Journal of Circuits, Systems, and Computers


# A NOVEL VISION BASED FINGER-WRITING CHARACTER RECOGNITION SYSTEM

LIANWEN JIN, DUANDUAN YANG
*School of Electronic and Information, South China University of Technology, 381 WuShan Road, GuangZhou, 510640/GuangDong, P.R.China*

LI-XIN ZHEN JIAN-CHENG HUANG
*Motorola China research center,*
*ShangHai, 200002/ShangHai, P.R.China*

Received (received date)
Revised (revised date)
Accepted (accepted date)

A novel video based finger writing virtual character recognition system (FVCRS) is described in this paper. With this FVCR system, a human can enter character into a computer by just using the movement of fingertip, without any additional device such as a keyboard or a digital pen. This provides a new wireless character-inputting method. A simple but effective background model is built for segmenting human-finger movements from cluttered background. A robust fingertip detection algorithm based on feature matching is given, and recognition of the finger-writing character is by a DTW based classifier. Experiments show that the FVCRS can successfully recognize finger-writing uppercase and lowercase English alphabet with the accuracy of 95.3%, 98.7% respectively.

## 1. Introduction

Vision-based HCI (Human Computer Interaction) is an important technology in making machines more intelligent [1,6]. One type of vision-based HCI application is the tracking and detection of finger or fingertip movement. There have been a significant research in this field: Gesture recognition[6], sign-language recognition[5], finger mouse[2,3], augmented desk interface systems [4,15,16], Window system control (such WWW navigator, menu control)[12], finger-painting system[3] and so on. Fingertip movement could also be used to write characters, virtually on the air, and through computer vision and pattern recognition technologies, it is possible for computer to recover and recognize the finger-writing characters. Nevertheless, the current published in the literature has few reports on using fingertips to write characters for human-computer interaction.

On the other hand, character-input HCI is being increasingly important because of the popularity of many portable devices, such as PDA, pocket PC, smart mobile phone, etc. Conventionally, character-input is usually achieved by keyboard, or pen-based computing technology (such as handwritten character recognition based on touch screen or digital tablet). For mobile application, the human-machine interface based on keyboards and screens is not effective since the dimensions of the fingers limit the minimum size of keyboards. Although handwriting has been recognized as one main character inputting modality for many mobile devices [18], it is not convenient for mobile application when the size of screen is limited. Moreover, special hardware (touch screen, pen, or digital tablet etc) is needed. Therefore, there is a need for alternative method for human-machine communications that involve smaller hardware for portable application. Rather than using the traditional tablets and touch-sensitive screens, handwriting can also be captured using a video camera. As camera is becoming more and more popular in many portable device (eg. mobile phone), the camera-based human-machine interface have a significant role to play in the development of the interfaces for the future[13], because cameras can be miniaturized thus making the interface much smaller.

In this paper, we propose a novel vision-based finger-writing virtual character recognition system (FVCRS). The basic idea of FVCRS is that people can write characters virtually by just using the movement of his finger-tip (We call this as "finger-writing". In this system, the trajectories of the finger-tip are detected and tracked in real time, which provides a kind of inkless character pattern. Since there is no ink when we use the fingertip to "write", we call the reconstructed character as "virtual character". The virtual character is finally recognized by a classifier. In this way, a human can enter character into a computer by just using the movement of his/her fingertip, providing an interesting wireless character inputting modality for HCI application. The diagram of our fingertip based virtual character inputting HCI system is given in figure 1.

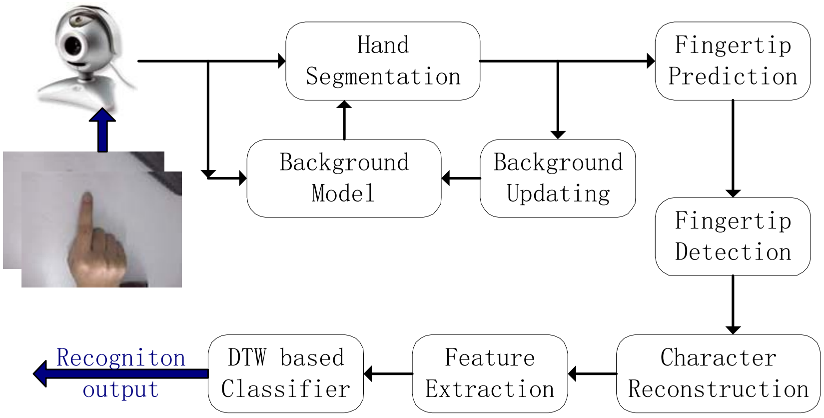



Fig 1. Diagram of our FVCR system.

