Real Time Hand Marker Tracking as a User Input Device

Pongsatorn Chawalitsittikul and Nikom Suvonvorn*
Department of Computer Engineering, Faculty of Engineering, Prince of Songkla University,
Hat Yai, Songkhla, 90112
E-Mail: armclashblack@hotmail.com, kom@coe.psu.ac.th*

Abstract

Conventional motion capture systems have used widespread especially in the entertainment industry. However, their usage as real time user input devices has been limited by their price. In this paper, we propose an easy-to-use and inexpensive system that facilitates 2-D user-input using video-based markers analysis captured from a consumer grade webcam.

Our system detects and tracks a specific pattern of color marker on cloth glove for hand pose analysis. Some features are observed allowing us to recognize the particular commands. We evaluated the performance of system under real time constraint in real environment with a practical application.

Keyword
Marker analysis, Color tracking, Command recognition

1. Introduction

Current trends in human machine interface have developed quit rapid in consumer devices such as multi-touch of iPhone, motion sensing devices of Wii, and etc. However, the hand tracking systems as user input have been limited due to the technical constraints and prices. In this paper, we introduce a hand motion capture system using a single camera that enable to track 2D hand pose and interact with applications in real-time. Our goal is aimed to design the system that uses a consumer grade webcam as a low-cost approach while having to deal with the robustness, precision, and real-time constraints. However, with this approach we require user to wear a glove with color maker of specific pattern. Our technique is then try to extract the patterned glove that represents the pose of hand, describing by position, axis, and zoom features, which are used later in the command recognition process. There are many research try to solve the same problem. Wang and Robert [1][2] proposed complex patterns of glove with specific tracking methods in order to retrieve the 3D hand pose for manipulating object in three dimension. Sýkora and all [3] focused on how to track the color ball for augmented reality application.

A webcam generally sits on top of the computer screen looking down towards the marker of user's hands. Figure 1 shows example of the pattern of color maker. The XY is the webcam coordinate system, where the X- and Y-axes are parallel to the camera view plane. Our related work is presented in the following sections: system overview, design, evaluation, and future work.

2. Theories and Principles

This section we present specific theories on Camshift algorithm using for color tracking and kalman filter for hand pose tracking respectively.

2.1 Color tracking algorithm

Camshift [4] is a non-parametric technique using for color tracking, which is derived from Mean Shift algorithm. It is suitable for tracking the rigid object in video sequence that its histogram is quite constant. The color histogram of the region of interest (ROI) in specific frame can be represented into the form of probability distribution function (PDF) via hue component of HSV color space. This distribution is systematically followed on the next frame in video sequence using mean shift algorithm. The mean shift vector, which is an important step of algorithm, is necessary to be determined. It is aimed for finding an optimized path of color gradient on PDF representing the way from current maximum value to the nearest dominant peak. By using the back-projection technique, object corresponding to the tracking distribution can be reconstructed into object in the spatial domain. The center of mass and size of object can be computed and used as object features. On the next round of algorithm, the current location of object is used later as the searching window on the next video frame. The process is repeated continuously.

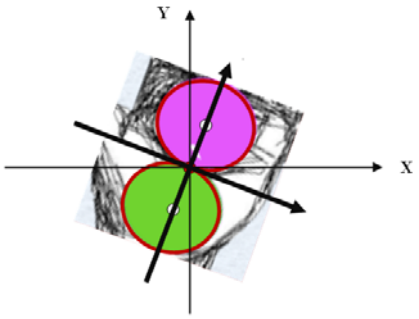

Figure 1: Example of marker with pattern of colors.

2.2 Kalman filtering

The Kalman filter [5] 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. The algorithm can be described as the following.

Step1. Initialization state

Step2. Prediction state

1) Estimate the next state $\hat{x}_k^- = A\hat{x}_{k-1} + Bu_{k-1}$

2) Estimate the error covariance of next state

$$P_k^- = AP_{k-1}A^{-1} + Q$$

Step3. Measurement state

1) Compute the Kalman gain

$$K_k = P_k^- H^T (HP_k^- H^T + R)^{-1}$$

2) Correct with measurement w_k

$$\hat{x}_k = \hat{x}_{k-1}^- + K_k(z_k - Hu_{k-1}\hat{x}_k^-)$$

3) Update the error covariance

$$P_k = (I - K_k H)P_k^-$$

Step4. Repeat the prediction at step 2.

3. Work process

3.1 System overview

Our work is focus under the idea that a distinctive maker simplifies hand pose reference parameters that is used later as general input device of interactive applications. Image sequence from consumer grade webcam attached over the computer screen is captured and analyzed for recognizing the specific pattern of color maker, which provides the essential parameters necessary for command recognition. To achieve this, many process need to be clarified. How to design reliable maker for the detection step? How to detect and tracking the color pattern? How to determine hand pose parameters and design a robust command recognition system? These questions will be examined in details in the following sections.

