Design and Hardware Implementation of a Fast Responsive Handsfree Mouse Using Robust Object Tracking Algorithms

KaziTauseef Mohammad[1]

Abstract—In this paper an improved pen mouse is implemented on hardware to demonstrate a robust approach for controlling mouse movements and functions. Using an illuminated marker of specific shape, color and size; actions of a mouse cursor can be effectively performed. Existing approaches involve gesture recognition of hand and eye movement to control functionality of a computer mouse. Outcome of such approaches are severely affected by the presence of background objects and surrounding light conditions. This project takes a different course towards changing the hardware design in order to use the mouse freely in the air. This technology can greatly boost the Man-Machine Interaction and allow the user of computer ease and freedom while interfacing with their devices. Our method uses combination of computer vision technology and simple inexpensive hardware that together can efficiently perform the Graphical User Interface (GUI) tasks of current commercial mouse devices (such as left and right clicking, double-clicking, and scrolling).

Index Terms—Pen mouse,Illuminated marker,Gesture recognition (GR),Man-Machine Interaction,Interfacing,Computer vision technology,Graphical User Interface (GUI),Commercial mouse

1 Introduction

In modern world computers are indispensable to our daily lives. From workplaces to education, research to entertainment we cannot imagine a single day without computers. Interaction of human with machine has taken new turns in this age of information technology. People have embraced computer as their friend. Just as interaction with a friend has to be straightforward and comfortable, Man-Machine Interaction is also evolving to allow the user to communicate with computers with simple and natural actions.

This work is greatly inspired by the current trend in developing user friendly devises and software for computers. The touch technology is constantly being upgraded to allow easy access to mobile device operation. Although touch technology has been implemented on computers but mouse functionality has still remained the main means of Graphical User Interfacing with the computer. This project aims to change the way of controlling the mouse by making its controls similar to touch actions.

2 Related Works

In his work Erden et al [1] has implemented image segmentation and gesture recognition to control the actions of a mouse. A webcam was used to capture the movement of finger tip, which is interpreted as cursor position and clicking was performed by placing another finger on top of a certain point on the captured screen. Similar works were performed Hojoon Park [2] where he used skin color detection to detect the center and position of the hand. The use of convex Hull algorithm detected the finger tips and number of pointing fingers was used to instruct certain mouse click events. The work of ShanyJophin et al [3] was an accumulation of previously mentioned works.Another approach was developed by Chu-Feng Lien[4] where heused only the finger-tips to control the mouse cursor andclick. The clicking method was based on image density, andrequired the user to hold the mouse cursor on the desiredspot for a short duration of time. The method used by Bhavana [5]suggested the use of color pointers placed on the finger tips to control mouse movement. Certain detected colored tips were used to perform certain clicking actions.The work of Abhik Banerjee et al [6] used similar tracking of colored tapes on fingers to move and control the mouse. AnkurYadav and AnandPandey [7] proposed the “Sixth Sense Technology” which suggested many uses of the gesture recognition via camera and projector. Their work proposed actions like taking pictures, computer screen navigation, painting on the screen etc with the recognized pattern of colored markers placed on the finger tips. In the work of Akhil Gupta [8]basic head and face detection algorithm and SSR filter were used to detect eye movement, which was used to control the mouse actions.

• *KaziTauseef Mohammad is currently appointed as Lecturer at Ahsanullah University of Science and Technology, and is pursuingMSc in EEE from Islamic University of Technology, Bangladesh, PH-01676027290. E-mail:kztauseef.kt@gmail.com*

3 Introduction to the System

The designed prototype hardware has a red colored backlit circular marker at the front end. The position of this marker is effectively tracked by the built-in or external webcam of the computer. The streaming image acquisition and image processing codes are written in C++ language using Visual Studios Ultimate 2012. The movement of the marker is converted to coordinates of the computer screen and the position of the mouse cursor is placed on the screen according to the movement of the marker. The basic mouse functions like clicking; scrolling is done by a micro controller which sends the input commands by a Radio Frequency (RF) transmitter. The commands are received by the computer through a USB RF receiver.This designed system has many advantages over the previous works. The features of the designed system and its edge over the other works are presented below.

