International Journal of Internet and Distributed Systems, 2023, 5, 1-17
https://www.scirp.org/journal/ijids
ISSN Online: 2327-7165
ISSN Print: 2327-7157

A Study on the Development of Touchless Mouse by Object Tracking through Webcam

Mohammad Alauddin[1], Shamim Ahmed[2], Rokshana Akter[2], Md. Amanat Ullah[1], Md. Abdul Mannan[2*]

[1]Department of Computer Science & Engineering, Uttara University, Dhaka, Bangladesh
[2]Department of Mathematics, Uttara University, Dhaka, Bangladesh
Email: alauddinsorwer@yahoo.com, shamim99990000@gmail.com, kssroksana@gmail.com, *mannan.iu31@gmail.com

How to cite this paper: Alauddin, M., Ahmed, S., Akter, R., Ullah, Md.A. and Mannan, Md.A. (2023) A Study on the Development of Touchless Mouse by Object Tracking through Webcam. *International Journal of Internet and Distributed Systems*, **5**, 1-17.
https://doi.org/10.4236/ijids.2023.51001

Received: November 6, 2022
Accepted: February 25, 2023
Published: February 28, 2023

Abstract

In this paper, we study the continuous growth of computer technology and the increasing importance of human-computer interaction. Interactive touch-less is now an undergoing developing technology in real life. Touchless technology introduces a new way of interacting with computers by object tracking method. Nowadays most mobile devices are using touchscreen technology. However, this technology is still not cheap enough to be used in desktop systems. Designing a touchless device such as a mouse or keyboard using a webcam and computer vision techniques can be an alternative way of touch screen technology. Recent trends in technology aim to build highly interactive and easy-to-use applications as a replacement for conventional devices. Such a device is touchless mouse. Its development is completed on the MATLAB platform. The overall objective is to apply image processing techniques from video to track the movement of color which is captured by a webcam and that is converted into mouse movements and operations to control the system. The sub-system which is implemented here would allow a person to control his/her mouse without any input other than the marker movements. We use three fingers (with color) as three color markers (red, green, blue) for completing the activities of a mouse. My first goal was to successfully track the marker color and the second goal was to track the marker position from the acquired image frame for performing the mouse operations. The webcam is used to capture the information on the marker and trigger the associated actions. I use java.awt. Robot file for performing the mouse operations using the acquired data from the image frame.

Keywords

MATLAB, GUI, RGB, DIP, Matrix

1. Introduction

This paper aims to treat a review of existing knowledge of image processing which is relevant to this project. It begins a review of existing image enhancement methods designed to prove image enhancement in general. These include specialized image acquisition method, image conversions and more about digital image processing methods. From the theory and literature review, we can conclude that digital image processing allows for a more flexible and practical method of improving image quality. The following sections briefly covered the various data classes, how to perform image acquisition, aspects of image processing and image representation in MATLAB ([1] [2]). As time is proceeding ahead, technology is improving and evolving every single moment. No one can claim something to be the latest because we can see the presence of something newer and better in front of our own eyes. Two of the basic fundamental intentions of technology are to make things that are not complicated to be understood by the user and make the user's work more convenient. Things are simple when the interface between humans and technology is least complex. A mouse is typically used with a computer to control the motion of a cursor in a graphical user interface (GUI) ([3] [4]). There are some main operations that a mouse can perform. Firstly, pointing-and-clicking can select files, programs or actions from a list of menus, or through icons. It can also trigger the floating menu which only appears by right clicking. More particularly in clicking, a user can right click, single click (left click), and double click etc. Secondly, drag-and-drop can move the object by holding the mouse button down while moving the cursor to a different location. In this view, my work proposes a touchless finger mouse, where the users are able to perform mouse operations simply by showing their finger with color in front of a webcam. Touch-less technology helps people, who work in production plants, engineering sectors, research centers, and especially those disabled ones who don't have to touch on any kind of hardware. Existing Touch-less technologies are developed on measuring temperatures, sensors, flash scanning of fingerprints and finger gestures via image processing method. However, touch screens cannot be applied to desktop systems because of cost and other hardware limitations. This system will complement the touch screen technology, *i.e.*, [5] [6].

- The main objective of the project is to develop a touchless mouse, which allows human to control their system easily without any conventional mouse.
- Since the work is developed on the MATLAB platform, one of the goals of the project is to analyze and implement the various results of the image acquisition and processing toolbox in MATLAB.
- Performing Color code Calibration and mouse interactivity on the system.
- To analyze the vast and various possibilities and functionalities of Image Processing using MATLAB.

2. About the Touchless Mouse

A touchless mouse is software that allows users to give mouse inputs to a system

without using an actual mouse. To the extreme it can also be called as hardware because it can be used replacing traditional mouse. A touchless mouse can usually be operated with multiple input colors, which may include all actual mouse operation. Touch fewer mice which use webcam and works with the help of different image processing techniques. A webcam is set to take images continuously. The user must have a particular color in his finger so that when the web camera takes image it must be visible in the image. This color is detected from the image pixel and the pixel position is mapped into mouse input. Depending upon the size of the image taken by camera, various scaling techniques are used because the pixel position in the image will not have a correspondence with screen resolution.

2.1. Digital Image

An image may be defined as a two-dimensional function, f(x, y), where x and y are spatial plane coordinates, and the amplitude of f at any pair of coordinates (x, y) is called the intensity or gray level of the image at that point. When x, y, and the amplitude values of f are all finite, discrete quantities, we call the image a digital image in Figure 1.

2.2. Pixel

A pixel or picture element is a smallest individual part of a graphic image. Graphics monitors display pictures by dividing the display screen into thousands (or millions) of pixels arranged in rows and columns. The pixels are so close together that they appear connected. The quality of a display system largely depends on its resolution, how many pixels it can display, and how many bits are used to represent each pixel.

