Vision-Based Virtual Control Mechanism via Hand Gesture Recognition

Chia-Hung Yeh[1,*], Shu-Jhen Fan Jiang[1], Jia-Chi Bai[2], Jia-Sian Liou[2],
Ruey-Nan Yeh[3], Sun-Chen Wang[3], and Po-Yi Sung[3]

[1] Department of Electrical Engineering

Sun Yat-Sen University

Kaohsiung 804, Taiwan, ROC

yeh@mail.ee.nsysu.edu.tw

[2] Department of Computer Science and Information Engineering

National Dong Hwa University

Hualien 972, Taiwan, ROC

[3] Material and Electro-Optics Research Division

Chung-Shan Institute of Science and Technology

Taoyuan 325, Taiwan, ROC

Received 5 May 2010; Revised 8 June 2010; Accepted 30 June 2010

Abstract. As web cameras cost is much lower and easier to get nowadays, the ways to develop an interesting applications and interaction methods are hot research topics now. In this paper, an effective real-time virtual control mechanism based on hand gesture recognition is proposed. Through developed algorithms, hand gesture can be recognized efficiently to further control the application without any additional equipments or devices. Furthermore, the disturbance of human face and arm that have the same feature, skin color, with hand is excluded to enhance the accuracy of hand gesture recognition. A ball game is designed to evaluate the performance of the proposed system. Experimental results that the proposed system provides high accuracy in virtual control and has high potential for various kinds of applications.

Keywords: Vision-based, virtual control, hand gesture, virtual control, face exclusion, arm exclusion, background updating

1 Introduction

In this era, digital cameras become more and more popular, vision-based virtual control devices are also getting higher attention and demands and might become another input device that could be used in real-world in the future. However, due to the limitations of complicated environment such as non-static background or luminance condition, the feasibility is still not good enough for real-world usage. Traditional control mechanisms such as keyboard and mouse are generally used all over the world now. In most case, they are quite enough to fit the requirements of normal usage, and the cost of those basic input devices are much more friendly than other high-level input devices. Nevertheless, they need particular space for deployment despite wireless keyboard and mouse.

Mouse and keyboard have long dominated the realm of computer input mechanism due to their effectiveness and ease-of-use. However, those conventional input devices need quite space to deploy. If available space is insufficient to operate conventional input devices, or for the sake of mobility, proper substitutes are required for different situations. Trackballs and touch-pads are designed for such situations. However, the trackball is not as easy as a regular mouse, and is not widely accepted as a mouse. Although touch-pad is now the standard equipment for laptop computers, it suffers similar problems as trackballs for inconvenience to perform some basic mouse actions such as drag-and-drop. Another solution which began to draw extensive attention during recent years is the touch screen. Touch screen is much more user-friendly than trackballs and touch-pads. It can also play the role as the keyboard by showing characters on the screen for the user to touch and type. Several leading companies in the computer industry like Apple or Microsoft have announced to give strong support to such devices.

Recently, technological advances of CCD/CMOS sensor arrays have lowered the price of cameras significantly. Cameras are now used for all kinds of purposes, including photography, video recording, video confer-

* Correspondence author

ences, video surveillance, and etc. With the popularity and pervasion of camera, using cameras as input devices has now become a feasible option. By simply posing and moving hand gestures, the user can issue commands to control devices or type characters, thus eliminating the need of extensive space occupied by physical input devices. In this paper, a vision-based virtual control mechanism is proposed to provide an effective way for virtual input. The efficient image processing and visual content analysis techniques are employed to analyze hand gestures such as touching and grabbing in order to perform specific purposes. Experimental results show that the efficiency of the proposed system.

The rest of the paper is organized as follows. Relevant work regarding multimodal input mechanisms is discussed in Sec. 2. The preprocessing steps, including dynamic background updating, skin-color detection and primary component selection are explained in Sec. 3. In Sec. 4, the core algorithms of the proposed system are explicated. Experimental results are shown in Sec. 5 to demonstrate the efficiency of the proposed system. Finally concluding remarks are given in Sec. 6.

2 Relevant Works

In past decades, vision-based input mechanism is one of the major research topics in computer vision field. Great amount of efforts have been devoted to explore new possibilities of human-computer interaction. Prevalent approaches include color segmentation, image differencing, correlation, contour analysis, model fitting, motion analysis, stereovision, and so on. Since many systems combine different approaches to achieve better results, it is sometimes difficult to strictly classify certain hybrid methods. The following sections study relevant literatures.