## 2. Fingertip Segmentation from Cluttered Background

Background modeling is one important computer vision problem. Although many methods have been proposed to address the problem (eg. [7~10, 19]), it may be safe to say a universal model for foreground detection from arbitrary background has not been developed. Pixel intensity is one of the most commonly used features in background modeling. The pixel intensity is often modeled with some statistical distribution, whose parameters are obtained after a process of training and recursive updating. A Gaussian distribution based model (mixed Gaussian model is often used, since single Gaussian model is not good enough) is one popular background model for many applications[8]. Nevertheless, models based on mixed Gaussian are computational intensive and not fit for real-time application without special hardware support. Besides the Gaussian based model, there are many other models have been proposed, such as the temporal deviation model in W4[10], non-parametric model[9], and so on. But from experiments we found most of these background modeling methods are not very suitable for real time indoor application such as our system.

In our FVCR system, an ordinary USB PC camera is used in indoor sites for video capture. Although the background of indoor scene is not as complex as that of an outdoor scene, but there are still many issues that should be taken into consideration, which include shadows, flash light, etc. In order to solve the problem of indoor background modeling, we proposed a simple but effective background model to segment the human hand from a cluttered background. Our background model $M_t(x)$ is given by:

$$\mathrm{M}_t(\mathrm{x}) = \begin{bmatrix} \mathrm{m}_t(x) \\ d^t_{median} \\ d^t_{\max} \end{bmatrix} = \begin{bmatrix} \mathrm{m}_t(x) \\ median_z\left\{\left|I^t(\mathrm{z}) - I^{t-1}(\mathrm{z})\right|\right\} \\ \max_z\left\{\left|I^t(\mathrm{z}) - I^{t-1}(\mathrm{z})\right|\right\} \end{bmatrix} \quad (1)$$

Where $z$ is pixel index over the current frame. Here, parameters $d^t_{median}$ and $d^t_{\max}$ are the median and max values of intensity change over the current frame, and are updated frame by frame. $I^t(\mathrm{x})$ denotes the intensity value of pixel $x$ at frame $t$. $m_t(x)$ is the mean of intensity value of background model of pixel $x$ at time $t$, and it is updated by the equation (2).

$$\mathrm{m}_t(x) = \begin{cases} \mathrm{I}^t(\mathrm{x}) & if(pixel(x) \in update - region) \\ \mathrm{m}_{t-1}(x) & if(pixel(x) \in hand - region) \end{cases} \quad (2)$$

Where: $\mathrm{m}_0(x) = \frac{1}{N}\sum_{i=1}^{N}\left|I^i(x)\right|$.

$N$ is the number of initial frames (typically 20-40) without foreground. In eq.(2), the hand-region is automatically determined by a connection-region detection algorithm, starting from the detected fingertip pixel. The update-region is then defined as the region excluding the hand region (see fig. 2).


(a) (b) (c)

Fig. 2 Definition of update region. (a). Original image. (b). Hand connect-region detection. (c). Update region

After $M_t(x)$ is determined, the following equation is used to judge whether a pixel $x$ belongs to foreground or not:

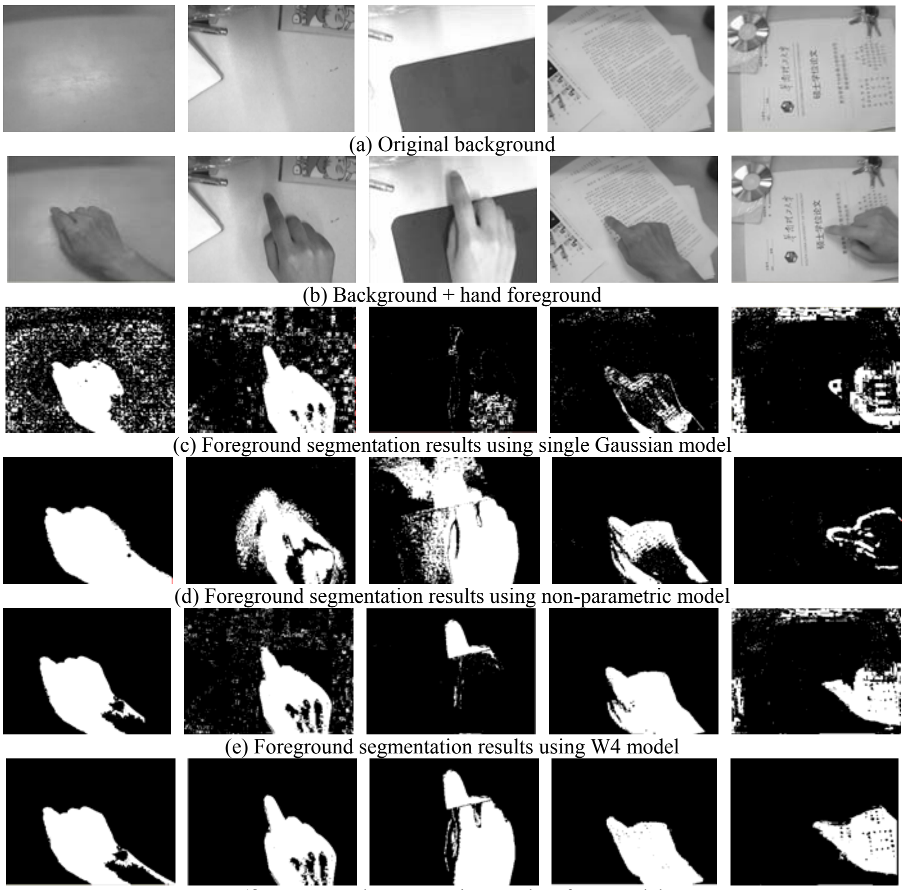
$$B(x)=\begin{cases} 0 \quad background \cdot if & I^t(x)-m(x)<k_{median}\cdot d^t{}_{median} \\ & \vee I^t(x)-m(x)<T \quad \vee I^t(x)-m(x)<k_{\max}\cdot d^t{}_{\max} \\ 1 \quad foreground \quad otherwise & \end{cases} \tag{3}$$

