A FINGER-TRACKING VIRTUAL MOUSE REALIZED IN AN EMBEDDED SYSTEM

Wai-Wah Martin Tsang and Kong-Pang Pun
s016404@ee.cuhk.edu.hk and kppun@ee.cuhk.edu.hk
Department of Electronic Engineering
The Chinese University of Hong Kong, Shatin, N.T., Hong Kong

Abstract - **In this paper, a virtual mouse that is based on tracking the movement of a finger is presented. The idea is for users to control their computer or TV system simply by moving their fingers, without any contact on realistic objects. A fast and robust method that can trace the position of finger tip is proposed. The method consists of four steps. First, the skin-color pixels of figures are detected by color segmentation using the chrominance component of the input from a CMOS image sensor. Second, density regularization processes are carried out to reinforce the regions of skin-colour pixels. Third, an effective window search technique is applied for minimizing computational cost. Fourth, a finger-tip tracking algorithm is applied to find out the finger-tip position. Clicking action is also implemented by a specific movement. The virtual mouse has been designed in an embedded Linux system. The device works successfully, the response is quick and accurate positioning is obtained.**

I. INTRODUCTION

The mouse interface of a computer system or other electronic appliances is traditionally based on the direct contact of some realistic objects. The freedom of moving fingers through the air to control the computer is frequently imagined in some popular movies. The main objective of this work is to design a device which allows users to operate their mouse through the movement of their fingers in the air. We call this device as a "virtual mouse". The virtual mouse can find its applications in situations where physical contact is highly desirable, for instance, for use of disabled people.

A main problem in designing the virtue mouse is the accurate positioning. In this paper we use the finger tip as the pointer of the mouse, and the movement of the finger is tracked by a fast and robust detection algorithm. Further more, the mouse clicking functions are also implemented. All these algorithms are realized in an embedded Linux system rather than in a powerful personal computer system.

The rest of this paper is arranged as follows. Section II proposes the finger tracking algorithm. Section III presents the system architecture, both hardware and software. Section IV shows the experimental results of the prototype virtual mouse, and lastly section V concludes the paper.

II. THE FINGER TRACKING METHOD

The proposed finger tracking method consists of five steps, namely, the RGB to YUV conversion, colour segmentation, density regularization, window searching and finger tracking as illustrated in Fig.1. The first three steps are used to extract the pixel of the same colour range with the input YUV parameter. The last two steps are applied to extract the finger (X,Y) coordinate.

A. RGB to YUV conversion

The first stage of this YUV finger tracking algorithm is to have the YUV data. The first step is to convert RGB image raw data to YUV data. To have this conversion, the equation can be found from the document BT.601-5 from ITU. There is a standard of transforming RGB to YUV. Since the formula is suitable for 24bit image only, the 16bit image is first converted to 24bit.

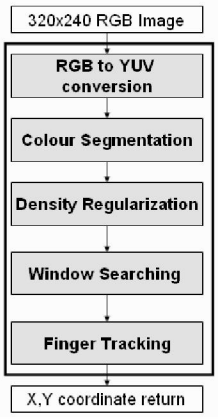

Figure 1. YUV colour space finger tracking flows.

B. Color Segmentation

The first stage of the algorithm involves the use of colour information in a fast, low-level region segmentation process. The aim is to classify pixels of the input image into skin colour and non-skin colour. To do so, we have devised a skin-colour reference map in YCrCb colour space.

A skin-colour region can be identified by the presence of a certain set of chrominance (i.e., Cr and Cb) values distributed narrowly and consistently in the YCrCb colour space.

We denote and as the respective ranges of Cr and Cb values that correspond to skin colour, which subsequently define our skin-colour reference map [10] . The ranges found to be most suitable for all the input images that we have tested are [200 150] and [110 70].

With this skin-colour reference map, the colour segmentation can now begin. Since we are utilizing only the colour information, the segmentation requires only the chrominance component of the input image.