2.1 Model Fitting Approach

Model fitting, no matter in 2D or 3D, is a widely adopted strategy for hand gesture recognition. It requires prior knowledge about the geological structure of hand gestures, usually in the form of pre-constructed frameworks or models. Then, the model is fitted in the candidate iteratedly to see if its geological structure matches the prior knowledge the model carries. Triesch *et al.* fit the target image with labeled graphs that are composed of Gabor-filter nodes called jets [1]. After constructing the standard graph from sample images, graphs can be fitted in the candidate to see if it matches well. During the matching process, nodes in the graph are allowed to have slight distortion. This method is robust to complicated backgrounds, but it not applicable to real-time process. Lee *et al.* proposed a scheme using the 3D structure to model the bones and joints of human hands [2]. The fitting process is done by searching a proper joint angle configuration to match the seven characteristic points on a hand image. However, because the proposed model has 27 degrees of freedom (DoF), enumerating and testing every joint angle combination would be not impractical. A set of joint movement constraints are therefore applied to exclude impossible configurations. Angle limits are carefully defined according to the physical properties of each joint. In model fitting process, all seven characteristic points can be matched with least error. If more than two characteristic points cannot locate their proper positions, minor error is allowed, and a weight is given to each point so that the model can be fitted according to their importance. For example, the palm has a much higher weight than fingers because the positions of fingers are strongly associated to the position of the palm. Even with all those efforts, fitting the fitting of 27-DoF model is still extremely computational intensive. Different tolerance levels to achieve a tradeoff between precision and performance are tested in the experiment. For a rough matching, the fitting process completes in 0.023 second in average. For a precise matching, it takes 45 minutes even for a single sample. Mitra et al. [3] provide a survey on the gesture recognition which including hand, arm, face, head.

2.2 Shape-Based Approaches

A typical shape-based approach usually includes feature extraction from gesture shape and analyzing the feature patterns according to predefined rules or specially trained pattern recognizers. Dhawale *et al.* proposed an efficient 3D interactive system called HAND3D [4]. The system is capable of giving elementary 3D input to the computer by operating bare hand with a single camera above the tabletop. The shape of the hand is analyzed according to standard human body proportions and its size in the camera view to determine the corresponding input. Due to its simplicity, the system can easily achieve real-time performance with high accuracy. Hardenberg *et al.* proposed another real-time interaction system with several applications [5]. In the system, an input image is first applied to extract the region of interest with a smart image differencing technique. Then, fingertips are located according to the shape features. The positions of fingertips are then tracked and used as the means of real-time interaction. Several applications are demonstrated, such as virtual painting on the wall, GUI control,

and moving virtual objects. Although many proposed approaches are just to improve the performance of previous works, particularly of those works in certain aspects. For example, Wilson proposed a prototype called "TAFFI" for robust gesture input mechanism by detecting pinching action of thumb and forefinger [6]. When a pinching action occurs, an enclosed space is created by thumb and forefinger. Pinching action is detected by looking for a cavity inside the hand shape. In the experiment with the application of satellite imagery viewing, TAFFI provides instinctive panning, rotation and moving operations over the virtual map.

2.3 Stereo Vision Approach

Stereo vision simulates human eyes by two cameras together. A disparity map, containing the information of depth, is produced by comparing the amount of local shift between the slightly different images from the two cameras. Stereo vision is often combined with other existing approaches. With the aid of depth information, the robustness of the algorithm can be improved, and new fields of application can be explored. Malik and Laszlo proposed a 3D interactive interface called "Visual Touchpad" which provides 3D operations of object manipulation [7]. The user operates over a black touchpad with bare hands. Two cameras are situated above the touchpad to record the movement of user's hands. Since the background is a black touch pad, a simple background subtraction can be done by excluding black area. User's hands are tracked by flood-filling and size filter, and each fingertip is detected through contour analysis. Since each camera produces a slightly different image, the disparity of each fingertip can be used to generate depth information to detect touching events. An application of virtual album is created to demonstrate the power of the Visual Touchpad interface. Users can drag, rotate and release photos by touching the pad and control the orientation of the finger. There are also two-handed operations for more convenient operation, such as pie menu and virtual keyboard.

Another work is proposed by Nickel and Stiefelhagen to recognize pointing gestures and estimate the direction where the hand points [8]. In this system, two cameras with the capability of stereo vision are used to track the user and locate hands and head according to 3D spatial relations. The pointing action is detected by three specially trained HMMs. When a pointing action occurs, the system analyzes the 3D coordinates from the forearm part and applies PCA to determine its orientation. The head-hand direction is also estimated by lining up the coordinates from the head to the pointing hand. A magnetic orientation sensor is put on the user's head as a benchmark to compare the precision of different schemes. According to their experiments, the head-hand direction is more representative than the other two methods with lower error angle and higher rate of target identification.