3.1 Dependency of background light

The previous works were all heavily dependent on lighting condition and brightness of the background. This drawback was mentioned in most of the previous works. Due to this light dependency of the program while detecting colored markers or skin color the mouse actions have to be performed in certain illumination of the background. This sometimes makes the technology very impractical to use; for example while using the computer at dim light or using it against the outdoor background. The work presented in this paper overcomes this problem by using an illuminated Red colored marker at the tip of the designed device. As the marker itself is a luminous object it is not affected by the variation of lighting condition of the surrounding.

3.2 Color detection and background elimination

All the previous approaches applied any one of the following methods for detection.

1. Skin color detection
2. Colored marker detection
3. Pattern recognition of hands or colored markers

In each case the user had to place his hand against a white or light colored background of low brightness so that the algorithm can effectively detect the hand. Presence of other objects of similar color in the background creates false detection which results in unintended outcomes of the program. In this work an effective process of separation of foreground objects from background objects was applied. Background elimination was achieved using median blur filter, Gaussian filter and canny edge detection algorithms after detection of Red colored marker from the continuous image.

3.3 Shape detection

Even after background elimination the presence of same shades of colored objects in the foreground can cause false detection. To avoid this shape detection algorithm was applied based on Hough circle transform. Besides that the algorithm was manipulated so that only the circles of certain range of radius can be detected. Even more adjustments were made to the algorithm so that once it detects the illuminated marker it does not detect any new objects of similar shapes. The result of such shape detection procedure ensured optimized tracking of the illuminated color marker.

3.4 Basic mouse functions

The approach of the previous works was to use image processing to recognize mouse events like left and right click. Wrong detection of hand and finger positions, failure to recognize gestures and misinterpretation of gestures make the basic operation of the mouse very clumsy. Above all the computational time of image processing for both position detection of the cursor and gesture recognition of the hand make the mouse operation impractically delayed.

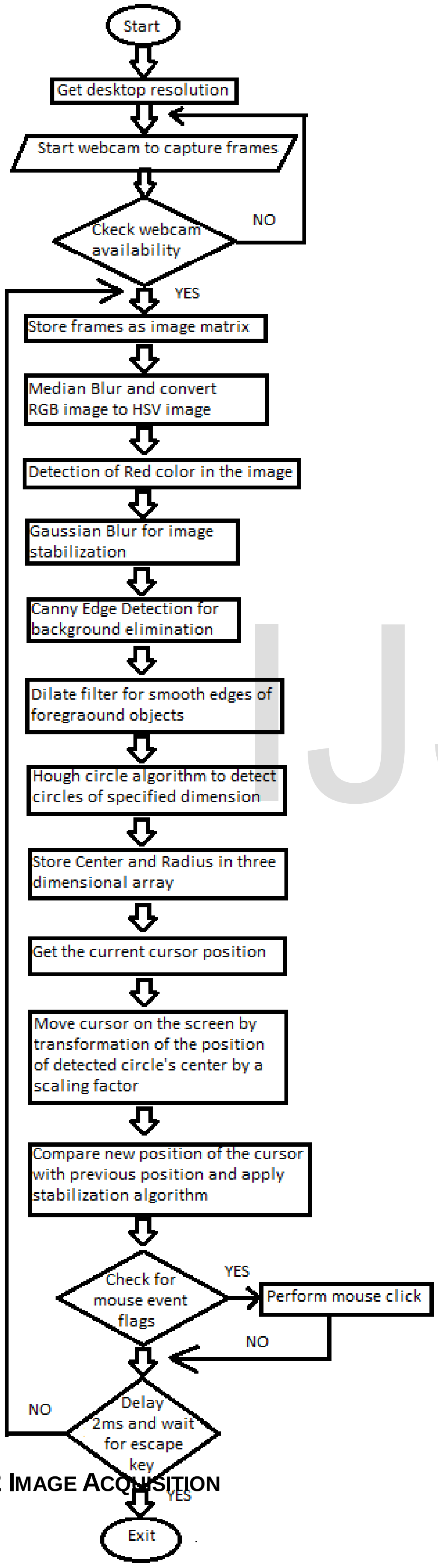
The different approach of this work is to modify the hardware of the mouse and incorporate image processing. This provides the user a virtual screen in the space in front of the computer where they can move the mouse and at the same time perfrom basic clicking and scrolling functions of the mouse with fast responsiveness of a commercially available mouse.