The equation below is used for colour segmentation.

$$O_1(x,y) = \begin{cases} 1, if[Cr(x,y) \in R_{Cr}] \bigcap [Cb(x,y) \in R_{Cb}] \\ 0, otherwise \end{cases}$$

C. Density Regularization

This stage considers the image produced from the previous stage to contain the facial region that is corrupted by noise. The noise may appear as small holes on the finger region due to the undetected finger features such as edges, shadow area, or it may also appear as object with skin-colour appearance in the background scene. Therefore, this stage performs simple morphological operations such as *dilation* to fill in any small holes in the facial area and *erosion* to remove any small object in the background area. The intention is not necessarily to remove the noise entirely but to reduce its amount and size.

To distinguish these two areas, it is first needed to identify regions of the bitmap that have higher probability of being the facial region. The probability measure is to derive from observation that the finger colour is very uniform, and therefore the skin-colour pixels belonging to the finger region will appear in a large cluster, while the skin-colour pixels belonging to the background may appear as large clusters or small isolated objects.

$$D(x,y) = \sum_{i=0}^{1}\sum_{j=0}^{1} O_1(2x+i, 2y+j)$$

$$O_2(x,y) = \begin{cases} 1, if D(x,y) = 4 \\ 0, otherwise \end{cases}$$

D. Window Searching

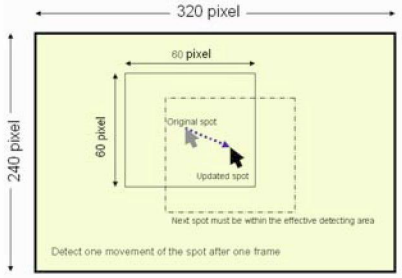

Figure 2 Search Window

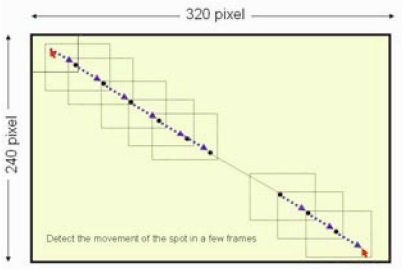

Figure 3 Movement of searching window

In an embedded system, colour comparing and finger tracking for all pixels are computationally expensive. Due to the limited processing power of the application processor, an exhaustive search of the whole frame is impossible. It is necessary to reduce the searching area. This is done by defining a certain search region around the last detected position of the finger. Another benefit of this method is to reduce the detection error. The project first assumes that the displacement of the finger in between two consecutive frames will not excess a particular searching region. Any movement outside the search region is not seen as the displacement of a finger. In other words, it is neglected. Therefore, detection error can be eliminated. Another assumption is that the time for a finger to move from one corner to the diagonal corner will not access the time pre-set so that the searching region in that time will cover the displacement of the finger.

The algorithm does not calculate the velocity and acceleration of the finger. However, it presets the search window size is 60pixels x 60 pixels where the centre of the window is the last known finger position. In the next frame, the possible finger position that can be detected must not excess 30pixels in both x, y direction. For a movement from one corner to another diagonal corner, the x – direction movement will be the determining steps because the longest path has 320 pixels. Assuming that the frame rate for the CMOS sensor is 24frames/second, the fastest movement of the finger should not excess the time calculated below:

$$T_{MIN} = T_{one\ frame} \times \text{No. of Frame} = \frac{1}{24} \times \frac{320}{30} = 0.4444s$$

The minimum time for the movement should be no less than 0.4444s, which is appropriate for the real time case. Any fast movement requiring less time will be fail to be detected. The search window size algorithm is shown here:

$$|X_{t-1} - X_t| < \frac{60}{2} \qquad |Y_{t-1} - Y_t| < \frac{60}{2} \qquad \forall x, \forall y$$

Any x and y satisfy the above equations will be within the searching window.

E. Finger Tracking

This step is to find out the exact fingertip position from the window that is found in the previous step. In the 60X60 pixels size of window, it is assumed that nothing will be appeared exactly as the finger shape after applying the previous steps. By this approach, it is easy to find out the fingertip with the peak of the finger. This is done by searching the $O_2(x,y) = 1$ from the top to the bottom.

