Vis Comput (2016) 32:123–136
DOI 10.1007/s00371-014-1060-0

ORIGINAL ARTICLE

A visual attention-based method to address the midas touch problem existing in gesture-based interaction

Huiyue Wu · Jianmin Wang

Published online: 23 January 2015

Abstract The “Midas Touch” problem has long been a difficult problem existing in gesture-based interaction. This paper proposes a visual attention-based method to address this problem from the perspective of cognitive psychology. There are three main contributions in this paper: (1) a visual attention-based parallel perception model is constructed by combining top-down and bottom-up attention, (2) a framework is proposed for dynamic gesture spotting and recognition simultaneously, and (3) a gesture toolkit is created to facilitate gesture design and development. Experimental results show that the proposed method has a good performance for both isolated and continuous gesture recognition tasks. Finally, we highlight the implications of this work for the design and development of all gesture-based applications.

Keywords Gesture-based interaction · Dynamic gesture · Visual attention · Midas Touch problem

1 Introduction

With the rapid development of human–computer interaction (HCI), there has been a surge of interests in research on freehand gesture as an input modal in mobile, immersive and ubiquitous computing environments. Compared with regular mouse and keyboard inputs, freehand gesture inputs provide end users with more space, more freedom, and more lifelike interactive experiences. Therefore, freehand gestures are widely used by HCI practitioners to accomplish a variety of interactive tasks instead of traditional mouse or keyboard, including pointing and drawing [2,19,35], manipulating virtual objects [5,6,20,22,29], interacting with large displays [26,34]. Outside the scope of PC-based applications, freehand gestures have also been exploited in some specific domains like controlling home electronic devices, such as TV and CD player [33].

Although great progress has been made in recent years, gesture-based inputs have long been suffering from a difficult problem called the “Midas Touch” [9]. With direct sensing, the user’s actions could potentially always be “active”, i.e., everything the user does is interpreted as an interaction. To deal with this problem, many researches have been done by practitioners in HCI. However, traditional approaches, such as hidden Markov model (HMM), neural network (NN), and dynamic time warping (DTW), generally use a bottom-up or data-driven paradigm. Systems developed based on these methods lack the guidance of higher level modules and cannot tolerate inaccurate inputs from lower level modules. Therefore, the recognition accuracy will dramatically decrease due to incorrect spatiotemporal spotting result in a real setting.

To address this problem, this paper proposes a visual attention model that combines top-down and bottom-up attention using “what” and “where” information from the perspective of cognitive psychology. The primary contributions of our work are: (1) a visual attention model based on the combination of top-down and bottom-up attention, (2) a framework for dynamic gesture spotting and recognition, and (3) a toolkit for gesture design and development.

The rest of this paper is structured as follows: Sect. 2 reviews related work, Sect. 3 introduces the visual attention model, Sect. 4 presents the framework for gesture spotting

H. Wu (✉)
The School of Communication and Design,
Sun Yat-Sen University, Guangzhou, China
e-mail: wuhuiyue@gmail.com

J. Wang
User Experience Lab, The College of Arts and Media,
Tongji University, Shanghai, China
e-mail: wangjianmin@tongji.edu.cn

and recognition, Sect. 5 provides the toolkit and discusses the experimental results. After the discussion of the implications and limitations of our research, we conclude the paper with future research directions.

2 Related work

Despite advances in gesture-based interaction techniques, gesture spotting and recognition remains a challenging task in most of real-world scenarios. For example, system designers must avoid what is often called the "Midas Touch" problem, which refers to the phenomenon that every "active" hand action, even unintentional, could be recognized as a command by vision-based technology. Existing methods to address this problem can be roughly divided into two categories: (1) methods based on bottom-up visual cues reasoning, and (2) methods based on top-down semantic constraints.

2.1 Methods based on bottom-up visual cues reasoning

Some methods were adopted to address the Midas Touch problem based on bottom-up visual cues reasoning. For example, [16] proposed an HMM-based threshold model (i.e., garbage model or non-gesture model) for gesture spotting and recognition. Although good performance is reported on isolated gesture recognition tasks, their method is limited to off-line gesture recognition and sensitive to complex environmental conditions. Systems presented by Kölsch et al. [14] and Shen et al. [28] can recognize on-line gestures in unconstrained environments, but their systems only focused on static gesture recognition. In comparison, Yang et al. [37] proposed a method based on time-delay neural network, which can be used to recognize 2D dynamic gestures. Pedersoli et al. [23] developed an open-source package for recognition of both static hand postures and dynamic hand gestures. Similar to Lee et al.'s method [16], Yang et al. [36] and Peng and Qian [24] categorized gestures into communicative gestures and non-communicative gestures (garbage-gesture) and then used an HMM network for gesture spotting from live video feeds.

In summary, all the systems mentioned above adopt data-driven or bottom-up methods for gesture analysis and information reasoning, and lack effective top-down modules to accommodate inaccurate and ambiguous inputs from bottom-up modules.

2.2 Methods based on top-down semantic constraints

Other methods were proposed to overcome the Midas Touch problem using a mouse or keyboard button combined with top-down semantic constrains to initiate actions. The Camera Mouse [1], for example, used a combination of physical button and a dwell time threshold strategy to select items on the screen. However, the input speed will be mitigated if the dwell time is too long. On the contrary, the error rate will increase due to unintentional selection if the dwell time is too short. Compared with the strategy based on dwell time, the strategy based on spatial proximity was used for selection by other systems. The Shared Space [11], for example, supports virtual object manipulation based on object proximity and spatial relations. However, similar limitation exists in the Shared Space. Because the distance between the physical prop and the intended object is difficult to measure, a nearby unintended object is easy to be selected if the distance is too short. On the contrary, the intended object is difficult to select if the distance is too long. Recently, Liang et al. [17] exploited both temporal constraints and spatial features of input stream for gesture recognition. Different from the dwell time- and spatial proximity-based strategies, Kjeldsen et al. [12] designed some interface widgets to help the user perform different types of interactive tasks. A widget is related to some kind of interaction technique, such as trigger a command or adjust a parameter value. However, the number of widgets on the interface increases as the number of system commands and parameter values increases. Consequently, the increased number of widgets increases the system's space requirements and the user's cognition burden. Recently, a probability statistical method similar to the one presented in this paper is proposed by Kristensson and Nicholson [15]. Using a probabilistic reasoning algorithm in their system, the user's intended gestures are incrementally predicted while they are still being articulated. However, their system does not support continuous gesture recognition.

In summary, "Midas Touch" is a common problem existing in human–computer interaction. Conventional methods based on lower level data-driven or high-level semantic constraints primarily focus on improving the performance of algorithms from the perspective of data identification. Compared with human–computer interaction, the "Midas Touch" problem rarely appears in human–human interaction. Cognitive psychology studies have shown that we human beings have a visual processing system existing in our brains. Under the control of visual attention, the processing resources are distributed to key information while ignoring irrelevant information. This paper attempts to find a solution for the "Midas Touch" problem from a perspective of cognitive psychology. In contrast to previous work, we emphasize the hierarchical parallel perception model constructed by combining top-down and bottom-up visual attention.

3 The visual attention model

Attention is the behavioral and cognitive process of selectively focusing on one thing while ignoring others. It also refers to the allocation of processing resources. Generally speaking, visual attention is thought to operate as a two-stage

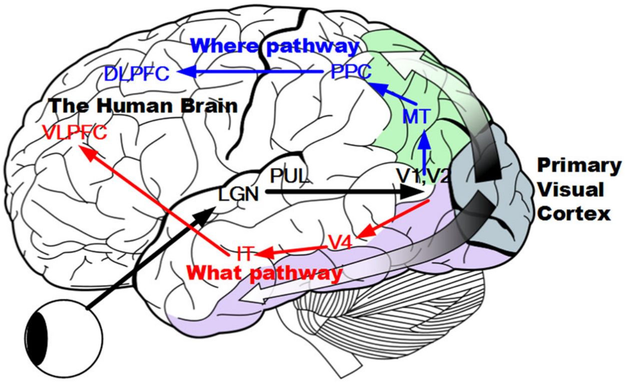


Fig. 1 "What" and "where" pathways

process [10,31]. In the first stage, attention is distributed uniformly over the external visual scene and visual information is processed in parallel. In the second stage, attention is concentrated on a specific area of the visual scene. According to [32], visual information is gathered through two types of pathways as being complementary, namely "what" and "where" pathways. The "what" pathway is thought to operate as a data-driven or bottom-up processing of information [13], and the "where" pathway is thought to operate as a goal-driven or top-down processing of information [8]. Visual information transferred through the two pathways indicating what objects in visual scene and where they are (Fig. 1). This theory of visual system shares the same opinion with Marr's [18].

