Journal of Information & Computational Science 11:4 (2014) 1037–1045 March 1, 2014
Available at http://www.joics.com

Vision-based Hand Gesture Interaction Using Particle Filter, Principle Component Analysis and Transition Network ⋆

Zijian Wang [a], Zuo Zhang [a], Fang Wang [b], Yaoru Sun [a,*]

[a] *Department of Computer Science and Technology, Tongji University, Shanghai 200092, China*

[b] *Department of Information Systems and Computing, Brunel University, Uxbridge, United Kingdom*

Abstract

Vision-based human-computer interaction is becoming important nowadays. It offers natural interaction with computers and frees users from mechanical interaction devices, which is favourable especially for wearable computers. This paper presents a human-computer interaction system based on a conventional webcam and hand gesture recognition. This interaction system works in real time and enables users to control a computer cursor with hand motions and gestures instead of a mouse. Five hand gestures are designed on behalf of five mouse operations: moving, left click, left-double click, right click and no-action. An algorithm based on Particle Filter is used for tracking the hand position. PCA-based feature selection is used for recognizing the hand gestures. A transition network is also employed for improving the accuracy and reliability of the interaction system. This interaction system shows good performance in the recognition and interaction test.

Keywords: Human Computer Interaction; Hand Gesture Recognition; Hand Tracking; Principle Component Analysis; Particle Filter

1 Introduction

Compared with using traditional mechanical interaction devices, vision-based Human-computer Interaction (HCI) offers a more natural and friendly way to interact with computers. In recent years, there has been a growing interest in vision-based HCI because it's an inexpensive and non-obstructive solution for HCI [1, 2].

Many studies of human-computer interaction were published in past years. Jakub Segen [3] used a predefined background that could facilitate segmentation of hand region. Hand gestures are recognized according to the hand region. Using markers or auxiliary gloves can make the design of hand interaction systems easier and improve the efficiency of the system [4, 5]. We

⋆Project was supported by the grants from National Natural Science Foundation of China (61173116 and 60970062) and Specialized Research Fund for the Doctoral Program of Higher Education (20110072110014).

*Corresponding author.
Email address: yaoru@tongji.edu.cn (Yaoru Sun).

DOI: 10.12733/jics20102890

adopted an approach without restrictions of auxiliary tools. Our system can work in some usual backgrounds, and a user can interact with a computer using a bare hand in real time.

With a low-cost webcam, users can use pre-designed hand gestures to control the cursor and perform conventional operations: moving the cursor, left click, left-double click and right click. The system can work in real time on a conventional processor. We propose a color-based segmentation method based on the segmentation approach in [6]. Particle Filter algorithm [7] is employed to locate hand position accurately and efficiently. Principle Component Analysis (PCA) [8] is used to reduce the dimensions of hand images, and a minimum distance classifier is used for recognition.

The remainder of this paper is organized as follows:

(1) The hand region extraction method is reported in Section 2.1.

(2) The hand tracking algorithm based on Particle Filter algorithm is described in Section 2.2.

(3) The recognition method and five different hand gestures on behalf of five operations as moving, left click, left-double click, right click and no action (shown in Fig. 1) are detailed in Section 2.3.

(4) A transition network with anti-shake measure and usage of an auxiliary no-action hand gesture are proposed in Section 2.4.

(5) Recognition and interaction tests are detailed in Section 3.

(6) Finally, we offer some conclusions in Section 4.

The schematic diagram of the HCI system is shown in Fig. 1.

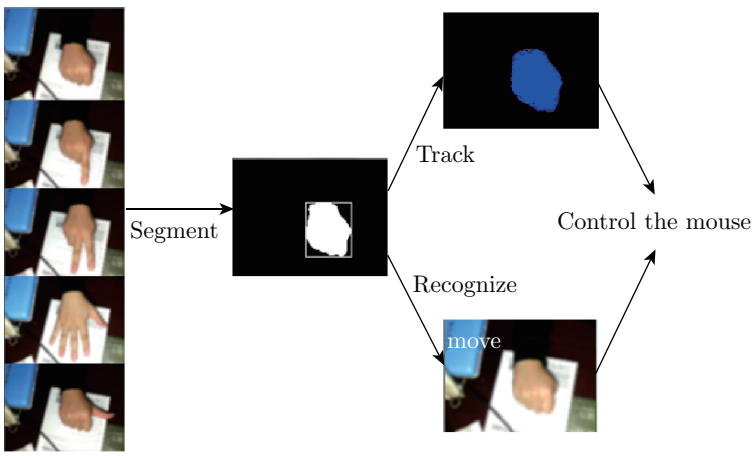

Fig. 1: The schematic diagram of the method. Five hand gestures are designed on behalf of five mouse operations (images in the left column). The cursor is controlled according to the tracking and recognition results. The tracking and recognition processes are based on the result of segmentation