In equation (3), $d^t_{median}$ is median value of $I^t(x)$-m(x), $d^t_{\max}$ is max value of $I^t(x)$-m(x), $k_{median}$ (typically 1.5-1.8 ) and $k_{max}$ (typically 0.2-0.3) are the constant weights of $d^t_{median}$ and $d^t_{\max}$. T is a threshold constant (typically 25~28).

Although our background model based on eq.(1)~eq.(3) is just a simple model without high computation, experiments show that it works very well with indoor scenes. Compared with some other similar background model such as W410, non-parametric model9, single Gaussian model, it works even better. Fig.3 shows some experimental results of our model and previous models. In figure 3, the 1st column from left show a normal situation with simple background. The 2nd column from left shows a situation with heavy shadow disturbance. The 3rd column shows a situation when a flash light was shining by. The 4th and 5th column gives some situations where the background is composed of complex texture objects. From figure 3, it can be seen that our background model can adapt to intensity changes coursed by the flash light, shadows, complex texture background more effectively.

### 3. Fingertip Detection

In early finger-based HCI system, the problem of fingertip detection was easily solved by using special marked gloves and color makers (eg.[14]) and some basic image-processing technology, such as histogram analysis and template matching. But for more natural HCI interaction, accurate and robust fingertip localization is the key to building a bare-hand interaction system. Many approaches have been proposed to solve this problem, such as the 3-D fingertip model or the 2-D fingertip model. As the 3-D fingertip model needs more than one camera and thus heavy computation will be involved, the 2-D appearance-based fingertip model is more suitable than the 3-D fingertip model for real-time application with only a singe camera. For 2-D fingertip position localization with the bare hand, previous



(a) Original background

(b) Background + hand foreground

(c) Foreground segmentation results using single Gaussian model

(d) Foreground segmentation results using non-parametric model

(e) Foreground segmentation results using W4 model

(f) Foreground segmentation results of our model

Fig.3 Foreground segmentation results of our model and some previous models.

fingertip localization methods mainly include contour analysis[23], template match[3], heuristics method[17], and so on. Unfortunately, most of these approach can only works well under some special constraints, such as white background[17], clear contrast between hand and background[16], hand motion cannot be very fast[4,16] (usually the speed of hand motion should less than 10fps), the direction of fingertip must be upward[24,15], etc. To overcome these constraints mentioned above, we propose a new approach for robust fingertip detection, which consists of two stages. In the first stage, re-sampling process is used to solve the problem of blur and the rough location of the fingertip is detected. In the second stage, a circle cross feature vector is extracted from the fingertip image for accurate fingertip position location. The diagram of our fingertip detection approach is shown in figure 4. Details are described in the following sections.