Figure 2 show the overview of our system. Simple of marker colors are firstly loaded to determine histogram for tracking using Camshift. The center of mass of two ROIs is then tracked by kalman algorithm in order to filter the outlier data. The necessary parameters as features of hand pose are extracted from the tracking area such as axis, distance, and angle. These parameters are then applied for command recognition using a dynamic grid system.

3.2 Color maker design

This section offers a description of how to design the pattern of colors for marker. The design of pattern is considered under the conditions that necessary parameters must be resolved. Considering parameters are: (1) hand position (2) principle axis of hand as rotation angle of XY-axis to the horizontal and vertical respectively (3) zoom factor representing how close of hand to the camera. The important criteria are that these parameters must be robust and reliable. Figure 1 show the designed pattern which consists of two cycle marker with difference colors. Only the center of cycle of markers will be used for determining the principle axis (Y-axis) of hand position, X-axis is then determined easily. The middle of these points is considered as hand position. The angle of Y-axis to the vertical line is rotation parameter and the distance between points will use for zoom factor.

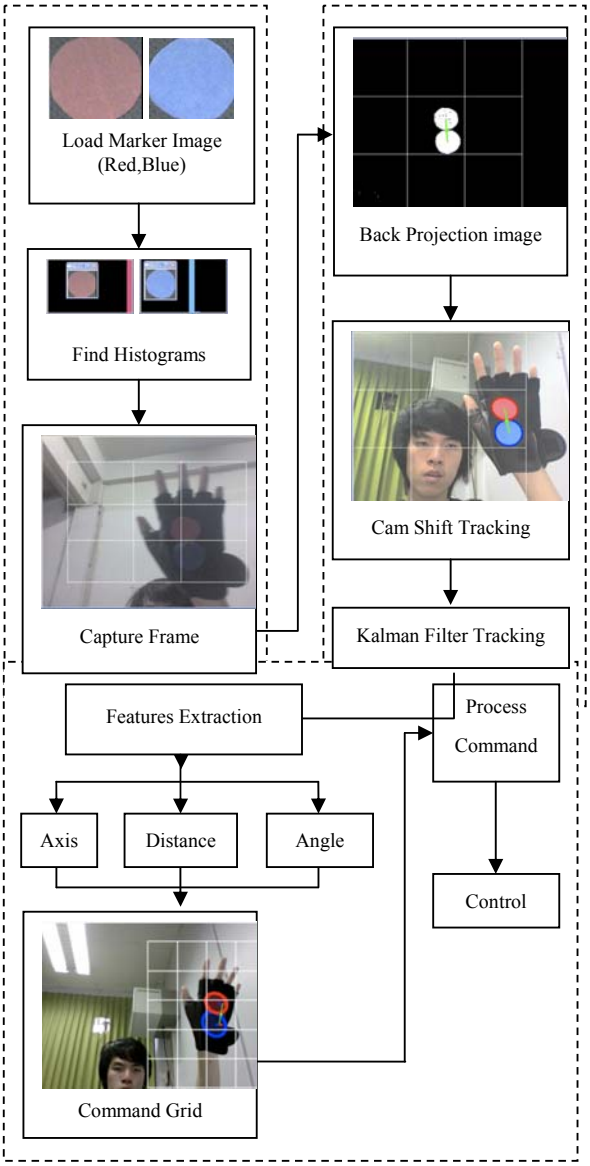

Figure 2: System overview

One of important steps is how to find the center of cycle. In order to separate better these two makers, colors are chosen from H space of HSV color model in the opposite way, so S is located in maximum zone. For example, if H color at 60 degrees is for upper maker, the H color at 240 degrees must be used for lower one, shown in figure 3. The maker must be separated very well form background, so the color of glove is defined in black, which V is zero

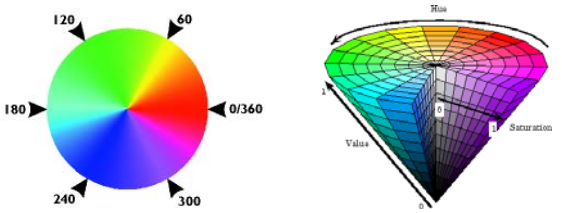

Figure 3: Maker color selection

3.3 Color maker analysis

Based on the designed maker, we can determine the area of interest occupied by the hand using the Camshift method. The initialization of density probability function of Camshift algorithm is done by using a simple of color marker from captured frame in testing environment. Each marker is detected and tracked separately. Figure 4 (left) represents the back-projection image from the color probability distribution of tracking object. Each marker, the cycle enclosing the maker in the back-projection image and its center of mass are computed by the following:

$$x_c = \frac{M_{10}}{M_{00}} \text{ and } y_c = \frac{M_{01}}{M_{00}}$$

where $M_{00} = \sum_x \sum_y I(x,y)$ is the zero moment, and

$$M_{10} = \sum_x \sum_y xI(x,y) \quad \text{and} \quad M_{01} = \sum_x \sum_y yI(x,y)$$

are first order moment. $(x_1, y_1)_c$ and $(x_2, y_2)_c$ are the center of mass of upper and lower marker respectively. Figure 4 (right) shows the result of the detected maker in red and blue cycle on RGB image space.

