Real Time Hand Tracking as a User Input Device

Kittasil Silanon
Department of Computer Engineering
Faculty of Engineering, Prince of Songkla University,
Hat Yai, Thailand
jalego3@hotmail.com

Nikom Suvonvorn
Department of Computer Engineering
Faculty of Engineering, Prince of Songkla University,
Hat Yai, Thailand
kom@coe.psu.ac.th

***Abstract* — In this paper, we propose a system that facilitates the two dimension user-input as general mouse device for controlling applications. The method is based on hand movement analysis by applying image processing technique. The Haar-like with a cascade of boost classifiers is applied for hand detection, and then tracked with skin color using CamShift. Extracted hand features are computed and used for recognizing commands via a finite state machine. We evaluated the performance of system under real time constraint in real environment with demonstration application.**

Keywords- Hand detection; Tracking; Command Recognition

I. INTRODUCTION

Trends in human machine interfacing have invented very rapid in consumer devices such as multi-touch of mini-computer like iPad, mobile phone iPhone, motion sensing of game devices, and etc. Relatively, many applications to support these platforms are actively developed. In this paper, we introduce a hand motion capture system using single camera that enables to track 2D hand position for commanding the desirable applications in real-time. The system offers very easy way of short range human-machine interfacing, which is very effective in terms of application constraints and prices, and generally very appropriate for the PC-based applications without any additional integrated material devices. There are many research try to solve the same problem. Wei Du [1] introduced a system that can serve as a user input device, replacing mouse and trackball through hand gesture recognition and hand tracking. Nebojsa Jojic [2] developed a real-time system for detecting pointing gestures and estimating the direction of pointing using stereo cameras for controlling the cursor on a wall screen. Juan [3] proposed a vision-based system that can interpret a user's hand gesture to manipulate objects and optimized for the medical data visualization environment.

We propose a system that used only a webcam as additional material, installed on top of the computer screen looking down towards the user's hands, for interfacing with applications. Our main contribution is then focused on the image processing techniques that try to detect hand pose and extract hand features which are used later in the command recognition process. Figure 1 shows example of system.

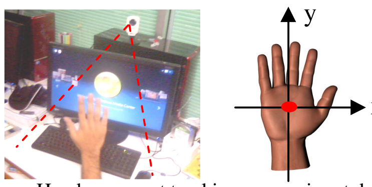


Figure 1. Hand movment tracking as user input device.

II. PROPOSED SYSTEM

In this section, we introduce the system for command recognition based on hand motion by analyzing the image sequences, obtained from the webcam attached over the computer screen. There are two main parts of our system: hand detection and tracking, and command recognition. The overall of system is illustrated in the figure 2. First part, a hand detector is implemented using a cascade of boost classifiers, which allows obtaining very robust object detector. Two types of hand pose with vertical position are considered such as open and close hand, which is necessary for executing a command as start and stop symbol respectively. However, hand pose can have many figures caused by translation and rotation in 3D, which needs to be found. The tracking method, CamShift and Kalman filter, is then applied in order to extract hand through image sequence. In the second part, hand parameters are determined such as position, axis, area, and angle, which are defined as hand feature. A Finite Stage Machine is established based on these features for recognizing commands.

Figure 2 (left) shows the sequence of steps for hand detection and tracking; figure 2 (right) shows the recognition process. These two parts will be detailed in the section 3 and 4 respectively. In section 5, the experimentation result is discussed for evaluating the performance of system. For testing in real environment, we have implemented the system in windows 7, which our module can control any applications as a mouse interface. In the last section, we conclude our method.

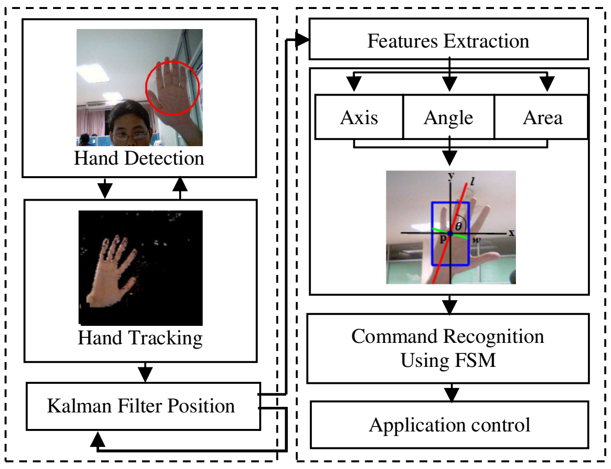


Figure 2. System overview : (left) detection and tracking, (right) recognition process.