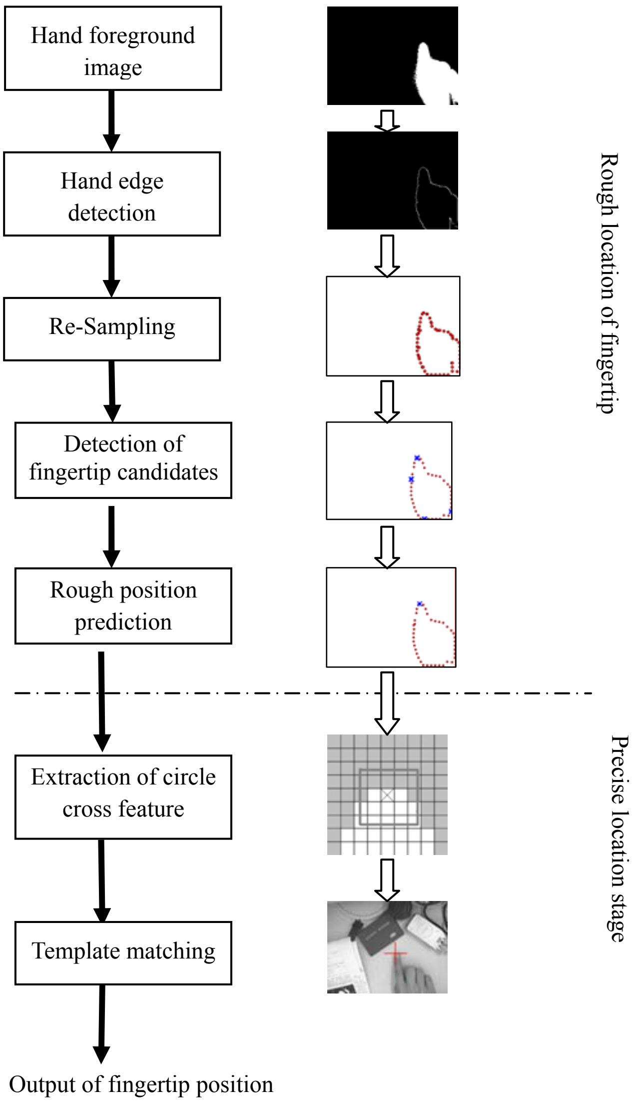



Fig. 4. The flow-chat of fingertip detection process.

### 3.1 *Rough localization of fingertip*

The contour of the hand is used to determine the fingertip position. However, if we use the contour of the whole "hand" to determine the position of the fingertip directly, too much computation would be required and failure can occur because of sometime the contour of hand can be blurred caused by fast motion. So we propose a rough fingertip localization method based on re-samples of the edge image (contour) of the hand. As shown in figure 5, firstly we got the hand-edge image (with size of 320×240 pixles) from the segmented foreground using edge operator (see figure 5(c) ). Then grids of size 10×10 are marked for the edge image (figure 5(d)). Each grid of the edge image will be finally mapped to a while or black pixel according to whether there are edge pixels in each grid or not (figure 5(e). In this way, the original 320×240 hand contour is mapped to a re-sampled 32×24 contour. From figure 6, it can be seen that after the re-sampling process of the original hand contour, some heavily blurred contour become clear and smooth without loss of useful fingertip information. This is very helpful for further fingertip detection. Obviously, another merit of the re-sampling process is that the size of hand contour is reduced greatly (from 320×240 to 32×24), so the amount of computation required is greatly reduced.

In our FVCR system, it is found that the rough position of the tip of the pointing finger must be one of four directional peaks of the closed curve of the finger contour. So the four peaks are treated as the candidates for the rough position of the fingertip (see figure 5(e)). The overall shape of a human finger can be approximated by a cylinder with a hemispherical cap. We then select several sample-point pairs starting from each candidate peak anticlockwise and clockwise. The variance of the distances of these point-pairs around each peak is calculated. The peak corresponding to the minimal variance is regarded as the rough fingertip position. (see figure 5(f)).

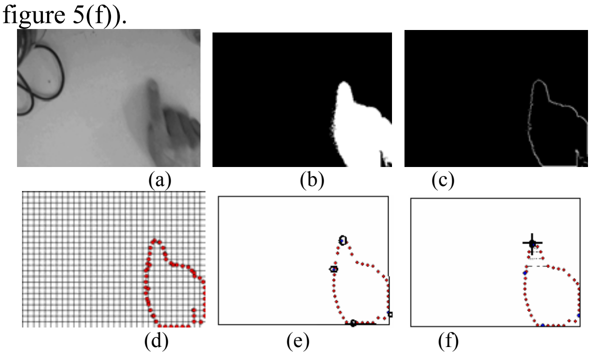



Fig. 5. Processing of rough fingertip detection. (a).original image (b) segmented hand foreground. (c). Edge of hand. (d). Griding re-samping of hand contour. (e) Detect four peaks form the re-sampled contour (f) Rough fingertip detection

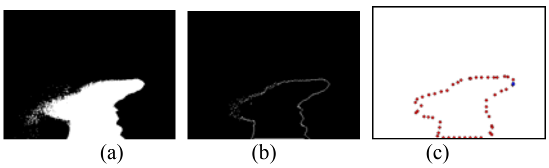



Fig. 6. An example of re-sampling of a blurred image. (a) foreground. (b) the contour image (c) the re-sampling contour image

**3.2** ***Precise localization of fingertip***

It should be noticed that the fingertip position detected by the stage mentioned in section 3.1 is not accurate enough, since the hand contour may not be accurate due to disturbance from shadows, or mis-segmentation of the correct hand foreground, according to the complexity background. So we need to detect the precise fingertip position from the original image. In our FVCR system, the moving fingertip may point to different directions in different frames. Experiments show that when the fingertip is not pointed upward, traditional template matching cannot locate the finger accurately, even with multiple templates. To solve the rotation-invariant problem, we propose a method based on circle cross feature matching. As shown in figure 7, a set of rectangles (typically 12~18) are drawn around the fingertip, with the number of white pixels from each rectangle passed as an accumulated feature. All features will finally form a feature vector, which we call a **circle cross feature vector**. For example, the circle cross feature along the three rectangles shown in figure 7 is 5,7 and 7 respectively. To find the precise location of the fingertip, the circle cross feature vector of fingertip template is matched with an unknown image around the region of the rough fingertip position. The position that produces the highest matching score will be regarded as the fingertip location.