2.3. Image Processing and Analysis

Image processing is a form of signal processing in which the input is an image, such as a photograph or video frame. The output of image processing may be either an image or a set of characteristics or parameters related to the image. Most image-processing techniques involve treating the image as a two-dimensional

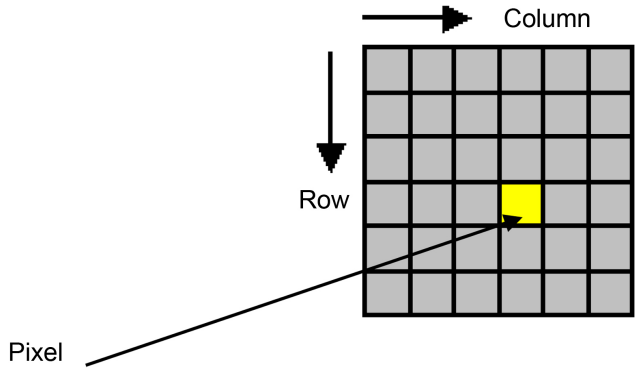

Figure 1. A digital image.

signal and applying standard signal-processing techniques to it. Image analysis is the extraction of meaningful information from images; mainly from digital images by means of digital image processing techniques. Then each pixel is retrieved from the image and extracts the red, green and blue values (RGB) from each pixel. Now we can easily detect a particular color since all the colors are combinations of RGB values. Here we just try to detect only Red Green and Blue colors.

2.4. Digital Image Processing

Digital Image Processing means processing of digital image on digital hardware usually on computer. More simply, Digital image processing encompasses processes whose inputs and outputs are images and, in addition, encompasses processes that extract attributes from images, up to and including the recognition of individual objects. Digital image processing is the use of computer to perform image processing on digital images. As a subcategory or field of digital signal processing, digital image processing has many advantages over analog image processing. It allows a much wider range of algorithms to be applied to the input data and can avoid problems such as the build-up of noise and signal distortion during processing. A process flow diagram has been included to have a better understanding of the multiple steps in digital image processing (see Figure 2).

The fundamental steps are described briefly in the following:

1) ***Image Acquisition***

This is the first step or process of the fundamental steps of digital image processing. To acquire a digital image we require an imaging sensor and the capability to digitize the signal produced by the sensor. The available Image acquisition toolbox in MATLAB helps to acquire images from a real live video.

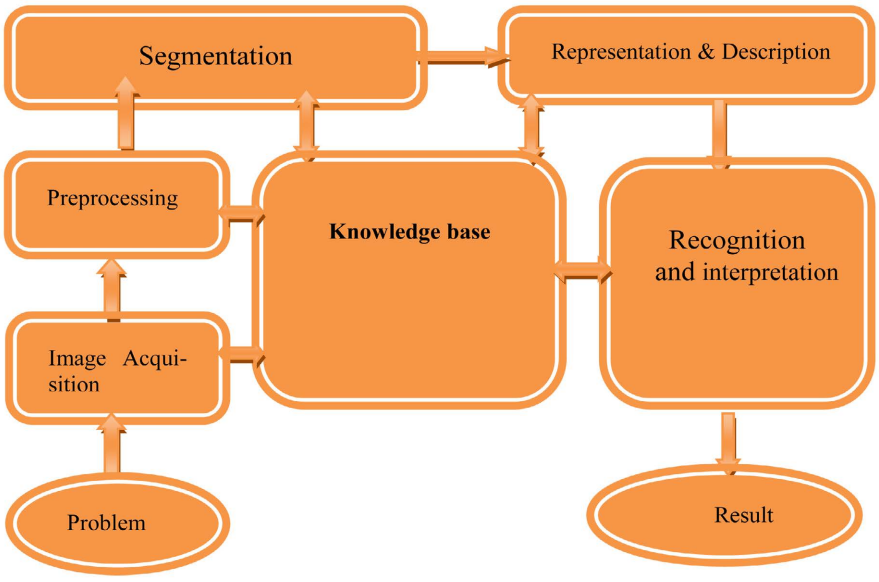

Figure 2. Fundamental steps of image processing.

2) *Image Preprocessing and enhancement*

The processes consist of a collection of techniques that seek to improve the visual appearance of an image or to convert the image to a form better suited for analysis by a human or a machine. The common image enhancement techniques:

a) *Contrast stretching*

b) *Noise smoothing*

c) *Low-pass*, *High pass Filtering etc.*

3) *Segmentation* [7]

In general, autonomous segmentation is one of the most difficult tasks in digital image processing. A rugged segmentation procedure brings the process a long way toward successful solution of imaging problems that require objects to be identified individually. It is one of the most difficult tasks in DIP [8]. Segmentation kinds:

a) *Autonomous Segmentation*

b) *Rugged Segmentation* (*long process to get successful solution*)

c) *Erratic Segmentation*

4) *Representation and Description*

Representation and description almost always follow the output of a segmentation stage, which usually is raw pixel data, constituting either the boundary of a region or all the points in the region itself. That means representation makes a decision whether the data should be represented as a boundary or as a complete region.

5) *Image recognition and interpretation*

Image recognition is to assign a label to an object based on the information provided by its descriptors. Interpretation involves assign meaning to an ensemble of recognized objects.

2.5. Components of a Digital Image Processing System

1) Physical devices that are sensitive to the energy radiated by the object (see Figure 3).

2) Digitizer that converts the output of the physical sensing device into digital form (see Figure 4).