Based on the "what–where" information pathways' theory, we propose a hierarchical parallel perception model that integrates top-down and bottom-up inference for visual information (Fig. 2).

As shown in Fig. 2, visual information flows both top-down and bottom-up. In the bottom-up direction, static features such as color, brightness, and shape are extracted for hand detection ("what" information) from the video input. Then the features are fed into the subsequent hand tracker for spatial localization (lower level "where" information). In the top-down direction, priori knowledge provided by the gesture and non-gesture models is used as a guidance for gesture spotting, i.e., when a dynamic gesture starts and ends (higher level "where" information). Finally, the dynamic gesture trajectory is fed into the pattern recognition module for gesture classification.

Accordingly, selective attention, pre-attention, and sustained attention modules are defined to constitute such a visual information processing system by integrating the theory of "what–where" pathways with visual attention mechanism:

- The selective attention module consists of color detector and position detector. The idea in selective attention is that not all objects in the visual scene give us information and focusing only on the relevant parts of the scene while ignoring other irrelevant stimuli. The selective attention procedure is performed with feature extraction and feature integration modules. Similar to the V_1 region in the human brain's visual information processing system (Fig. 1), the

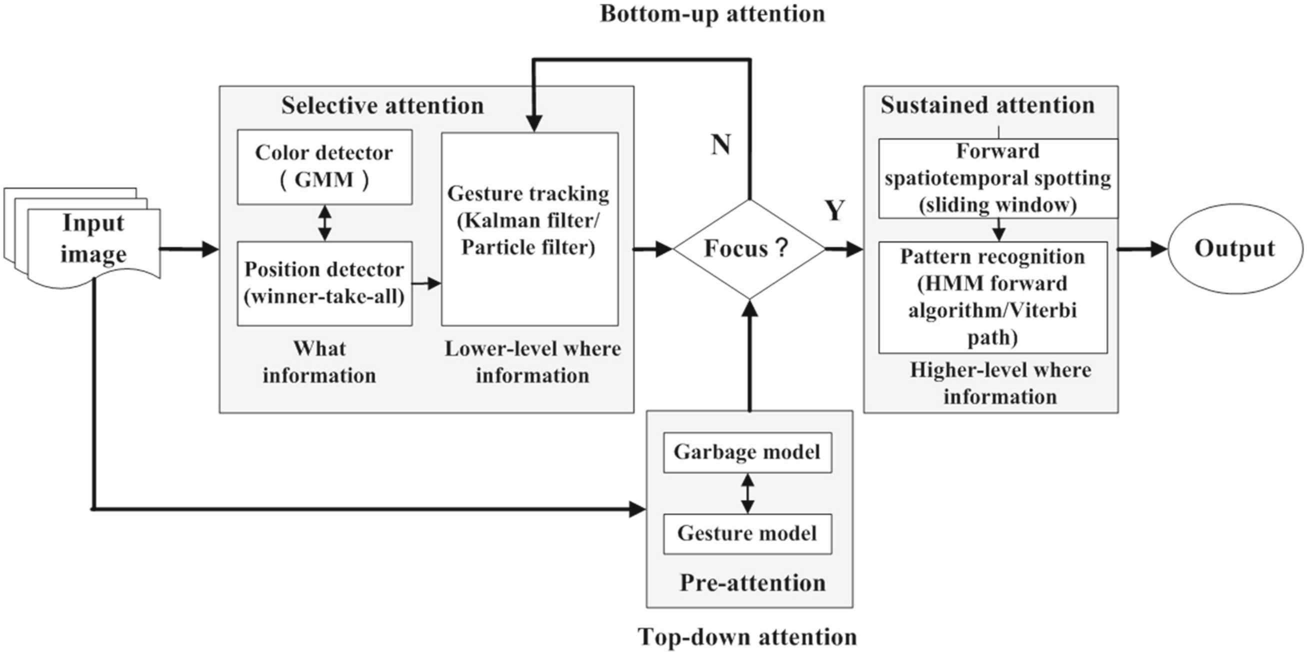

Bottom-up attention
Selective attention
Color detector (GMM)
Position detector (winner-take-all)
Gesture tracking (Kalman filter/ Particle filter)
What information
Lower-level where information
Input image
N
Y
Focus ?
Sustained attention
Forward spatiotemporal spotting (sliding window)
Pattern recognition (HMM forward algorithm/Viterbi path)
Higher-level where information
Output
Garbage model
Gesture model
Pre-attention
Top-down attention

Fig. 2 Architecture of the visual attention model

feature extraction module primarily extracts lower level visual cues such as skin color, brightness, shape, edge, location and orientation using relevant machine vision algorithms (e.g., Gaussian mixture model). Similar to the V_2–V_3 regions, the feature integration module identifies the hand from the background based on the reasoning of different visual cues using appropriate heuristic algorithm or inference rule (e.g., a "Winner-Take-All" strategy).

- The pre-attention module is responsible for task- or goal-relevant information extraction. It can be used to guide bottom-up attention and direct attention to the sustained attention region. Heuristic algorithms are commonly used by the pre-attention module to mimic task-driven or top-down attention mechanism. In this study, a garbage model is constructed in the pre-attention module to guide higher level "where" information extraction.
- The sustained module consists of spatiotemporal spotting and pattern recognition modules for higher level "where" information processing. In the spatiotemporal spotting module, predictive information provided by the pre-attention module is used for both spatial and temporal gesture spotting. Spatial gesture spotting refers to the detection of where a dynamic gesture occurs, and temporal gesture spotting determines when a dynamic gesture starts and ends. Once a start point is detected, the segmented part is incrementally sent to the pattern recognition module until an end signal is received. The final gesture classification result is determined using appropriate pattern recognition algorithm (e.g., HMM, DTW, or NN).

We apply the Bayes' rule to construct the visual attention model. Let G be a dynamic gesture, T is the gesture type, $L = (L_s, L_e)$ denotes the higher level "where" information, where L_s and L_e is the start and end point, respectively. Given the environment context information E, the conditional probability of the dynamic gesture G is computed as:

$$P(G|E) = P(L|T, E)P(T|E) \tag{1}$$

where, the likelihood functions $P(T|E)$ and $P(L|T, E)$ correspond to the pre-attention and sustained attention stage, respectively. Given environmental context information E, the likelihood function $P(T|E)$ provides a priori knowledge for the most likely gesture type T. If the estimated value of $P(T|E)$ is greater than a threshold value ξ, then activate the sustained attention module and distribute processing resources to the sustained attention region that is most likely to contain the dynamic gesture (higher level "where" information). Otherwise, stop sustained attention. $P(T|E)$ is computed as follows:

$$P(T|E) = \frac{P(E|T)P(T)}{P(E|T) + P(E|\neg T)P(\neg T)} \tag{2}$$

Given environmental context information E and gesture type T, the likelihood function $P(L|T, E)$ denotes the probability of a dynamic gesture articulated along the path L. Under the influence of $P(L|T, E)$ the higher level "where" information (i.e., dynamic gesture trajectory) is extracted from the lower level "where" information and fed into the recognition module. $P(L|T, E)$ is computed as follows:

$$P(L|T, E) = \frac{P(L, E|T)}{P(E|T)} \tag{3}$$

4 The framework for gesture spotting and recognition

4.1 Selective attention module

Selective attention module consists of a color detector and a position detector. The color detector is responsible for skin color detection using the Gaussian mixture model in YC_rC_b color space. To reduce the negative impact caused by changing lighting conditions, we use only the chrominance components C_rC_b while ignoring the brightness component Y. Let $\gamma = [C_r, C_b]^{\mathrm{T}}$ be the chrominance component of an input pixel, u_0 be the mean (C_r, C_b) of the input pixel, and k_0 be the covariance of the input pixel's distribution. Then, each pixel is classified as skin or non-skin by using the Gaussian mixture model based on a large database of skin and non-skin pixels, respectively. The K-means clustering algorithm is used for model training. Given a m dimensional observation vector X, the Gaussian density model is established as follows:

$$P(X) = \frac{\exp\left[-\frac{1}{2}(X - u_0)^{\mathrm{T}} k_0^{-1}(X - u_0)\right]}{(2\pi)^{\frac{m}{2}} |k_0|^{\frac{1}{2}}} \tag{4}$$

