# An HMM-Based Threshold Model Approach for Gesture Recognition

Hyeon-Kyu Lee and Jin H. Kim

**Abstract**—The task of automatic gesture recognition is highly challenging due to the presence of unpredictable and ambiguous nongesture hand motions. In this paper, a new method is developed using the Hidden Markov Model based technique. To handle nongesture patterns, we introduce the concept of a *threshold model* that calculates the likelihood threshold of an input pattern and provides a confirmation mechanism for the provisionally matched gesture patterns. The threshold model is a weak model for all trained gestures in the sense that its likelihood is smaller than that of the dedicated gesture model for a given gesture. Consequently, the likelihood can be used as an adaptive threshold for selecting proper gesture model. It has, however, a large number of states and needs to be reduced because the threshold model is constructed by collecting the states of all gesture models in the system. To overcome this problem, the states with similar probability distributions are merged, utilizing the *relative entropy* measure. Experimental results show that the proposed method can successfully extract trained gestures from continuous hand motion with 93.14 percent reliability.

**Index Terms**—Hand gesture, gesture spotting, Hidden Markov Model, segmentation, pattern recognition, relative entropy, state reduction, threshold model.

## 1 Introduction

Human gestures constitute a space of motion expressed by the body, face, and/or hands. Among a variety of gestures, the hand gesture is the most expressive and the most frequently used one [1], [2], [3], [4], [5]. In this paper, we define a gesture as a meaningful part of the hand motion to communicate with a computer. The interaction using hand gestures has been studied as an alternative form of human-computer interface by a number of researchers, including Quek [6], Kjeldsen and Kender [7], and Starner and Pentland [8].

The task of locating meaningful patterns from a stream of input signal is called *pattern spotting* [9]. *Gesture spotting* is an instance of pattern spotting where it is critical to locate the start point and the end point of a gesture pattern. It has been regarded as a highly difficult task mainly due to two aspects of signal characteristics: *segmentation ambiguity* [10] and *spatio-temporal variability* [11].

The segmentation ambiguity problem concerns how to determine when a gesture starts and when it ends in a continuous hand trajectory. As the motion switches from one gesture to another, the hand makes an intermediate movement between the two gestures. Without knowing the gesture boundaries, reference patterns have to be matched with all possible segments of input signals, as shown in Fig. 1. In this course, transitional motions may be mistaken as meaningful ones. The other difficulty of gesture spotting comes from the fact that the same gesture varies dynamically in shape and duration, even for the same gesturer. An ideal recognizer will extract gesture segments from the continuous input signal and match them with reference patterns allowing a wide range of spatio-temporal variability.

Recently, the HMM has attracted the attention of many researchers as a useful tool for modeling the spatio-temporal variability of gestures [12]. However, it has some limitation in representing nongesture patterns. In the study of pattern spotting, each reference pattern is defined by a keyword model and all the other patterns are modeled by a single HMM called a *garbage model* or a *filler model* [13]. The garbage model in speech recognition represents acoustic nonkeyword patterns. It is usually trained using a finite set of nonkeyword samples. It is not easy, however, to obtain the set of nongesture training patterns because an almost infinite number of meaningless motions can be obtained. To overcome this problem, we make use of the *internal segmentation property*[1] of the HMM and introduce an artificial threshold model that consists of the state copies of all trained gesture models in the system. The artificial threshold model is an HMM that yields positive matching result with the patterns generated by the combination of subpatterns of the predefined gesture patterns in arbitrary order. One simple architecture of the threshold model is an ergodic model constructed by fully connecting all the states from all the gesture models in the system. We believe that such a threshold model can provide a confirmation mechanism for the provisionally matched gesture patterns.

The new ergodic model can match any described gestures. However, a gesture is better described by the dedicated model because the temporal order of subpatterns

• *H.-K. Lee is with Microsoft Korea, 6th Fl. POSCO Center, 892 Daechi-dong, Kangnam-gu, Seoul 135-777, Korea. E-mail: hklee@microsoft.com.*
• *J.H. Kim is with the Department of Computer Science, KAIST, 373-1 Kusung-dong, Yusong-ku, Taejon 305-701, Korea. E-mail: jkim@cs.kaist.ac.kr.*

*Manuscript received 26 Feb. 1999; revised 10 June 1999.*
*Recommended for acceptance by M. Shah.*
*For information on obtaining reprints of this article, please send e-mail to: tpami@computer.org, and reference IEEECS Log Number 107833.*

1. The property that the states and transitions of a trained HMM represent subpatterns of a gesture and their sequential order.

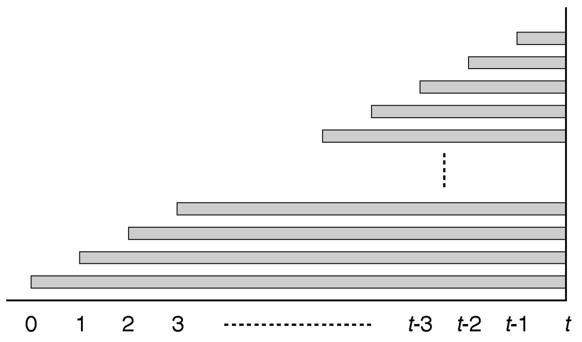


Fig. 1. All possible segmentations for a gesture with varying start points and the fixed end point $t$. The gesture recognizer will match all reference patterns for each of the segments.

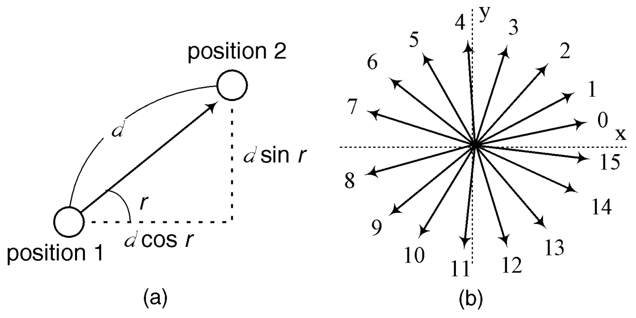


Fig. 3. Feature vector and directional codewords. A feature vector is a unit vector $(\cos\theta, \sin\theta)$ which consists of normalized elements in x and y directions. (a) Feature vector. (b) Directional codewords.

is better described in the model. These characteristics illustrate that the likelihood of the new model can be used for an adaptive threshold of the likelihood of gesture models. In this sense, the new ergodic model is termed *threshold model*. The threshold model differs from the garbage model in that its likelihood provides a confidence limit for the likelihoods calculated by other gesture models, while that of the garbage model is just a similarity measurement.

A potential weakness of the threshold model is in the spotting speed. The threshold model usually has a large number of states in proportion to the number of the gesture models in the system. Accordingly, the computational requirement increases exponentially and the spotting speed slows down. In this paper, this problem is alleviated by reducing the number of states of the threshold model based on the relative entropy, which has been often used as a measure of the distance between two probability distributions [14]. Pairs of states with the least distance are merged repeatedly until the number of states reaches an experimentally determined value.

The gesture spotting procedure in this paper is illustrated in Fig. 2. A hand location is detected whenever a new image frame is grabbed and is used to build up a hand trajectory. Here, the hand trajectory is automatically projected into a 2D-plane because the location is expressed by x- and y-coordinates of the center of the hand in a frame. In short, the problem of this paper is how to locate predefined gesture patterns from the 2D projected hand trajectory that extends at each time step.

A gesture in this paper is described as a spatio-temporal sequence of feature vectors that consist of the direction of hand movement, as shown in Fig. 3a. A feature vector should be converted to one of the 16 directional codewords (Fig. 3b) by a vector quantizer because we make use of the discrete HMM-based approach.

In this study, we chose the 10 most frequently used browsing commands of PowerPoint™, and defined a gesture for each of them, as shown in Table 1. The shapes of the gestures were chosen to be distinct from each other while keeping the naturalness of hand motion. However, this choice includes a lot of confusions between gestures, as shown in Fig. 4. The causes of the confusion are the extra movement to locate the hand to start position of each gesture (Fig. 4a) and the similarity of simple gestures to some parts of complex gestures (Fig. 4b).