3 Proposed Algorithms

Fig. 1. shows the flowchart of our proposed system. In the preprocessing stage, the foreground scene is extracted by the background subtraction technique, and the non-skin color parts are excluded by the skin color detection; then the primary skin-color component can be determined. In the analysis stage, palm part has to be extracted to estimate the curving coefficient for hand gesture recognition when face and arm are excluded effectively.

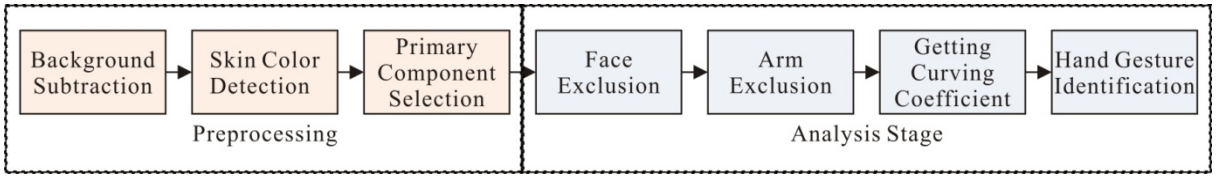


Fig. 1. Flowchart of the proposed vision-based virtual control system

3.1 Preprocessing

Before hand gesture recognition, some preprocessing operations should be first applied to raw data captured by camera to make further analysis easier as well as improve the robustness of our system. These common operations are organized in the following section. Raw image is classified into different regions by its identities. An example of the preprocessed image is shown in Fig. 2. Green area represents background pixels, while blue represents foreground pixels. Red represents skin-color areas that are not selected as the primary component. The original color of the primary component remains unaltered.

3.1.1 Background Subtraction

In the beginning, the proposed system separates foreground object from background image such that the region of interest (ROI) region would be extracted clearly. If we perform skin color detection directly on captured frames without this procedure, background objects with similar skin color will be included. The most straightforward way of background subtraction is to use a predefined image as background image that is subtracted from target image (captured image) to obtain foreground object. The predefined image is usually the first frame of captured images. However, this approach suffers from many limitations such as non-static background environment, luminance change, or occlusion. For solving these problems, we develop an adaptive algorithm for dynamic background updating.

We first define static pixel that the pixel values at the same position of two continuous frames remain unchanged. Static pixel can be expressed as

$$if \quad \left\| P_n(x, y) - P_{n-1}(x, y) \right\| < T,\ P_n(x, y) \in static\ pixel\ , \tag{1}$$

where n is the frame number. $P_n(x,y)$ and $P_{n-1}(x,y)$ mean the pixel values of the position (x, y) of the current frame and the previous frame, respectively and T is a predefined threshold. The basic concept is to restore a background image which updates at any moment and compares it to each input frame. We can recognize foreground objects and background by Eq. (2)

$$P_n(x, y) \in \begin{cases} foreground & if \quad \left\| P_n(x, y) - P_{n-1}(x, y) \right\| > T \ or \ \ P_n(x, y) \in static\ pixel \\ background & otherwise \end{cases}, \tag{2}$$

where P_B is the background image.

If a pixel is classified into foreground and remains unchanged, it has potential to be a background pixel called *potential background pixel*. We calculate the time that being unchanged for potential background pixels between two consecutive frames.

$$R'_B(x, y) = \begin{cases} 0 & initialize \\ R_B(x, y) + 1 & if \ P_n(x, y) \in background \\ R_B(x, y) - 1 & if \ P_n(x, y) \in potenial\ background \\ R_B(x, y) & otherwise \end{cases}, \tag{3}$$

where R_B represents the frame counter of the pixel. If the pixel value at (x, y) matches one at the same position in the background image, it gains a positive vote and vice versa. When the counter reaches zero and less than counter of original background pixel, it will be replaced by latter. It means that the time of latter that being unchanged is longer than former; we should reset it as the new background pixel for updating purpose. In the initial step, we have no updated background image, so we use the first frame as our initial background image. Here, an upper-bound value R_{Max} is set for the repetition frequency. Without such limitation, the repetition frequency will grow infinitely, and new background may take unreasonable long time to become the estimated background.

Furthermore, the images captured from cameras usually carry much noise, especially general consumer-level webcams. To tackle this down, we introduce the weighted average method [9] as follows,

$$R'_B(x, y) = \alpha R_B(x, y) + (1 - \alpha) P_n(x, y)\ , \tag{4}$$

where α is a weighting coefficient while $P_n(x,y)$ belongs to background pixels. After updating background several times, we could obtain a better background image and subtraction results. It also reduces the probability of false detections caused by noise. Finally, some morphological operators such as dilation and erosion are used to clean up the rest of scattered small noise for further refinement. Fig. 2. shows the result of the proposed background updating scheme. Foreground objects are shown in blue, green for background and red represents skin-color areas that were not selected as the primary component.