The position detector is used to extract the spatial location information of an input pixel. Combining the color information and the position information, an input pixel is described as a six-tuple $\langle Y, C_b, C_r, x, y, z\rangle$. In real settings, there may be n candidate regions in the same frame, such as a hand, a face or other kind of skin color distractors. Feature vector Q_{ij} = (color, position, velocity) denotes the candidate region j in frame i, where, color denotes skin color characteristic, position $= (x_{ij}, y_{ij}, z_{ij})$ denotes the centroid of the candidate region j, and velocity $= (u_{ij}, v_{ij}, w_{ij})$ denotes the motion speed of the centroid in the world coordinate system.

In the selective attention module, we associate each candidate region in the visual field with a value that is a function of the response of the color and position detectors and the relative importance of the particular feature to the target task being solved. When the system attempts to determine whether an object is a hand, it runs through all existing candidate regions, and then find the strongest of these responses

Table 1 A truth table for identifying a hand

Candidate region	Color	Position	Velocity
ID #1	+	Ambiguous	+
ID #2	−	Ambiguous	+
ID #3	+	−	Ambiguous
ID #4	Ambiguous	−	Ambiguous

and this region becomes the focus of attention using a "truth table" and the "Winner-Take-All" strategy. In addition to the proposed method, the reader can also refer to other proposals for selective attention processing in computer vision, such as the saliency map used by Treisman and Gelade [31], Itti [8], Salah et al. [27], and Tian [30], and some competitive schemes adopted by Koch and Ullman [13], and Itti [7].

In the example shown in Table 1, when the color characteristic is compared, #2 is rejected and #1, #3 and #4 are left as candidates. Comparing the position characteristic, #3 and #4 are eliminated. And now #1 is still left as an option, because a positive response is still missing. Therefore, the process continues until #1 has a positive response by comparing the velocity characteristic. Since it is the only positive response left in the candidate set, #1 is identified as a hand. Consequently, feature vector $Q^* = \max[Qij]$ is computed as the "what" information and lower level "where" information of the hand.

Once a hand is successfully identified, a blob is produced to analyze the edge, bounding box, and centroid of the hand region. Each blob has a detailed representation of its appearance and shape. In the tracking phase, the hand region is re-estimated and the centroid point is calculated in a new image using the Kalman filter in conjunction with the Camshift algorithm. Consequently, connecting all the centroid points produces the hand motion trajectory in the visual scene (lower level "where" information).

4.2 Sustained attention module

Compared with Lee et al.'s method [16], a forward spotting strategy is used in this paper. Under the guidance of the pre-attention module, the sustained attention module focuses on pattern recognition of the dynamic gesture trajectory segmented from the lower level "where" information.

Location, velocity, and orientation are three basic features for a dynamic gesture. Among them, the orientation feature is proved to be the best feature in terms of recognition accuracy [4,16,21]. Therefore, we use the orientation feature as a main local feature in this study. The orientation of hand movement is determined by the angle θ between two consecutive points $P_t = (x, y)$ and $P_{t+1} = (x', y')$ of the hand motion trajectory:

$$\theta = \tan^{-1}\left(\frac{y' - y}{x' - x}\right) \tag{5}$$

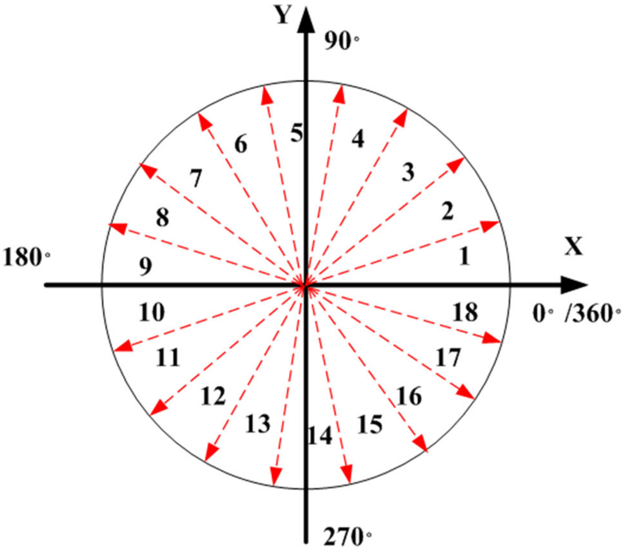


Fig. 3 Directional codewords

Consequently, a feature vector is produced by converting the orientation to one of the 18 directional codewords (Fig. 3), and then fed into the gesture HMM.

Considering the strong sequential relationship between two consecutive strokes in a dynamic gesture, the HMM based on left–right banded (LRB) topology is used for gesture modeling (Fig. 4). A state in a gesture HMM corresponds to a stroke in a dynamic gesture. Each state can move to the next state or stay unchanged as time increases. The transition between two states represents the sequential relationship between two strokes of a gesture. The number of the states in a gesture HMM is determined by the complexity of the gesture shape. All gesture HMMs are trained using the Baum-Welch algorithm.

When the system is running, an observation sequence $O = O_1 O_2 O_3 \ldots O_T$ is obtained by extracting feature information from the input video. The Viterbi algorithm [25] is used for gesture recognition:

Step 1. Initialization ($t = 1$; $1 \leq i \leq N$):

$$\delta_1(i) = \prod_i b_i(O_1)$$
$$\psi_1(i) = 0$$

Step 2. Recursion ($2 \leq t \leq T$; $1 \leq j \leq N$):

$$\delta_t(j) = \max_i \delta_{t-1}(i) a_{ij} b_j(O_t)$$
$$\psi_t(j) = \arg\max_i \delta_{t-1}(i) a_{ij}$$

Step 3. Termination:

$$p(O|\lambda_k) = \max_i \delta_T(i)$$
$$q_T^* = \arg\max_i \delta_T(i)$$

Fig. 4 The gesture HMMs

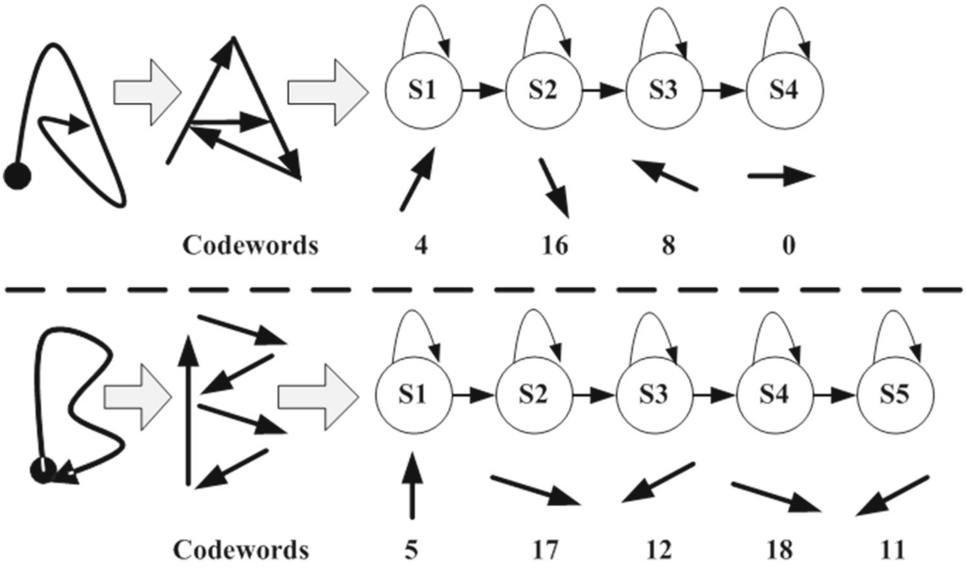
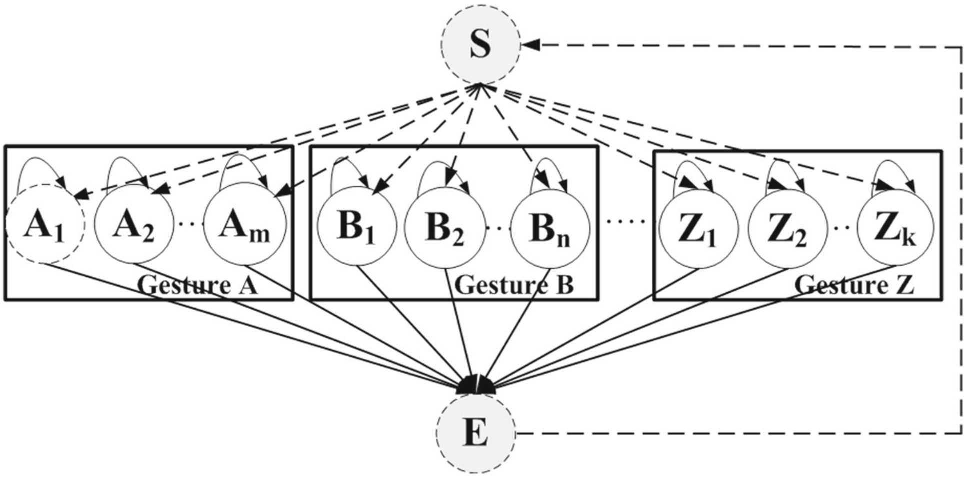


Fig. 5 The garbage model



where, Π_i is the initial probability distribution, a_{ij} is the transition probability between state i and j, $b_j(O_t)$ is the observation probability of state j at time t, $\delta_t(j)$ is the maximum likelihood value of state j at time t, N is the sum of states of all gesture HMMs.

4.3 Pre-attention module

In Eq. 2, the likelihood function $P(T|E)$ is computed based on a training process. The output is used as a priori knowledge for the determination of what types of gestures are most likely to appear in such environmental context E. The training set for $P(E|T)$ consists of a large number of video clips that contain the complete dynamic gesture trajectory. Therefore, we approximate the priori probability by $P(T) = P(\neg T) = 50\,\%$.

In this study, a fully connected garbage model is constructed to represent all the meaningless motion trajectories (e.g., the transition motion trajectory before or after a dynamic gesture). In the garbage model, each state can be reached from all other states [16,36]. The procedure for constructing the garbage model is given as follows:

Step 1. Copy all states i from all gesture HMMs, each with its observation probability $b_i(k)$. Then use a Gaussian filter to smooth the observation probabilities.

Step 2. Define two dummy states, named *Start* state (S) and *End* state (E). Then connect the two states with the other states produced in Step 1, see Fig. 5.

Step 3. Set the self-transition probability of each state to be the same as in the gesture HMMs.

Step 4. Compute the transition probability between a dummy state and a non-dummy state as follows:

$$\begin{aligned} a_{Si} &= \frac{1}{N} \\ a_{iE} &= 1 - a_{ii} \end{aligned} \tag{6}$$

where, a_{Si} is the transition probability from the dummy state S to non-dummy state i, a_{iE} is the transition probability from non-dummy state i to the dummy state E, a_{ii} is the self-transition probability of state i, N is the sum of states of all gesture HMMs.

Step 5. The transition probability between two non-dummy states i and j is computed as:

$$\overrightarrow{a_{ij}} = \frac{1 - a_{ij}}{N - 1}, \quad \text{where } i \neq j \tag{7}$$

where $\overrightarrow{a_{ij}}$ is the transition probability of the garbage model from state i to j, a_{ij} is the transition probability of the gesture HMMs from state i to j, N is the sum of states of all gesture HMMs.

As shown in Fig. 5, the number of states of the garbage model increases as the number of the gesture HMMs increases. Since there are many states with similar distribution in the garbage model, the increase of the number of states will cause noting but a waste of time and space [16]. Therefore, the relative entropy measure [3] is used to reduce the states in the garbage model. Let $M = \{M(k)|k \in \aleph\}$ and $N = \{N(k)|k \in \aleph\}$ be two discrete probability distributions, where $\aleph$ is the set of observation symbols, the symmetric relative entropy $D(M||N)$ is defined as:

$$D(M||N) = \frac{1}{2}\sum_{k}\left(M(k)\log\frac{M(k)}{N(k)} + N(k)\log\frac{N(k)}{M(k)}\right) \tag{8}$$

Then the state-reduction procedure is given as follows:

Step 1. Initialization.

Compute the symmetric relative entropy for each pair of observation probability distributions $M^{(i)}$ and $N^{(j)}$of state i and j.

$$D(M^{(i)}||N^{(j)}) = \frac{1}{2}\sum_{k}\left(M(k)^{(i)}\log\frac{M(k)^{(i)}}{N(k)^{(j)}} + N(k)^{(j)}\log\frac{N(k)^{(j)}}{M(k)^{(i)}}\right) \tag{9}$$

Step 2. Comparison and selection.

Compare all state pairs and find $(\bar{i}, \bar{j}) = \arg\min D(M^{(\bar{i})}||N^{(\bar{j})}), \bar{i} \neq \bar{j}$

Step 3. Merging and update.

Merge state $\bar{i}$ and $\bar{j}$ and update the probability distribution. $M_k^{(\bar{\bar{i}})} = \frac{M_k^{(\bar{i})}+N_k^{(\bar{j})}}{2}$

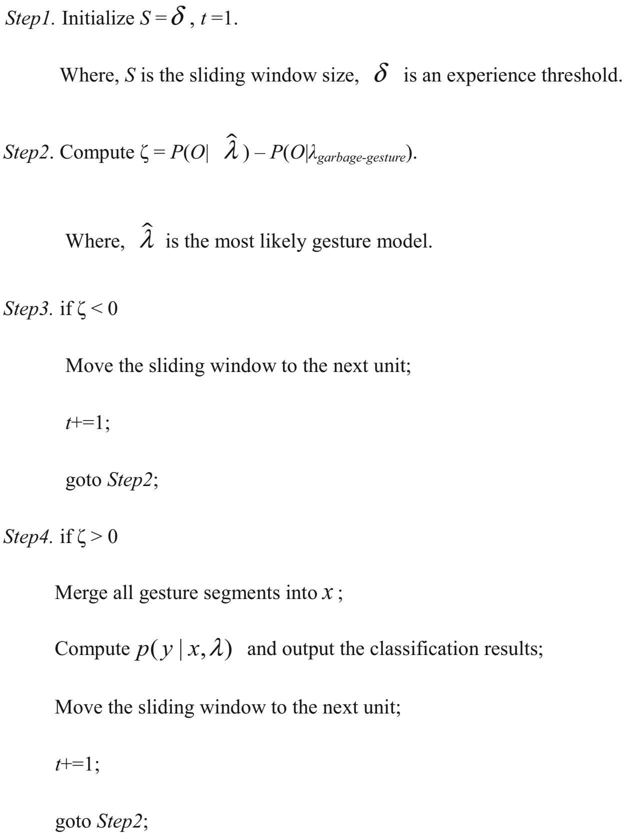
Step 4. Loop until $N < \delta$, where N is the sum of states in the garbage model, δ is a threshold value.