For each gesture, we design a model using the left-right HMM utilizing the temporal characteristics of gesture signals. A unique initial state and a unique final state are in the model as shown in Fig. 5. The number of states in a model is determined based on the complexity of the corresponding gesture. It has been reported that the error rate seems to reach the minimum at a specific value of $N$ [13], the number of states, although the model likelihood improves further as $N$ increases to some extent. It is, however, necessary to limit the number of parameters in the model with limited training samples. The number of states in our gesture models ranges from five to eight, depending on the complexity of the gesture shape. The gesture models are trained using Baum-Welch reestimation algorithm.

To evaluate the proposed method, we have developed a prototype system called *PowerGesture* with which one can browse PowerPoint™ slides using gestural commands. The rest of this paper is organized as follows: In Section 2, we describe the background of the research, including previous

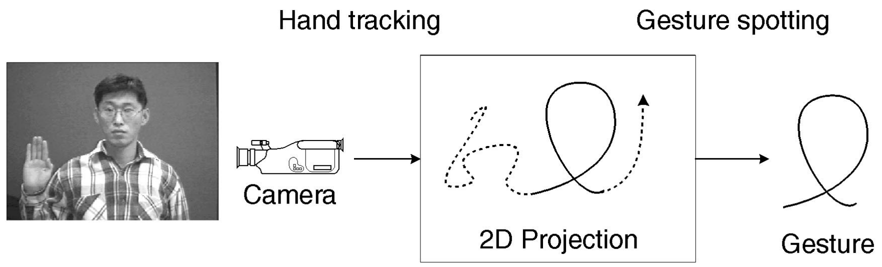
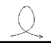
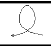
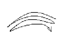
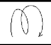
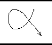


Fig. 2. Gesture spotting procedure.

TABLE 1
Gestures Used

| Gesture | Command | Command Description |
|---|---|---|
|  | Last | Go to last slide. |
|  | First | Go to first slide. |
| | Next | Go to next slide. |
| | Previous | Go to previous slide. |
| | Start | Start presentation. |
|  | Bye | Quit PowerPoint$^{TM}$. |
| | White | Fill screen with white color. |
| | Black | Fill Screen with black color. |
|  | Hidden | Show hidden slide. |
|  | Stop | End presentation. |

gesture recognition studies, and the HMM. Section 3 is devoted to the detailed description of the threshold model and the end-point detection. Experimental results are provided in Section 4, and concluding remarks in Section 5.

## 2 Background

### 2.1 Related Works

Gestures in general are defined as finite characteristic motions made in 2D or 3D space using a suitable input device. The 2D gestures are associated with a mouse or stylus on a tablet. In this paper, we will consider only 3D hand gestures captured with a camera. Hand gestures can be further classified into two types: a posture and a gesture. The posture is the static finger configuration without hand movement, while the gesture is the dynamic hand movement with or without finger motion.

This paper explores a vision-based analysis of hand gestures that are viewed as spatio-temporal patterns. Major approaches for analyzing spatial and temporal patterns include Dynamic Time Warping (DTW) [10], Neural Networks (NNs) [1], [7], and Hidden Markov Models (HMMs) [8], [12]. The DTW is a template-based dynamic-programming (DP) matching technique that has been applied to problems with temporal variability [10]. Although it has been successful in small vocabulary tasks, the DTW needs a large number of templates for a range of variations. Furthermore, it cannot handle undefined patterns.

Takahashi et al. [10] proposed a spotting algorithm, called continuous dynamic programming (CDP), to recognize gestures of the body and arms. In the method, the predefined pattern corresponding to a gesture is described with a spatio-temporal vector field derived by the three-directional gradient ofan image sequence. An input pattern is compared with the predefined patterns by the CDP matching method. The CDP matching works as follows: An input pattern is generated from an input image sequence at each time $t$. They regard the time $t$ as a possible end point of a gesture and compare the input pattern with all predefined patterns. They make use of the dynamic programming for calculating the distance between two patterns regardless of the time difference. Since the distance drops down to minimum value at the end point of the corresponding gesture, the distances with all gestures in the system are recorded and observed in each time step. Once a distance moves to the minimum value, the corresponding predefined pattern (a gesture) is fired. The weakness of the approach is that it performs not so robustly with respect to shape variations.

As larger data sets become available, more emphasis is being placed on Neural Networks. For representing temporal information with NNs, there are two approaches: recurrent neural networks and feed-forward networks with a complex preprocessing phase. The problem of the neural networks arising in the context of gesture spotting is how to model nongesture patterns. Although the neural networks have shown effectiveness in recognizing the static patterns (the postures), they are not suited for the dynamic patterns

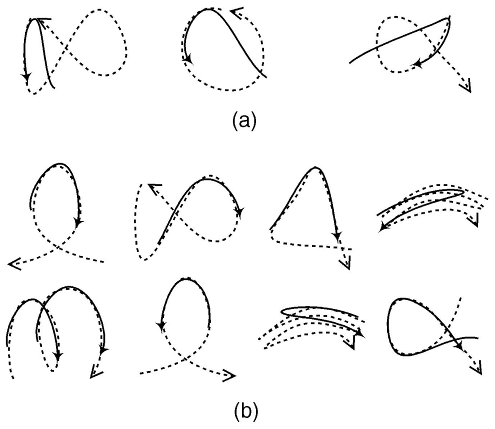


Fig. 4. Confusion between gestures. (a) Confusion caused by the extra hand movement and (b) by the similar shapes. Each dotted curve represents a gesture pattern and the dark curve represents a plausible gesture which should be ignored.

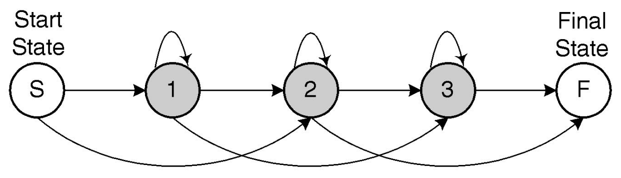


Fig. 5. An example of the gesture models.

(the gestures) [2]. Kjeldsen and Kender [7] developed a gesture-based interface using the neural networks for window system control. The system has two functional layers: hand tracking layer and action layer. The hand-tracking layer identifies and tracks the user's hand in real-time. In this layer, they extract the hand by taking advantage of the unique coloration of human skin, which is a kind of color histogramming technique. The action layer is based on a grammar to map an event (identified by the hand-tracking layer) to an action of the system. The grammar also utilizes both the motion and the pose of the hand. The authors use neural networks to classify hand poses only when needed.

The Hidden Markov Model (HMM) is another statistical modeling tool that can be applied to analyzing time-series with spatial and temporal variability [12]. In spite of the success in online handwriting recognition [16], [17] and speech recognition [9], [13], only a few researchers have used the HMM to recognize hand gestures [8], [18]. Starner and Pentland implemented an HMM-based system for recognizing sentence-level American Sign Language (ASL) without explicitly modeling fingers [8]. In their work, the subject wears distinctly colored gloves on both hands and sits in a chair in front of the camera to aid hand tracking. They chose an eight-element feature vector consisting of each hand's x- and y-coordinates, axis angle of the least inertia, and the eccentricity of the bounding ellipse. The HMM is used to recognize data strings and is combined with statistical grammars to bring the context during training and recognition. For recognition, the Viterbi algorithm is used both with and without a strong grammar based on the known forms of the sentences. However, once the recognizer starts, the subject must conduct only sign languages because it cannot separate undefined hand motions.

We choose the HMM-based approach because it can be applied to analyzing time-series with spatio-temporal varibilities and can handle undefined patterns. The following section describes the definition and the topologies of the HMM.

### 2.2 Hidden Markov Model (HMM)

The HMM is rich in mathematical structures; it serves as the theoretical basis for a wide range of applications. It can model spatio-temporal information in a natural way. It also has elegant and efficient algorithms for learning and recognition, such as the Baum-Welch algorithm and Viterbi search algorithm [12].