2.6. Pixel

A pixel p at coordinates (x, y) has four horizontal and vertical neighbors whose coordinates are given by (x + 1, y), (x − 1, y), (x, y + 1), (x, y − 1). This set of

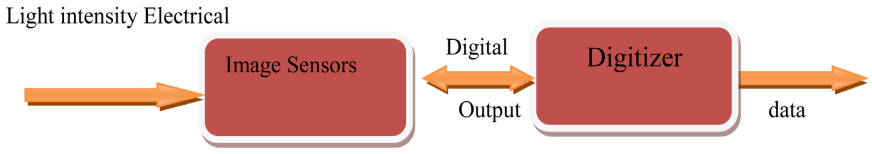

Figure 3. Digital video camera.

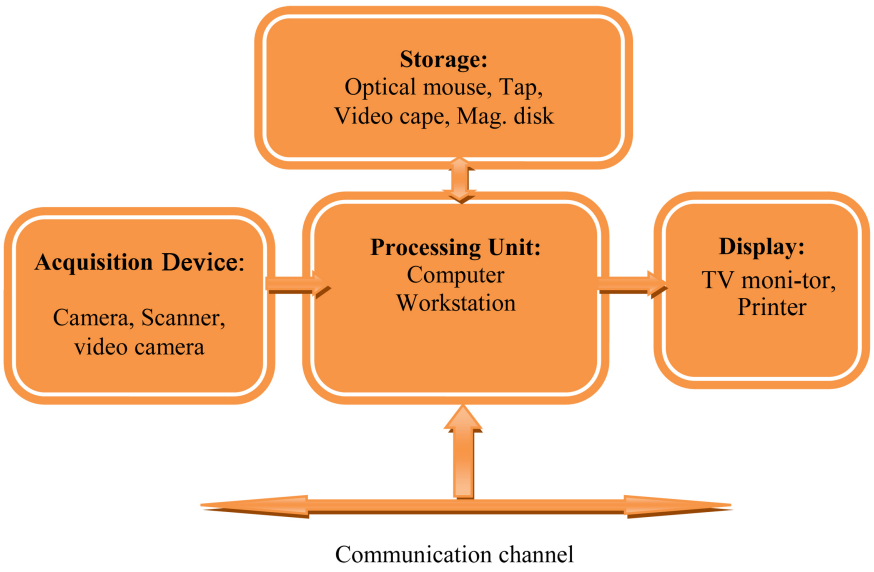

Figure 4. Basic element of digital image processing system.

pixels, called the 4-neighbors of p, is denoted by $N_4(p)$. Each pixel is a unit distance from (x, y), and some of the neighbors of p lie outside the digital image if (x, y) is on the border of the image. The four diagonal neighbors of p have coordinates (x + 1, y + 1), (x + 1, y − 1), (x − 1, y + 1), (x − 1, y − 1).

And are denoted by ND(p). These points, together with the 4-neighbors, are called the 8-neighborsof p, denoted by $N_8(p)$. As before, some of the points in ND(p) and $N_8(p)$ fall outside the image if (x, y) is on the border of the image, *i.e.* [9] (see **Figure 5**).

2.7. Adjacency, Connectivity, Regions, and Boundaries

Connectivity between pixels is a fundamental concept that simplifies the definition of numerous digital image concepts, such as regions and boundaries. To establish two pixels are connected, it must be determined if they are neighbors and if their gray levels satisfy a specified criterion of similarity (say, if their gray levels are equal). For instance, in a binary image with values 0 and 1, two pixels may be 4-neighbors, but they are said to be connected only if they have the same value.

Let V be the set of gray-level values used to define adjacency. In a binary image, V = {1} if we are referring to adjacency of pixels with value 1. In a grayscale image, the idea is the same, but set V typically contains more elements. For example, in the adjacency of pixels with a range of possible gray-level values 0 to 255, set V could be any subset of these 256 values. We consider three types of adjacency:

1) *4-adjacency. Two pixels p and q with values from V are* 4*-adjacent if q is in the set* $N_4(p)$

2) 8*-adjacency. Two pixels p and q with values from V are* 8*-adjacent if q is in the set* $N_8(p)$

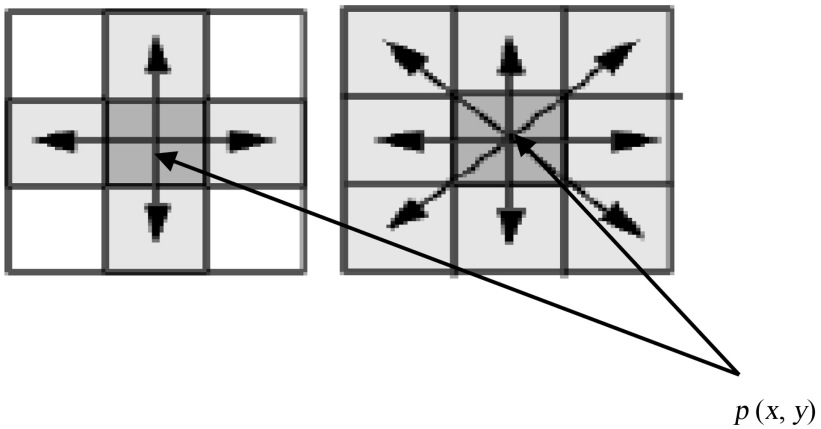

Figure 5. The border of the image.

3) *m-adjacency* (*mixed adjacency*). *Two pixels p and q with values from V are m-adjacent if* (*i*) *q is in* $N_4(p)$, *or*

4) *q is in ND*(*p*) *and the set has no pixels whose values* $N_4(p) \cap N_4(q)$.