2 Hand Gesture Interaction

2.1 Hand Segmentation

The hand region is segmented from the background before tracking and recognition. Hand segmentation methods based on skin color have been used as efficient approaches in the previous research [6, 9, 10] on the premise that human skin color is distinct from the background color. An efficient skin-color segmentation method based on HSV (Hue, Saturation and Value) color space is implemented in our system because different skin color has a similar range in the HSV color space [10].

An image is binarized according to the HSV color range of hand skin: the pixels whose color is in the skin color range are set as "1" while the others are set as "0". But some "1" pixels outside the hand region could be mistaken as a part of the hand region by the tracking algorithm. To resolve this problem, the "1" region enclosed by the longest "1" contour is filled with "1" and the others are filled with "0". Finally, a new binary image whose "1" pixels stand for the hand region is obtained. The hand segmentation algorithm and the results are shown in Fig. 2.

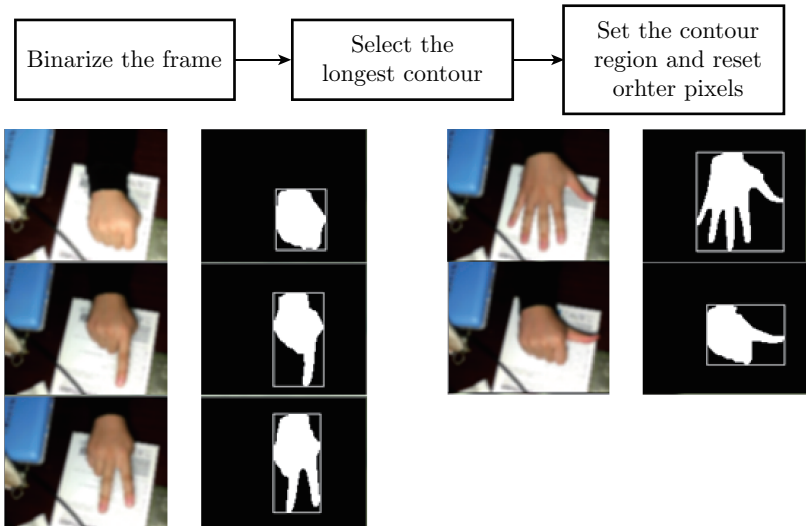

Fig. 2: The hand segmentation algorithm and the results. Five hand gestures are shown. The left images are video-input images and the rights are binarized images

2.2 Hand Tracking

In order to obtain the accurate hand position, an accurate and effective algorithm based on Particle Filter algorithm is used. This algorithm has three steps: initial sampling, resampling and hand position calculating.

Initial sampling runs in processing the first frame. N sample points are put randomly in the binary image (the result acquired in Section 2.1). The size of the binary image is $W \times H$ pixels. Coordinates of the N sample points are $< x_i, y_i > (1 \leq i \leq N)$.

Resampling runs in every frame. The sample points in the current frame are resampled according to the confidence values and posterior probabilities of the sample points in the previous

frame. The confidence is calculated by the following equation:

$$confidence_i = \begin{cases} 1 & \text{if} < x_i, y_j >= 1 \\ 0 & \text{if} < x_i, y_j >= 0 \end{cases} \quad (1)$$

This equation assigns 1 to the confidence values of the sample points in the hand region and 0 to the others. The posterior probability (PP) is given by:

$$PP_i = \frac{confidence_i}{\sum_{i=1}^{N} confidence_i} \quad (2)$$

It's obvious that PP of a sample point in the background is 0, but PP of one in the hand region is: $\frac{1}{\sum_{i=1}^{N} confidence_i}$. The sample points in the hand region are selected for resampling. To prevent the deterioration of Particle Filter Algorithm, each of them is resampled for RN times ($RN = \frac{N}{\text{“1” pixel number}}$). The new position of each resample point is selected randomly in the rectangle range from $< x_i - W/5, y_i - H/5 >$ to $< x_i + W/5, y_i + H/5 >$.