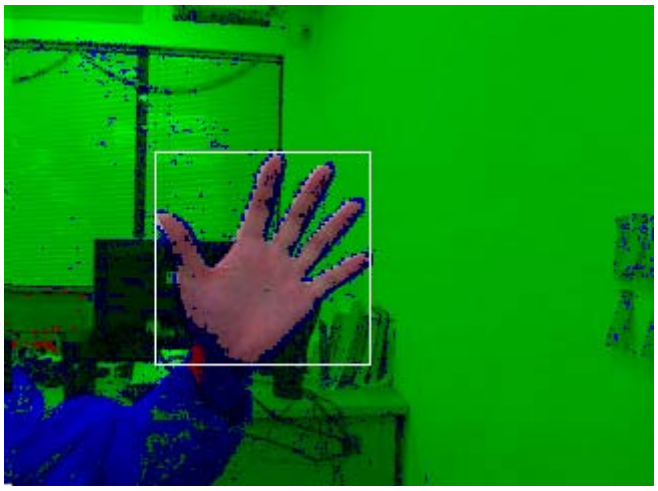


Fig. 2. The result of our background updating and skin-color detection algorithm

3.1.2 Skin Color Detection and White Balance

Skin color detection is another important function in the preprocessing stage. By filtering out the non-skin color pixels of an image, we can classify the region of interest. As mentioned above, the color of objects in the background may be similar to skin color and tough to eliminate. After the separation of foreground and background regions, it is much easier to perform skin color detection on foreground regions. The luminance condition also affects skin color detection seriously. For example, brighter or dimmer skin regions have different brightness values although they have the same hue. To reduce such effects, we should choose a color space with its brightness value independent of other components. Many candidates such as normalized RGB, YUV, HSV, YCbCr[10], and etc. have been proposed. Here, HSV is chosen for skin color detection in our proposed system after evaluating the balance between precision and performance. The HSV color space represents colors with hue, saturation and luminance, in a manner similar to human vision. By separating the brightness component, skin color detection in the HSV color space is more robust against the luminance change. To detect skin color, we use simple decision rules instead of other computational intensive approaches to lower the amount of computation. The rules are summarized as follows:

$$\begin{aligned} &0 < H < 51, \\ &100 < V < 226, \\ &0.0028 < S < 0.724. \end{aligned} \tag{5}$$

These empirical rules mark the distribution of human skin color. The first rule defines a rough range of skin color hue, while the second rule excludes pixels that are either too bright or too dark to be skin color. At last, the third rule examines the saturation of the pixel to ensure that color is not too gray (<0.0028) nor too unnatural (>0.724). The bounds of H, V and S in Eq. (5) are 0 to 360, 0 to 1.0 and 0 to 255, respectively.

The luminance provided by the environment also affects skin color detection in the aspect of white balance. The same object may present different colors under different light sources. For example, a piece of white paper looks exactly white under a 6500K flourish light but yellow under a light bulb. Fig. 3. shows an example of two images derived from the same image with different white balance settings. The paper in each image can be perceived white by human eye. However, the paper on the left have a bluish tone, while the paper on the right have a yellowish tone. Human eyes automatically correct such bias with prior knowledge to perceive the colors as the same. Such bias must be corrected through the means of white balance adjustment. White balance can be performed automatically, manually, or use dedicated preset values for certain given scenarios. Automatic white balance is commonly used on digital cameras for quick snap shots with imprecise but acceptable outcome. Manual white balance gives the best outcome but requires user to adjust the settings manually. Preset white balance mechanism uses pre-adjusted values for each kind of scenario for user selection, so that good results can be achieved without much user intervention. For the precision of skin-color detection, we use manual white balance. Automatic white balance is not preferable due to precision issues as well as the interference to background subtraction mechanism caused by dynamic color adjustments.