The HMM is a collection of states connected by transitions [12], [15]. Each transition has a pair of probabilities: a *transition probability* (which provides the probability for taking the transition) and an *output probability* (which defines the conditional probability of emitting an output symbol from a finite alphabet given a state). A formal characterization of HMM is as follows:

- $\{s_1, s_2, s_3, \ldots, s_N\}$—A set of $N$ states. The state at time $t$ is denoted by the random variable $q_t$.
- $\{v_1, v_2, v_3, \ldots, v_M\}$—A set of $M$ distinct observation symbols, or a discrete alphabet. The observation at time $t$ is denoted by the random variable $O_t$. The observation symbols correspond to the physical output of the system being modeled.
- $A = \{a_{ij}\}$—An $N \times N$ matrix for the state transition probability distributions where $a_{ij}$ is the probability of making a transition from state $s_i$ to $s_j$:

$$a_{ij} = P(q_{t+1} = s_j \mid q_t = s_i).$$

- $B = \{b_j(k)\}$—An $N \times M$ matrix for the observation symbol probability distributions where $b_j(k)$ is the probability of emitting $v_k$ at time $t$ in state $s_j$:

$$b_j(k) - P(O_t = v_k \mid q_t = s_j).$$

- $\pi = \{\pi_i\}$—The initial state distribution where $\pi_i$ is the probability that the state $s_i$ is the initial state:

$$\pi_i = P(q_1 = s_i).$$

Since $A$, $B$, and $\pi$ are probabilistic, they must satisfy the following constraints:

- $\Sigma_j a_{ij} = 1 \; \forall i$, and $a_{ij} \geq 0$.
- $\Sigma_k b_j(k) = 1 \; \forall j$, and $b_j(k) \geq 0$.
- $\Sigma_i \pi_i = 1$ and $\pi_i \geq 0$.

Following the convention, a compact notation $\lambda = (A, B, \pi)$ is used which includes only probabilistic parameters.

Some examples of the HMM topologies are illustrated in Fig. 6. The left-right model as shown in Fig. 6a is good for modeling order-constrained time-series whose properties sequentially change over time [15]. Since the left-right model has no backward path, the state index either increases or stays the same as time increases. In other words, the state proceeds from left to right or stays where it was. On the other hand, every state in the ergodic or fully connected model can reach every other state in a single transition, as shown in Fig. 6b.

## 3 Spotting with Threshold Model

### 3.1 Threshold Model

For correct gesture spotting, the likelihood of a gesture model for a given pattern should be distinct enough. Unfortunately, although the HMM recognizer chooses a model with the best likelihood, we cannot guarantee that the pattern is really similar to the reference gesture unless the likelihood value is high enough. A simple thresholding for the likelihood often does not work. Therefore, we propose a new concept, called *threshold model*, that yields the likelihood value to be used as a threshold. A gesture is

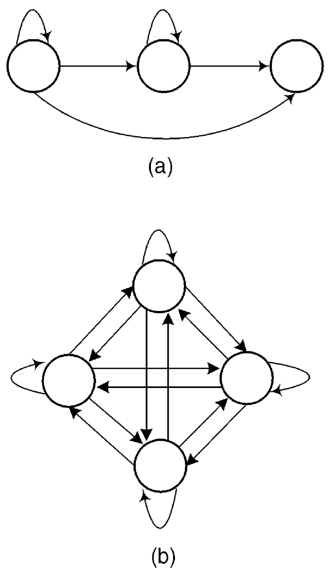


Fig. 6. Types of the HMM: (a) left-right model (b) ergodic model.

recognized only if the likelihood of the best gesture model is higher than that of the threshold model.

The HMM's internal segmentation property implies that each state with its self-transition represents a segmental pattern of a target gesture and that outgoing transitions represent a sequential progression of the segments in a gesture. With this property, we can construct an ergodic model with the states copied from all gesture models in the system and then fully connect the states (Fig. 7). In this model, each state can be reached by all other states in a single transition. Output observation probabilities and self-transition probabilities in this model are kept as in the gesture models, but all outgoing transition probabilities are equally assigned as

$$a_{ij} = \frac{1 - a_{ij}}{N-1}, \quad \text{for all } j, i \neq j,$$

where $a_{ij}$ is the transition proability from state $s_i$ to $s_j$ and $N$ is the number of states (the sum of all states excluding the start and final states). The start and final states produce no observation. Fig. 7 shows the threshold model as a simplified version of the ergodic model in Fig. 6b.

Maintaining the output probability distributions and the self-transition probabilities makes the states represent any subpattern of reference patterns and the ergodic structure makes it match well with any patterns generated by combining the subpatterns in any order. The likelihood of the model, given a gesture pattern, would be smaller than that of the dedicated gesture model because of the reduced forward transition probabilities. Consequently, the likelihood can be used as an adaptive threshold for selecting the proper gesture model (Fig. 8). For this reason, we call it the *threshold model*. The threshold model acts as a base-line. A candidate gesture is found when a specific gesture model rises above the threshold model.

## 3.2 Gesture Spotting Network

In many gesture recognition tasks, a gesture spotter receives a stream of continuous hand motion with an intermittent sequence of several gestures. To spot them in the input stream, we have constructed a circular *gesture spotting network (GSN)*, as shown in Fig. 9. In the figure, S is the dummy start state.

The gesture spotting network finds the start and end points of gestures embedded in the input stream. For this, it is desirable to know how and in what state sequence the model produces the most likely observation sequence. We can uncover a state sequence using the *Viterbi algorithm*, where we adopt the optimality criterion of maximizing the probability of a state sequence that produces the observation sequence.

To find the single best state sequence, $Q_{1,t} = q_1 q_2 \dots q_t$, for the given observation $O_{1,t} = O_1 O_2 \dots O_t$, we need to define the quantity:

$$\delta_t(i) \cong \max_{Q_{1,t}} P(Q_{1,t-1}, q_t = s_i, O_{1,t} \mid \lambda)$$

with the highest probability along a single path arriving at $s_i$ at time $t$ and accounting for the first $t$ observations. We have, by induction:

$$\begin{array}{ll} \delta_1(j) = \pi_j b_j(O_1) & 1 \leq j \leq N, \\ \delta_t(j) = \max_i [\delta_{t-1}(i) a_{ij}] \cdot b_j(O_t) & 2 \leq t \leq T,\ 1 \leq j \leq N. \end{array}$$

For the backtracking information, we use $\psi_t(j)$ to keep the argument that maximizes $\delta_t(j)$ for each $t$ and $j$.

$$\psi_t(j) = \arg\max_i [\delta_{t-1}(i) a_{ij}] \qquad 2 \leq t \leq T,\ 1 \leq j \leq N.$$

In case of null transitions, the likelihood of the source state at time $t$ is simply maximized without time delay as

$$\begin{aligned} \delta_t(j) &= \max_i [\delta_t(i) a_{ij}], \\ i^* &= \arg\max_i [\delta_t(i) a_{ij}], \\ \psi_t(j) &= \psi_t(i^*). \end{aligned}$$

Finally, to uncover the most likely state sequence $Q^* = q*_1 \, q_2^* \dots q_T^*$ after the preceding computation, we must trace back to the initial state by following the Viterbi path of $\psi$s as:

$$\begin{array}{ll} q_T^* = s_N, & \\ q_t^* = \psi_{t+1}(q_{t+1}^*) & t = T-1, T-2, \dots 1. \end{array}$$

This algorithm described above is known as the *Viterbi algorithm* [19]. It can be efficiently implemented using a lattice structure. We use this algorithm for calculating likelihood and finding the start point from the end point.