III. HAND DETECTION AND TRACKING

A. *Hand Detection*

A Cascade of Boost Classifier [4][5] for object detection is originally developed by Viola and Jones. This method uses Haar-like features and a cascade of boosted tree classifier as a supervised statistical model of object recognition. Haar-like feature consists of two or three connected "black" and "white" rectangles. The feature value is defined by the difference between the sum of pixel values within the black and white rectangles. Figure 3 shows a basic set of Haar-like features.

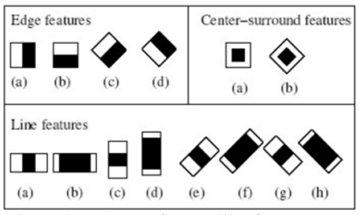


Figure 3. A set of Haar-like feature.

The AdaBoost algorithm is introduced to improve the classification performance that designed to select rectangle feature which best separates the positive and negative example. In the first iteration, the algorithm train a weak classifier *h(x)* using one Haar-like feature that achieves the best recognition performance for the training samples. The classifier consist of a feature *f(x)*, a threshold θ_t and parity p_t indicating the direction of inequality sign.

$$h_t(x)=\begin{cases}1 & \text{if} \quad p_t f_t(x)<p_t\theta_t \\ 0 & \text{otherwise}\end{cases}$$

In the second iteration, the training samples that were misclassified by the first weak classifier receive higher weights. The iteration goes on and the final result is a cascade of linear combinations of the selected weak classifiers $H(x)$, which achieves the required accuracy.

$$H(x)=\sum_{t=1}^{T}\alpha_t h_t(x)$$

In the detection process by using the trained cascade, the sub-windows must be test each stage of the cascade. A negative outcome at any point leads to the immediate rejection of the sub-window.

In our technique, a command begins with open hand posture and stop using close hand posture. Each frame, these two gestures must be detected. In the training processing, we have collected around 5,000 positive samples for each hand style from students in our university in various conditions such as indoor with neon light and outdoor with natural light. Systematically, around 10,000 negative samples are selected from landscape, building, and human faces or body images. Figure 4 show some of these samples. Each posture type was trained two rounds. The first round is for discriminating quickly the target posture from the outliers with small samples. By observing the false positive and false negative, we determine the additional positives and negative samples in order to overcome the better recognition rate while reducing noise as shown in figure 5.

Figure 4. (left) positive samples, (right) negative samples.

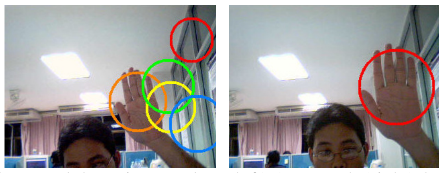


Figure 5. Hand detection results : (left) 1st round. (right) 2nd round.

Table 1 show the detection rate results form our experimentation of the two hand postures with respective configurations. We notice that a good detection rate is rised at least more than 91% with 16 trainning stage at minimum.

Table1. Hand detection rate.

Hand Posture	**Window Size**	**Training and Performance**			
		Positive	***Negative***	***Stage***	***Rec Rate***
Open	32x32	5,058	10,448	19	91.17%
Close	32x32	5,678	10,448	16	99.06%

B. *Hand Tracking*

CamShift [6] is a non-parametric technique using for color object tracking deriving from the Mean Shift algorithm. The main difference between CamShift and Mean Shift algorithm is that CamShift updates continuously its probability distributions; in generally the target object in image sequences changes significantly its shape size or color, while Mean Shift is based on static distributions. That why CamShift is suitable for tracking the rigid object. In the hand tracking, the process can be described as the following.

Step 1: the color probability distribution of detected hand image is determined from its histogram via hue component of HSV color space, related to skin color.