4 Description of the system

The implemented system is divided into two basic parts

I. Software part- it includes image acquisition, image processing and accessing the mouse driver for controlling the position and event of the mouse.
II. Hardware part- It includes the illuminated marker, webcam, PCB and chipset of the mouse, RF transmitter and receiver module. The entire project is run on a Microsoft Windows 8.1 Pro Operating system with Intel Core(TM) i5-5200U CPU @ 2.2 GH_3. The image processing task was very smoothly6 performed by the CPU. Later the image processing load was shifted to the inbuilt NVIDIA GeForce 920M Graphics card, which made the process even faster as the processor was free to perform other tasks. Although there was no noticeable change during the running of the program as both CPU and graphics card performance were very smooth.

4.1 Flow Chart of the Process


Start
Get desktop resolution
Start webcam to capture frames
Ckeck webcam availability
NO
YES
Store frames as image matrix
Median Blur and convert RGB image to HSV image
Detection of Red color in the image
Gaussian Blur for image stabilization
Canny Edge Detection for background elimination
Dilate filter for smooth edges of foregraound objects
Hough circle algorithm to detect circles of specified dimension
Store Center and Radius in three dimensional array
Get the current cursor position
Move cursor on the screen by transformation of the position of detected circle's center by a scaling factor
Compare new position of the cursor with previous position and apply stabilization algorithm
Check for mouse event flags
YES
Perform mouse click
NO
NO
Delay 2ms and wait for escape key
YES
Exit

4.2 Image Acquisition

To input of image frames the implemented system applies the built in webcam of the laptop which is currently available in all models of laptop computers. However external webcam works with the system just as fine. The user has to place the external webcam on top of the screen to get the desired experience of freely moving the mouse in the air. The demonstrated syetem acquires images at a resolution of 640 x 480 with a frame rate of 50 fps. However the resolution and frame rate can be altered according to the user or system requirements. The continuous video stream is captured as frames of images which are converted to image matrix of three channel RGB image for storage and image processing.

4.3 Median Blur and Conversion of RGBImage to HSV Image

Smoothing, also called *blurring*, is a simple and frequently used image processing operation [9]. The median filter runs through each pixelin the image matrix and replaces them with the median of their neighbourhood pixels. The neighbourhood pixels are located in square neighbourhood around the evaluated pixel. This smoothen the image and improve its quality by eliminating the pixels with defects of high variation from its neighbouring pixels.

In the stored image matrix the pixel values are kept as BGR rather than RGB. Each pixel is converted to HSV value using the following algorithms [9]:

The *R,G,B* values are divided by 255 to change the range from 0..255 to 0..1:

$R' = R/255$

$G' = G/255$

$B' = B/255$

$Cmax = \max(R', G', B')$

$Cmin = \min(R', G', B')$

$\Delta = Cmax - Cmin$

Hue calculation:

$$H = \begin{cases} 0^\circ & \Delta = 0 \\ 60^\circ \times \left(\frac{G'-B'}{\Delta} mod6\right) & , C_{max} = R' \\ 60^\circ \times \left(\frac{B'-R'}{\Delta} + 2\right) & , C_{max} = G' \\ 60^\circ \times \left(\frac{R'-G'}{\Delta} + 4\right) & , C_{max} = B' \end{cases}$$

Saturation calculation:

$$S = \begin{cases} 0 & , C_{max} = 0 \\ \frac{\Delta}{C_{max}} & , C_{max} \neq 0 \end{cases}$$

Value calculation:

$V = Cmax$

After the conversion the colors of the image stands out which makes it easier to separate the basic color components from the image.