3. The Fundamental Principles of Color

Color or colour is the visual perceptual property corresponding in humans to the categories called red, blue, yellow, green and others. It is the aspect of things that is caused by differing qualities of light being reflected or emitted by them. To see color, you have to light. When light shines on an object some colors bounce off the object and others are absorbed by it. Our eyes only see the colors that are bounced off or reflected.

3.1. Color Fundamentals

The following terms are used to define color light:

1) ***Brightness or Luminance***: *This is the amount of light received by the eye regardless of color.*

2) ***Hue***: *This is the predominant spectral color in the light.*

3) ***Saturation***: *This indicates the spectral purity of the color in the light.*

3.2. Color Model

A color model is an abstract mathematical model describing the way in which colors can be represented as tuples of numbers, typically as three or four values or color components. When this model is associated with a precise description of how the components are to be interpreted (viewing conditions, etc.), the resulting set of colors is called color space. A color model is simply a way to define color. A model describes how color will appear on the computer screen or on paper. Popular color models are:

- *RGB* (*Red*, *Green*, *Blue*)
- *CMY* (*Cyan*, *Magenta*, *Yellow*)
- *HSI* (*Hue*, *Saturation*, *and Intensity*)

3.3. RGB & CMY Model

The color subspace of interest is a cube shown in Figure 6. Models in which RGB values are at three corners; cyan, magenta, and yellow are the three other corners, black is at their origin; and white is at the corner farthest from the origin. The gray scale extends from black to white along the diagonal joining these two points. The colors are the points on or inside cube defined by vectors extending from the origin.

Thus, images in the RGB color model consist of three independent image planes, one for each primary color. The importance of the RGB color model is that it relates very closely to the way that the human eye perceives color. RGB is a basic color model for computer graphics because color displays use red, green, and blue to create the desired color. Therefore, the choice of the RGB color space simplifies the architecture and design of the system. Besides, a system that is designed using the RGB color space can take advantage of a large number of existing software routines, because this color space has been around for a number of years. In the CMY color model gray scale extends from white to black along the line joining these two points and color are points on or inside the cube defined by vectors extending from origin. All values of CMY are assumed to be in the range [0, 1]. Most devices perform an RGB to CMY [10] conversion internally. This conversion is performed using the simple operation

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1-R \\ 1-G \\ 1-B \end{bmatrix}$$

3.4. HSI Color Model

HSI color space is an attractive Model in image processing applications since it represents colors same as of how human eye sense. This Model comes from the concept of hue, saturation and intensity. The color components of HSI model are defined with respect to color triangle as shown in Figure 7.

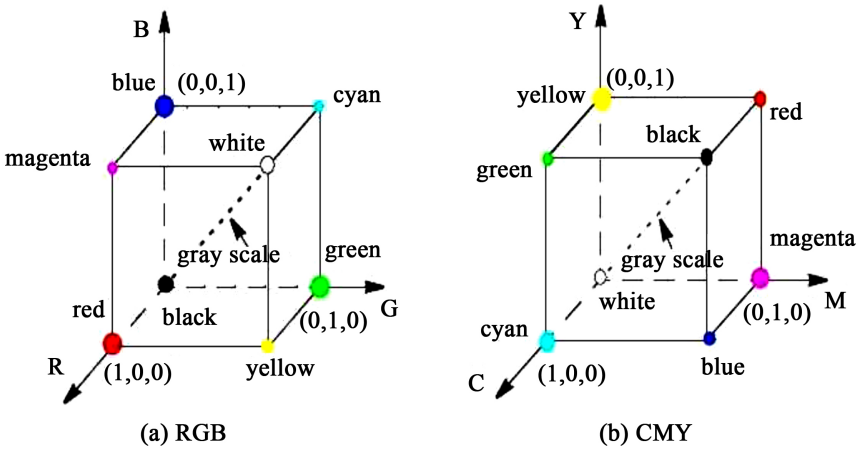

Figure 6. RGB and CMY color model.

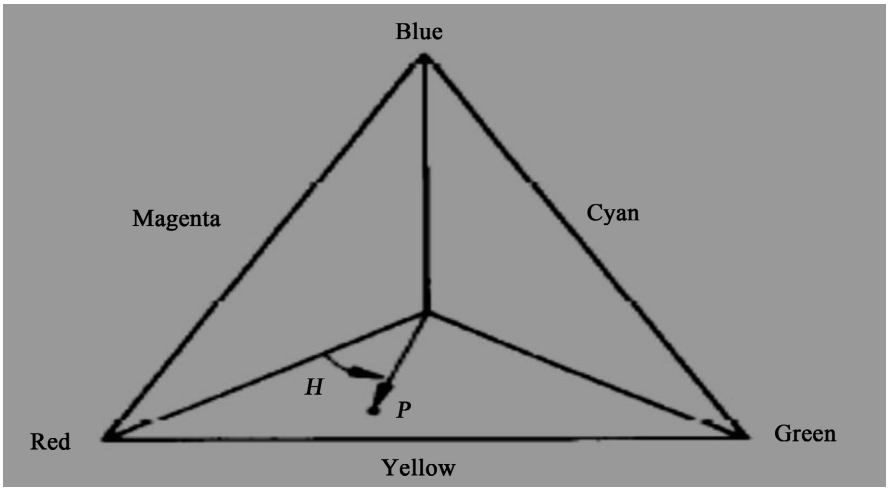

Figure 7. HIS color model.

In Figure 7, hue H of a color point P is the angle of the vector shown with respect to the red axis. Thus, when H = 0^0, the color is red, when H = 60^0, the color is yellow and so on. The saturation S of a color point P is the degree which the color point undiluted by white and proportional to the distance from p to the center of the triangle. The intensity I is measured with respect to a line perpendicular to the triangle and passing through its color.