Compared with the gesture HMMs, the garbage model is a weak model due to the smaller forward transition probability. Therefore, the garbage model provides a lower threshold to the model likelihood for a given hand motion trajectory to be accepted as a dynamic gesture. For gesture spotting, the garbage model and the gesture HMMs satisfy the following conditions:

$$\begin{aligned} P(O_g|\lambda_g)P(g) &> P(O_g|\lambda_{\text{garbage-gesture}})P(\hat{O}_k) \\ P(\hat{O}_k|\lambda_g)P(g) &< P(\hat{O}_k|\lambda_{\text{garbage-gesture}})P(\hat{O}_k) \end{aligned} \tag{10}$$

where, O_g denotes the observation sequence of a gesture pattern, λ_g denotes the gesture HMM, $\hat{O}_k$ denotes the observation sequence of a non-gesture pattern, $\lambda_{\text{garbage-gesture}}$ denotes the garbage model. As can be seen from in Eq. 10, the garbage model provides a confidence limit for rejecting or accepting a non-gesture pattern.

The sliding window technique is used to calculate the observation probability of the gesture HMMs and the garbage model. The procedure is given as follows:



```
Step1. Initialize $S=\delta$, $t=1$.
    Where, $S$ is the sliding window size, $\delta$ is an experience threshold.
Step2. Compute $\zeta = P(O|\hat{\lambda}) - P(O|\lambda_{garbage\text{-}gesture})$.
    Where, $\hat{\lambda}$ is the most likely gesture model.
Step3. if $\zeta < 0$
    Move the sliding window to the next unit;
    $t$+=1;
    goto Step2;
Step4. if $\zeta > 0$
    Merge all gesture segments into $x$ ;
    Compute $p(y|x,\lambda)$ and output the classification results;
    Move the sliding window to the next unit;
    $t$+=1;
    goto Step2;
```

5 System prototype and experimental evaluation

5.1 System prototype

Based on the visual attention model, we developed a toolkit for dynamic gesture training and recognition. We provide no detail about the implementation of the system in this paper,

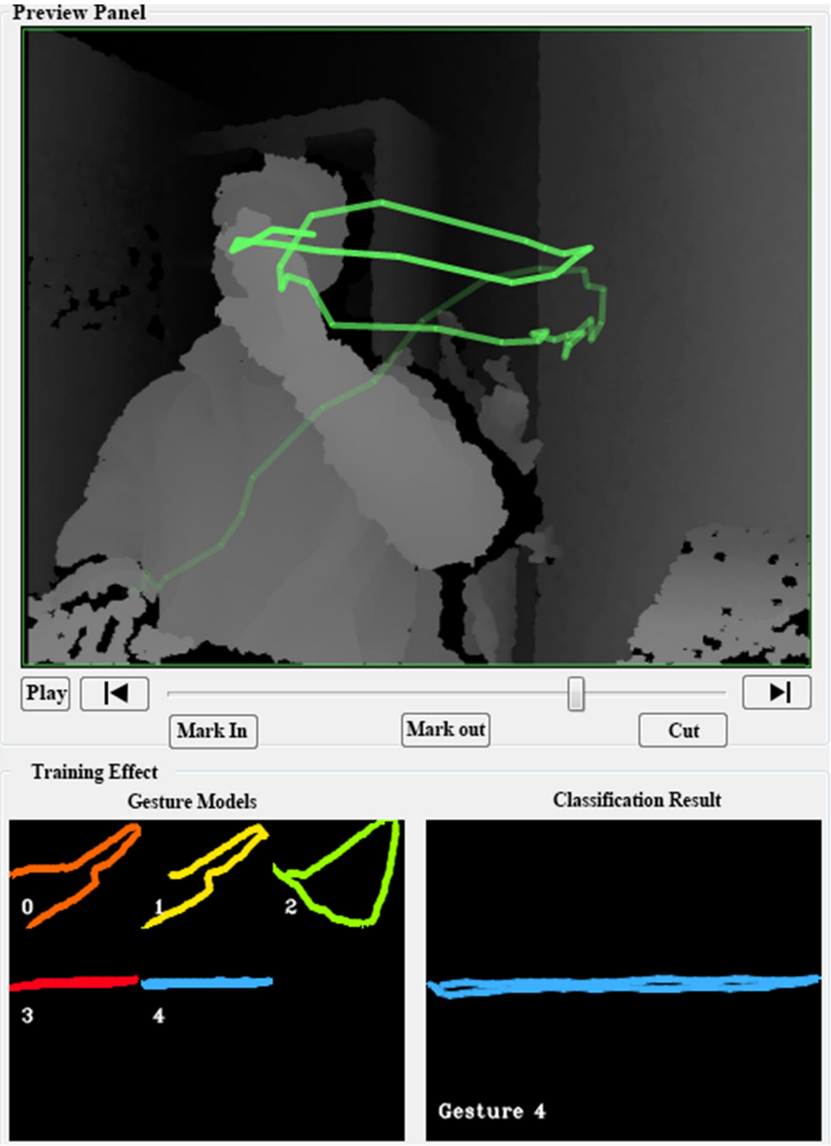


Fig. 6 Off-line gesture training based on the toolkit

because of our focus on this paper on the visual attention model, rather than the development of the system. Technical details of system implementation can be found in Wu et al. [38].

In the training part, the selective attention model, pre-attention model, and sustained attention model are off-line trained using the toolkit based on a large database of video clips collected from seventy subjects from a university (Fig. 6).

In the recognition part, the user's gestures are on-line recognized. The recognition results are encapsulated into higher level gesture events. A dynamic gesture event (GE) is defined as follows:

$$GE = \langle ID, Type, \text{“}name\text{”}, startPt, endPt, t, r, data, sampleRate\rangle$$

where *ID* is the unique identification of a dynamic gesture, *Type* is the gesture type, "*name*" is the name of a dynamic gesture, usually defined by the user, *startPt* and *endPt* are the start point and end point of the gesture trajectory determined by the pre-attention model, *t* is the duration of a dynamic gesture, which can be used to distinguish between short gestures and long gestures, *r* is a flag indicating whether a gesture is successfully recognized, *data* is a float array used to store the dynamic gesture trajectory, *sampleRate* is the sampling rate.

5.2 Experimental evaluation

An experiment was conducted to justify the proposed method. The test environment had a 42-in. LCD display, a depth camera, and a PC which hosted the system to process

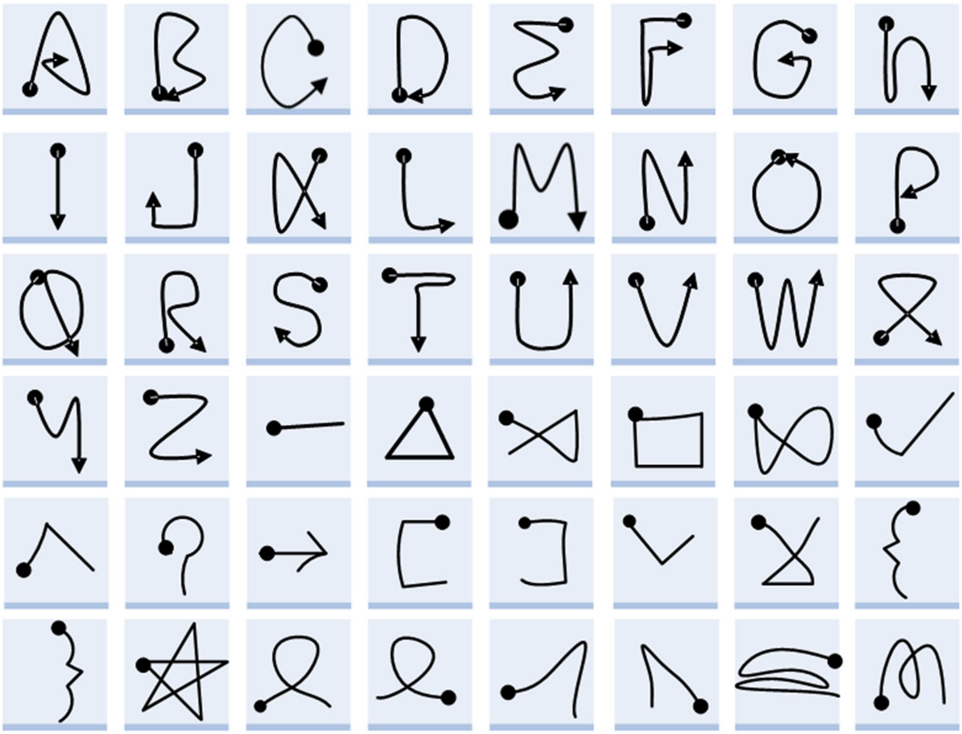


Fig. 7 Gesture vocabulary. The beginning of a gesture is indicated by a *solid dot*

gesture inputs from the camera and deliver contents to the LCD display. The PC had a 2.9 GHz CPU, 8 GB memory, and a 2T hard disk. The experiment consisted of two parts: off-line test and on-line test.

