IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-7, Issue-4, April 2019 E-ISSN: 2347-2693

# Green Virtual Mouse Using OpenCV

**Manne Vamshi Krishna[1*], Gopu Abhishek Reddy[2], B. Prasanthi[3], M. Sreevani[4]**

[1,2,3,4]Dept. of Computer Science, Mahatma Gandhi Institute of Technology, Hyderabad, India

*Corresponding Author: vamshikrishnamanne@gmail.com Tel:+91 8297574773*

DOI: https://doi.org/10.26438/ijcse/v7i4.575580 | Available online at: www.ijcseonline.org

Accepted: 11/Apr/2019, Published: 30/Apr/2019

***Abstract*---** There has been a greater development of virtual technologies in the recent arena. Some of them increased the computing performances of the functioning systems. One of those highly used virtualized technology is the virtual mouse. The moments that the mouse detects are converted into the pointer movements on a display that enables the management of Graphical User Interface (GUI) on a computer platform. This paper advocates an approach for Human-Computer Interaction (HCI) where a real-time camera is used in handling the cursor movements. The Virtual mouse colour recognition program acquires real-time images continuously which will then go through a series of filtration and transformation. As the process completes the program will apply an image processing technique to capture the coordinates of the position of the targeted colours from the changed frames. Then a set of different combinations of functions are operated and then by analyzing the set of different colours thereby a program will execute the mouse function and then it is translated as an actual mouse for user's machine.

***Keywords*-**Virtual Mouse, Graphical User Interface, Colour Recognition, Human-Computer Interaction, Calibration Phase, Recognition Phase

## I. INTRODUCTION

With reference to computing, a mouse is a pointing, hand-held device that identifies two-dimensional movements corresponding to a flat surface. It involves the controlling of Graphical User Interface (GUI) on a computer display using the movement of a pointer. There are several types of a mouse at present; initially, the mechanical mouse that determines the movements by a hard rubber ball that rolls around as the mouse is moved. Later the optical mouse was proposed to substitute the rubber ball with an LED sensor to recognize the table top changes and gives off the information to the computer to process. Later in the year 2004, the invention of laser mouse which improves the accuracy movement and overcomes the difficulties of previous versions by enabling to track high-gloss surfaces.
In spite of the limitations, the technology still continues to rise at a higher pace including the importance of Human-Computer Interactions (HCI). Ever since the evolvement of hand-held devices which can interact via touch screen technology, all the global nations are in demand for such technology on every device, including the desktop systems. But this could lead to a price factor where it cannot be afforded by every user.

Accordingly, a Virtual HCI device which is able to replace the physical mouse or keyboard by using a webcam or any other image capturing devices can be an alternate technology for touch screen devices. Employing this technology, a webcam will be continually used by software that keeps track of the gestures given by the user in order to process and translate them as a motion of points similar to a physical mouse.

This paper thereby gives a brief summary of the procedure involved in the setup of virtual mouse as the introduction in section I, motivation and scope in section II, it's functioning in the sections III, results and implementation issues in section IV, and their outcome and drawbacks in section V and overall conclusion in section VI respectively.

## II. BACKGROUND AND SCOPE

A. The motivation of Virtual Mouse
It can be said that the Virtual Mouse will soon be replacing the traditional physical mouse in the near future, where the users are intending interest towards technology that can be accessible even remotely without access of any external devices. This could increase the scope for convenience and cost-effectiveness.

**Cost Effective:** A quality physical mouse can have a variation of its cost depending on its features embedded. As the virtual mouse requires only a webcam, the investments spent on a physical mouse can be disregarded and a single webcam is adequate for the users to associate with the

 

computer system, while the portable systems such as a laptop have an in-built webcam which only needs the virtual mouse software for enabling the service.