Step 2: this target distribution of detected hand is tracked on the searching window of next frame in image using mean shift algorithm. The mean shift vector, which is aimed for finding an optimized path that climbs the gradient of a probability distribution to the mode (peak) of nearest dominant peak, is necessary to be computed.

Step 3: the back-projection technique, which associates the pixel values in the image (tracking hand) with the value of the corresponding distribution, is applied.

Step 4: the center of mass and size of tracking hand (projected image) is computed and defined as hand features. This step will be detailed in the next section (hand features).

Step 5: on the next iterative, the current position of hand in image is used for defining the searching window on the next frame. The process is repeated at step 2 continuously.

Note that the step 1 will be re-executed systematically if the detected hand by Haar-like features with boost cascade of classifier found in the searching window. We found that Haar-like method provides very accurate results when hand is paralleled to the vertical axis, compared with

CamShift, but missed mostly in other directions, so that CamShift is applied in order to solve the problem.

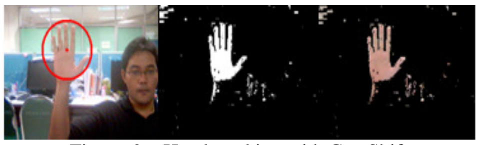


Figure 6. Hand tracking with CamShift

C. *Hand Movement Estimation*

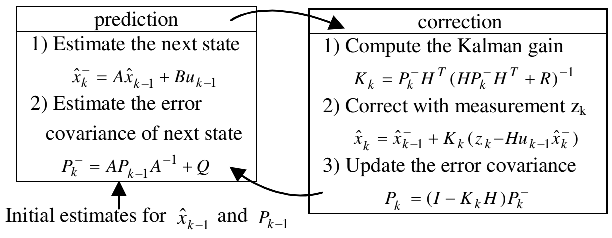
The Kalman filter [7] is a recursive linear filtering method. It addresses the general problem of trying to estimate the state of discrete time process that is described by the linear stochastic differential equation by the following.

$$x_k = Ax_{k-1} + Bu_{k-1} + w_{k-1}$$

With a measurement $z \in \Re^m$ that is

$$z_k = Hx_k + v_k$$

The random variables w_k and v_k represent Gaussian noise of the process and measurement respectively. The algorithm of Kalman filter estimates a process by using feedback control technique: estimating the process state by an appropriate model and doing feedback by noisy measurements. As such, the equations of Kalman filter are formed into two groups: prediction and correction equations. In the post tracking of hand, the algorithm can be described as the following figure 7.


prediction
1) Estimate the next state
$\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$
2) Estimate the error covariance of next state
$P_k^- = AP_{k-1}A^{-1} + Q$
correction
1) Compute the Kalman gain
$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$
2) Correct with measurement z_k
$\hat{x}_k = \hat{x}_{k-1}^- + K_k(z_k - Hu_{k-1}\hat{x}_k^-)$
3) Update the error covariance
$P_k = (I - K_k H)P_k^-$
Initial estimates for $\hat{x}_{k-1}$ and P_{k-1}
Figure 7. The operation of the Kalman filter.

Experimentally, we found that hand detector by Haar-like features with boost cascade of classifier cannot provoide total results and CamShift may wrongly track espectially when there are other parts of body such as face or background, having color in skin color range, move close to tracking hand. Therefore, the Kalman filter is applied in order to predict hand position in the next frame based on previous frame, obtained by CamShift. To apply the Kalman algorithm, two principle equations needed to be declared: tracking process and measurement. The state of tracking process is measured from hand poistion and velocity in each image frame. So, we define the process of state x_k by the following:

$$\begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_{k,i} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_{k-1,i} + w_{k-1,i}$$

Where x, y, v_x, v_y are the position and the velocity of hand in the k^{th} image frame respectively. Here, we assume that that the motion of hand between two successive frames can be uniformly approximated as straight line, the frame interval Δt is very short. For the measurement z_k, we define directly by the position value obtained from CamShift algorithm.

IV. Command Recognition

A. *Hand Features*

During the step 4 of hand tracking process by CamShift algorithm, in each frame, the hand characteristics are computed. The following parameters are considered: center of mass (p), axis angle (θ), area (z) and distance (d). The features of the hand in a frame can be easily computed from the moments of pixels in hand's region, which is defined as .

$$M_{ij} = \sum_x \sum_y x^i y^j I(x,y)$$

In the above equations, $I(x,y)$ is the pixel value at the position (x,y) of the image, x and y are range over the hand's region.