We tested the off-line recognition accuracy on two gesture sets: the set of one-handed gestures used in Kristensson et al.'s study, and the gesture set used in Lee et al.'s study. After excluding five duplicate gestures that appeared in both sets, we defined 48 gestures in the final gesture set (Fig. 7). For each gesture, seventy isolated gesture samples were collected using the toolkit shown in Fig. 6. Using a cross-validation method, the first 45 input samples were used for training and the last 25 input samples were used for testing. Therefore, our database contained $70 \times 48 = 3{,}360$ gesture samples, where 2,160 samples were used for training and 1,200 samples were used for testing.

Next, the on-line test was conducted based on the same gesture vocabulary. But different from the off-line test, continuous gesture samples were collected in this test (e.g., Fig. 8).

As shown in Fig. 8, the non-gesture (N-G) pattern has the greatest probability from frame 80 to 99, therefore, all gesture patterns are rejected by the system. To take a closer look at the input stream, a transition stroke (garbage-gesture) before the gesture "*S*" is performed in this stage. From frame 99 to 145, the probability of the non-gesture pattern nearly drops to zero and there comes the gesture "*S*". Next, the above curve from frame 145 to 163 indicates a transition stroke between two continuous gestures "*S*" and "*T*", the above curve from frame 163 to 184 indicates the gesture "*T*", and the above curve from frame 184 to 200 indicates a non-gesture pattern.

To test the performance of the spotting algorithm for continuous gestures, we defined three types of errors: insertion error (detect a non-existent gesture), deletion error (fail to detect a gesture), and substitution error (falsely classify a gesture). Following the convention, the system performance was measured in terms of the three types of errors and the reliability. The average recognition accuracy is 93.47 %, as shown in the bottom row of Table 2.

For a comparison with existing methods, we provide the results by the method proposed by Lee and Kim [16] and Kristensson et al. [15]. Lee et al. used a backward spotting algorithm which first detects the end point of a dynamic gesture and then tracks back to find the start point. Compared with Lee et al.'s work, a forward spotting algorithm is used for gesture spotting and recognition in this paper. Table 3 illustrates the difference between the two methods tested on the same ten gestures used in Lee et al's experiment. As for the recognition accuracy for isolated gestures and the spotting accuracy for continuous gestures, no significant difference is found between the two methods. However, the average speed of the forward spotting algorithm used in our method and the average speed of the backward spotting algorithm used in Lee et al.'s method are 0.33 seconds (SD = 0.10) and 0.71 s (SD = 0.14), respectively. Using a matched-pair t test, we find the difference in the spotting speed between the two methods is significant, $t_{25} = -23.047$, $p < 0.001$.

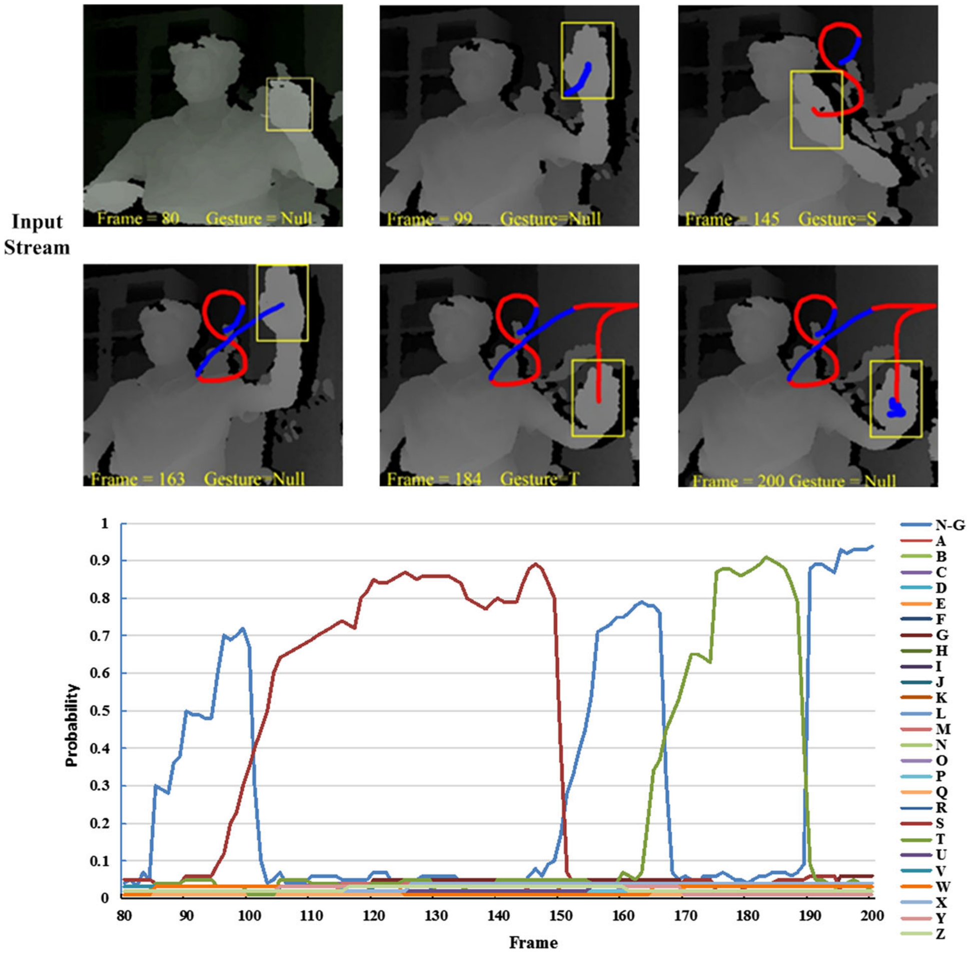


Fig. 8 Continuous gesture recognition

In a study with aspects similar to ours, Kristensson et al. (2012) used a probabilistic algorithm to incrementally predict the user's intended gestures while they are still being articulated. Table 4 shows the difference between Kristensson et al.'s method and our method tested on the same 43 one-handed gestures used in Kristensson et al.'s study.

We discuss the experimental results from the following four aspects:

(1) For isolated one-handed gestures, our method achieved higher recognition rate (97.21 %) than that obtained using Kristensson et al.'s method (92.7 %). For isolated two-handed gestures, the recognition accuracy is slightly higher than that obtained for isolated one-handed gestures in Kristensson et al.'s method. The primary reason is that additional information can be provided by two simultaneous input gestures to the recognizer under an appropriate probabilistic model. Our method, by contrast, does not support two-handed gesture recognition.

(2) Due to the lack of the guidance of pre-attention mechanism, Kristensson et al.'s method does not support continuous gesture recognition. In comparison, our method achieved an average recognition rate of 93.33 %.

(3) The average speed of gesture spotting is 0.35 s using Kristensson et al.'s method and 0.33 s using our method. No significant difference is found between the two methods. Benefited from the use of forward spotting scheme, both methods are faster than the method proposed by Lee et al. [16].

(4) Kristensson et al.'s method achieved an average prediction accuracy of 46 and 80 % when the complete gestures had been articulated 20 and 80 %, respectively. In comparison, the average prediction accuracy is 60 and 95 % using our method. But, it is worth noting that the prediction accuracy for some gestures, such as *I* and *J*, *N* and *M*, *O* and *Q*, and *V* and *W*, is well below the average value due to the sub-gesture problem, i.e., some gestures are very similar to sub-gestures of other gestures (Fig. 7). Therefore, the system is prone to make mistakes before a gesture is completed, for example, falsely matching the gesture model "*V*" with a sub-gesture of "*W*".