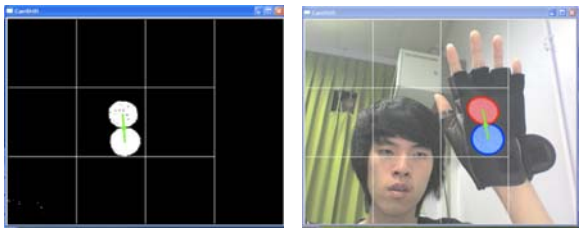

Figure 4: Maker detection and tracking

We found that Camshift provides quit good result of $(x_1, y_1)_c$ and $(x_2, y_2)_c$ when hand moves slowly. In opposite way, when hand moves faster the blur effect from motion induce the wrong tracking or lost totally tracking area in the worse case. In order to solve the problem, the histogram of marker is reinitialized whenever the tracking is null. However, the wrong tracking is always unsolved that led to the inaccuracy of the values of center of mass. We then introduce the post-tracking using Kalman filter.

To apply the algorithm of Kalman, two principle equations needed to be declared: tracking process and measurement. The state of tracking process is established in which the displacement of hand from frame to another is linear. So, the tracking equation is then defined by the following:

$$\begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_{k,i} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ v_x \\ v_y \end{bmatrix}_{k-1,i} + w_{k-1,i} \quad i \in [1,2]$$

For the measurement z_k, we define directly to the value obtained from Camshif algorithm. Two trackers are necessary in order to track our marker in the same time.

3.4 Parameters

As mention previously, three parameters needed to be determined: position of hand (p), zoom factor (z), and rotation angle (θ), shown in figure 5. Position of hand (p) is very simple to determined, it is the middle point $p(x_m, y_m) = ((x_1 + x_2)/2), (y_1 + y_2)/2)$ between center of mass of both markers. Distance between centers of markers (d) is defined by $d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$. This value is large when hand stays close to camera and small in the opposite way. Zoom factor (z) can then be defined by the ratio of these distance values at difference times, so $z = d_{t+1}/d_t$. If z is bigger than value one, it means zoom-in, and zoom-out in the opposite way. Rotation of hand is measured by the angle between vertical line and the principle axis of hand defined simply by following:

$$\theta = \tan^{-1}\left(\frac{x_1 - x_2}{y_1 - y_2}\right) = \tan^{-1}\left(\frac{x_1 - x_m}{y_1 - y_m}\right)$$

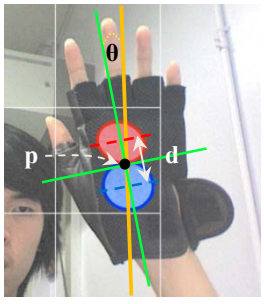

Figure 5: Parameter estimation

3.5 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 eight 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, considering parameters with respect to time t.

$$f(t) = [p_t(x_m, y_m) \quad \theta_t \quad z_t \quad d_t]$$

For displacement commands, only $\partial p_t / \partial t$ is used for constructing command gird. Because of the perspective transformation effect, the same motion of hand with different distances from camera will provides different value of parameters. If gird is static the user must move hand longer while staying far from camera in order to execute the same command. To resolve the problem, a dynamic grid is used for command recognition. When hand move to active area, a simple stop will mark the beginning of command, which the 5x5grid is created enclosing its center $p(x_m, y_m)$ with d x d square cell. Then, move hand from center to the neighborhood cells will activate the displacement command with respect to the specific direction. Figure 6 shows an example of dynamic grid built at difference distances of hand far from the camera.

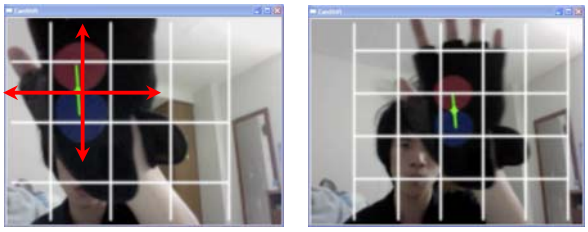

Figure 6: Dynamic grid for displacement recognition

Figure 7 show the example of hand movement, we can notice that marker detection and tracking is done perfectly even if it is blurred causing by motion effect – sensibility is generally quit low in consumer grade webcam.