Fig. 3. Effect of different white balance settings

3.1.3 Primary Component Selection

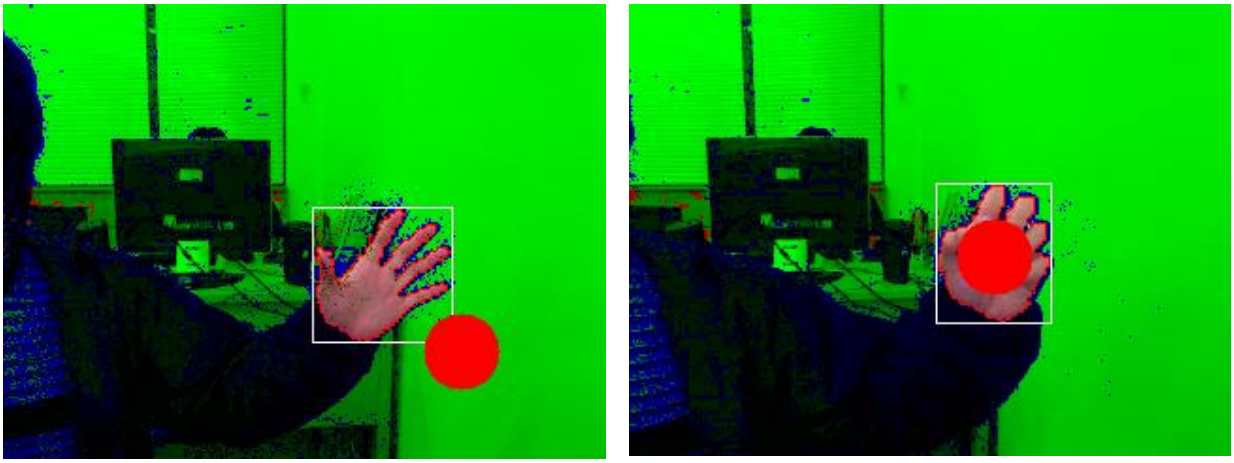
After skin color detection, all skin color pixels in the foreground are marked. However, if multiple objects are detected in a single frame, such as hands or faces of other persons passing through camera view, we must then determine which parts belongs to the primary component. First, we perform the 4-way connected components algorithm to connect the marked skin color pixels, and the pixels with the same connected number will form a component. After applying this algorithm, all isolated components are distinguished. Each component is marked with a distinct number. Then, we select the largest component as our primary component and ignore other un-important ones. Fig. 4(a) and 4(b) shows the original image and the result of primary component selection, respectively. In Fig. 4(b), the component is shown with the same gray value and the largest component is marked with a red rectangle. The size of the selected primary component is also checked whether the coverage area is greater than a predefined threshold in order to guarantee that the select component is meaningful. The selected primary component is then passed to the next stage: the analysis stage.

(a) (b)

Fig. 4. An example of connected components

3.2 Virtual Catch Ball Game

For catching ball, there are several basic commands such as catching, dragging, and tossing. In our system, catching and tossing event are simulated by holding and opening hand. Dragging event involves both of the two functions: hold your hand to select the object you want to drag, move the hand to where you want an object to be, and then release your hand, as shown in Fig. 5.



(a) (b)

Fig. 5. Examples of virtual catch ball actions (a) approaching and (b) grabbing and dragging

3.2.1 Face Exclusion

When capturing gestures, user's face will inevitably come into the camera view. Since face is usually larger than hand in the camera view, it may disturb the function of primary component selection. Thus, it is essential to consider face exclusion for virtual control mechanism. To detect human face, we must find out the common characteristics of facial features, such as eyes, ears, nose, or mouth. There is no need to employ complex algorithms here because the preprocessed data delivered to this stage are foreground objects independent of background interference. We only need to exclude everything except the hand. After several experiments, we observe human face exclusion by lip color detection is an efficient method with high accuracy [12]. Lips have a distinct chromatic distribution from ordinary skin color and are easy to be distinguished with a set of simple rules:

$$\begin{cases} R-G \geq 110 \\ G-B \geq 36 \end{cases}, \tag{6}$$

Although these rules do not offer perfect classification for lip color, they are good enough to distinguish the differences between face and hand. These rules are applied to all pixels in the primary component so that lip color can be detected. To reduce the error caused by noise, a pixel with lip color will be marked as a lip pixel only when at least one of its neighboring pixels also has lip color. All independent lip color pixels is regarded as noise and discarded.

The main idea of the face exclusion algorithm is to compare the area of lip pixels to the whole component. Lip pixels significantly occur in the face component. In contrast, lip pixels in a hand component seldom occur that most of them are discarded as noise. Let N_{All} and N_{Lip} be the number of pixels of the whole component and lip pixels in the component, respectively. Therefore, face component can be excluded according to the following rule:

$$\begin{cases} \dfrac{N_{Lip}}{N_{All}} \geq R_{Lip} \\ N_{Lip} \geq T_{Lip} \end{cases}, \tag{7}$$