Table 2 Spotting results with the visual attention model

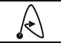
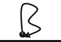
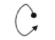
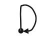
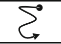
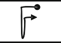
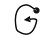
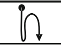
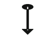
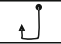
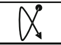
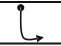
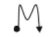
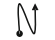
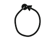
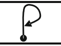
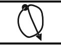
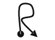
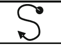
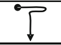
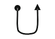
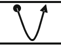
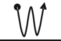
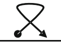
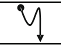
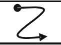
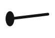
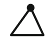
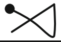
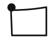
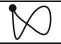
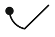
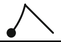
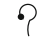
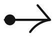
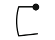
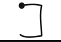
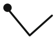
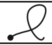
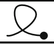
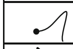
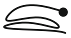
Gesture	Training Data	Isolated gestures			Continuous gestures						
		Test data	Correct	Rec. (%)	Test data	Insert error	Delete error	Substitute error	Correct	Det. (%)	Rel. (%)
	45	25	25	100	30	1	0	1	29	96.7	93.5
	45	25	24	96	30	0	1	1	28	93.3	93.3
	45	25	25	100	30	0	1	0	29	96.7	96.7
	45	25	24	96	30	1	1	3	26	86.7	83.8
	45	25	25	100	30	0	1	0	29	96.7	96.7
	45	25	23	92	30	0	0	3	27	90	90
	45	25	24	96	30	0	0	2	28	93.3	93.3
	45	25	25	100	30	0	1	0	29	96.7	96.7
	45	25	25	100	30	2	3	1	26	86.7	81.3
	45	25	25	100	30	1	1	1	28	93.3	90.3
	45	25	25	100	30	0	0	1	29	96.7	96.7
	45	25	25	100	30	0	2	0	28	93.3	93.3
	45	25	24	96	30	0	0	2	28	93.3	93.3
	45	25	24	96	30	0	0	2	28	93.3	93.3
	45	25	24	96	30	0	1	2	27	90	90
	45	25	24	96	30	1	1	2	27	90	87.1
	45	25	24	96	30	0	0	2	28	93.3	93.3
	45	25	24	96	30	0	1	0	30	100	100
	45	25	25	100	30	1	1	3	26	86.7	83.9
	45	25	24	96	30	1	2	2	26	86.7	83.9
	45	25	23	92	30	1	1	2	27	90	87.1
	45	25	23	92	30	0	0	3	27	90	90
	45	25	24	96	30	0	1	1	28	93.3	93.3
	45	25	25	100	30	0	0	1	29	96.7	96.7
	45	25	25	100	30	1	0	0	30	100	96.8
	45	25	24	96	30	0	1	1	28	93.3	93.3
	45	25	25	100	30	2	3	2	25	83.3	78.1
	45	25	25	100	30	0	0	1	29	96.7	96.7
	45	25	25	100	30	0	0	1	29	96.7	96.7
	45	25	24	96	30	0	1	1	28	93.3	93.3
	45	25	25	100	30	0	1	0	29	96.7	96.7
	45	25	23	92	30	1	2	3	25	83.3	80.6
	45	25	23	92	30	0	0	1	29	96.7	96.7
	45	25	24	96	30	1	0	0	30	100	96.8
	45	25	23	92	30	1	1	1	28	93.3	90.3
	45	25	25	100	30	0	0	1	29	96.7	96.7
	45	25	25	100	30	1	0	1	29	96.7	93.5
	45	25	23	92	30	2	2	4	24	80	75

Table 2 continued

	45	25	24	96	30	1	1	1	28	93.3	90.3
	45	25	25	100	30	0	0	0	30	100	100
	45	25	25	100	30	0	0	1	29	96.7	96.7
	45	25	25	100	30	0	0	0	30	100	100
	45	25	25	100	30	0	0	0	30	100	100
	45	25	24	96	30	1	0	1	29	96.7	93.5
	45	25	24	96	30	0	1	2	27	90	90
	45	25	25	100	30	0	0	1	29	96.7	96.7
	45	25	24	96	30	0	1	1	28	93.3	93.3
	45	25	24	96	30	1	1	2	27	90	87.1
Total	**2160**	**1200**	**1167**	**97.25**	**1440**	**21**	**34**	**61**	**1346**	**93.47**	**92.13**

Rec. = Number of correctly recognized gestures / number of input gestures
Rel. = Number of correctly recognized gestures/(number of input gestures + number of insertion errors)
Det. = Number of correctly recognized gestures/number of input gestures

Table 3 Comparison between Lee et al.'s method and the proposed method

	Lee et al.'s method	Proposed method
Recognition accuracy for isolated gestures (%)	98.19	98.0
Spotting results for continuous gestures		
Detection (%)	93.81	94.68
Reliability (%)	93.14	94.07
Speed of gesture spotting (s)	0.71	0.33

Recognition accuracy = number of correctly recognized gestures/number of input gestures
Reliability accuracy = number of correctly recognized gestures/(number of input gestures + number of insertion errors)
Detection accuracy = number of correctly recognized gestures/number of input gestures

Table 4 Comparison between Kristensson et al.'s method and the proposed method

	Kristensson et al.'s method	Proposed method
Recognition rate for isolated gestures (%)		
One-handed gesture	92.7	97.21
Two-handed gesture	96.2	Unsupported
Continuous gesture recognition	Unsupported	93.33
Speed of gesture spotting (s)	0.35	0.33
Prediction accuracy before the gesture is completed (%)		
1/5 of the complete gesture	46	60
4/5 of the complete gesture	80	95

6 Conclusion and future work

In gesture-based interaction in the real world, one of the challenges is to overcome the "Midas Touch" problem. In this paper, a hierarchical parallel perception model is proposed based on the human brain's visual attention mechanism. Different from previous work, visual information flows both top-down and bottom-up in the proposed model. Based on the visual attention model, a unified framework is introduced for hand detection, spatiotemporal spotting and pattern recognition using the selective attention module, pre-attention module, and sustained attention module, respectively. Experimental results show that the proposed method can achieve a high recognition performance for both isolated and continuous dynamic gestures.

The main contribution of this paper is the visual attention-based hand gesture recognition framework to address the Midas Touch problem existing in most of real-world scenarios. Without loss of generality, we provided some basic methods and techniques based on this framework for system designers to develop hand gesture-based applications. The proposed method can also work as a reference framework for the design and development of various hand gesture-based applications, such as intelligent household appliances (music player, air conditioning, etc.), PC-based interactive systems (computer games, virtual/augmented reality systems), and robots.

It should be noted that our method offers a general framework for developing hand gesture-based applications. The methods and techniques we presented here exemplify an implementation of this framework in design. Other alternatives, hand gesture-based algorithms and tools can also be used to replace our methods.

Our future work can be extended in the following directions. First, we will continue improving the performance of the proposed method and make it more efficient and robust in real settings. Second, we will extend our method to support two-handed gesture recognition and bimanual interaction technologies. Third, we will explore a reasonable mechanism for sub-gesture reasoning. Furthermore, we are also interested in verifying the performance of the proposed method for more general applications in real-world scenarios.

Acknowledgments We thank the financial support from the National Natural Science Foundation of China, No. 61202344; the Fundamental Research Funds for the Central Universities, Sun Yat-Sen University, No. 1209119; Special Project on the Integration of Industry, Education and Research of Guangdong Province, No. 2012B091000062; the Fundamental Research Funds for the Central Universities, Tongji University, No. 0600219052, 0600219053. We would like to express our great appreciation to editor and reviewers.