F. Click Recognition

As a mouse, click function is necessary for control. Using a finger as a mouse, click function is defined by a motion. The clicking motion is checked by an algorithm such that any motion that matches the clicking motion will be accounted as a CLICK.

The algorithm takes last 20 positions for the calculation. With these 20 positions putting in an array, if the array values matches the clicking motion, CLICK motion is detected.

III. SYSTEM ARCHITECTURE

A. Software System Architecture

In the beginning, CMOS sensor driver is needed to initial the CMOS sensor. With particular setting and programming, the CMOS sensor will send out 320X240 16bit image raw data to the memory. With a suitable YUV range parameter, the YUV finger-tracking algorithm can process the input image to extract out the fingertip position as the mouse coordinate. With the mouse coordinate, clicking recognition can check if a specify click motion is done or not. Afterwards, two different output methods will be selected according to the purpose. For USB mouse, a GPIO (General Purpose IO) driver will be applied to initial the USB mouse controller interface. Then, the hardware can be

connected to the computer through USB cable to facilitate the functioning of a mouse. For TV system DEMO, a TV card driver is implemented to initial the TV-out card. By controlling the frame buffer, a GUI interface with a mouse pointer can be programmed for the TV OUT display.

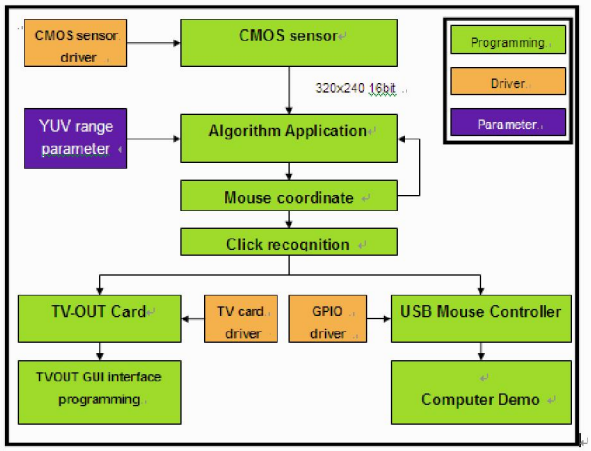

Figure 4 Software Architecture

B. Hardware System Architecture

In the development process, the base acts as the hardware system. All the software systems ands algorithms develop based on the hardware system.

The processor is Freescale® i.MX21 application processor. It has an operating frequency of 266MHz and a core of ARM 9. The ADS development board of the i.MX21 has 2X32MB SDRAM and 2X32MB NOR Flash onboard which has installed an embedded linux. There is also one 300k pixel CMOS sensor attached with the ADS. For display, it can output to an LCD panel or a TV-OUT card.

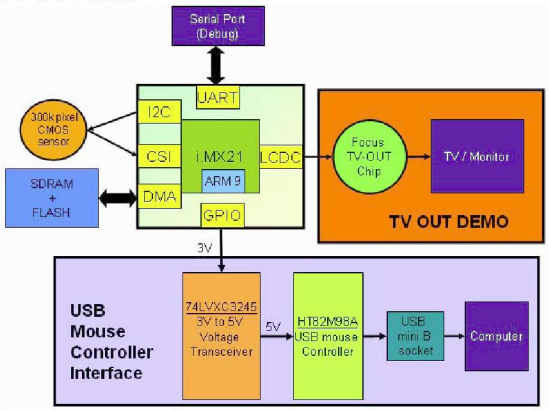

Figure 5 Hardware Architecture

IV. EXPERIMENTAL RESULTS

A. Color Segmentation

A skin-colour region can be identified by the presence of a certain set of chrominance (i.e., Cr and Cb) values within a narrow range. During the development, many sets of chrominance have been put to trials and their results are shown in the below table.