4.3 DETECTION OF RED COLOR FROM THE IMAGE

For detecting red color from the image two shades of red are defined in the program. The inRange function is called to check whether the stored image contains pixels within this range of the shades of red. For every pixel in the stored image matrix "source()" the function compares it with the lower bound ''lb()'' and upper bound "ub()" and stores them in another matrix ''shade1()'' in the following manner:

$$shade1(i) = lb(i) \leq source(i) \leq ub(i)$$

In this way two separate matrix are created which represents the range of the shades of red that can be detected by the program. Then these detected shades of red are joined together to obtain only red objects in the image and filter out the rest of the colors.

4.4 GAUSSIAN BLUR

Probably the most useful filter (although not the fastest). Gaussian filtering is done by convolving each pixel of the Red filtered image with a*Gaussian kernel* [9] and then summing them all to produce the output array.The pixel located in the middle would have the biggest weight. The weight of its neighbors decreases as the spatial distance between them and the center pixel increases.
The color filtered image is corrupted by anamolous pixels which are due to added noise. The Gaussian filter removes those anamolies by smoothing the image.

4.5 CANNY EDGE DETECTION

Canny Edge Detection [10] is a popular edge detection method which is a multi-stage algorithm. The method employes a Noise Reduction stage by a 5x5 Gaussian filter [11], [12].
The second stage involves finding intensity gradient of the image where smoothened image is filtered with a Sobel kernel in both horizontal and vertical direction and obtain first derivative in horizontal and vertical direction. From these two images the edge gradient and direction of each pixel can be found as follows. The gradient direction is always perpendicular to edge.
In the next stage Non-maximum Suppression is applied where a full scan of image is done to remove any unwanted pixels which may not constitute the edge.This is done by checking whether each pixel is a local maximum in its neighborhood in the direction of gradient.

In the final stage Hysteresis Thresholding is done. This stage decides which of the edges are really edges and which are not. For this, we need two threshold values, minVal and maxVal. Any edges with intensity gradient more than maxVal are sure to be edges and those below minVal are sure to be non-edges, so discarded. Those who lie between these two thresholds are classified edges or non-edges based on their connectivity. If they are connected to "sure-edge" pixels, they are considered to be part of edges, otherwise they are not edges.

4.6 DILATION AND HOUGH CIRCLE TRANSFORM

The Hough transform [13] can be used to determine the parameters of a circle when a number of points that fall on the perimeter are known. A circle with radius R and center (a, b) can be described with the parametric equations

$$x = a + R\cos(\theta)$$
$$y = b + R\sin(\theta)$$

When the angle θ sweeps through the full 360 degree range the points (x, y) trace the perimeter of a circle. If an image contains many points, some of which fall on perimeters of circles, then the job of the search program is to find parameter triplets (a, b, R) to describe each circle.

Thus the image after edge detection is dilated to fill the gaps between the detected edges and then Hough circle transform applied to the image to find whether any edge detected is a circle within a specified range of radius.

This step identifies the illuminated marker as a circle and stores its co ordinates on the screen as (a, b, R). Once the illuminated marker is detected, to avoid detection of any further circles is restricted in the program by fixing a large minimum distance between the centers of the detected circles.