**Convenient:** The users require an actual physical mouse to communicate with a computer system, which requires a certain phase of an area to operate, but it is not to specify the cable length constraints it suffers. But the virtual mouse makes it convenient as there is requirement of only a webcam which allows image capturing of user's hand position, the colour paper detection and in order to determine the position of the pointers that the user want it to be by moving fingers, thus eliminating the need to manually move the physical mouse.

B. Scope

To replace the physical mouse, the virtual mouse will be soon introduced; to improve convenience which will be able to precisely communicate and regulate the computer system. To process that feature, the software developed must be swift enough to capture and process every image, in order to successfully track the user's gestures and the colour papers used on them. Thereby, this software application is aided with latest software coding techniques and open source library known as the Open CV. The scope of this application enables various functionalities.

- Removes the requirement of having a physical mouse
- User-friendly application
- Real-time application

When the user's gesture is captured in real time by the webcam the application starts processing, where the captured image will be processed for segmentation in order to classify which pixels values equal to the defined colour's values. Once the process of segmentation is completed, the overall image will be converted to binary image where the identified pixels will be shown in white and the rest are black. The position of the white section in the image will be recorded and set as the position of the mouse pointer, thus results in simulating the mouse pointer without the use of a physical mouse.

C. Impact and its Significance

The virtual mouse application is capable of replacing all the current methods of utilizing a physical computer mouse where manual operations can be overridden. It offers an easy way to communicate with the computer system, where every assignment can be gesture performed. Besides, this application could direct the motor-impaired users by just showing the correct sequence of colours to the webcam.

## III. PHASES OF VIRTUAL MOUSE IMPLEMENTATION

In the process of colour recognition, it contains two major phases which are the calibration and recognition phases.

A. Calibration Phase

The calibration phase is used to allow the system to recognize the Hue Saturation Values of the colours chosen by the users, thereby it will store the values and settings to text documents which will be used on later in recognition phase. It has following steps involved in this particular phase.

- **Real-Time Image Acquisition**
  The program begins by capturing all real- time images via a webcam where it will wait for the user to give input colour. The size of the acquired image is then compressed to a reasonable size which reduces the processing loads of the processing pixels within the captured frame.
- **User's Colour Input Acquisition**
  This program acquires the frames that consist of input colours submitted by the users, the captured image is thereby sent for processing where it undergoes a series of transition and calculations to acquire the calibrated HSV values.
- **Frame Noise Filtering**
  Every captured frame consists of noises that effect the accuracy and performance of the entire program, therefore we need to maintain them noise free. For eliminating these noises filters are added to the captured frames to remove the unwanted noise. The Gaussian filter will be used to provide efficient smoothing and discard the noises in a frame.
  **GaussianBlur** (InputArray**src**, OutputArray**dst**, Size **ksize**, double **sigmaX**, double **sigmaY**=0, int **borderType**=BORDER_DEFAULT)
- **HSV Frame Transition**
  Any captured frame is translated from a BGR format to an HSV format.

  **cvtColor** (**src**, **dst**, **CV_BGR2HSV**)

- **HSV Values Extraction**
  The HSV values are acquired by splitting the converted frame into 3 single different phases. In order to do that the frame needs to be divided from a multiple-channel array into a single-channel array, which can be performed using split() function.

  **split** (const Mat&**src**, Mat* **mybeing**)
- **Standard Deviation Calculation**
  This methodology can be used to obtain the maximum and minimum of the HSV values, which is a measurement used to quantify the amount of variation/ dispersion among other HSV values. For obtaining an accurate range of values three sigma rule is required in the calculation so that the chances of the captured values have a very high possibility to fall within the three-sigma intervals.