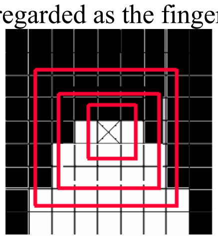
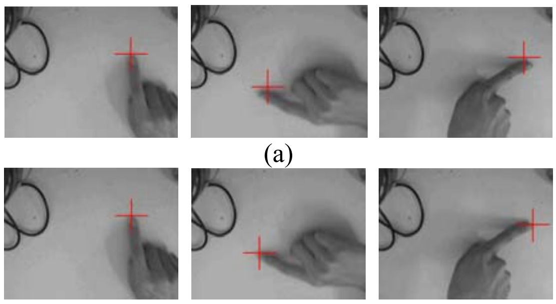


Fig. 7. Extraction of circle cross feature for fingertip



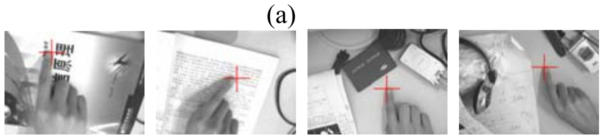
(a)

(b)

Fig. 8. Comparison of our fingertip detection method and conventional method. (a) Fingertip detection results using conventional template matching method, (b) Results using the proposed circle cross feature matching method.

Figure 8 shows a comparison of the fingertip detection results using the traditional template matching method and our circle cross feature matching method. It can be seen that when the orientation of fingertip is not upward, the traditional template method is not good enough. The circle cross feature matching method is robust when the fingertip points to different orientations. Figure 9 gives more experimental results.

(a)



(b)

Fig. 9. More fingertip detection results: (a) The segmented foreground. (b) The localization of fingertip in corresponding original images.

## 4. Virtual Character Reconstruction

Since the moving trajectories of the fingertip are inkless, we need to find a way to recover the character written by the finger. To link all trajectories of fingertip can be done in a simple and natural way, but there are three problems that need to be taken into account: First, how to decide the starting point of a virtual character. Second, how to decide the end of writing. And third, how to process some wild points (as shown in figure 10) caused by a mis-detected fingertip position. We use the following strategies to solve these problems:

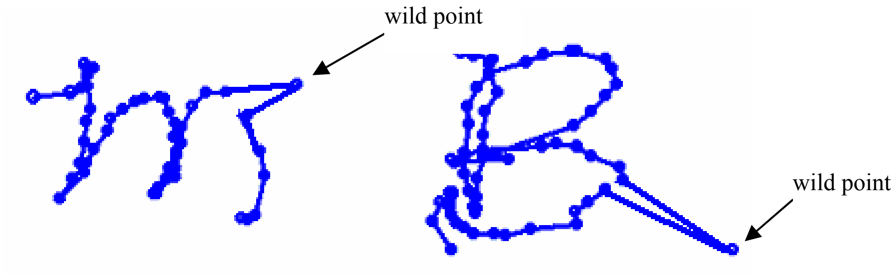



Fig. 10. Example of wild points alone the tracked fingertip trajectories.

Detection of start writing:

(i) The fingertip position is in the region from the top to the middle line; and

(ii) the vertical or horizon displacement is more than 15 pixels in 3 consecutive frames; and

(iii) the angle between the two vectors composed by the fingertip position in 3 consecutive frames is less than a given threshold.

Detection of end writing:

(i) The pixels belonging to the foreground less than 100, or

(ii) the position of fingertip stays in the same place for 15 frames.

Processing of wild points:

(i) A point will be considered as wild and be erased if both of the distances between it and the previous and the next one are more than 1.5 times the average distance.

(ii) For three consecutive points in a video sequence, if the distances from them and the corresponding predicted points (by Kalman filtering[16]) is more than the 1.5 times the average distance of the series, the points are considered as wild and erased.

By using the above strategies, the accuracy in detecting of the stat of writing is more than 96%, and the accuracy in detecting of the end of writing is more than 99%. Also, most wide points (more than 95%) are successfully removed.

The virtual character reconstruction is implemented by linking all consecutive detected fingertip positions together after applied the above processing. Then the reconstructed character is linearly normalized to an image with the size of $64 \times 64$ pixels. Examples of some reconstructed characters are given in figure 11.