In the last step, hand position is obtained by:

$$handposition = \sum_i < x_i, y_i > \times PP_i \qquad \text{where } i \text{ refers to all the remaining points} \quad (3)$$

The hand position is obtained by calculating sample points in this algorithm, but the number of the sample points may be reduced to 0 when the hand moves outside the camera window, which can lead the tracking to failure. To prevent it, the initial step will restart when the number of the sample points is reduced to 0. The sample points and the hand positions are shown in Fig. 3.

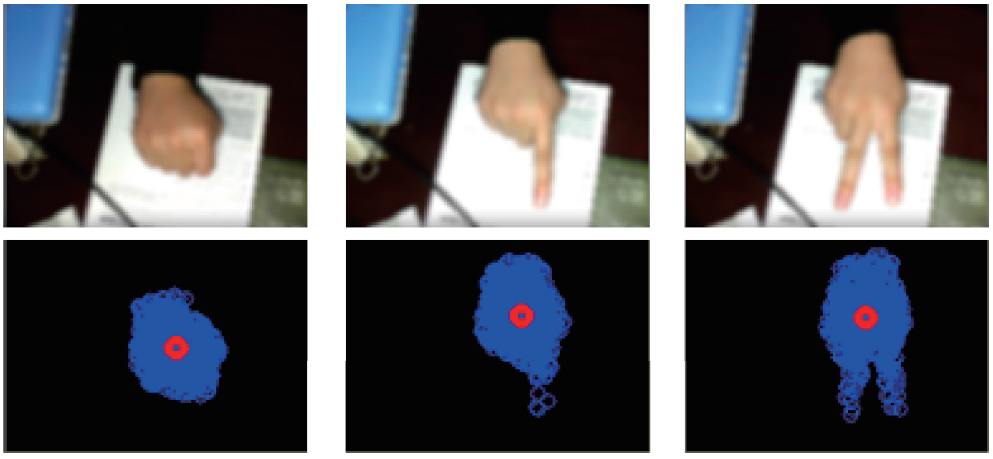

Fig. 3: Three hand images and the tracking results. Blue circles represent the sample points and red circles represent the hand positions

2.3 Hand Recognition

Five hand gestures stand for mouse operations: move, left click, left-double click, right click and no action. Every hand gesture has five training images shown in Fig. 4. It will cost a lot of time to process a $W_0 \times H_0$-pixel-size hand gestures image. The PCA algorithm is used to reduce the dimensions of hand image data. This algorithm is composed of a training phase and a recognition phase.

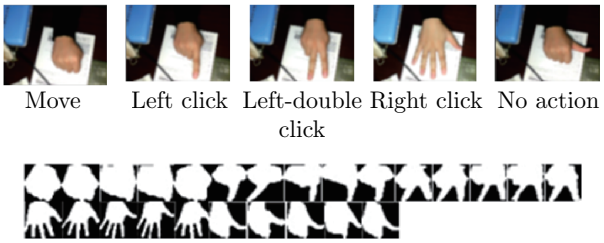

Fig. 4: The 5 hand gestures and 5 training images for each gesture

2.3.1 Training Phase

An eigenspace is built in the training phase. The eigen hands converted from the prepared training hands are located in the eigenspace.

First, the twenty-five $W_0 \times H_0$-pixel-size training images are converted into column vectors $x_i (1 \leq i \leq 25)$ by:

$$\begin{pmatrix} a_{11} & \dots & a_{1M_0} \\ \vdots & \ddots & \vdots \\ a_{H_01} & \dots & a_{H_0M_0} \end{pmatrix} = (a_{11}\ a_{21}\ \dots\ a_{H_01}\ a_{12}\ \dots\ a_{H_02}\ \dots\ a_{1M_0}\ \dots\ a_{H_0M_0}) \tag{4}$$

A difference image d_i can be calculated for each hand image by $d_i = x_i - \Psi$. The average image Ψ is defined by:

$$\Psi = \frac{1}{25} \sum_{i=1}^{25} x_i \tag{5}$$

Through the difference image, a covariance matrix C is built according to:

$$C = \frac{1}{25} A A^T \qquad \text{where } A = (d_1, d_2, \dots, d_{25}) \tag{6}$$

The eigenvalues and eigenvectors of C are used to build the eigenspace. Singular Value Decomposition (SVD) is used to calculate the eigenvalues λ_i and eigenvectors μ_i.