4.7 LOCATION OF THE CURSOR ON THE SCREEN

The certer of the detected circle is set as the physical coordinate of the cursor on the screen. The resolution of the captured frame is different from the resolution of the computer display. Since the detected circle center will only be within the window of the captured frame so to move the cursor over the entire computer screen proper scaling of the coordinates is required in both the x-axis and y-axis. The scale factor is obtained as:
scale factor in the x direction = width_com/width_window
scale factor in the y direction = height_com/height_window

where, width_com=width of computer screen
width_window=width of frame
height_com=height of computer screen
height_window=height of frame

The image of the camera is laterally inverted so left direction movement is interpreted as right direction movement and vice versa. To solve this problem image can be flipped first after acquisition from the video stream but flipping every image is very inefficient and costly in terms of processing time.

To solve the laterally inverted movement in the x direction the physical x position of the cursor is set as (width_com – x coordinate of the center of the circle). This simple coordinate geometry can solve the problem efficiently by saving computational time.

5 HARDWARE IMPLEMENTATION

The illuminated marker is made from a red led covered by red tracing paper to diffuse the red light from it.

The mouse button signals are sent by 2.4-GHz Universal Serial Bus Wireless Optical Mouse chipset MC68HC908JB8 and received by the USB RF receiver dongle MC68HC908JB16 [14].

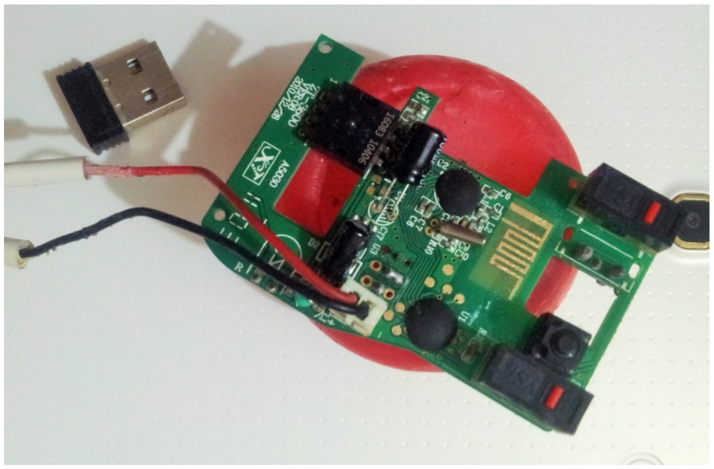


Figure (1): Optical Mouse chipset and USB Receiver dongle

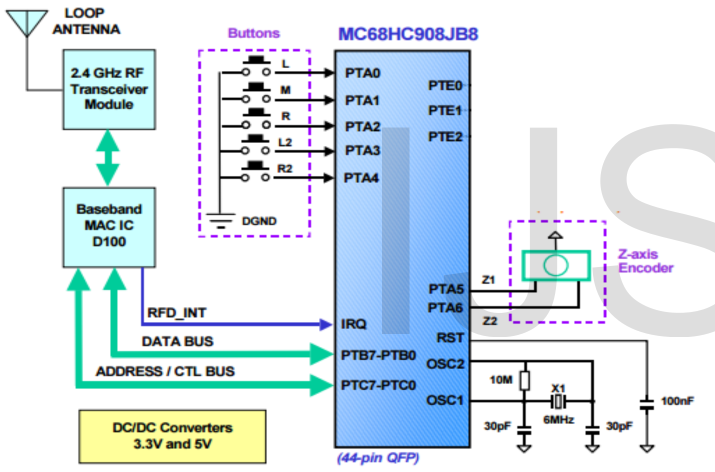


Figure (2): Mouse Block Diagram [14]

