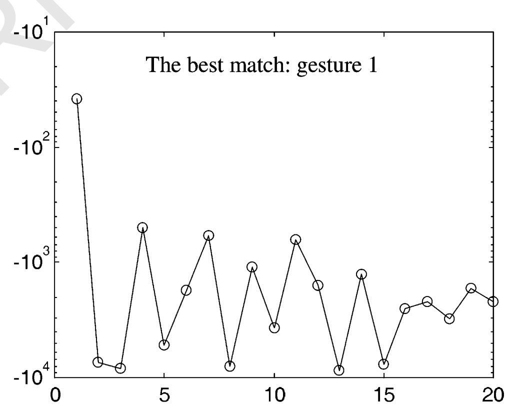
Each hand gesture is performed 3 times by 20 different individuals. There are 60 different image sequences captured for each hand gesture. There are twenty different gestures, and 1200 image sequences are used for training. The size of each gray-level image frame is 256×256, its frame rate is 30 frames/sec, and each gesture-making takes about one second. The input image sequence is divided into three different time intervals: in the first (begin) period, the sign language speaker remains silent (no gesture), then in the second (action) period, the speaker starts making one simple hand gesture, and finally, in the last (end) period, the speaker remains silent again.

In the experiments, six gestures have constant hand shape, whereas fourteen gestures have time-varying hand shape. They may have similar or different moving trajectories. The simply single hand gestures can be completed in less than one second. The host computer was equipped with a Pentium IV 1.2 GHz CPU and 128 MB main memory. In the experiments, the hand tracking and the handshape extraction are operating in real-time. The following feature extraction processes includes FD and motion analysis may finish in less than one second. Totally, the recognition system about one second from image sequence capturing to gesture recognizing. In the training stage, for each gesture, we have asked 20 different individuals to make the gestures three times, and for each gesture, we have 60 different training image sequences to generate the corresponding HMM.

Each input image sequence is pre-processed by hand region extraction process for contour information and coding. 1200 image sequences are used in training phase, and 1200 image sequences are used in testing phase. Our system consists of two methods: (1) using only contour information and (2) using combined contour information

Observation Symbol Sequence																														
57	41	41	41	41	41	42	42	45	44	45	43	38	39	38	38	38	36	36	36	39	39	39	41	39	49	23	49	23	39	39

(a)



(b) The maximum likelihoods generated by 20 HMMs

Fig. 21. The experimental results of recognizing the 1st gesture from a sequence of frames.

Table 3
The error rate of the gesture recognition system using FD and motion vector

Gesture	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20
Error (%)	7	1	0	11	12	5	4	7	11	6	5	11	3	0	10	4	13	15	0	3

and motion information. The extracted information is converted to vector sequences and then quantized into symbol sequences for both of the training and recognition processes.

The same gesture made by different individuals may looks different because of different hand-shapes and gesture speed. To design a robust recognition system, the training data are selected to cover all possible hand-shapes for each individual. Before using HMMs for training or recognition process, any vector sequence is preprocessed by VQ to an observable symbol sequence O. The codebooks are created based on their corresponding training data. The codebook size M, which is power of 2, is chosen by experiments. We have tried different codebook sizes, and find that $M = 64$ is the best choose because the recognition rate does not have any significant improvement for $M > 64$. Based on these training symbol sequences, we can effectively generate the 1st-order 4-state HMM for modeling the gesture. We have tested our system by using three different state number HMMs (3-state, 4-state and 5-state), and we found that the 4-state HMM has proved to generate the best performance.

(1) *Fourier descriptor (FD) only*. Totally 1200 image sequences are collected for 20 different gestures, thus each kind of gesture with 60 sequences in average, in training phase and other 1200 sequences are collected for test. The recognition rate of using training data for testing is 97%, and the recognition rate of using testing data is 90.5%. The error rates of recognizing gesture 9, 12, 17 and 18 are among the highest. This is because the extracted hand shape may be not precise and the hand-shapes of these gestures are similar to one another. Thus, we may combine the FD and motion vector as the feature vector for a better performance.

(2) *FD and motion vector*. We add motion information to the feature vector for our HMM modeling. We find that the recognition rate of using training data for testing is 98.5% and the recognition rate of using testing data rises to 93.5%. This method gains 3% improvement of the recognition rate using the testing data. The reason is that adding the motion vector improves the recognition rate for the 9th, 12th, 17th and 18th gestures in our vocabulary. However, for some gestures, there may be a slightly performance decrement due to the different experimental environments.

Fig. 21 shows the results of the gesture recognition of the 1st gesture in our vocabulary. Fig. 21(a) shows the sequence of observation symbols which is input to the hmm. Fig. 20(b) shows the output of the maximum likelihood of each HMM applied to the testing sequence. there are totally 20 HMMS in the recognition system of which first HMM generate the largest maximum likelihood. In our experiments, we have tested 20 different gestures from different signers, some gestures are not precise, and the recognition rate drops to 85% (see Table 3). We find that our recognition system is size and rotation insensitive, for small objects and for large objects, it can still effectively identify the correct gesture. We also find that when the symbol sequence has an error at frame 30 (symbol 35 is obtained instead of symbol 21), and the score of the HMM modeling gesture 10 is very close to the score of the HMM modeling gesture 13. Our system can still recognize the gesture correctly. However, if in the beginning, the system makes many error observations and generates wrong symbols, then the HMM models will not justify the correct recognition. Another reason for error recognition is that we don't have enough training data to make a good estimate of the HMM model parameters.