Color of the HSI model are defined with respect to normalized red, green and blues given in term of RGB primaries by

$$r = \frac{R}{R+G+B}, g = \frac{G}{R+G+B}, b = \frac{B}{R+G+B}$$

The intensity component in the HSI model is defined as:

$$I = \frac{R+G+B}{3}$$

4. Design and Methodology

This chapter aim is to discuss main design and methodology which include processing the real time images captured by a Webcam for Color Recognition and various mouse operations using MATLAB programming. In the touchless application, one of the major problems is color detection. A color finger has been used to make the color detection easy and fast. To simulate the click events of the mouse three fingers with three colors have been used.

4.1. Requirements

- *Webcam*
- *Red, Green and Blue color marker*
- *Windows XP/7/vista installed computer*

4.2. Research Process: The Basic Algorithm Is as Follows

Step-1: Webcam Initialization

Step-2: Acquiring image from the webcam

Step-3: Get the grey image of the RGB frame

Step-4: Required Color Extraction

Step-5: Filtering the Noise

Step-6: Acquiring Binary Image using proper threshold value.

Step-7: Define the region and the center of the pointer

Step-8: Window Screen Synchronization

Step-9: Move the cursor according to the position of the center of the pointer

Step-10: Perform the single, double, left and the right click, drugging of the mouse

4.3. Program Flowchart (Figure 8)

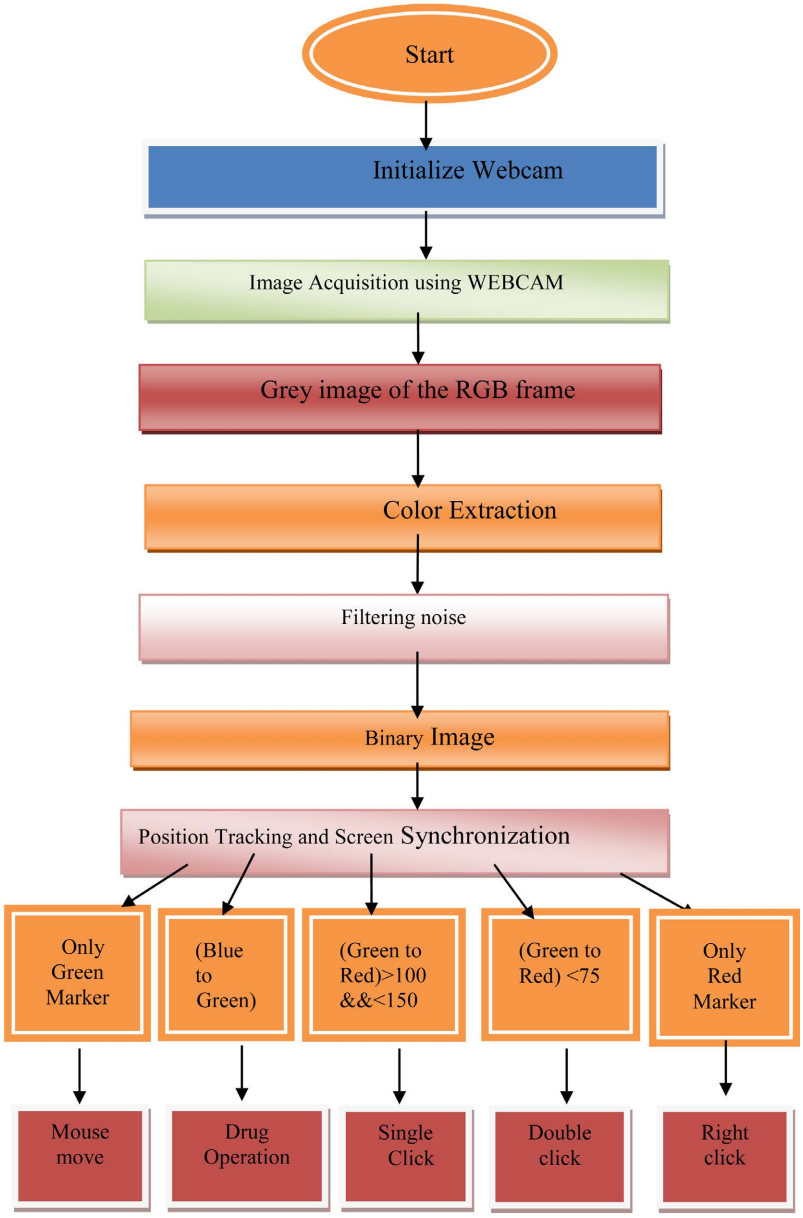

Figure 8. Program flowchart for webcam and color.

4.3.1. Webcam Initialization

Webcam Initialization is performed differently on various PC's or Laptops. It has to be configured based on the Webcam configuration in each system. To retrieve the properties of the webcam configuration in each system, type Matlab code, imaqhwinfo. It returns a structure that contains information about the image acquisition adaptors available on the system. An adaptor is the interface between MATLAB and the image acquisition devices connected to the system. The adaptor's main purpose is to pass information between MATLAB and an image acquisition device via its driver.

4.3.2. Acquiring Image from the Webcam in MATLAB

In my project I acquired image using webcam. In MATLAB, to recognize the video camera I have created an object using the command vid = videoinput ("winvideo"). Then MATLAB automatically find the webcam connected to the computer. Once it is an object in the workspace we can edit its settings, such as the number of frames per Trigger, Returned Color space and frame grab interval to optimize our project. We have to start the recording of the video using the command start(vid). The camera will be triggered and collects frame specified by the frame per trigger property. The data from the image frame is acquired using the getsnapshot() command. Each image frame is composed of an array of M × N pixels (concentration of "picture element") with M rows and N columns of pixels. Each pixel contains a certain value for red, green and blue (see **Figure 9**).