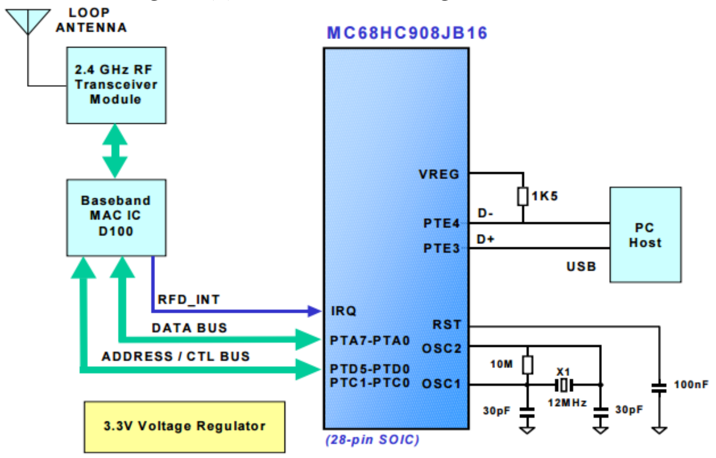


Figure (3): USB Receiver Block Diagram [14]

Both the illuminated marker and the mouse chipset is powered by a 6 V rechargeable lithium ion battery. There are two DC/DC converters; the 3.3-V supply is used for the baseband IC and RF module. The 5-V supply is only used for the MC68HC908JB8.

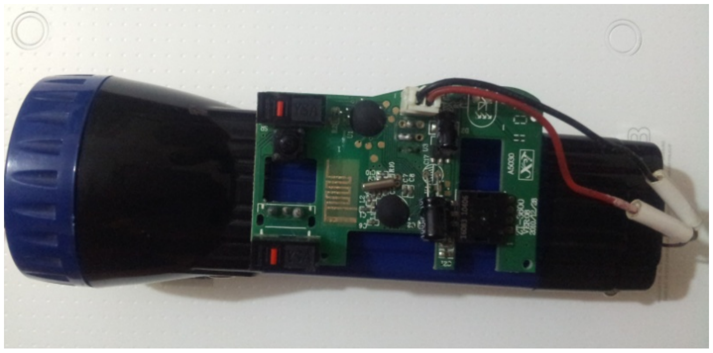


Figure (4): Assembled final prototype

5.1 Power Saving Features

The battery only has to power a small led and other power consumption only occurs when a mouse button is pressed so the battery once recharged can work for three to four weeks under normal operational condition.

6 Results of the Image Processing

The following figures would demonstrate the image processing stages and would show the results and effectiveness of the developed program.