The first ten eigenvalues and eigenvectors are selected in descending order. The bigger the eigenvalue is, the more important the corresponding eigenvector is. The ten eigenvectors make up the eigenspace ω by:

$$\omega = (\mu_1, \mu_2, \dots, \mu_{10}) \tag{7}$$

Every projection vector in the eigenspace represents a hand image. The training images are projected onto the eigenspace by the following equation:

$$\Omega_j = \omega^T d_j \qquad \text{where } j = 1, 2, \dots, 25 \tag{8}$$

2.3.2 Recognition Phase

Recognition phase runs in every frame. In this phase, the binarized hand image is recognized in real time. A rectangle is set to cover the hand region as a Region of Interest (ROI). It is

also scaled into an image of $W_0 \times H_0$ pixels that has the same size as training images before recognition.

Recognition is similar to the training phase. First, the normalized image is converted into a column vector x_0 and the difference image is calculated by $d_0 = x_0 - \Psi$. The hand image can be projected onto the eigenspace, and the projection vector Ω_0 is defined by:

$$\Omega_0 = \omega^T d_0 \tag{9}$$

Finally it is necessary to match the hand image with one of the twenty-five training images. The Euclidean distance between the hand image and training image is calculated by Eq. (10). The hand image matches with the training image that has the lowest ϵ_i.

$$\epsilon_i = \|\Omega_i - \Omega_0\| \qquad \text{where } i = 1, 2, \ldots, 25 \tag{10}$$

2.4 Cursor Control

Hand tracking provides the hand position, and hand recognition provides the category of the hand gesture. They are both parameters used in the cursor controlling function. A cursor action transition network based on the transition network proposed in [11] is implemented to improve the accuracy.

In the transition network, the cursor responses to the video input in a delay of one frame. It is operated based on parameters in the current, previous and the second previous frames. When the recognition results of the current, previous and the second previous frames are all "move", the cursor starts moving. When the recognition result of the second previous frame is "move" and the ones of current and previous frames are click-kind-action, the cursor will proceed the corresponding click-kind-action. But the no-action operation is unique. When the current gesture means no-action, the cursor will do nothing. The transition network is shown in Fig. 5.

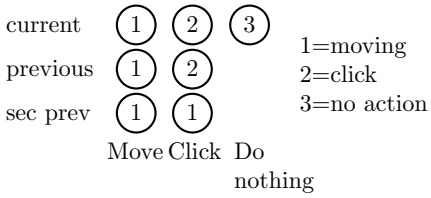

Fig. 5: Transition network. In this transition network, recognition results in 3 sequential frames to determine one cursor action (except no-action)

The new position is determined by the original cursor location and the scaling hand speed when the cursor moves. A measure for reducing the cursor shake is implemented. The hand position in the current, previous and second previous frame is denoted as C, P and S, respectively. The angle made by line SP and line PC is α. An abrupt change in the moving direction (small α) is probably caused by an unintentional hand shake, which could be blocked by setting a threshold of α. The moving command is valid only if α is bigger than 100° (Fig. 6).

To facilitate moving, a no-action gesture is proposed to help when the hand is moving to the edge of the camera window. When it's necessary to move the hand far away, even out of the camera window, users can move the hand to an appropriate position holding the no-action

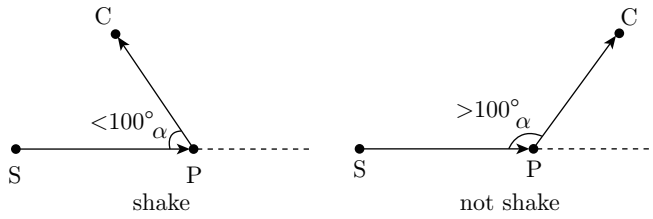

Fig. 6: The method to reduce cursor shake. The cursor moves only if α is bigger than 100°

gesture and then continue the unfinished movement. Besides, a deviation will occur when the hand moves into the camera window after the hand moves out. The system reinitializes tracking algorithm when the hand moves out of camera window to remove the deviation.

3 Performance

In this section, we test the accuracy and efficiency of our system. All participants in our tests wear long sleeves covering the arms. The application is implemented in Visual C++ using OpenCV libraries [12] on the laptop platform with Intel Core 2 CPU (2.0 GHz), and 3GB RAM. And the operating system is Windows 7. The average execution speed of the application is 21 frames per second. The processing rate of the webcam is 30 frames per second. The average execution speed is faster than the speed in [9] which is 10 frames per second.