For reliable spotting, the model transition probability into the threshold model $p(TM)$ is tuned to satisfy:

$$P(X_G \mid \lambda_G) p(G) > P(X_G \mid \lambda_{TM}) p(TM) \qquad (1)$$

$$p(X_{TM} \mid \lambda_G) p(G) < P(X_{TM} \mid \lambda_{TM}) p(TM), \qquad (2)$$

where $X_G$ denotes a gesture pattern, $X_{TM}$ a nongesture pattern, $\lambda_G$ the target gesture model, $\lambda_{TM}$ the threshold model, and $N_G$ the number of gesture models in the system. Inequalities (1) and (2) imply that a gesture should best match with the corresponding gesture model and a nongesture with the threshold model, respectively. The

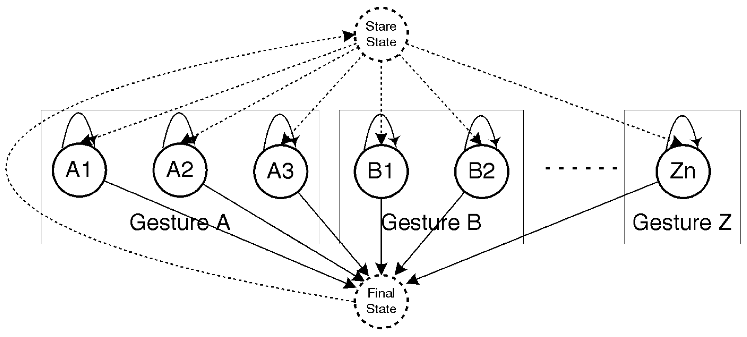


Fig. 7. A simplified structure of the threshold model. The dotted arrows are null transitions.

model transition probabilities into gesture model $p(G)$s are set equal using $p(TM)$ as

$$p(G) = \frac{1 - p(TM)}{N_G}.$$

With a sequence of spotting experiments, we have decided $p(TM)$ as the value generating the best result.

Inequality (1) says that the likelihood of a gesture model should be greater than that of the threshold model. The time satisfying such a condition can be called a *candidate end point (CEP)*. Once we obtain CEP, its corresponding start point can easily be found by backtracking the Viterbi path because the final state can only be reached through the start state in the left-right HMM. There are, in general, several such CEPs satisfying (1). Thus, the remaining problem is the determination of the right end point. This will be described in the next section.

## 3.3 End-Point Detection

If we look at time-evolution of the likelihood of individual models, the threshold model usually scores the best. As the forward pass gets close to the end of a gesture, the target gesture model soars up above the threshold. As shown in Fig. 10, the likelihood of the model *last* is initially lower than that of the threshold model. After time 12, however, the likelihood of the model *last* becomes greater. All points between 13 and 16 are candidate end points (CEPs) of the gesture *last*. Each time we encounter a CEP, the corresponding start point is determined by backtracking the Viterbi path.

The end-point detection is the process of choosing the best among these CEPs. It removes nested gestures and makes the spotter fire only once when a gesture is encountered. The process is initiated when the last CEP of the current gesture is found after the preceding gesture to

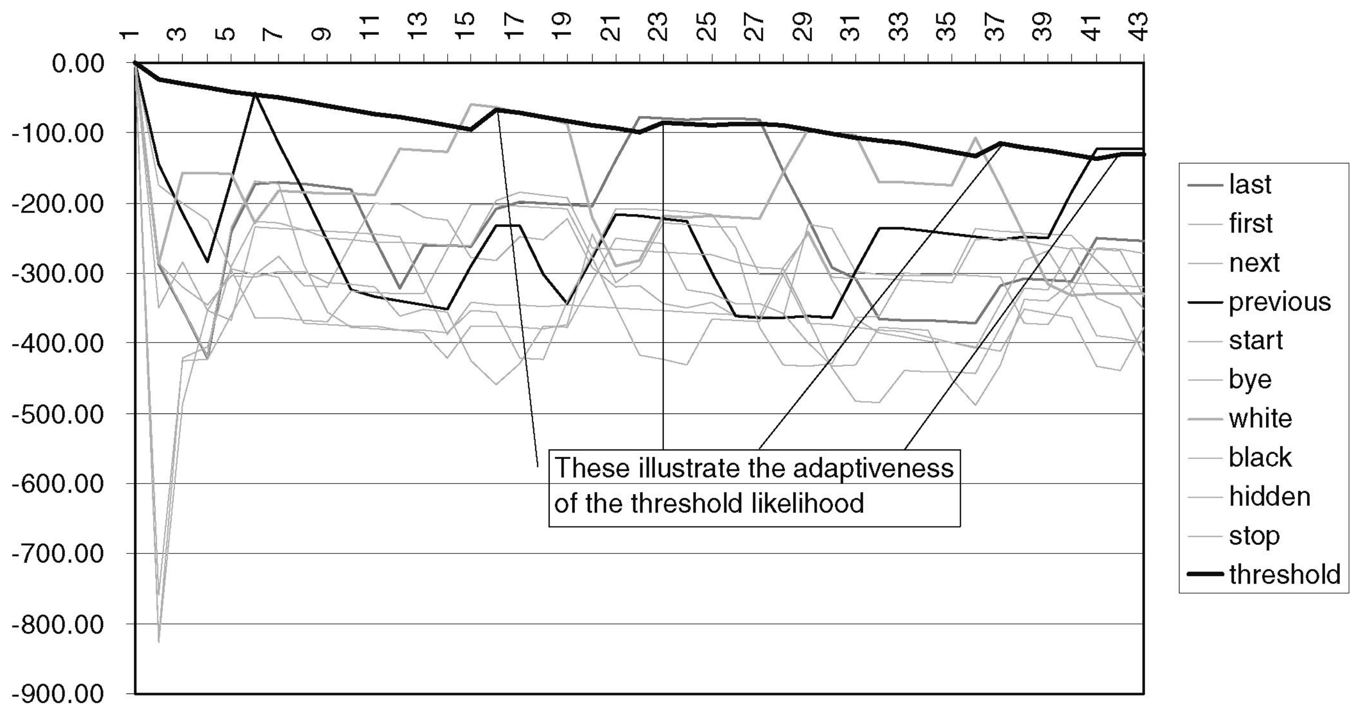


Fig. 8. A sample likelihood evolution graph of the gesture models and the threshold model demonstrating the adaptiveness of the threshold likelihood. The thick line is the threshold likelihood.

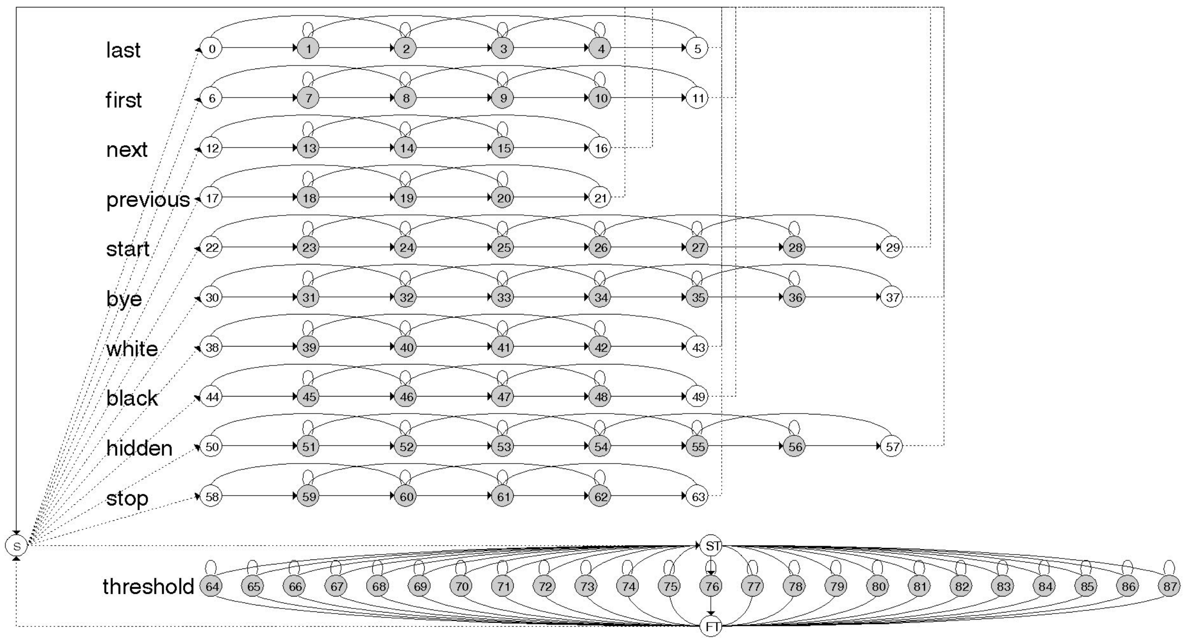
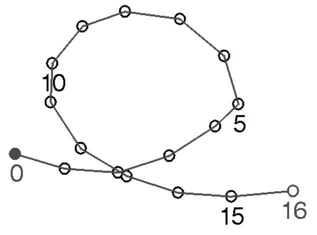
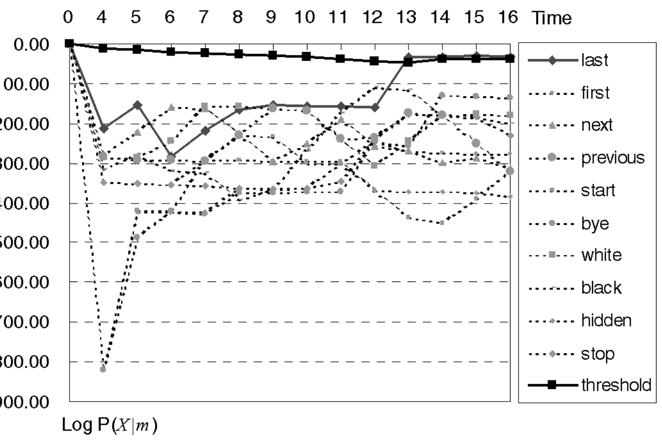