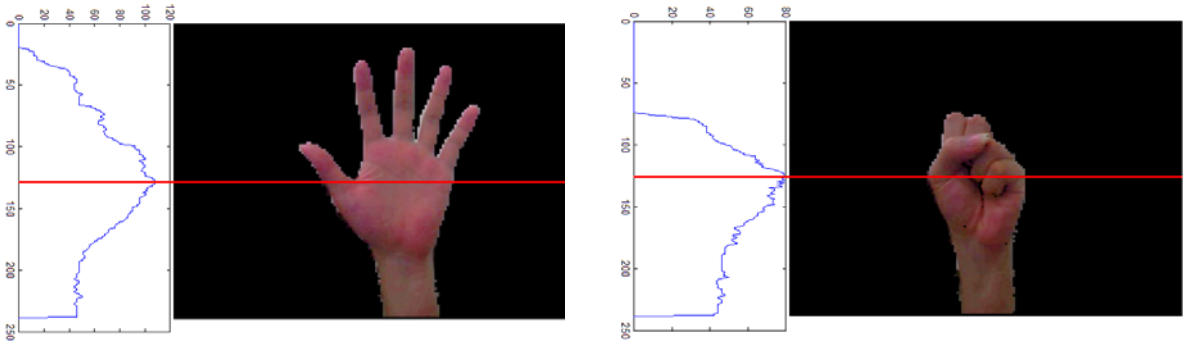
where R_{Lip} is the threshold of lip pixel ratio, and T_{Lip} is the threshold of the total lip pixels. In our system, we set $R_{Lip} = 0.001$ and $T_{Lip} = 8$. When the selected primary component fits the rules above, it will be discarded. The second largest component will be selected instead of the original one and again be examined by these rules. Such step is repeated until a hand component is found or no component is left.

3.2.2 Arm Exclusion

In this step, we separate hand part from the arm. The width of bounding box of the primary component is used as a cue to locate where the wrist is, and only the part above the wrist remains. Let D_{max} denote the vertical position where the maximum horizontal density occurs; if we analyze the horizontal density of the hand, we can find that D_{max} occurs in the middle part of the palm no matter what the user's gesture is (as shown in Fig. 6.). Then, we define the parameters w and h *as* follows: w is the width of the component, and h represents the distance from wrist to where the w is on the y-axis as shown in Fig. 7. The width of palm is usually nearly 2 times longer than h. According to this natural rule, let L_{Up} and L_{Down} represent the position of upper border and lower border of bounding box of entire hand component, respectively. Entire arm exclusion procedure can be expressed as

$$\begin{aligned} h &\approx 0.5w \\ Y_{wrist} &= Y_w + h' \end{aligned}, \tag{8}$$

where Y_{wrist} and Y_w are the y-positions of the wrist and w, respectively. The pixel belongs to skin-color and its y-position is below the Y_{wrist} will be removed. The rest of the component is the palm we need.



(a) (b)

Fig. 6. The property of horizontal density

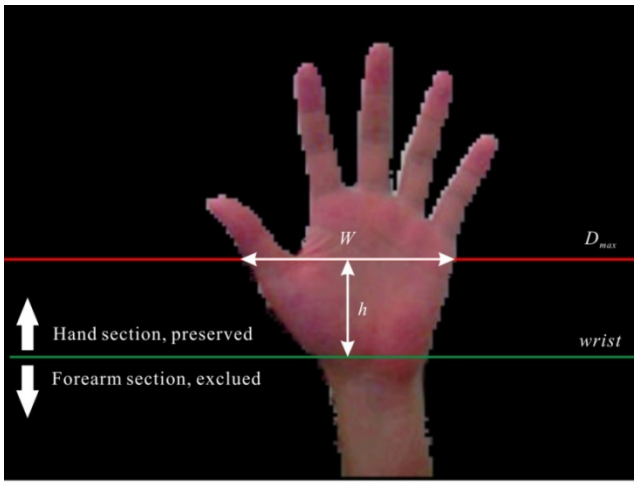


Fig. 7. Special relation between D_{max} and the wrist

3.2.3 Hand Gesture Identification

After obtaining palm region, we can recognize the hand gesture for control purpose. For each point Q on hand contour, there is a tangent unit vector $V_T(Q)$ and a vector $V_O(Q)$ from Q to the centroid. As we know, the inner product of two vectors is zero implies that they are orthogonal shown in Fig. 8(a). According to this attribute of linear algebra, we can get the cosine value of two vectors, $V_T(Q)$ and $V_O(Q)$, which are calculated by all points on the contour. When the value approximates to zero, the contour portrays nearly a circle which shown in Fig. 9. The curving coefficient C_{rv} can be calculated by

$$C_{rv} = \frac{\sum_{Q \in B_{hand}} \frac{(\overrightarrow{V_T(Q)} \cdot \overrightarrow{V_O(Q)})^2}{\left\|\overrightarrow{V_T(Q)}\right\|^2 \left\|\overrightarrow{V_O(Q)}\right\|^2}}{size(B_{hand})}, \tag{9}$$

where B_{hand} represents the border of the contour. The curving coefficient C_{rv} can be used to estimate the curving level of the contour. Once we obtain the coefficient through a predefined threshold Γ_{curve}, we can determine which the current hand state is as follows

$$state = \begin{cases} hold & if\ C_{rv} > \Gamma_{curve} \\ open & if\ C_{rv} \le \Gamma_{curve} \end{cases}. \tag{10}$$

In Eq. (10), *state* denotes the determined hand gesture. If C_{rv} is greater than Γ_{curve}, the state is set to be hold; otherwise it is set to be open.