3.1 Recognition Test

Ten participants took part in the test and each of them tested 500 hand images, 100 for each hand gesture. The test images included 50% normal hand images and 50% images of hand with a little bend or rotation. The results are shown in the Table 1. Five thousand hand images were tested in total and the global mean accuracy is 98.3%.

Table 1: Accuracy of hand recognition in the test

Participants	Move	Left Click	Left-double	Right Click	No Action
1	100%	97%	97%	99%	99%
2	100%	100%	96%	95%	96%
3	96%	100%	99%	100%	95%
4	100%	98%	97%	96%	100%
5	100%	100%	100%	95%	96%
6	100%	96%	100%	98%	100%
7	100%	100%	96%	99%	98%
8	99%	99%	96%	100%	96%
9	100%	97%	100%	99%	99%
10	100%	99%	95%	98%	100%
Average accuracy	99.5%	98.6%	97.6%	97.9%	97.9%

3.2 Interaction Test

We designed three interaction tasks to test the performance of our interaction system. Participants performed the tasks using our interaction system and a touch tablet on a laptop respectively.

In the first task, participants created a text file by a right click on the desktop, opened it by a left-double click and then close it by clicking the left button.

In the second task, participants entered disk D through the windows explorer and viewed the attribute of the first file.

In the third task, participants calculated 1 plus 2 using the calculator.

The average completion time with our interaction system is 16.5 sec, 14.2 sec and 9.5 sec while the time with a touch tablet is 12.4 sec, 9.8 sec, and 5.8 sec, respectively. Considering the maturity of tablets interface design, the performance of our vision-based hand gesture interaction system is acceptable and has potential for improvement in the future.

4 Conclusion

In this paper, we proposed a detailed framework of a vision-based real-time hand gesture interaction system. We used a tracking algorithm based on particle algorithm for tracking the hand and used the PCA algorithm for recognition. A cursor action transition network was implemented to improve the accuracy of the application. This application could work in real time of 21 frames per second. The experimental results showed a high recognition accuracy of 98.3%. The interaction tests showed a good performance.

References

[1] Vladimir I. Pavlovic, Rajeev Sharma, Thomas S. Huang, Visual interpretation of hand gestures for human-computer interaction: A review, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 19, No. 7, 1997, 677-695

[2] Ronald Poppe, Vision-based human motion analysis: An overview, Computer Vision and Image Understanding, Vol. 108, No. 1, 2007, 4-18

[3] Jakub Segen, Senthil Kumar, Shadow gestures: 3D hand pose estimation using a single camera, IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, 1999

[4] Pattie Maes, SixthSense: Integrating information and the real world, 8th IEEE International Symposium on Mixed and Augmented Reality, ISMAR 2009, IEEE, 2009

[5] James M. Rehg, Takeo Kanade, Digiteyes: Vision-based hand tracking for human-computer interaction, Proceedings of the 1994 IEEE Workshop on Motion of Non-Rigid and Articulated Objects, IEEE, 1994

[6] Cristina Manresa et al., Hand tracking and gesture recognition for human-computer interaction, Electronic Letters on Computer Vision and Image Analysis, Vol. 5, No. 3, 2005, 96-104

[7] Michael Isard, Andrew Blake, Condensation-conditional density propagation for visual tracking, International Journal of Computer Vision, Vol. 29, No. 1, 1998, 5-28

[8] Svante Wold, Kim Esbensen, Paul Geladi, Principal component analysis, Chemometrics and Intelligent Laboratory Systems, Vol. 2, No. 1, 1987, 37-52

[9] Thomas Coogan et al., Real time hand gesture recognition including hand segmentation and tracking, Advances in Visual Computing, 2006, 495-504

[10] Y. Shen, S. K. Ong, A. Y. C. Nee, Vision-based hand interaction in augmented reality environment, Intl. Journal of Human-Computer Interaction, Vol. 27, No. 6, 2011, 523-544

[11] Yasushi Hamada, Nobutaka Shimada, Yoshiaki Shirai, Hand shape estimation using sequence of multi-ocular images based on transition network, Proceedings of the International Conference on Vision Interface, 2002

[12] Gary R. Bradski, Vadim Pisarevsky, Intel's Computer Vision Library: Applications in calibration, stereo segmentation, tracking, gesture, face and object recognition, Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, Vol. 2, 2000