 

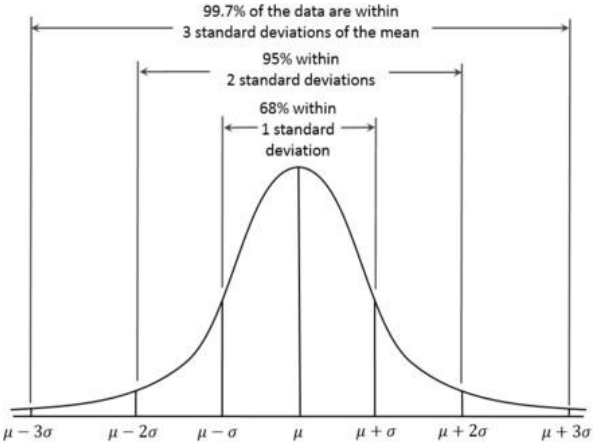

Figure 1. Distribution of Three-Sigma Rule

B. Recognition Phase

In the recognition phase, the system records values during the calibration phase, starts capturing frames and find for colour input, based those values. It has following steps involved in this particular phase.

- **Webcams and Variables Initialization**
  At the commencement of the recognition phase, the application shall initialize all the essential variables used for holding variant frames and values where each of them will be used to carry out a certain task. Moreover, this is the part where the application gets the calibrated HSV values and settings where they are used later for the Binary Threshold step.
- **Real-Time Image Acquisition**
  A webcam is used to capture the real-time images using (**cv::VideoCapture cap(0);**) where each of the images captured are stored using a frame variable (**cv::Mat**), which are then compressed and flipped into a fair size that lessens the processing load.
- **Frame Noise Filtering**
  Frame noise filtering resembles the filtering in that of the calibration phase. To overcome the current noise of the captured frames, Gaussian filters are employed.
  **GaussianBlur** (InputArray**src**, OutputArray**dst**, Size **ksize**, double **sigmaX**, double **sigmaY**=0, int **borderType**=BORDER_DEFAULT)
- **HSV Frame Transition**
  Conversion of every captured frame from a BGR to an HSV format is required.
  **cvtColor** (**src**, **dst**, **CV_BGR2HSV**)
- **Binary Threshold Transition**
  The HSV frames thus converted encounter a series of checks whether the HSV values of converted frame lie within the values of HSV variables associated in the calibration phase. As a result of the range check, the binary frame is converted into a Binary Threshold, where one part of the frame is set to 255 (1 bit) if given values range in between the given HSV values, else the frame is set to 0 (0 bit).
- **Binary Threshold Morphological Transformation**
  On obtaining the Binary Threshold value, the frames undergo a process called Morphological Transformation, which is a structuring method used to eliminate holes and tiny objects hiding about the forefront processes. The translation aids of two morphological operators, namely Erosion and Dilation. The erosion operator is required for erasing the borders of the foreground object, diminishing the region of a binary threshold, which becomes beneficial for reducing meager noises. Dilation principle is in contrary to Erosion which raises the region of binary threshold and enables the eroded object to retire into its legacy form.
- **Colour Combination Operation**
  Once the Morphological Transformation Process gives the results, the program calculates the remaining objects by highlighting them as blobs which needs a cvBlobwhich is an extension to OpenCV. To analyze the functionality of mouse functions based on colour combinations found within the captured frame, the results of the calculation are sent for comparison.
- **Colour Coordinates Acquisition**
  For every object contained in the binary threshold range, the program will highlight the overall state of the object where it determines the complete area of the shape and midpoint coordinates of the shapes. These coordinates will be utilized later in either placing the cursor positions or measure the length between two points to execute various mouse functions based on results gathered.
- **Execution of Mouse Action**
  The mouse actions are executed based on the colour combinations existing in the processed frame. The mouse actions are thereby performed according to the coordinates allocated by the program, and then the program will continue on acquire and process the oncoming real-time image until the user terminates from the program.

## IV. RESULTS AND DISCUSSION

The Virtual Mouse Colour Recognition must be capable of identifying the colors given by users with high precision, consistency, and insignificant performance influence on others. However, any variations in the attributes of the captured frames; such as brightness, weather and background stem in varied recognition results. The following circumstances may result in inaccurate detection during the recognition phase in real-time:

a) The image varies with the brightness of the environment, dark or bright.
b) Due to color conflicts in the image with the backdrop.