Step 1: the center of mass are $Xc = \frac{M10}{M00}$, $Yc = \frac{M01}{M00}$

Step 2: the angle of hand is

$$\theta = \frac{1}{2}\arctan\left[\frac{2\left(\frac{M11}{M00} - XcYc\right)}{\left(\frac{M20}{M00} - Xc^2\right) - \left(\frac{M02}{M00} - Yc^2\right)}\right]$$

Step 3: the long-axis (l), short-axis (w) of tracking hand can be presented as :

$$l = \sqrt{\frac{(a+c) + \sqrt{b^2 + (a-c)^2}}{2}}, \quad w = \sqrt{\frac{(a+c) - \sqrt{b^2 + (a-c)^2}}{2}}$$

$$a = \frac{M20}{M00} - Xc^2 \ , \ b = 2\left(\frac{M11}{M00} - XcYc\right), \ c = \frac{M02}{M00} - Yc^2$$

When used in tracking, the above equations give us angle, length-axis and width-axis. Then, we use axis for determined area (z) is l x w and distance (d) is $\sqrt{l^2 + w^2}$.

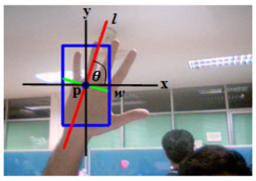


Figure 8. Features of the hand's region.

By considering the parameters with respect to time t, we formulate the hand feature $f(t)$by the following :

$$f(t) = \left[p_t(x_c, y_c) \quad \theta_t \quad z_t \quad d_t\right]$$

B. *Command Recognition*

A simple recognition system of commands based on grid is established in order to interface the hand movement as user input of interactive application. The following commands are considered: (1) cursor displacement in four directions, and (2) clicks: left, middle, right. The main idea for designing the command recognition system is that to use the extracted parameters as features f.

For displacement commands, only $\partial p_t / \partial t$ is used for constructing command gird. A dynamic grid is used for command recognition. When hand has detected, which the 3x3 grid is created enclosing its center $p(x_c,y_c)$.Then, move hand from center to the neighborhood cells will activate the displacement command with respect to the specific direction. For the click commands, we consider the value of $\partial\theta_t/\partial t$ and $\partial z_t/\partial t$ defined by the following: click left if only if $\partial\theta_t/\partial t$ is positive, click right if only if $\partial\theta_t/\partial t$ is negative, and middle click if only if $\partial z_t/\partial t$ is less than $z/2$ as in figure 9.

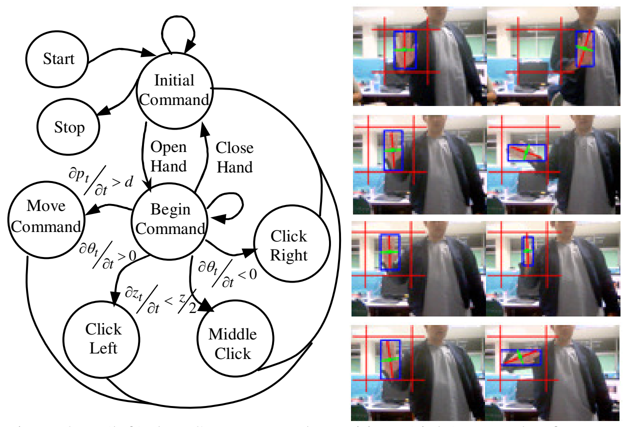

Start
Initial Command
Stop
Open Hand
Close Hand
$\partial p_t/\partial t > d$
Move Command
Begin Command
Click Right
$\partial\theta_t/\partial t > 0$
$\partial\theta_t/\partial t < 0$
$\partial z_t/\partial t < z/2$
Click Left
Middle Click

Figure 9. (left) the FSM states and transitions (right) example of move and click left, middle click, and click right command respectively.

V. System Performance and Discussion

Our system is implemented using OpenCV library. Testing system is run on Intel processor Core 2 Duo, 2 Ghz, 2 GB memory. The video sequence is analyzed with image resolutions 320x240 pixels at 30 fps.