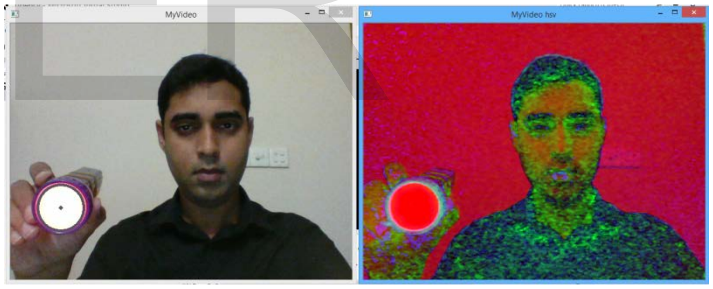


Figure (5): Conversion to HSV image

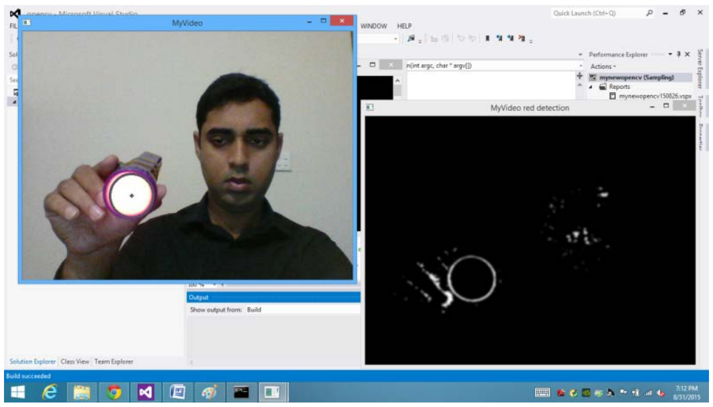


Figure (6): Detection of red color

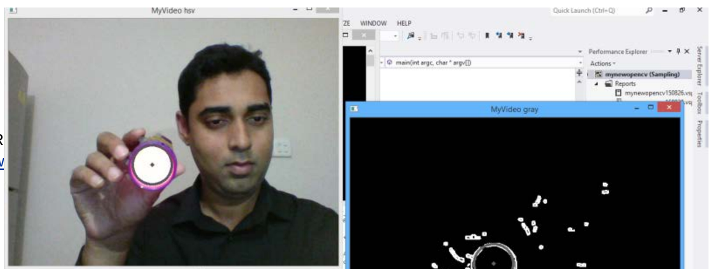


Figure (7): Canny Edge detection

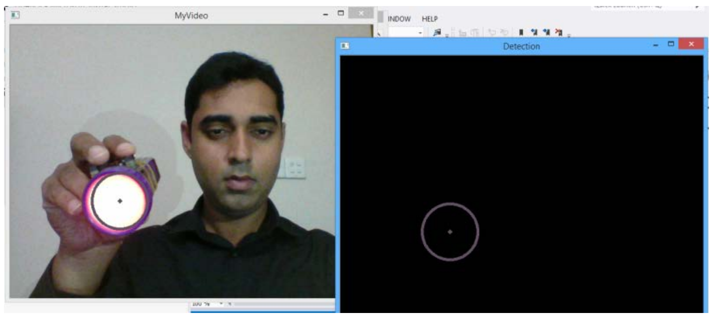


Figure (8): Detection of the circle

6.1 Explanation of the Processing Stages

Conversion of RGB image to HSV image highlights the basic colors as shown in figure (5). Figure (6) clearly demonstrates the detection of the red color from the entire pixel of the image. The effectiveness of canny detection algorithm is shown in figure (7) where the sharp edges of the foreground are identified. As the image can have many red colored objects in it so Hough circle transform algorithm only detects the presence of the circular red marker on the screen in Figure (8).

6.2 Robustness of the Detection Stage

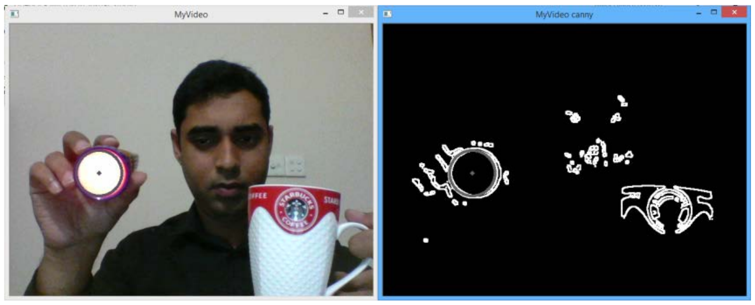


Figure (9): Robustness against foreground objects

Any red item placed in the foreground does not affect the outcome of the detection stage. This fact is demonstrated by the red colored mug being in the image in figure (9) but the marker is still detected as shown by the grey circle.

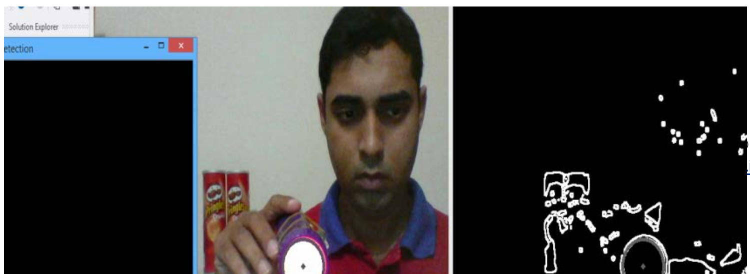


Figure (10): Robust detection in any background condition

The image in figure (10) proves the robustness of the object tracking algorithm against a harsh condition. The presence of many red colored objects in the image is detected by the color and edge dectector as shown on figure (10) right. The shape detection and background object filtering algorithm effectively detects the illuminated marker, as shown on figure (10) right.