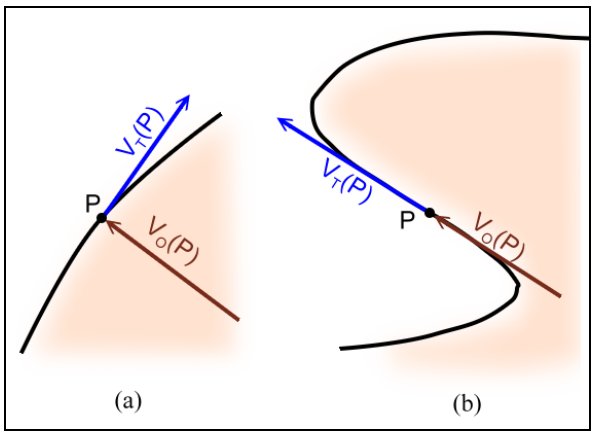


Fig. 8. (a) Two vectors at point P orthogonal to each other (b) Two vectors at point P parallel to each other

(a) (b)

Fig. 9. Tangent vector (yellow) and unit vector (red) are shown at each sampling point

4 Experimental Results

The proposed system is built on Microsoft DirectShow API [13, 14, 15] and Visual Studio 2005. A virtual ball game program is implemented by the proposed algorithms shown in Fig. 10. In this program, user can control the soccer on the application window. User holds his hand forming a fist to grab the ball, and drag to move the

ball with users' fist. While user opens his hand, the ball is released and bounce according to the direction and velocity at the releasing moment.

In aspect of hardware, we just simply need single consumer-level webcam with normal resolution, 320×240. The one we used in system is Microsoft LifeCam VX-6000 type. The application is tested on the computer with Pentium 4 Core 2 Duo 3.4Ghz CPU, 1GB RAM and Windows XP SP2. The accuracy of hand gesture detection is about 90% in normal luminance conditions, but according to rules of thumb, accuracy decreases when in environment illuminated by halogen bulb. The average frame rate of the system reaches about 27 fps, and average CPU usage is below 30%. However, such a high performance relies on webcam with higher default frame rate. Another consumer-level webcam was used during the development, but its hardware default frame rate is lower than Microsoft's, thus the average frame rate reaches only about 17 fps. Even with the barely satisfactory frame rate, it is still fast enough to meet the demand for real-time application.

$$accuracy = \frac{number\ of\ true\ positives + number\ of\ true\ negatives}{number\ of\ true\ positives + number\ of\ false\ positives + number\ of\ true\ negatives}. \quad (11)$$

Table 1 shows the accuracy of the system by Eq. (11). It was tested two environment, without human face and with face exclusion mechanism, in the screen in normal luminance environment. We perform 100 operations each actions on both tests in the same luminance condition and environment. Despite the accuracy of the system with face exclusion mechanism is lower than test 1, it is still accurate enough to control the system. All these experimental results prove the reliability and efficiency of our system.

Table 1. Accuracy analysis of the proposed system

Action	Without human face	With face exclusion
Select	96%	90%
Release	98%	93%
Drag	95%	92%

Fig. 10. Illustration of virtual catch ball game

DirectShow also provides capability of data output in RS-232 format. In this system, we output centroid of the hand component to a DSP simulator in RS-232 format shown in Fig. 11. Such high performance and flexibility promise its widen applications in multimedia and human-machine interaction fields. The proposed system might become the prototype of advanced interface of human-machine interaction in the near future.

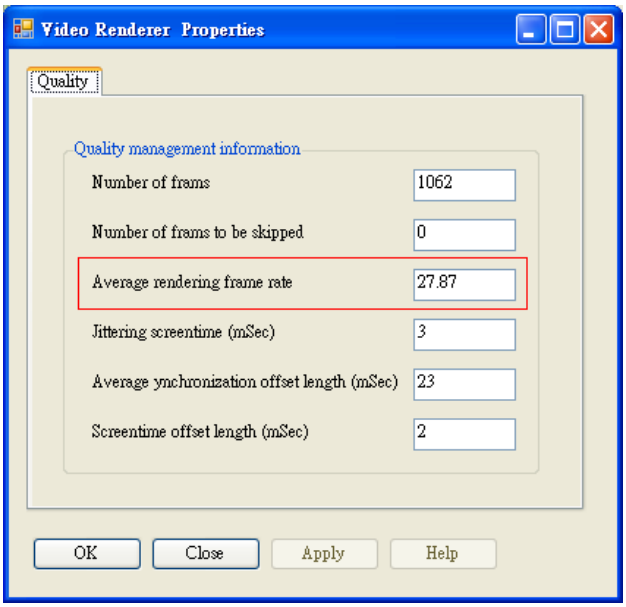


Fig. 11. Average rendering frame rate retrieved from an original DirectShow filter in utility which is called GraphEdit