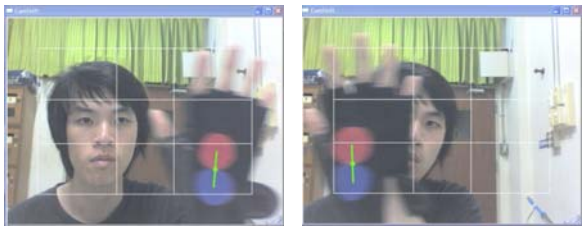

Figure 7: Examples of displacement command: left-to-right movement.

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
- Middle click if only if $\left(\partial z_t / \partial t - 1\right)$ is positive

Figure 8 show the example result of click commands, for click right, middle click, and click left respectively.

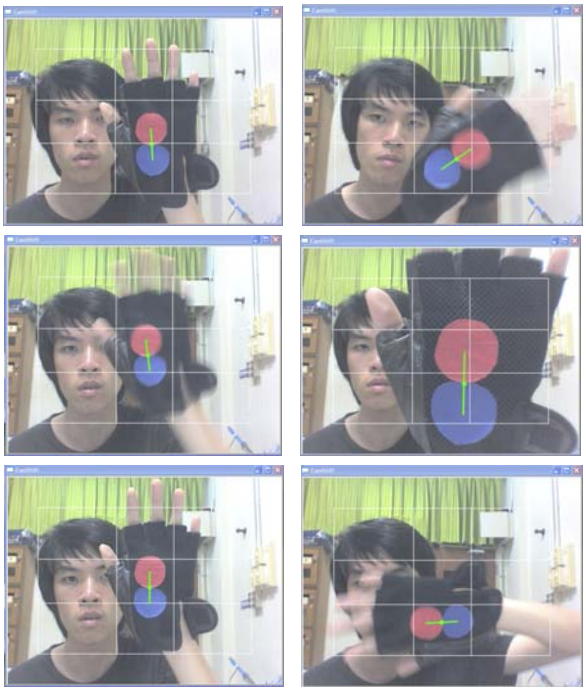

Figure 8: Example of click right, middle click, and click left commands respectively.

4. System performance and evaluation

Our system is implemented using OpenCV library [6]. Its performance is tested depends on two main 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. Testing system is run on Intel processor Core 2 Duo, 2 Ghz, 2 GB memory. The video sequence is analyzed with image resolutions 360x240 pixels at 30 fps.

Table 1 shows our testing results. We can notice that the analysis at near distance provides command recognition more accurate than far distance. When color region in image is too small, the color probability became too low, so hand cannot be detected. We also found that the low speed of hand movement offers commands more precise than high speed one. Intuitively, during recognition process more frames are analyzed under low

speed case that increases eventually the effectiveness of the method.

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

Table 1: Experimentation results

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://fivedots.coe.psu.ac.th/~kom/?p=305.

Figure 9: Our module is applied for controlling Window Media Center of Windows 7.

5. Conclusion and discussion

We have introduced maker detection and tracking technique as a user input device using consumer grade webcam that allows estimating 2D positions of hand. Important parameters are extracted from double cycle style marker that allows forming the specific commands. A simple and effective recognition process as a dynamic grid is proposed in order to translate commands precisely in real-time. Our system is tested with satisfaction results and combined with Windows 7 to facilitate 2-D touch input of Windows Media Center.

6. Reference

[1] Wang, R. Y. and Popović, J. 2009. Real-time hand-tracking with a color glove. In ACM SIGGRAPH 2009 Papers (New Orleans, Louisiana, August 03 - 07, 2009). H. Hoppe, Ed. SIGGRAPH '09. ACM, New York, NY, 1-8. DOI= http://doi.acm.org/10.1145/1576246.1531369.

[2] Robert Y. Wang, Real Time Hand Tracking as a User Input Device, ACM Symposium on User Interface Software and Technology (UIST), October 19-22, 2008, Monterey, CA

[3] Daniel Sýkora, David Sedláček and Kai Riege, Real-time Color Ball Tracking for Augmented Reality, Proceedings of the 14th Eurographics Symposium on Virtual Environments, pages 9-16, Eindhoven, The Netherlands, May 2008.

[4] Gary R Bradski, Computer Vision Face Tracking For Use in a Perceptual User Interface, Intel Technology Journal, No.Q2. 1998

[5] Greg Welch and Gary Bishop, An Introduction to the Kalman Filter, TR 95-041, University of North Carolina at Chapel Hill, July 24, 2006

[6] Gary Bradski and Adrian Kaebler, Learning OpenCV : Computer Vision with the OpenCV Library, O'Reilly, Inc, 2008.