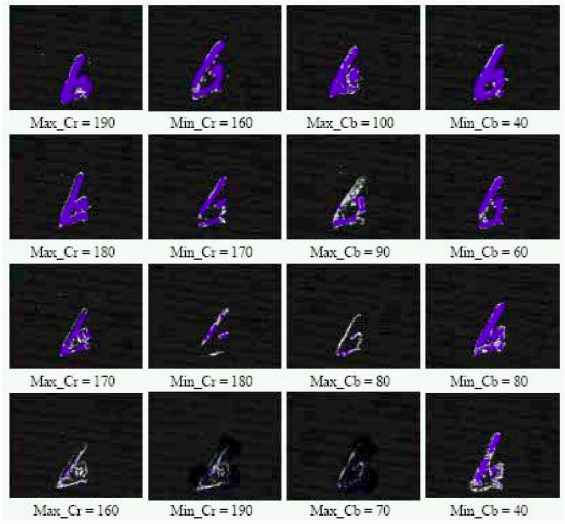

Figure 6 Different Set of Colour Segmentation

After many set of testing, the range of Cr is [200 150] and Cb is [110 70]. In this range, nearly all of the background is removed and only the hand is left.

However, different lighting condition may slightly affect the effectiveness of the colour segmentation by this colour range. Therefore, it is better to find out the most suitable colour range in different condition so that the best performance can be achieved.

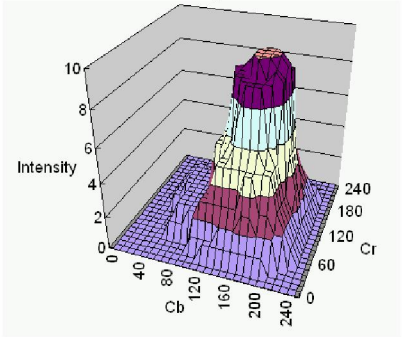

Figure 7 Skin intensity under different Cr and Cb values

B. Density Regularization

The purpose of density regularization is to fill in any small holes in the facial area and *r*emove any small objects in the background area. By the equation stated in the section II, the image density from the colour segmentation will then be regularized. An example is shown by figure 3.7 in which the finger is white at first. After density regularization, the effective area of the finger is reduced to the blue area. This can remove some noise in the background and be able to fill up some small holes inside the finger shape area.

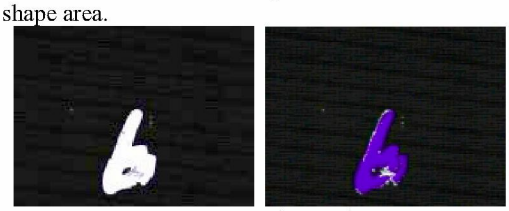

Figure 8 Images before and after density regularization

C. Window Searching

If the searching window is small, fast movement will not work as the finger is not in the window to be found. For normal speed movement, the mouse pointer can still follow the movement of the finger, meaning that the finger is still moving within the searching window.

If the searching window is too large, there will be pros and cons. The advantage is that fast movement can also be found. One of the disadvantages is the increased processing time and resources. Another disadvantage is an increase in the error detection possibility. Since the recognition is still under improvement, the window size can be optimized when the recognition part is fully developed.

D. Finger Tracking

This method is to search the fingertip from the top of the searching window. When the finger appears in the searching window, the top of the finger tips will be tracked and returned the position as mouse coordinate.

This method works when there is no similar colour object in the background or when the noise is very small. Or else, the mouse pointer will try to follow the other object instead of the finger if it finds another pixel within the same searching window.

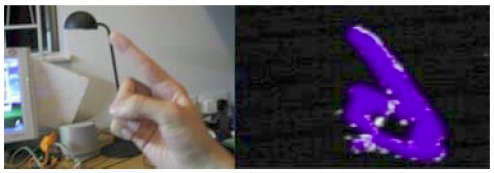

Figure 9 Images before and after finger tracking algorithm

V. COMPARSION TO OTHER EXISTING METHOD

There are many other existing finger tracking methods that has been proved to be worked. They are Colour Tracking System, Correlation Tracking System and Contour Based Tracking System. This paper belongs to Colour Tracking System.