4.3.3. Getting the Grey Image from the RGB Frame [11]

Conversion is performed from obtained RGB true color image into grayscale intensity image. The function rgb2gray (RGB) is used to convert RGB images to grayscale by eliminating the hue and saturation information while retaining the luminance (see **Figure 10**).

4.3.4. Color Extraction

Color extraction is a process of extracting color from an image. The RGB format is a practical method to represent color images. Matlab creates three matrices (or three M × N arrays) with each matrix represents normalized component of red, green or blue to read and store each of the frames of the video (see **Figure 11**).

To detect color of the pointer, MATLAB's built in "imsubtract" function has been used. Imsubtract function can be used as,

$$Z = \text{imsubtract}(X, Y),$$

where, it subtracts each element in array Y from the corresponding element in array X and returns the difference in the corresponding element of the output array Z. For identifying the red color we subtract the grayscale image from its RGB image (:, :, 1). Similarly we can identify the green and blue color.

4.3.5. Filtering Noise

After detecting the required color in the input image, a median filter has been used to filter out the noise. Median filtering is a nonlinear operation often used

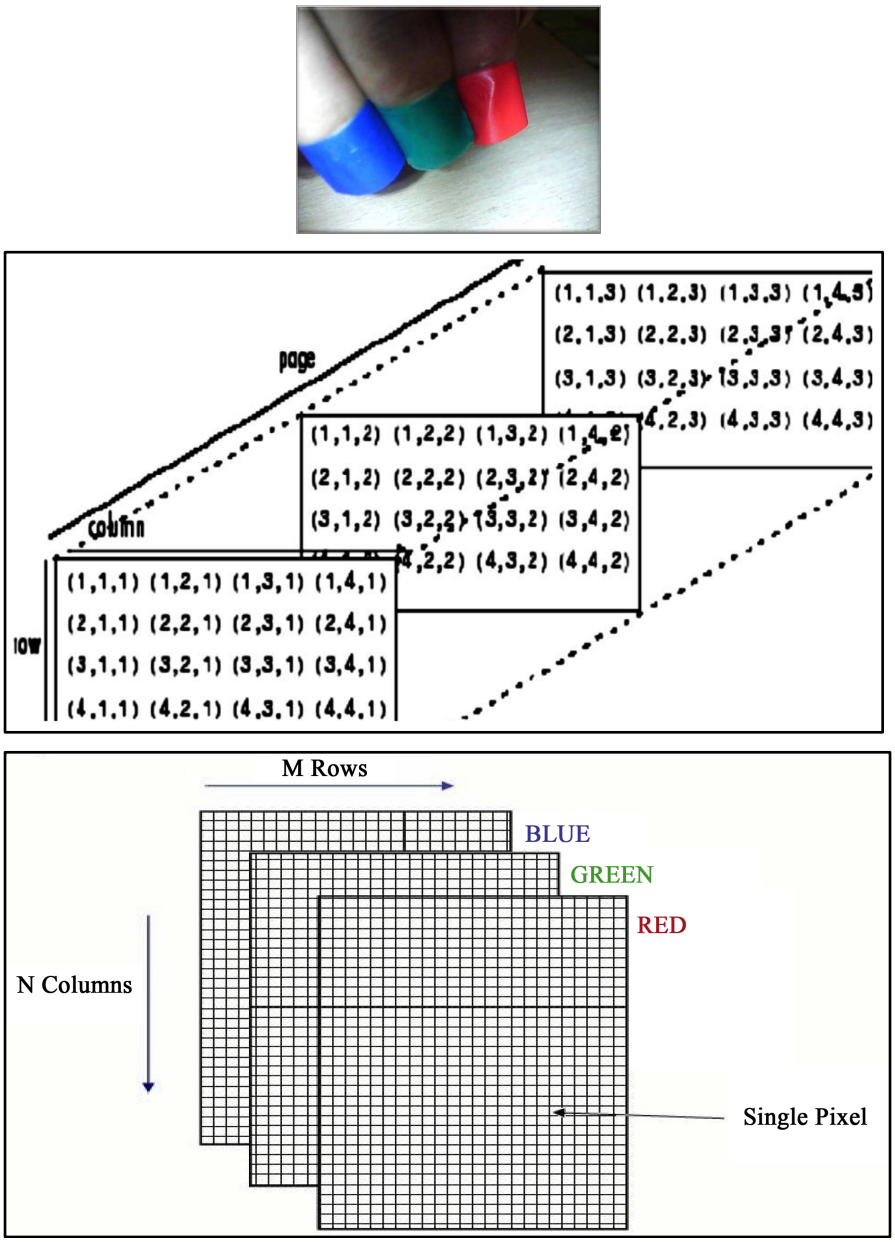

Figure 9. Image and corresponding 3D matrix and layer in MATLAB.

Figure 10. Conversion of RGB frame to grayscale image.

Figure 11. Conversion of gray to red component.

in image processing to reduce "salt and pepper" noise. A median filter is more effective than convolution when the goal is to simultaneously reduce noise and preserve edges. As (see **Figure 12**) really there is no noise as the illumination was enough when we were getting this snapshot sample.

4.3.6. Gray Scale Image to Binary Image Conversion

Gray Scale Image to Binary Image conversion will be performed after using threshold level and that level can be used to convert an intensity or RGB image into a binary image using code, im2bw(im,th), where im and th are gray scale image and threshold value respectively. A normalized intensity value that lies in the range of [0, 1] and threshold value is used in my project without any method. In our study, the threshold 0.15 gave the best result for the large range of illumination change. The threshold black and white pixels interclass variance will be reduced by choosing the threshold level. Matlab code "im2bw" converts an RGB image into binary image through the two steps. First it converts the image into a grayscale format and then converts the grayscale image into a binary format using that threshold level. Below (see **Figure 13**) shows HSI to binary image conversion.