7 Conclusion

The developed prototype mouse can effectively perform the general functions of a mouse with precision. As the mouse click events are sent via RF signal so there is no delay of clicking action of the developed mouse and any commercially available mouse. Although the prototype is larger in shape but the actual model of the mouse can be made into a pen shape in future. This mouse has the potential to have great impact in PC based gaming experience. It is also extremely useful in education, like teaching from sides and giving presentation. The design is very cost effective and has more functionality than normal mouse. Moreover, it is more precise and faster than the hand jesture recognition based cursor movement algorithm. The C++ code of the mouse is converted to a driver software and by installation of this driver in any PC with a webcam can easily use the implemented mouse.

8 Acknowledgment

I would like to thank my younger brother Kazi Waseef Mohammad for his efforts in coding. His support played a vital role in completion of this project.

References

[1] A. Erdem, E. Yardimci, Y. Atalay, V. Cetin, A. E, "Computer vision based mouse, Acoustics, Speech, and Signal Processing," 2002. Proceedings. (ICASS). IEEE International Conference.

[2] Hojoon Park, "A Method for Controlling Mouse Movement using a Real-Time Camera,"www.cs.brown.edu/research/pubs/theses /masters/2010/park.pdf2010.

[3] ShanyJophin, Sheethal M.S, Priya Philip, T M Bhruguram,"Gesture Based Interface Using Motion and Image Comparison," International Journal of Advanced Information Technology (IJAIT) Vol. 2, No.3, June 2012

[4] Chu-Feng Lien, *Portable Vision-Based HCI - A Real-time Hand Mouse System on Handheld Devices*.

[5] Bhavana I V "Virtual Mouse Based on Colour Pointer Tracking," *International Journal of Science and Research (IJSR) ISSN (Online): 2319-7064 Impact Factor (2012): 3.358*

[6] Abhik Banerjee, AbhirupGhosh, KoustuvmoniBharadwaj, HemantaSaikia, "Mouse Control using a Web Camera based on Colour Detection," *International Journal of Computer Trends and Technology (IJCTT) – volume 9 number 1– Mar 2014*

[7] AnkurYadav, AnandPandey, Avinash Singh, Ajit Kr Singh, "Computer Mouse Implementation using Object Detection and Image Processing," *International Journal of Computer Applications (0975 – 8887) Volume 69– No.21, May 2013*

[8] Akhil Gupta, AkashRathi, Dr. Y. Radhika, "HANDS-FREE PC CONTROL" CONTROLLING OF MOUSE CURSOR USING EYE MOVEMENT EYE," *International Journal of Scientific and Research Publications, Volume 2, Issue 4, April 2012 1 ISSN 2250-3153*

[9] Richard Szeliski, "Computer Vision: Algorithms and Applications© 2010," *Microsoft ResearchElectronicdraft:September 3, 2010 pages 111-127*

[10] Canny, John, "A Computational Approach to Edge Detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on (Volume:PAMI-8 , Issue: 6)*

[11] Bao, P., "Canny edge detection enhancement by scale multiplication," *Pattern Analysis and Machine Intelligence, IEEE Transactions on (Volume:27 , Issue: 9)*

[12] Bing Wang, "An Improved CANNY Edge Detection Algorithm," *Computer Science and Engineering, 2009. WCSE '09. Second International Workshop on (Volume:1)*

[13] Harvey Rhody, "Hough circle theorem," *Chester F. Carlson Center for Imaging Science Rochester Institute of Technology, rhody@cis.rit.edu October 11, 2005*

[14] Dennis Lui Ernest, Chan W.S. Wong, "2.4-GHz Wireless Optical Mouse and Multimedia Keyboard Solution Designer Reference Manual — Rev 0," *http://www.freescale.com/pages/2.4ghz-wireless-optical-mouse-and-multimedia-keyboard-reference-design:RD68HC908WOMK2,2004-2015*