 

c) Distance variations when the user interacts with the application.
d) The images are rotated in a clockwise or anti-clockwise rotation.

To obtain higher precision and consistency throughout the recognition cycle, implementing the verification plan is required for a faultless performance of the program. Below are some of the test cases which work over different conditions.

- **Bright environment condition**

Figure 2. Bright Environment

In a bright environment, the program was effective and all the colors have been identified precisely. The two highlighted squares indicate that the targeted colors are recognized, distinguished, and executed subsequently. However, minimal modification of HSV track-bars is required.

Figure 3. Brighter Environment

In a much brighter environment, the intensity of brightness was too high that exceedingly changes the actual RGB values of the targeted colors, making it unrecognizable.

- **Dark environment condition**

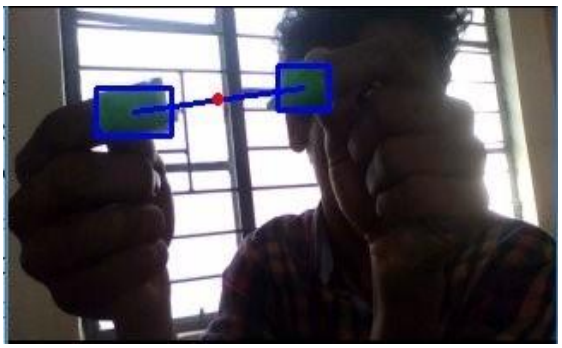

Figure 4. Dark Environment

In a dark environment, when tested, the application was efficient, an all the colors have been identified precisely. The two highlighted squares indicate that the targeted colors are recognized, distinguished, and executed subsequently.

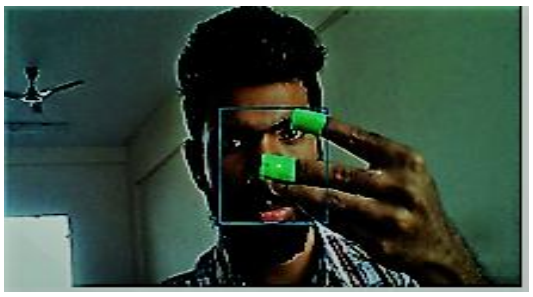

Figure 5. Darker Environment

In a darker environment, some colors are not correctly recognized, because the intensity of brightness was too low that exceedingly changes the actual RGB values of the targeted colors, making it unrecognizable.

- **Color conflicts condition**

Figure 6. Colour Conflict condition

Color inputs are successfully disregarded. The conflicts were identified, letting it cease executing outcast mouse functions until the color conflicts are no longer in the frame.

- **Various distance between webcam**

 

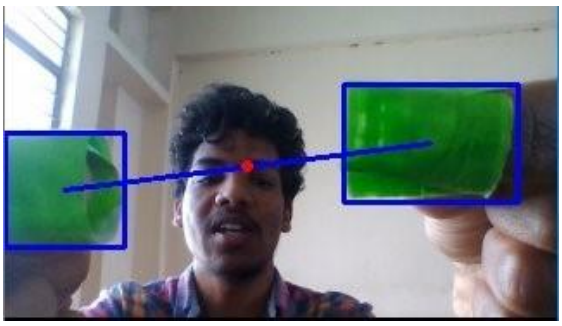

Figure 7. 15cm away from the webcam

Although the two pointers are brought very close to the webcam where they actually look larger than just points, the colors have been identified precisely.

Figure 8. 35cm away from the webcam

Even though, the pointers are moved farther than the usual range of usage, 20cm to 30cm from the camera, the application was able to identify the colors precisely.

- **Rotation**

Figure 9. Tilted and Rotated

If the position of the webcam is tilted or rotated, the program runs by changing its axis and functions as above and identifies colours precisely for performing the mouse operations.