Fig. 9. Gesture Spotting Network. A label denotes the name of a gesture and each dotted line represents a transition between models.





(a) (b)

Fig. 10. A sample gesture and its likelihood evolution. (a) Trajectory of a sample gesture for *last*. (b) The likelihood evolution of the gesture models and the threshold model for a sample gesture.

fire or when the elapsed time steps since the last gesture are greater than a given length. The last condition implements a duration constraint.The detection criterionis defined as follows:

1. When the immediately following pattern B is not a gesture, as in Fig. 11a, the last CEP of the preceding gesture A is determined as the end point. A is reported.
2. When the immediately following pattern B is by itself a gesture, there are two alternatives:
   a. When the start point of B precedes the first CEP, as in Fig. 11b of A, A is regarded as a part of a larger current gesture (B) which includes A and extends beyond the CEP of A. All the CEPs of A are ignored.
   b. When B starts between the first and the last CEPs of A, as in Fig. 11c, the immediately preceding CEP is chosen as the end point of A.

We cannot fire a gesture immediately at a CEP because the detected gesture may be a part of a larger gesture, as in Fig. 11b. Since the delayed response may cause one to wonder whether one's gesture has been recognized correctly or not, we resolve it with two approaches. One is limiting the delay time to some extent by introducing a maximum length of the nongesture pattern that is longer than the largest gesture. Another is taking advantage of heuristic information to catch one's completion intentions, such as moving the hand out of the camera range or freezing the hand for a while. Once the intention is detected, all input signals are removed and the remaining gestures are retrieved. Fig. 12 illustrates the flowchart of the end-

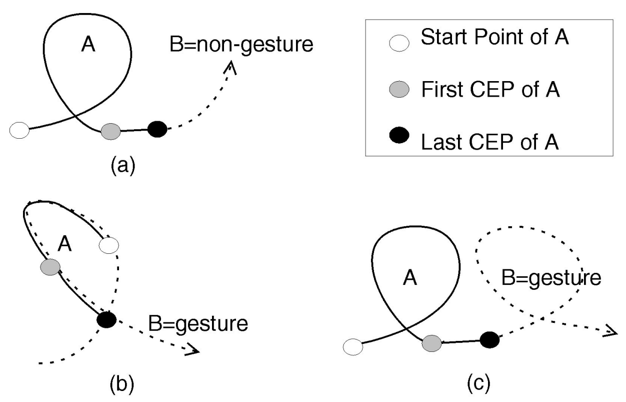


Fig. 11. Determination of an end point. (a) Nongesture is following. (b) Nested gesture. (c) Gesture is following. Each dark curve A represents the preceding gesture that may be reported and the dotted curve B represents the immediately following pattern or gesture.

point detection routine. In the figure, we introduce a stack to check the gesture embedding.

### 3.4 Enhancing the Performance

Since we constructed the threshold model by combining all the gesture models in the system, the number of states in the threshold model is equal to the sum of the states of all gesture models excluding the start and final states. This means that the number of states in the threshold model increases as the number of gesture models increases. Consequently, the increased number of states increases time and space requirements. To overcome this problem, we propose using *relative entropy* to reduce the number of states of the threshold model.

The *entropy* $H(X)$ of a discrete random variable $X$ is defined by:

$$H(\boldsymbol{X}) = -\sum_{x \in X} p(x) \log p(x),$$

where $p(x)$ is a probability mass function. The entropy measures the expected uncertainty of the source that generates the possible event $\boldsymbol{X}$. The more uncertain, the bigger the entropy.

Consider two discrete probability distributions $\boldsymbol{p} = \{p_1, p_2, \ldots, p_M\}$ and $\boldsymbol{q} = \{q_1, q_2, \ldots, q_M\}$. One problem is how to measure the difference between $\boldsymbol{p}$ and $\boldsymbol{q}$. This leads to the idea of the relative entropy of $\boldsymbol{p}$ given $\boldsymbol{q}$, or vice versa. The *relative entropy* was introduced by Kullback [20]; it is also discussed in the literature by other names such as the *Kullback-Leibler information criterion*, *cross entropy*, or *directed divergence*. The relative entropy $D(\boldsymbol{p}\|\boldsymbol{q})$ between two discrete probability distributions is defined by

$$D(\boldsymbol{p}\|\boldsymbol{q}) = \sum_{i=1}^{M} p_i \log \frac{p_i}{q_i},$$

where $0 \log \frac{0}{q_i} = 0$ and $p_i \log \frac{p_i}{0} = \infty$. The relative entropy is always nonnegative; it is zero if and only if $\boldsymbol{p}$ equals $\boldsymbol{q}$ [14].

To be precise, the relative entropy is not a true distance between distributions since it is not symmetric and does not satisfy the triangle inequality. Nonetheless, it is often useful to think of relative entropy as the distance between distributions because the ordering of distributions can be predefined and the calculation is very simple [21]. Since, the relative entropy is not symmetric, in other words

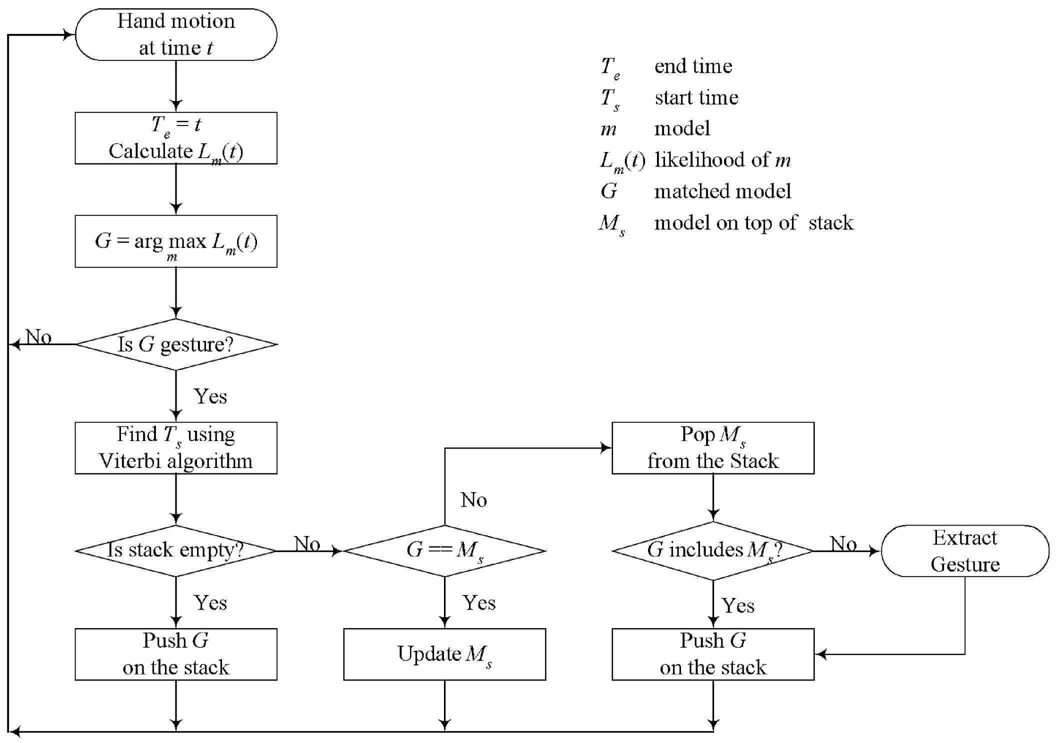


Fig. 12. The procedure of the end-point detection.