4.3.7. Position Tracking

In my project, for position tracking, I have used MATLAB function "regionprops". Using this function we collect the centroid value of the position of the detected color marker in the acquired image frame. We use this data for performing mouse operations. From the difference of the centroid value of the color marker in the consecutive acquired image frame we have to detect the motion of the marker movement (see **Figure 14**).

4.3.8. Window Screen Synchronization

Based on the co-ordinates of the location of the identified color which is calibrated earlier, the window screen mouse co-ordinates (X, Y) are to be mapped accordingly. This is done by co-relating the Matlab video frame co-ordinates to Window screen co-ordinates mathematically. This centroid point is mapped to the computer screen by calculating the ratio of the screen resolution to the

Figure 12. Before and after filtering the image using median filter.

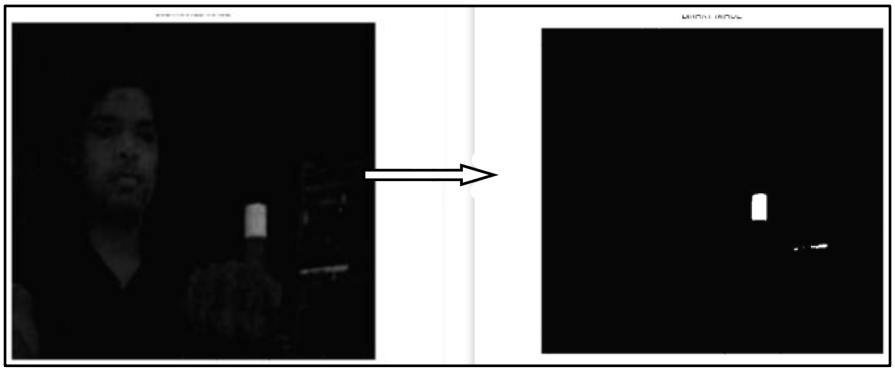

Figure 13. Gray scale image to binary image conversion.

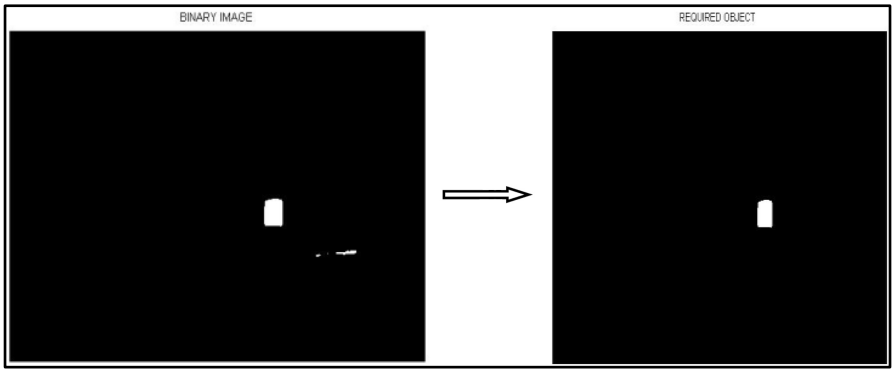

Figure 14. Binary to required object conversion.

webcam resolution. We need to subtract the coordinate of the centroid point in relative to the total resolution to overcome the mirror effects. As we are facing a webcam, it is just like we are facing a mirror. Hence, when we are moving to the right in our plane, the mirror is actually moving in the reverse direction in its own plane. The mapping calculation is given in Equation (4.1)

$$X = sx - (sx/cx) \times x_1,\ Y = (sy/cy) \times y_1; \tag{4.1}$$

where X and Y are the cursor point on computer screen, cx and cy are the resolution of webcam, sx and sy are the resolution of computer screen. Below Matlab

codes perform the window screen synchronisation to move the mouse pointer based on this co-relation:

```
new_size=get(0,'ScreenSize');
  sx=new_size(3); sy=new_size(4);
camera_size=get(vid,'VideoResolution');
  cx = camera_size(1); cy = camera_size(2);
  rx=sx/cx; ry=sy/cy;
jRobot.mouseMove(sx-rx*x1,ry*y1);
```

4.3.9. Mouse Operations

In my project for mouse operation we have imported a java file in MATLAB. For mouse cursor movement the centroid value is used for the green color marker. The value of the centroid is used as parameter in jRobot.mousemove. As the value of the centroid of the green color marker changed, then the cursor position is also changed according to the present centroid value of the color. It then selects its position on the monitor screen. In this way we can move the mouse cursor on the screen. For performing the single click, we take the distance of the centroid value of red and green marker. When the distance between this two centroids is within specified range then we will able to perform single click using imported java file. When the difference between this two centroid is less than a specified value then we will be able to perform double click. Only red color is used for right click and green color is used for cursor movement in the same way. In order to perform the drug operation, we take the distance of the centroid value of green and blue marker. When the difference between this two centroid is less than a specified value then we will able to perform drug operation using imported java file [12] (see **Figure 15**).