## V. OUTCOMES AND DRAWBACKS

By the tests carried out on various conditions, the application recognizes colours and distinguishes them correctly in both brighter and darker environments than normal, but the threshold limit is until the RGB values stay unchanged by the effect of brightness. Also, it works from a distance of pointers varying from 10cm to 35 cm from the webcam. It also works even though the webcam isn't on the axis, it may be tilted to any angle, but the application sets it correctly.

There are problems in this project that limit the results of colour recognition. Distance is one of those problems that may affect the results, as the current detection region can support up to 1m radius, any display of colours that exceed the mentioned distance will be considered as noise and be filtered off. The performance of the program is dependent on the users' hardware, as processor speed or resolutions taken from the webcam effect the performance load. Therefore, the slower the processing speed or the higher the resolutions, the longer time are required to process a single frame.

## VI. CONCLUSION AND FUTURE SCOPE

To sum up, the virtual mouse replaces the present physical mouse, where all mouse movements can be carried out without compromising accuracy and efficiency. For these attributes to be maintained, few techniques were required. Firstly, the colour coordinates in-charge of handling cursor movements are averaged to reduce unwanted cursor movements and stabilize it. As a different distance triggers different mouse functions, several colour combinations were implemented with the distance between colours in the combination. Accordingly, actual mouse functions can be triggered precisely with the least examination.

Besides, to develop effective and amenable tracking of colours, calibrations phase was implemented, in which the users choose colours for different functions and they don't fall in the same RGB value range. Additionally, adaptive calibrations were also implemented as well, which allows the program to save a different set of HSV values from different angles where it will be used during the recognition phase.

There are some innovations and enhancements needed in order for the program to be more user-friendly, dependable, and flexible in multiple environments. Better performance of the application can be achieved with better hardware as the response time is highly dependent on it, which involves the speed of the processor, RAM availability and webcam features.

 

## REFERENFCES


[1] Banerjee A, Ghosh A, Bharadwaj K,& Saikia H, "Mouse Control using a Web Camera based on color Detection", IJCTT vol. 9 no. 1,March 2014.

[2] Piyush Kumar, Siddharth S. Rautaray, Anupam Agrawal., "Hand data glove: A new generation real-time mouse for Human-Computer Interaction", IEEE; 07 May 2012.

[3] R. Meena Prakash, T. Deepa, T. Gunasundari, N. Kasthuri,"Gesture recognition and fingertip detection for human computer interaction", ICIIECS; 18 Mar 2017.

[4] "Kalman Filtering: Theory and Application", IEEE Press, 1985.

[5] R. A. Brooks, "The Intelligent Room project, Proceedings of the 2nd International Conference on Cognitive Technology" (CT '97), p.271, August 25-28, 1997

[6] Michael Coen, Brenton Phillips, Nimrod Warshawsky, Luke Weisman, Stephen Peters, and Peter Finin, "Meeting the computational needs of intelligent environments: The meta glue system." In Proceedings of MANSE'99, 1999.

[7] Donald J. Cohen, Laurence Prusak, "In good company: how social capital makes organizations work", Ubiquity, January 2001.

[8] Michael Dertouzos. "The future of computing", Scientific American, 1999.

[9] Krzysztof Gajos, Rascal - "A Resource Manager for Multi Agent Systems in Smart Spaces, Revised Papers from the Second International Workshop of Central and Eastern Europe on Multi-Agent Systems: From Theory to Practice in Multi-Agent Systems", p.111-120, September 26-29, 2001.

[10] Ajay Kulkarni. "A reactive behavioural system for the intelligent room". Technical report, MIT AI Lab, 2002.

[11] "Employees on the move, Steelcase Workplace Index Survey", April 2002.

[12] Max Van Kleek, "Intelligent environments for informal public spaces: the Ki/o Kiosk Platform". M.Eng. Thesis, Massachusetts Institute of Technology, Cambridge, MA, February 2003.

[13] P Viola and M Jones. "Rapid object detection using a boosted cascade of simple features. In Proceedings of Computer Vision and Pattern Recognition, IEEE 2001


 