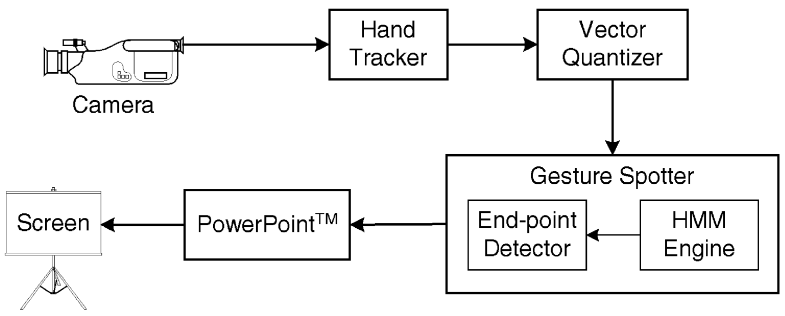


Fig. 13. Block diagram of the Power Gesture system.

$D(\boldsymbol{p}\|\boldsymbol{q}) \neq D(\boldsymbol{q}\|\boldsymbol{p})$, let us change *relative entropy* for symmetricity slightly as:

$$D(\boldsymbol{p}\|\boldsymbol{q}) = \frac{1}{2}\sum_{i=1}^{M}\left(p_i \log\frac{p_i}{q_i} + q_i \log\frac{q_i}{p_1}\right). \tag{3}$$

The proposed state reduction procedure is based on (3) and the algorithm follows.

**[Algorithm]** State reduction in the threshold model.

Let $\boldsymbol{p}^{(i)}$ and $\boldsymbol{q}^{(j)}$ be discrete probability distributions of $M$ observation symbols in state $i$ and state $j$, respectively.

**Step 1.** Calculate the symmetric relative entropy for each pair $p$ and $q$ of output probability distributions as

$$D(\boldsymbol{p}^{(i)}\|\boldsymbol{q}^{(j)}) = \frac{1}{2}\sum_{k=1}^{M}\left(p_k^{(i)} \log\frac{p_k^{(i)}}{q_k^{(j)}} + q_k^{(j)} \log\frac{q_k^{(j)}}{p_k^{(i)}}\right).$$

**Step 2.** Find an $(i, j)$ pair with the minimum symmetric relative entropy $D(\boldsymbol{p}^{(i)}\|\boldsymbol{q}^{(j)})$.

**Step 3.** Merge two states and recalculate the probability distribution of $M$ observation symbols in the merged state as:

$$p_k^{(i)\prime} = \frac{p_k^{(i)} + q_k^{(j)}}{2}.$$

**Step 4.** If the number of states is greater than an experimentally determined value, then go to Step 1.

Through a set of experiments, we have found that the suitable number of states of the threshold model without severe degradation in spotting power is about one and a half the number of observation symbols. The time complexity of the gesture spotter, $C$, is defined as follows:

$$C = K \times l\overline{N}T + N_{th}^2 T,$$

where $K$ is the number of gesture models in the system, $l$ is the number of transitions per state, $\overline{N}$ is the average number of states of gesture models in the system, $N_{th}$ is the number of states of the threshold model, and $T$ is the length of an observation sequence. The first term is the sum of the time

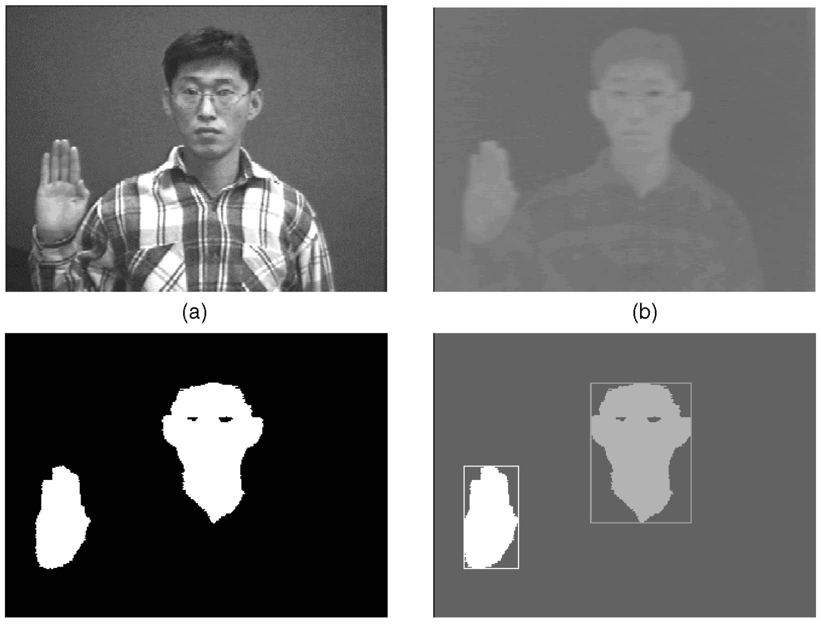


Fig. 14. Hand extraction procedure. (a) Camera image. (b) I-component image. (c) Binary image. (d) After segmentation.

TABLE 2
Isolated Gesture Patterns for the HMM Training and Testing

| Command | Training Data | Test Data |
|---|---|---|
| Last | 196 | 54 |
| First | 195 | 55 |
| Next | 198 | 52 |
| Previous | 195 | 55 |
| Start | 202 | 48 |
| Bye | 203 | 47 |
| White | 196 | 54 |
| Black | 202 | 48 |
| Hidden | 212 | 38 |
| Stop | 205 | 45 |
| **Total** | **2004** | **496** |

TABLE 3
The Reults of the Threshold Model Test

| Command | Test Data | Correct | Error | Recognition (%) |
|---|---|---|---|---|
| Last | 54 | 54 | 0 | 100.00 |
| First | 55 | 54 | 1 | 98.18 |
| Next | 52 | 51 | 1 | 98.08 |
| Previous | 55 | 55 | 0 | 100.00 |
| Start | 48 | 46 | 2 | 95.83 |
| Bye | 47 | 45 | 2 | 95.74 |
| White | 54 | 53 | 1 | 96.30 |
| Black | 48 | 46 | 2 | 95.83 |
| Hidden | 38 | 38 | 0 | 100.00 |
| Stop | 45 | 45 | 0 | 100.00 |
| **Total** | **496** | **487** | **9** | **98.19** |

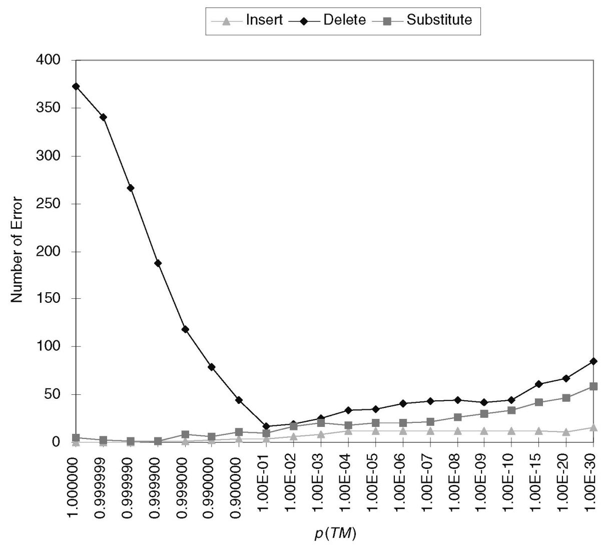
complexity of gesture models in the system and the second is that of the threshold model. In our gesture spotter, we quantize the direction of hand movement in 16 values and the number of states in the threshold model is reduced from 44 to 24. Consequently, the expected saving of the matching time with such a reduction is 63.91 percent.