References

1. Betke, M., Gips, J., Fleming, P.: The Camera Mouse: visual tracking of body features to provide computer access for people with severe disabilities. IEEE Trans. Neural Syst. Rehabil. Eng. **10**(1), 1–10 (2002)
2. Colaco, A., Kirmani, A., Yang, H.S., Gong, N.W., Schmandt, C., Goyal, V.K. Mime: Compact, low-power 3D gesture sensing for interaction with head-mounted displays. In: Proceedings of the ACM Symposium of User Interface Software and Technology (UIST'13), pp. 227–236 (2013)
3. Cover, T.M., Thomas, J.A.: Entropy, Relative Entropy and Mutual Information. Elements of Information Theory. Wiley, New York (1991)
4. Elmezain, M., Hamadi, A.A., Michaelis, B.: Improving hand gesture recognition using 3D combined features. In Second International Conference on Machine Vision, pp. 128–132 (2009)
5. Feng, Z.Q., Zhang, M.M., Pan, Z.G., Yang, B., Xu, T., Tang, H.K., Li, Y.: 3D-Freehand-pose initialization based on operator's cognitive behavioral models. Vis. Comput. **26**(6–8), 607–617 (2010)
6. Hilliges, O., Izadi, S., Wilson, A.D., Hodges, S., Mendoza, A.G., Butz, A.: Interactions in the air: adding further depth to interactive tabletops. In: Proceedings of the ACM Symposium of User Interface Software and Technology (UIST' 09). pp. 139–148 (2009)
7. Itti, L., Koch, C., Niebur, E.: A model of saliency-based visual attention for rapid scene analysis. IEEE Trans. Pattern Anal. Mach. Intell. **20**(11), 1254–1259 (1998)
8. Itti, L.: Models of Bottom-Up and Top-Down Visual Attention. California Institute of Technology, Pasadena (2000)
9. Jacob, R.J.K.: Eye Movement-Based Human-Computer Interaction Techniques: Toward Non-Command Interfaces. Advances in Human-Computer Interaction, vol. 4, pp. 151–190. Ablex Publishing Co., Norwood (1993)
10. Jonides, J.: Further towards a model of the mind's eye's movement. Bull. Psychon. Soc. **21**(4), 247–250 (1983)
11. Kato, H., Billinghurst, M., Poupyrev, I.: Virtual object manipulation on a table-top AR environment. In: Proceedings of the ISAR2000, pp. 111–119 (2000)
12. Kjeldsen, R., Levas, A., Pinhanez, C.: Dynamically reconfigurable vision-based user interfaces. Mach. Vis. Appl. **16**(1), 6–12 (2004)
13. Koch, C., Ullman, S.: Shifts in selective visual attention: towards the underlying neural circuitry. Hum. Neurobiol **4**, 219–227 (1985)
14. Kölsch, M., Turk, M., Höllerer, T.: Vision-based interfaces for mobility. In: Proceedings of IEEE International Conference on Mobile and Ubiquitous Systems (Mobiquitous'04), pp. 86–94 (2004)
15. Kristensson, P.O., Nicholson, T.F.W., Quigley, A.: Continuous recognition of one-handed and two-handed gestures using 3d full-body motion tracing sensors. In: Proceedings of the 17th International Conference on Intelligent User Interfaces (IUI'12), pp. 89–92 (2012)
16. Lee, H., Kim, J.: An HMM-based threshold model approach for gesture recognition. IEEE Trans. Pattern Anal. Mach. Intell. **21**(10), 961–973 (1999)
17. Liang, H., Yuan, J.S., Thalmann, D., Zhang, Z.Y.: Model-based hand pose estimation via spatial-temporal hand parsing and 3D fingertip localization. Vis. Comput. **29**(6–8), 837–848 (2013)
18. Marr, D.: Vision: A Computational Investigation into the Human Representation and Processing of Visual Information. W.H.Freeman, San Francisco (1982)
19. Mo, Z.Y., Lewis, J.P., Neumann, U.: SmartCanvas: a gesture-driven intelligent drawing desk. In: Proceedings of the ACM Symposium of User Interface Software and Technology (UIST'05), pp. 239–243 (2005)
20. Mujibiya, A., Miyaki, T., Rekimoto, J.: Anywhere touchtyping: text input on arbitrary surface using depth sensing. In: Proceedings of the ACM Symposium of User Interface Software and Technology (UIST' 10), pp. 443–444 (2010)
21. Nianjun, L., Brain, C.L., Peter, J.K., Richard, A.D.: Model structure selection $ training algorithms for a HMM gesture recognition system. In: International IWFHR, pp. 100–106 (2004)
22. Pan, Z.G., Li, Y., Zhang, M.M., Sun, C., Guo, K.D., Tang, X., Zhou, S.Z.Y.: A real-time multi-cue hand tracking algorithm based on computer vision. In: Proceedings of the 2010 IEEE Virtual Reality Conference, pp. 219–222 (2010)
23. Pedersoli, F., Benini, S., Adami, N., Leonardi, R.: XKin: an open source framework for hand pose and gesture recognition using Kinect. Vis. Comput. **30**(10), 1107–1122 (2014)
24. Peng, B., Qian, G.: Online gesture spotting from visual hull data. IEEE Trans. Pattern Anal. Mach. Intell. **33**(6), 1175–1188 (2011)
25. Rabiner, L.R.: A tutorial on hidden Markov models and selected applications in speech recognition. Proc. IEEE. **77**(2), 257–286 (1989)
26. Rovelo, G., Vanacken, D., Luyten, K., Abad, F., Camahort, E.: Multi-viewer gesture-based interaction for omni-directional video. In: Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'14), pp. 4077–4086 (2014)
27. Salah, A.A., Alpaydin, E., Akarun, L.: A selective attention-based method for visual pattern recognition with application to handwritten digit recognition and face recognition. IEEE Trans. Pattern Anal. Mach. Intell. **24**(3), 420–425 (2002)
28. Shen, Y., Ong, S.K., Nee, A.Y.C.: Vision-based hand interaction in augmented reality environment. Int. J. Hum. Comput. Interact. **27**(6), 523–544 (2011)
29. Song, P., Goh, W.B., Hutama, W., Fu, C.W., Liu, X.P.: A handle bar metaphor for virtual object manipulation with mid-air interaction. In: Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'12), pp. 1297–1306 (2012)
30. Tian, M.: Top-down attention motivated research on perception model. Ph.D. thesis, Beijing Jiaotong University, China (2007)
31. Treisman, A.M., Gelade, G.: A feature-integration theory of attention. Cognit. Psychol. **12**(1), 97–136 (1980)
32. Ungerleider, L.G., Mishkin, M.: Two cortical visual systems. In: Ingle, D.J., Goodale, M.A., Mansfield, R.W. (eds.) Analysis of Visual Behavior, pp. 549–586. The MIT Press, Cambridge (1982)
33. Vatavu, R.D.: User-defined gestures for free-hand TV control. In: Proceedings of the 10th European Conference on Interactive TV and Video (EuroITV'12), pp. 45–48 (2012)
34. Walter, R., Bailly, G., Muller, J.: StrikeAPose: revealing mid-air gestures on public displays. In: Proceedings of the ACM Conference on Human Factors in Computing Systems (CHI'13), pp. 841–850 (2013)
35. Wilson, A.D.: Robust computer vision-based detection of pinching for one and two-handed gesture input. In: Proceedings of the ACM

Symposium of User Interface Software and Technology (UIST' 06), pp. 255–258 (2006)

36. Yang, H.D., Park, A.Y., Lee, S.W.: Gesture spotting and recognition for human-robot interaction. IEEE Trans. Robot. **23**(2), 256–270 (2007)
37. Yang, M.H., Ahuja, N., Tabb, M.: Extraction of 2D motion trajectories and its application to hand gesture recognition. IEEE Trans. Pattern Anal. Mach. Intell. **24**(8), 1061–1074 (2002)
38. Wu, H.Y., Zhang, F.J., Liu, Y.J., Hu, Y.H., Dai, G.Z.: Vision-based gesture interfaces toolkit for interactive games. J. Softw. **22**(5), 1067–1081 (2011)

Huiyue Wu is a lecturer at the School of Communication and Design, Sun Yat-Sen University, China. His research focuses on human–computer interaction and vision-based interfaces.

Jianmin Wang is a professor at the College of Arts and Media, Tongji University, China. His research focuses on human–computer interaction, usability engineering, information architecture design, and engineering of digital life.