5 Conclusions

In this paper, a real-time virtual control system is proposed via hand gesture recognition. The proposed system provides an effective way for virtual input by image processing and visual content analysis techniques. The proposed system efficiently excludes the disturbance of human face and arm to make the system more reliable. According to experimental results, the proposed system is fast, simple and efficient. A virtual ball game program that is implemented on DirectShow platform demonstrates the control behavior of the proposed algorithms. The proposed system needs neither auxiliary sensors nor high computing power, thus minimizing the additional hardware cost to encourage its widespread use among consumer grade devices.

In this work, the camera is considered to be static so that adaptive background subtraction can be performed without difficulty. In the future, we will modify and extend our system to non-static camera so that we can port our work on the mobile devices.

Acknowledgement

The authors would like to thank the Chung-Shan Institute of Science and Technology for financially supporting this research under Contract No. BV97K01P and XV98K61P134PE.

References

[1] J. Triesch and C. V. D. Malsburg, “Robust Classification of Hand Postures against Complex Backgrounds,” *Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition*, pp. 170-175, 1996.

[2] J. Lee and T.L. Kunii, “Model-Based Analysis of Hand Posture,” *IEEE Computer Graphics and Applications*, Vol. 15, No. 5, pp. 77-86, 1995.

[3] S. Mitra and T. Acharya, “Gesture Recognition: A Survey,” *IEEE Transactions on Systems, Man, and Cybernetics*, Part C: Applications and Reviews, Vol. 37, No. 3, pp. 311-324, 2007.

[4] P. Dhawale, M. Masoodian, B. Rogers, “Bard-Hand 3D Gesture Input to Interactive Systems,” *Proceedings of the 7th ACM SIGCHI New Zealand Chapter’s International Conference on Computer-Human Interaction: Designed Centered HCI*, ACM International Conference Proceeding Series, Vol. 158, pp. 25-32, 2006.

[5] C. V. Hardenberg and F. Berard, "Bare-Hand Human-Computer Interaction," *Proceedings of the 2001 Workshop on Perceptive User Interfaces*, Vol. 15, pp. 1-8, 2001.

[6] A.D. Wilson, "Robust Computer Vision-Based Detection of Pinching for One and Two-Handed Gesture Input," *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*, pp. 255-258, 2006.

[7] S. Malik and J. Laszlo, "Visual Touchpad: A Two-Handed Gestural Input Device," *Proceedings of the 6th International Conference on Multimodal Interfaces*, pp. 289-296, 2004.

[8] K. Nickel and R. Stiefelhagen, "Pointing Gesture Recognition based on 3D-Tracking of Face, Hands and Head Orientation," *Proceedings of the 5th International Conference on Multimodal Interfaces*, pp. 140-146, 2003.

[9] T. Yang, S.Z. Li, Q. Pan, J. Li, "Real-Time and Accurate Segmentation of Moving Objects in Dynamic Scene," *Proceedings of the ACM 2nd International Workshop on Video Surveillance & Sensor Networks*, pp. 136-143, 2004.

[10] D. Chai and A. Bouzerdoum, "A Bayesian Approach to Skin Color Classification in YCbCr Color Space," *Proceedings of the IEEE Region 10 Conference*, Vol. 2, pp. 421-424, 2000.

[11] R.L. Hsu, M. Abdel-Mottaleb, A.K. Jain, "Face Detection in Color Images," *Proceedings of the IEEE International Conference on Image Processing*, Vol. 1, pp. 1046-1049, 2001.

[12] C.C. Chiang, W.K. Tai, M.T. Yang, Y.T. Huang, C.J. Huang, "A Novel Method for Detecting Lips, Eyes and Faces in Real Time," *Journal of Real-Time Image*, Vol. 9, No. 4, pp. 277-287, 2003.

[13] M. Blome and M. Wasson, "Core Media Technology in Windows XP Empowers You to Create Custom Audio/Video Processing Components," *MSDN Magazine of Microsoft Developer Network***,** No. 2002, 2002.

[14] Microsoft Developer Network, Windows Media Developer Center, "DirectShow," http://msdn.microsoft.com/en-us/library/ms783323(VS.85).aspx.

[15] Wikipedia, "DirectShow," http://en.wikipedia.org/wiki/DirectShow.