6. Conclusions

We have developed a method to recognize the unknown input gestures by using HMMs. Since the variation of the hand gestures is usually large, the transition between states is necessary in each gesture for an effective hand tracking. We apply this system to recognize the single gesture. In the experiments, we assume stationary background so that our system will have smaller search region for tracking. With a larger training set and context modeling, lower error rates are expected and generalization to user independent gesture recognition system should be developable. Once we add a new gesture into the system, we only need to re-train another HMM for the new gesture, since the relationships between new model and the original models are independent.

References

[1] V.I. Pavlovic, R. Sharma, T.S. Huang, Visual interpretation of hand gestures for human-computer interaction, A Review, IEEE Transactions on Pattern Analysis and Machine Intelligence 19 (7) (1997) 677–695.

[2] T. Baudel, M. Baudouin-Lafon, Charade: remote control of objects using free-hand gestures, Comm. ACM 36 (7) (1993) 28–35.

[3] D.J. Sturman, D. Zeltzer, A survey of glove-based input, IEEE Computer Graphics and Applications 14 (1994) 30–39.

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

[5] A.F. Bobick, A.D. Wilson, A state-based technique for the summarization and recognition of gesture, Proceedings fifth international conference on computer vision (1995) 382–388.

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

[7] J. Davis, M. Shah, Visual gesture recognition, IEE Proceedings - Vision Image Signal Process 141 (2) (1994).
[8] Darrell, T., Pentland, A., Recognition of space-time gestures using a distributed representation, MIT Media Laboratory Perceptual Computing Group Technical Report. 1992 No197.
[9] T. Starner, A. Pentland, Visual recognition of american sign language using hidden Markov models, Proceedings of International. Workshop on Automatic Face- and Gesture-Recognition, Zurich, Switzerland, 1995
[10] L.W. Campbell, D.A. Becker, A. Azarbayejani, A.F. Bobick, A. Plentland, Invariant features for 3-D gesture recognition, Proceedings IEEE Second International Workshop on Automatic Face and Gesture Recognition, 1996.
[11] J. Schlenzig, E. Hunter, R. Jain, Recursive identification of gesture inputers using hidden Markov models, Proceedings Second Annual Conference On Applications. Of Computer Vision (1994) 187–194.
[12] R.H. Liang, M. Ouhyoung, a real-time continuous gesture recognition system for sign language, Proceedings IEEE Second International Conference on Automatic Face and Gesture Recognition, 1998, pp. 558–565.
[13] Y. Cui, J.J. Weng, Hand sign recognition from intensity image sequences with complex backgrounds, Proceedings IEEE Second International Conference on Automatic Face and Gesture Recognition, 1996.
[14] A. Ohknishi, A. Nishikawa, Curvature-based segmentation and recognition of hand gestures, Proceedings Annual Conference On Robotics Society of Japan, 1997, p. 401.
[15] S. Nagaya, S. Seki, R. Oka, A theoretical consideration of pattern space trajectory for gesture spotting recognition, Proceedings IEEE Second International Workshop on Automatic Face and Gesture Recognition, 1996.
[16] T. Heap, D. Hogg, Towards 3D hand tracking using a deformable model, Proceedings IEEE Second International Conference on Automatic Face and Gesture Recognition, 1996.
[17] S.C. Zhu, A.L. Yuille, FORMS: a flexible object recognition and modelling system, Proceedings Fifth International Conference on Computer Vision (1995) 465–472.
[18] R. Lockton, A.W. Fitzgibbon, Real-time gesture recognition using deterministic boosting, Proceedings of British Machine Vision Conference (2002).
[19] E. Persoon, K.S. Fu, Shape discrimination using fourier descriptor, IEEE Transactions SMC 7 (3) (1977) 170–179.
[20] D. Shridhar, A. Badreldin, High-Accuracy character recognition algorithm using fourier and topology descriptors, Pattern Recognition 17 (1984) 515–524.
[21] B. Horn, B.G. Shunck, Determining optical flow, Artificial Intelliegence 17 (1981) 185–203.
[22] A. Kundu, Y. He, P. Bahl, Recognition of handwritten word: first and second order hidden Markov model based approach, Pattern Recognition 22 (3) (1989) 283–297.
[23] J. Yamato, J. Ohya, K. Ishii, Recognizing human action in time-sequential images using hidden Markov models, Proceedings IEEE Conference on Computer Vision and Pattern Recognition (1992) 379–385.
[24] J. Schlenzig, E. Hunter, R. Jain, Recursive identification of gesture inputs using hidden Markov models, Proceedings Second Annual Conference on Applications of Computer Vision (1994) 187–194.
[25] Y. Linde, A. Buzo, R. Gray, An algorithm for vector quantizer design, IEEE Transaction on Communications COM-28 (1) (1980) 84–95.
[26] L.R. Rabiner, S.E. Levinson, M.M. Sondhi, On the application of vector quantization and hidden Markov models to speaker-independent, Isolated Word Recognition, Bell System Technology Journal 62 (4) (1983) 1075–1105.
[27] N. Otsu, A thresholding selection method from gray-level histogram, IEEE Transactions System Man Cybernet 9 (1) (1979) 62–66.
[28] G.D. Forney, The viterbi algorithm, Proceedings of the IEEE 48 (1973) 268–278.
[29] L.R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, Proceedings of The IEEE 77 (2) (1989) 257–285.