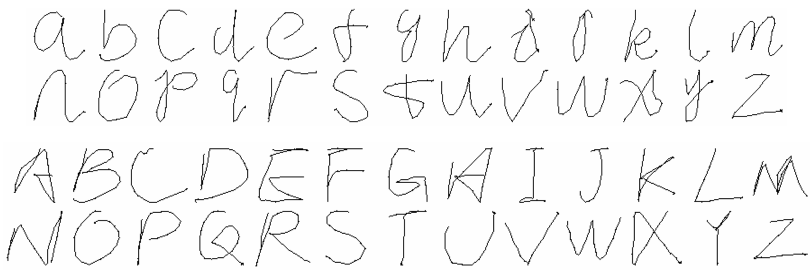


Fig. 11. Reconstructed virtual character samples

## 5. Character Recognition

One special property of a reconstructed virtual character is that all its strokes are connected (since it is difficult to distinguish between upwards and downwards movement of the finger). So it is a kind of one-stroke type of character, which is different from conventional online character captured by touch screen of digital pen. To solve the problem of the one-stroke style character recognition in our FVCR system, we proposed an on-line one-stroke cursive character recognition approach using a combination of directional and positional features, and a DTW(Dynamic Time Warping [22])-based classifier.

### 5.1. *Feature selection and extraction*

#### 5.1.1. *Normalized discrete directional feature (NDDF)*

In our recognition system, templates are presented by strings of normalized discrete directional angle values. The angle value are calculated as shown in Fig.12. An angle represents the direction of writing direction at an extracted feature point in a handwritten character`s stroke. The value is an integer between 0 and 255 linearly mapped from $0^o$ to $359^o$ in order to reduce the required storage.

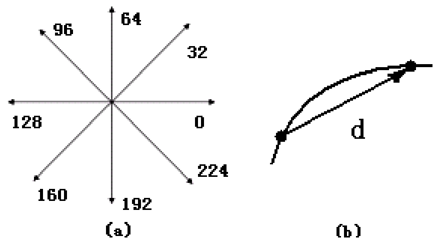



***Fig. 12.*** *(a) Normalized discrete directional angle value. (b) Two successive feature points extracted along the stroke`s curve*

5.1.2. *Sampling positional feature based on NDDF*

There is no positional feature stored in our templates, since we found that the coordinates ( $x_i, y_i$ ) of a character could be reconstructed from NDDF templates as:

$$x_i = x_i + d \cdot \cos(2\pi\theta_i / 256) \tag{4}$$

$$y_i = y_i + d \cdot \sin(2\pi\theta_i / 256) \tag{5}$$

where $\theta_i$ is the directional angle value of the *i*-th feature point, *d* is the sampling distance (a value between 3 and 5). The starting point's coordinates are initialized to 0. After calculating every point's coordinates by (4) and (5), the starting point's coordinates can then be figured out by (6) and (7).

$$x_0 = 0 - MinX \tag{6}$$

$$y_0 = 0 - MinY \tag{7}$$

where $MinX$ and $MinY$ are the minimal values along *x* and *y* axis respectively.

From the reconstructed character, the positional feature points can be sampled along the strokes. By using this technique, the size of template files are reduced without significant loss of recognition accuracy.

### 5.2. *DTW-based classifier design*

The classifier used in our work is based on prototype matching by DTW[22]. The k-nearest neighbor (k-NN) rule is used as the decision criterion. When performing the directional feature matching by DTW, the local distance is calculated by a quadratic curve equation (8):

$$d(i,j)=\begin{cases}(\Delta\theta)^2 & 0\le\Delta\theta<64\\ -(\Delta\theta-128)^2+8192 & 64\le\Delta\theta<128\end{cases} \quad (8)$$

where $\Delta\theta=\begin{cases}|\theta_i-\theta_j| & 0\le|\theta_i-\theta_j|<128\\ 256-|\theta_i-\theta_j| & 128\le|\theta_i-\theta_j|<256\end{cases}$, $d(i, j)$ is the local distance between the $i$-th angle in the template and the $j$-th angle in the input sample, $\theta_i$ is the normalized angle at the $i$-th feature point. From experiments, we found that the quadratic distance is more robust than traditional Euclidean distance [20]. Comparing with the Euclidean distance, an improvement of about 1% recognition accuracy is obtained when using the quadratic curve distance.

Since directional feature alone cannot present all features of a character, we also include positional feature in our classifier. We use positional features for the final classification. After directional-feature matching by DTW is performed, all the characters that can be written in a similar stroke order congregated in front of the candidates queue. Let $C_i$ be the $i$-th candidates, $D_k$ be the global distance of the $k$-th candidates, $\Delta D_k = D_k+1 - D_k$. $\beta$ and $\gamma$ are two parameters where $\beta > 1$, and $0 < \gamma < 1$ (We chose $\beta = 1.2$ and $\gamma = 0.5$ in our system.). Then a parameter K is computed by equation (9) in order to determine a candidate set G given by equation (10).

$$K=\arg\min_k\left\{\text{as long as } \Delta D_k > D_1\cdot\beta \text{ or } {\Delta D_{k+1}}/{\Delta D_k}<\gamma\right\} \quad (9)$$

$$G=\{\, C_i \mid i\le K\} \quad (10)$$

Once the set of candidates has been determined by (10), a combined score is computed for each candidate in this set, as:

$$S_{total}=S_{dir}+\alpha\cdot S_{pos} \quad (11)$$

Where $S_{dir}$ is the global directional distance of the candidate, $S_{pos}$ is the global positional distance, and $\alpha$ is a constant value chosen according to experiments (We chose $\alpha=1$ in our experiments). All the candidates in the set $G$ are rearranged in ascendant order by $S_{total}$. The first candidate is taken as the result of recognition.