A. *Hand Tracking*

We evaluate the precision of trajectory obtained from hand tracking process. The test is done by executing the hand movement following three types of ideal path: straight line, curve and v-shave respective. Then, we compute the errors, defined by the difference between ideal path and tracking position. Table 2 shows results of ten times test with average. We found that the error in average of curve is bigger than v-shape and line respectively.

Table2. Hand tracking with errors

No	**Testing errors (pixels)**		
	Line	***Curve***	***V-shape***
1	1.54	3.79	8.05
2	1.70	3.61	3.31
3	1.39	6.87	4.22
4	1.59	6.68	9.13
5	1.97	7.55	4.98
6	0.97	8.93	3.51
7	1.40	8.09	4.31
8	2.02	4.23	4.44
9	1.76	6.05	7.05
10	1.84	6.97	6.38
μ	1.61	6.27	5.53
σ	0.29	1.74	1.89

B. *Command recognition*

The performance of command recognition is tested depends on two factors: distance of hand from camera, and hand speed for executing a command. Testing is done under indoor environment, no change of light condition, with the following test: three fix distances with three levels of speed, ten times of testing in each case.

Table3. Experimentation results

Distance (cm)	Hand speed (s)		
	0.5s	1s	2s
30	10/10	10/10	10/10
60	10/10	10/10	10/10
90	7/1	8/10	9/10

Table 3 shows our testing results. We note that at low speed commands can be executed more precise than high speed one. Intuitively, during recognition process more frames are analyzed under low speed that increases eventually the effectiveness of the method. We found that the commands at near distance provide high recognition rate than far distance. Certainly, when the image region of hand in image is too small, then hand cannot be detected.

We also demonstrate our system as a user input that interface to Window Media Center application on Windows 7. You can see video showing the real-time interactive at http://www.youtube.com/watch?v=loVxPEP5fME.

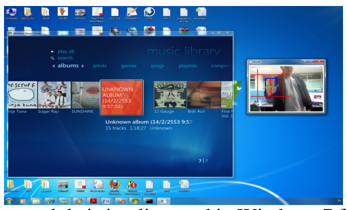


Figure 10. Our module is implimented in Windows 7 for controlling Window Media Center.

VI. Conclusion

We have introduced the image processing methods for hand detection tracking and recognition aimed for human applications interfacing like a general mouse input. The hand trajectory and other features is precisely extracted, using for command recognition. Eight basic commands are recognized at very high rate with respect to different speeds and distances. The implementation of our system on Windows 7 shows an example test under real-time constraint with real application.

References

[1] Wei Du and Hua Li , "Vision based gesture recognition system with single camera", 5th International Conference on ICSP, vol.2, no., pp.1351-1357 vol.2, 2000

[2] Jojic N., Brumitt B., Meyers B., Harris S. and Huang T. , "Detection and estimation of pointing gestures in dense disparity maps," Fourth IEEE International Conference on Automatic Face and Gesture Recognition, vol., no., pp.468-475,

[3] J. Wachs, H. Stern, Y. Edan, M. Gillam, C. Feied, M. Smith, and J. Handler,"A Real-Time Hand Gesture System Based on Evolutionary Search," Vision. vol. 22, no. 3, Oct. 2006, Dearborn, Mich, Society of Manufacturing Engineers.

[4] Viola, P. and Jones M., "Rapid object detection using a boosted cascade of simple features," Proceedings of the IEEE CVPR, vol.1, no., pp. I-511- I-518 vol.1, 2001

[5] Lienhart R. and Maydt J., "An extended set of Haar-like features for rapid object detection," International Conference on Image Processing, vol.1, no., pp. I-900- I-903 vol.1, 2002

[6] G. R. Bradski, "Computer vision face tracking for use in a perceptual user interface," Intel Technology Journal, Q2 1998.

[7] Greg Welch and Gary Bishop, "An Introduction to the Kalman Filter," Annual Conference on Computer Graphics & Interactive Techniques. ACM Press, Addison-Wesley, Los Angeles, CA, USA (August 12–17), SIGGRAPH 2001 course pack edition