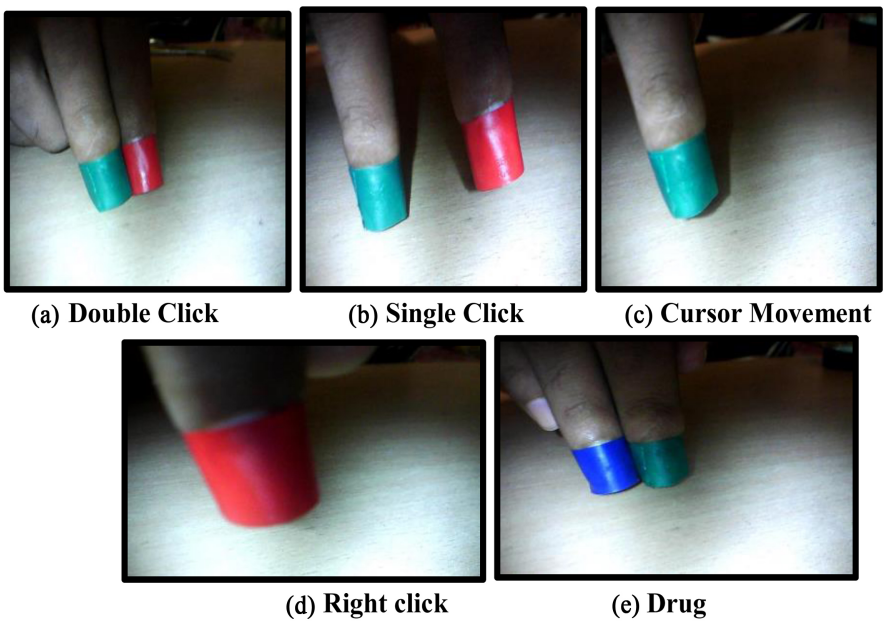

Figure 15. Various operation of touchless mouse according to their color and distance.

5. Result and Discussion

After conducting a user acceptance survey targeting 20 students and 20 academicians in our university, it shows that 75% of them are satisfied with the overall performance of the system. 70% of them feel comfortable with changing the colour using finger marker. 62.5% of them are willing to substitute the mouse with the system instead.

5.1. Limitations

- *Due to brightness and contrast sometimes webcam can hardly detect the expected color.*
- *If the color object moves fast, then webcam cannot detect the expected color.*
- *If the background color and the object color become similar than webcam detects unexpected pixels.*
- *Highly configured pc is needed to get expected result.*

5.2. Future Plane

- *Fast and smooth movement of mouse.*
- *Overcome the brightness and contrast effect.*
- *Add facilities to use as application software.*
- *Operate mouse with finger without any color.*

6. Conclusion

In this study, mouse application has been developed and implemented using a webcam. The proposed prototype can control a lot of operations as a normal mouse can perform. It is also affordable to almost everyone since it only requires a simple webcam and color marker. Overall, we are happy with the results of my project and also glad. All the suggestions forwarded during the software proposal have been successfully completed and the final threshold of application has been crossed. We always enjoy learning a new programming language and being able to design a complex program with it. Lastly, we were pleased with the results of my project, though there are some problems that are not major problems at this point and my code works efficiently and effectively. It is believed that more enhancements could be done in the future and instead of a color pointer directly finger can be detected.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Katona, J. (2021) A Review of Human-Computer Interaction and Virtual Reality Research Fields in Cognitive InfoCommunications. *Applied Sciences*, **11**, Article 2646. https://doi.org/10.3390/app11062646

[2] Gonzalez, R.C. and Woods, R.E. (2018) Digital Image Processing. Pearson, London.

[3] Vonach, E., Gerstweiler, G. and Kaufmann, H. (2014) ACTO: A Modular Actuated Tangible User Interface Object. *Proceedings of the Ninth ACM International Conference on Interactive Tabletops and Surfaces*, November 2014, 259-268. https://doi.org/10.1145/2669485.2669522

[4] Processing Graphics Librarly. http://processing.org/reference/libraries

[5] Zhu, X.X., Riener, A. and Hausen, D. (2018) Gesture-Based Interaction for Touchless Automotive HMI.

[6] Prajapati, R., Pandey, V., Jamindar, N., Yadav, N. and Phadnis, P.N. (2018) Hand Gesture Recognition and Voice Conversion for Deaf and Dumb. *International Research Journal of Engineering and Technology*, **5**, 1373-1376.

[7] Sood, A. and Mishra, A. (2016) AAWAAZ: A Communication System for Deaf and Dumb. 2016 5*th International Conference on Reliability*, *Infocom Technologies and Optimization* (*Trends and Future Directions*) (*ICRITO*), Noida, 7-9 September 2016, 620-624. https://doi.org/10.1109/ICRITO.2016.7785029

[8] Harish, N. and Poonguzhali, S. (2015) Design and Development of Hand Gesture Recognition System for Speech Impaired People. 2015 *International Conference on Industrial Instrumentation and Control* (*ICIC*), Pune, 28-30 May 2015, 1129-1133.

[9] Shangeetha, R.K., Valliammai, V. and Padmavathi, S. (2012) Computer Vision Based Approach for Indian Sign Language Character Recognition. 2012 *International Conference on Machine Vision and Image Processing* (*MVIP*), Coimbatore, 14-15 December 2012, 181-184. https://doi.org/10.1109/MVIP.2012.6428790

[10] Tripathi, K. and Nandi, N.B.G.C. (2015) Continuous Indian Sign Language Gesture Recognition and Sentence. *Procedia Computer Science*, **54**, 523-531. https://doi.org/10.1016/j.procs.2015.06.060

[11] Suzuki, S. and Keiichi, A. (1985) Topological Structural Analysis of Digitized Binary Images by Border Following. *Computer Vision*, *Graphics*, *and Image Processing*, **30**, 32-46. https://doi.org/10.1016/0734-189X(85)90016-7

[12] Li, Y.P. (2022) Cellular Mechanism of Mouse Atrial Development. *Open Journal of Regenerative Medicine*, **11**, 1-24. https://doi.org/10.4236/ojrm.2022.111001