## 4 Experimental Results

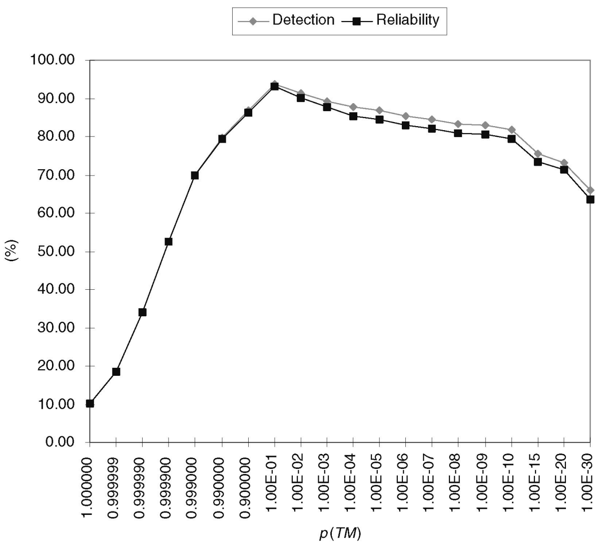
We implemented the *PowerGesture* system with which one can browse PowerPoint™slides using gestural commands. For the experiments, we have chosen 10 most frequently used browsing commands of PowerPoint™and defined a gesture for each of them, as shown in Table 1.

Fig. 13 shows the cascade architecture of PowerGesture. It is built on a Pentium Pro PC with Windows 95. It grabs image frames using the Matrox Meteor™ board. The Hand Tracker finds the hand in a sequence of image frames captured by a camera and makes a (sin, cos) pair as a directional feature for the hand's moving direction. The Vector Quantizer assigns one of the 16 codewords for a directional feature. Then, the Gesture Spotter detects individual gestures using the end-point detection routine. As a recognition result, the corresponding PowerPoint™ commands are generated for the gestures. Finally, PowerPoint™ controls the slide presentation.

Fig. 14 shows the hand-tracking process. It converts an RGB color image (Fig. 14a) to a YIQ color image (Fig. 14b). Since the I-component in YIQ color space is sensitive to skin colors [22], we simply apply a threshold to the I-component to produce a binary image (Fig. 14c). The hand region is then detected using the *one-pass labeling algorithm* [22], as shown in (Fig. 14d). In order to make the image processing simple, we assume that only one hand is used to make gestures in the simple background.



(a)



(b)

Fig. 15. Spotting results associated with $p(TM)$. (a) The number of errors. (b) The detection ratio and the reliability in percentage.

TABLE 4
Spotting Results with the Reduced Threshold Model ($p(TM) = 0.1$)

| Command | Number of Gestures | Results | | | | | |
|---|---|---|---|---|---|---|---|
| | | Insert | Delete | Substitute | Correct | Detection (%) | Reliability (%) |
| Last | 41 | 1 | 1 | 2 | 38 | 92.68 | 90.68 |
| First | 48 | 0 | 3 | 2 | 43 | 89.58 | 89.58 |
| Next | 57 | 1 | 0 | 1 | 56 | 98.25 | 96.55 |
| Previous | 41 | 1 | 4 | 1 | 36 | 87.80 | 85.71 |
| Start | 28 | 0 | 3 | 0 | 25 | 89.29 | 89.29 |
| Bye | 35 | 0 | 2 | 1 | 32 | 91.43 | 91.43 |
| White | 45 | 0 | 0 | 0 | 45 | 100.00 | 100.00 |
| Black | 49 | 0 | 3 | 1 | 45 | 91.84 | 91.84 |
| Hidden | 39 | 0 | 1 | 1 | 37 | 94.87 | 94.87 |
| Stop | 37 | 0 | 0 | 0 | 37 | 100.00 | 100.00 |
| **Total** | **420** | **3** | **17** | **9** | **394** | **93.81** | **93.14** |

In the following experiments, we carry out two tests. The first is an isolated gesture recognition test. This test examines whether the trained gesture model is acceptable and whether the threshold model generates appropriate threshold values. The other test observes the overall performance of the gesture spotter. In addition, we will see how the state reduction of the threshold model affects the performance and speed of the system.

### 4.1 Isolated Gesture Recognition Test

For a single gesture, we have collected 250 isolated samples from eight subjects (in total, 2,500 isolated samples for all gestures in the system) and have partitioned them into training and test sets, as shown in Table 2. Since the success of the gesture spotter greatly depends on the discrimination power of the gesture models and the threshold model, we carry out an isolated gesture recognition task carefully. The result is shown in Table 3. Most errors come from the failure of hand extraction that distorts the hand trajectory data. In case of error, gestures are rejected due to the lower likelihood of the target gesture model than that of the threshold model. With the reduced threshold model, we got exactly the same result as with the original one. This demonstrates that the discrimination power of the gesture spotter is rarely affected by the state reduction for isolated gesture samples.

### 4.2 Gesture Spotting Test

Since the second test evaluates the capability of the gesture extraction for continuous hand motions, we have collected 60 test samples from one person. Each sample is a sequence of 200 image frames (about 30 sec.) which contains more than one gesture and natural hand motions such as moving the hand to the start position of a gesture. In addition, some gestures can be embedded in a larger gesture.

In the gesture spotting task, there are three types of errors: The insertion error occurs when the spotter reports a nonexistent gesture, the deletion error occurs when the spotter fails to detect a gesture, and the substitution error occurs when the spotter falsely classifies a gesture. The detection ratio is the ratio of correctly recognized gestures over the number of input gestures:

$$\text{Detection ratio} = \frac{\text{correctly recognized gestures}}{\#\text{ of input gestures}}.$$

In calculating the detection ratio, the insertion errors are not considered. The insertion errors are likely to cause the deletion errors or the substitution errors because they often force the spotter to remove all or part of the true gestures from observation. To take into account the effect of the insertion errors, another performance measure, called *reliability*, is introduced that considers the insertion errors as follows.

$$\text{Reliability} = \frac{\text{correctly recognized gestures}}{\#\text{ of input gestures} + \#\text{ of insertion errors}}.$$

We have counted errors and calculated the detection ratio and reliability by varying model transition probability, $p(TM)$, as shown in Fig. 15. A high $p(TM)$ makes the threshold likelihood high; thus, many gestures are rejected. On the other hand, a low $p(TM)$ reduces the likelihood and, therefore, more gestures are likely to be reported. In Fig. 15a, as $p(TM)$ decreases between 1.0 and 0.1, the deletion errors decrease sharply. However, as $p(TM)$ passes 0.1, the deletion errors begin to increase because the increase of the insertion errors causes more deletion errors. The deletion errors directly affect the detection ratio, whereas the insertion errors do not. However, it should be noted that many insertion errors are not totally independent of the detection ratio because some insertion errors cause the deletion errors or the substitution errors (Fig. 15b).

We have conducted the same test for the reduced threshold model, and have obtained almost the same results. Table 4 shows the spotting performance of the reduced threshold model with $p(TM) = 0.1$. The experiments showed 93.38 percent reliability with the original threshold model and 93.14 percent reliability with the reduced threshold model. This confirms that the state reduction minimally affects the reliability of gesture spotting.

### 4.3 The Speed of Gesture Spotting

The PowerGesture system executes on a 200 MHz Pentium Pro PC with Windows 95. It has been developed to browse the slides of PowerPoint™ with hand gestures in real-time. Thus, the speed of gesture spotting is of practical importance. By the state reduction of the threshold model, 3.11 percent speed-up was observed at the cost of 0.26 percent reduced reliability.

The average spotting speed with the state reduction is 218 ms/frame, whereas the original is 225 ms/frame. Here, guessing the speed with a real-time processing environment[2] would help the real-world applicability of the Power Gesture system. The frame grabbing time occupies 65.6 percent (143 ms/frame) in the spotting time. If we adopt the real-time frame grabber that captures 33 ms/frame, the rate would be decreased to 30.6 percent and the spotting speed would be 108 ms/frame. In summary, the spotting speed corresponds to 0.15 real-time processing speed with our frame grabber and to 0.30 real-time processing with the real-time frame grabber.