### 5.3. *Recognition experimental results*

As there is no public finger-writing character database available, we collected 69 sets of uppercase and lowercase English letters written by different people using our camera user interface. We use templates constructed by hand to test the efficiency of character recognition. The size of the template files are 3.86K and 4.88K respectively.

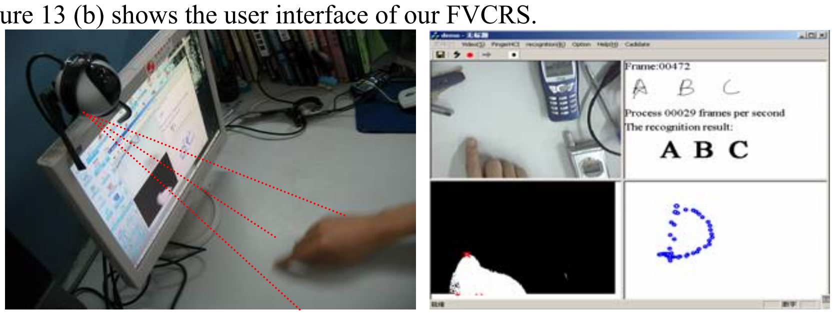
An experiment was conducted to compare different strategies and feature-matching methods. The experimental results are shown in Table 1. It can be seen that directional feature is better than positional feature when recognizing uppercase letters while positional feature is better for lowercase letters. This is because more lowercase letters have similar stroke direction such as “a”, “d” and “q” and the directional feature can hardly classify them. The results also show that the combination of these two features is a success. The recognition rate is raised by more than 10% after directional and positional feature are combined for lowercase letters.

Table 1. The recognition results of English letters by different strategy and feature matching. (DF=directional feature only; PF=positional feature only; CF=combined both features)

| Feature / Recognition rates | DF | PF | CF |
|---|---|---|---|
| Uppercase letters | 95.0 | 93.0 | 98.7 |
| Lowercase letters | 80.9 | 84.8 | **95.3** |

## 6. System Implementation

The proposed FVCRS system was implemented on a PC with Pentium IV 3.0G processor and 512M memory. We use an ordinary low resolution USB PC camera to capture the fingertip video, where the lens is integrated with the camera. The resolution of image is 320×240. The user can write with fingertip on a nature desk plane. There may be nothing else on the desk (which forms a simple background), or there may be some other objects on the desk, such as pen, book, phone, etc. (which forms a complex/cluttered background). In both cases our system works well and can detected the fingertip from background successfully with an average accuracy of 98.5%. There is also no special requirement for the “virtual finger-writing” space, as long as it is convenient for the user to write naturally. Usually an area of 9cm×9cm is large enough. Figure 13(a) shows the system setup. Figure 13 (b) shows the user interface of our FVCRS.




(a) (b)

Figure13. (a). System setup. (b). User interface of our FVCR system[a]

[a] An AVI demonstration of the FVCRS could be downloaded from: http://218.192.171.88/staff/lianwen/demo_fvcrs.html

The speed of the system is about 28~30fps (depending on the complexity of the background), suggesting that it is suitable for real time application. Our system can attain a recognition rate of above 95% at about 50 letters per minutes writing speed. Although this speed is lower than a PC based keyboard typing speed, it is comparable with the speed of the system based on online character recognition algorithm [21].

## 7. Summary

In this paper, we have described a background model for fingertip segmentation from a cluttered background, a new fingertip detection method based on circle cross feature matching, and an approach for one-stroke virtual character recognition. Based on these technologies, a novel vision-based finger-writing virtual character recognition system (FVCRS) has been built. With the FVCRS, a human can input character into a computer by using just the movement of a fingertip. The FVCRS only use an ordinary cheap PC USB camera as sensor, without any additional device such as keyboard, touch screen or digital pen.

The proposed FVCR technology can be used to make mobile character-inputting device much smaller, since the camera can be miniaturized and integrated with the portable device. Another potential application of the FVCRS is in the area of electronic leaning and entertainment. For example, it can be used to develop novel educational learning machine/system which can help children to learn handwriting in an interesting fashion.