There is one project use an infrared camera. Infrared cameras can detect light emitted from a surface with a certain temperature range. By setting this range to approximate human body temperature, the hand region can be segmented with near perfect accuracy. This method consists of expensive hardware.

Some method uses Correlation Tracking System. This method works well with slow movements and the background is relatively uniform.

Several methods use Contour Based Tracking System to extract hand posture information. This method is very restrictive on background conditions.

There is also a method using region-based algorithm. It first creates a hypothesis about hand state to build a hand model. To find out the finger, it has to calculate the similarity between the hand model and the image.

This paper proposed a method allowing the finger tracking to work well under a non-restrictive background and with quick finger movement.

Most of these methods use a USB camera together with a computer. This method is low cost which consists of a low quality CMOS image sensor and a low processing power application processor. It allows the mouse to be a standalone hardware.

VI. CONCLUSION

This paper presents a new approach to use the free movement of a finger in the air as the mouse control. It consists of several image processing algorithms, including color segmentation, window searching and a finger tracking method. The image processing is based on YUV colour space. The pixels belonging to skin colour, in YUV color space, exhibit similar chrominance values. With this skin-colour map, the pixels of the input image can be classified into skin colour and non-skin colour. Consequently, the resulted image can be used for finger tracking. A mouse coordinate is extracted after having tracked the finger tip. A clicking recognition algorithm is also designed.

A prototyping device is built with a USB mouse controller interface. It can connect to a computer becoming a real mouse. The functionality of the virtual mouse is also demonstrated in an AV control system. The prominence of the work is that the whole system is running on an embedded system which is far cheaper and highly mobile. The design can be implemented into a number of other products, such as those for use of disabled people.

ACKNOWLEDGMENT

The authors thank Prof. K.N. Ngan of CUHK for his technical advices on the project.

References:

[1] Foley, J. and van Dam, A., Fundamentals of Interactive Computer Graphics, Addison-Wesley, 1982.

[2] Christian von Hardenbery, Fingertracking and Handposture Reconition for Real-Time Human-Computer Interaction, MPhil thesis, der Technischen University Berlin, 2001

[3] Marchand-Maillet, Stephane, Binary digital image approaching: a discrete approach/Stephane Marchand-Maillet, Yazid M. Sharaiha, San Diego, Calif., London:Academic, c2000.

[4] Kiyoharu Aizawa, Katsuhiko Sakaue, Yashuhito Suenaga , Image processing technologies : algorithms, sensors, and applications, New York : Marcel Dekker, 2004.

[5] Gibson, J., *The Perception of the Visual World,* The Riverside Press, 1950.

[6] Kulessa, T. and Hoch, M., *Efficient Color Segmentation under Varying Illumination Conditions*, IEEE Image and Multidimensional Digital Signal Processing Workshop (IMDSP), Alpbach, 1998.

[7] Stafford-Fraser, J., *Video-Augmented Environments*, PhD thesis, Gonville & Caius College, University of Cambridge, 1996.

[8] Zhu, X., Yang, J., and Waibel, A.. *Segmenting Hands of Arbitrary Color*, International Conference on Automatic Face and Gesture Recognition (AFGR), Grenoble, 2000.

[9] H. Freeman, Computer processing of line-drawing images. ACM Computing Surveys, 6(1):57-98, March 1974.

[10] Douglas Chai and King N. Ngan, Face Segmentation Using Skin-Color Map in Videophone Applications, IEEE Transactions on Circuits and Systems for Video Technology, VOL. 9, No. 4, JUNE 1999

[11] Karim Yaghmour, Building Embedded Linux Systems, O'Reilly, April 2003

[12] S. Marchand-Maillet & Y.M. Sharaiha, Binary Digital Image Processing –A Discrete Approach, Academic Press, 2000.