## 5 Concluding Remarks

This paper describes an HMM-based gesture spotting system with a *threshold model* that calculates the threshold likelihood given an input pattern. The threshold model approves or rejects the pattern as a gesture. For gesture segmentation, it detects the reliable end point of a gesture and finds the start point by backtracking the Viterbi path from the end point. The model performs gesture spotting with 93.14 percent reliability and the processing time was 218 ms/frame. When we reduced the number of states in the threshold model, we achieved 3.11 percent speed-up at a negligible cost of reliability. In summary, the threshold model is simple and easy to design, but highly effective in providing a confirmation mechanism for the provisionally matched gesture patterns.

However, the proposed method has a problem in that the system cannot report the detection of a gesture immediately after the system reaches its end point. It is because the end-point detection process postpones the decision until the detection of the next gesture in order to avoid premature decision. The delayed response may cause one to wonder whether one's gesture has been recognized correctly or not. For this, we used heuristic information to catch one's intentions, such as moving the hand out of the camera range or freezing the hand for a while. Since such a heuristic is not very natural to humans, the problem of preserving naturalness of the gesture interface is left for future study.

2. The environment with the real-time frame grabber. It can capture 30 frames/sec, or 33 ms/frame.

## Acknowledgments

The authors greatly appreciate the contributions of Byung-Ro Moon, Bong-Ki Sin, and Jae-Hun Yang for their insightful comments on previous versions of the article. The authors are also grateful to the reviewers for their detailed review reports. The authors would like to thank Kyung-Won Kang for his help.

## References

[1] C. Maggioni, "A Novel Gestural Input Device for Virtual Reality," *Proc. IEEE Virtual Reality Ann. Int'l Symp.*, pp. 118-124, Seattle, Wash., 1993.
[2] K. Vaananen and K. Boehm, "Gesture Driven Interaction as a Human Factor in Virtual Environments—An Approach with Neural Networks," *Virtual Reality Systems*, R. Earnshaw, M. Gigante, H. Jones, eds., chapter 7, pp. 93-106. Academic Press, 1993.
[3] J. Davis and M. Shah, "Visual Gesture Recognition," *IEE Proc. Visualization and Image Signal Processing*, vol. 141, no. 2, pp. 101-106, Apr. 1994.
[4] C. Cedras and M. Shah, "Motion Based Recognition: A Survey," *Image and Vision Computing*, vol. 13, no. 2, pp. 129-155, Mar. 1995.
[5] M. Shah and R. Jain, *Motion-Based Recognition.* Kluwer Academic, 1997.
[6] F. Quek, "Toward a Vision-Based Hand Gesture Interface," *Proc. Virtual Reality Software and Technology Conf.*, pp. 17-31, Singapore, 1994.
[7] R. Kjeldsen and J. Kender, "Visual Hand Gesture Recognition for Window System Control," *Proc. Int'l Workshop Automatic Face- and Gesture-Recognition*, pp. 184-188, Zurich, Switzerland, 1995.
[8] T. Starner and A. Pentland, "Real-Time American Sign Language Recognition from Video Using Hidden Markov Models," Technical Report TR-375, MIT's Media Lab., 1995.
[9] R.C. Rose, "Discriminant Wordspotting Techniques for Rejection Non-Vocabulary Utterances in Unconstrained Speech," *Proc. IEEE Int'l Conf. Acoustics, Speech, and Signal Processing*, vol. II, pp. 105-108, San Francisco, 1992.
[10] K. Takahashi, S. Seki, and R. Oka, "Spotting Recognition of Human Gestures from Motion Images," Technical Report IE92-134, The Inst. of Electronics, Information, and Comm. Engineers, Japan, pp. 9-16, 1992 (in Japanese).
[11] T. Baudel and M. Beaudouin-Lafon, "CHARADE: Remote Control of Objects Using Free-Hand Gestures," *Comm. ACM*, vol. 36, no. 7, pp. 28-35, 1993.
[12] X.D. Huang, Y. Ariki, and M.A. Jack, *Hidden Markov Models for Speech Recognition.* Edinburgh: Edinburgh Univ. Press, 1990.
[13] L.D. Wilcox and M.A. Bush, "Training and Search Algorithms for an Interactive Wordspotting System," *Proc. Int'l Conf. Acoustics, Speech, and Signal Processing*, vol. II, pp. 97-100, San Francisco, 1992.
[14] T.M. Cover and J.A. Thomas, "Entropy, Relative Entropy and Mutual Information," *Elements of Information Theory*, pp. 12-49. John Wiley & Sons, 1991.
[15] L.R. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proc. IEEE*, vol. 77, pp. 257-285, 1989.
[16] S.H. Lee and J.H. Kim, "Augmenting the Discrimination Power of HMM by NN for On-Line Cursive Script Recognition," *Applied Intelligence*, vol. 7, no. 4, pp. 304-314, 1997.
[17] B.K. Sin and J.H. Kim, "Ligature Modeling for Online Cursive Script Recognition," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 19, no. 6, pp. 623-633, June 1997.
[18] J. Yamato, H. Ohya, and K. Ishii, "Recognizing Human Action in Time-Sequential Images Using Hidden Markov Model," *Proc. 1992 IEEE Conf. Computer Vision and Pattern Recognition*, pp. 379-385, 1992.
[19] A.J. Viterbi, "Error Bounds for Convolution Codes and an Asymptotically Optimum Decoding Algorithm," *IEEE Trans. Information Theory*, vol. 13, pp. 260-269, 1967.
[20] S. Kullback, *Information Theory and Statistics.* New York: Dover Publications, 1968.

[21] M. Hwang, X. Huang, and F.A. Alleva, "Predicting Unseen Triphones with Senones," *IEEE Trans. Speech and Audio Processing,* vol. 4, no. 6, pp. 412-419, 1996.

[22] I.J. Ko and H.I. Choi, "Hand Region Detection by Region Extraction and Tracking," *Proc. Korea Information Science Soc. Spring Conf.,* pp. 239-242, Taegu, Korea, 1996 (in Korean).

**Hyeon-Kyu Lee** received the BS degree in computer engineering from Seoul National University, Korea, in 1985 and the MS degree in computer science from KAIST, Korea, in 1987. In 1995, he undertook a PhD degree program in the Department of Computer Science at KAIST. He completed his PhD studies in August 1998. From 1987 to 1990, he was a junior researcher in the Division of Software Quality Assurance, Korea Telecom, Korea. In 1991, he joined the HandySoft Corp., Korea, as one of the founders and developed a Korean wordprocessor running on Windows™, Handy*Word Arirang. Notably, he received a government award from the Ministry of Science and Technology for the development of the word processor in 1994. Since September 1997, he has been in charge of the Department of Technology Planning at HandySoft. As an MSDN (Microsoft Developer's Network) Regional Director of Korea, he has supported Korean software developers who develop software products using Microsoft platforms from 1998 to March 1999. Since March 1999, he has worked for Microsoft Korea as a senior consultant.

**Jin H. Kim** received the BS degree in engineering from Seoul National University in 1971 and the MS and PhD degrees in computer science from the University of California, Los Angeles, in 1979 and 1983, respectively. He was a research engineer at the Korea Institute of Science and Technology (KIST) from 1973 to 1976, an engineering programmer for the County of Orange, California, from 1976 to 1977, and a senior staff member in computer science at Hughes Artificial Intelligence Center, Calabasas, California, from 1981 to 1985. He joined the faculty of KAIST in 1985. He was a visiting scientist at the IBM T.J. Watson Research Center in 1990. From 1995 to 1999, he was the president of Korea R&D Information Center (KORDIC). He was on several editorial boards, such as the *International Journal of Information Processing & Management*, the *International Journal of Autonomous Systems*, and the *International Journal of Chinese and Oriental Language Processing*. His research interests include pattern recognition, intelligent man machine interface, and AI for education.