## Acknowledgement

This work has been supported in part by Motorola Human Interface Lab Research Foundation (No.D84110) and Natural Science Foundation of China (No.60275005). The authors would like to thank the reviewers for their constructive comments which significantly improved the presentation quality of the paper.


## Reference


1. V. Pavlovic, R. Sharma and Thomas S. Huang, “Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review”, *IEEE Transactions on PAMI*, *Vol. 19, No. 7, 677-695，1997。*
2. Quek, F., T. Mysliwiec, and M. Zhao. “Finger mouse: A freehand pointing interface”, in *Proc. Int'l Workshop on Automatic Face and Gesture Recognition*, June 1995, pp. 372--377..
3. Christian von Hardenberg, François Bérad, “Bare-Hand Human-Computer Interaction”, *Proceeding of the ACM Workshop on Perceptive User Interfaces*, Orlando, Florida, 2001. pp.1-8 .
4. Crowley, J., Bérard, F., and Coutaz, J. “Finger tracking as an input device for augmented reality”, in *Proc. International Conf. Automatic Face and Gesture Recognition*, Zürich, 1995.pp195-200.
5. Kazuyuki Imagawa, Shan Lu, Seiji Igi, “Color-Based Hands Tracking System for Sign Language Recognition”, in *Proc. International Conf. on Automatic Face and Gesture*

*Recognition*, pp. 462--467, 1998.

6. Kenji Oka, Yoichi Sato, Hideki kOike, "Real-time Fingertip Tracking and Gesture Recognition", *IEEE Computer Graphics and Application*, vol.22, no.6, 2002, pp. 64 -71.
7. C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland, "Pfinder: Real-time tracking of human body," *IEEE Trans. Pattern Anal. Machine Intelligence.*, vol. 19, pp. 780–785, July 1997.
8. N. Friedman and S. Russell, "Image Segmentation in Video Sequences : A Probabilistic Approach," In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, San Francisco, 1997.
9. Elgammal,R.Duraiswami,D.Harwood and L.S.Davis, "Background and Foreground Modeling Using Nonparametric Kernel Density Estimation for Visual Surveillance",*Proceedings. of IEEE*, vol..90,no. 7, JULY 2002, pp.1152-1163.
10. Haritaoglu, D.Harwood,and L.S.Davis, "W4:Real-Time Surveillance of People and their Activities", *IEEE Trans. On Pattern analysis and machine intelligence*, vol.22, no. 8, Aug 2000. pp.809-830.
11. Kenji Oka, Yoichi Sato and Hideki Koike, "Real-time Tracking of Multiple Fingertips and Gesture Recognition for Augmented Desk Interface Systems", *Proc. IEEE Intl Conf. Automatic Face and Gesture Recognition (FG 2002) , pp. 429-434， May 2002.*
12. M.S. Lee, D.Weinshall, et al, "A computer vision system for on-screen item selection by finger pointing", *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition*, 1997. pp.1027-1033.
13. M.E. Munich,; P. Perona, "Visual input for pen-based computers", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 24 no. 3, 2002. pp. 313 -328 .
14. J Davis, M Shah**, "V**isual gesture recognition" *IEE Proceedings of Vision, Image and Signal Processing*, Vol.141 , no.2 , April 1994, Pages:101 – 106.
15. Dorfmuller-Ulhaas. K, Schmalstieg.D*,* "Finger tracking for interaction in augmented environments", *Proceedings IEEE and ACM International Symposium on Augmented Reality*, Oct. 2001 ,Pages:55 – 64.
16. Brown.T, Thomas. R.C, **"**Finger tracking for the Digital Desk**",** in *Proc. Of First Australasian User Interface Conference*, 2000. AUIC 2000 ,Pages:11 – 16
17. Tomita.A, Jr.Ishii. R, "Hand shape extraction from a sequence of digitized gray-scale images", *20th International Conference on Industrial Electronics, Control and Instrumentation (IECON '94)*, 1994, Vol.3 , Pages:1925 – 1930.
18. R. Plamondon and S. N. Srihari, "On-line and Off-line Handwriting Recognition: A Comprehensive Survey", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, January 2000. pp. 63–84.
19. Kentaro Toyama, John Krumm,el.al., "Wallflower: Principles and practice of background, maintenance", in *Proc. of ICCV*, September 1999. pages 225—261.
20. Teng LONG, Lian-wen Jin, Li-Xin Zheng, Jian-Cheng Huang, "One Stroke Cursive Character Recognition Using Combination of Directional and Positional Features", in *Proceedings of IEEE ICASSP*, March, 2005, Page: 449-452.
21. C.C. Tappert**,** C.Y. Suen, Wakahara**,** T, "Online handwriting recognition-a survey", *Proceedings of the 9th International Conference on Pattern Recognition*, vol.2, Rome Italy, 1998. pp1123-1132.
22. L Rabiner, BH Juang , Fundamentals of speech recognition, Prentice-Hall, Inc. NJ, USA, 1993.