[[1]](#footnote-1)

AI Virtual Mouse using OpenCV

Kiran More, Aditya Anasune, Aditya Patil, Aditya Suresh, Tejas Adsare, Aditya Bhoite, Aditya Kirar

Department of Engineering, Sciences and Humanities (DESH)

Vishwakarma Institute of Technology, Pune-411037, Maharashtra, India

*Abstract* — *Hand Gesture Recognition plays a key role in human-computer interactions. In a step towards making human and computer interactions beneficial, we propose an interactive computer application which can operate without any kind of need to have physical contact with keyboard and mouse and thus revolutionizing the way computers are controlled.* *The proposed system will require just the webcam as the sensor and will process the data with the help of fed algorithm. The output will be shown on the screen and the way of communication will just be very natural and effective. The application will be implemented using the python and various other modules in python like OpenCV, pyautogui and Numpy.*

***Keywords — Computer Vision, Hand Gestures, Image Capture, OpenCV, Processing, Virtual Mouse.***

# INTRODUCTION

We all are experiencing the era of technological development where we are striving to develop every field and every aspect of human life to make it easier to do things than they were did before. Going with that flow, we are introducing this project which proposes a *web-cam based mouse control system* which will allow user to control the cursor and carry out various mouse callback functions without any need of physical contact. It will facilitate efficient way to control mouse and easily modifiable to add more functions to it. The only restriction to upgrade the project will the hardware limitations as we didn’t include any high end PC requirements for running our project.

In this project, a hand detector module is used which is created using the *mediapipe module developed by Google, inbuilt modules like time and math module and the cv2 module* as well. The Hand detector module will be able to detect the presence of human hand and fingers and will be sued to draw lines over the detected part. The whole process goes through frame capture, frame processing, frame embedding, output frame and commands. Frame capture involves capturing real-time video from the camera available on the system and frame processing involves reading of images with the given algorithms to search for the presence of human fingers. Then it embeds the required line to show that the detection has been carried out successfully. After these steps, the final output of the processing is displayed on the screen whereas the changes detected is converted into system commands to control the mouse functions.

The accuracy of the project solely depends on the system used, the lighting available, the quality of image capture and the level of processing done by the system. The more favorable all these factors are, the more efficient will be the output. However, more work can still be done to make improvements so as to minimize time for user comfort and work on overall user experience with the help of a proper package making it easily installable and usable.

With the help of this paper, we are moving one step ahead towards making a hand recognition software for Laptops and PCs with built-in webcams or USB web-cam with the help of which the user will be able to control his system and perform various functions without any need of physical mouse. It will be performing simple yet necessary functions like left click and right click along with side functions like scrolling and taking screenshots.

# Literature Review

Current existing systems comprise of gesture recognition but they need to be done with a typical colored finger ring to be worn for the project to be able to recognize the presence of the finger. It doesn’t have a proper algorithm to look specifically for a finger instead it searches for a specific color which makes it redundant considering that the mere presence of similar color in the background will affect the results and might result in misclicks happening without the user’s consent. Some systems allow user to perform some functions but have restricted movement of hand allowing user to draw some shapes with their fingers of hand to call a function which itself has many gesture restrictions as well as chances of confusion. Some existing systems limit user with the options of functions as implementing multiple functions in one project might lead to slower process and less functionality of system with limited frames. Some projects do implement volume control with the help of finger movement but it is restricted to that part only and it’s not seen to be expanding in the field of mouse callback functions.

# Methodology

In this section of methodology, the method used in each component as well as the working of each subsystem will be explained. Following are the specified subsystems:

## Camera Modifications

All the video input for the project is done with the help of webcam connected to the PC or Laptop. To capture real-time video, Video Capture object is used. It is passed with an index number indicating the camera which we want to use where ‘0’ indicates default camera. If we have multiple cameras then passing 2 or 3 or so on will use that camera as input source. It’s also easy to modify the color coding of the input video with the help of cvtColor inbuilt function.

## 

**Fig -1.** Flowchart for the program

## Capturing of frames

The capturing of frames is done with a ‘while True:’ infinite loop which captures every frame it detects, which may max out at the frame rate limit of the camera device. Each frame captured is converted into RGB as the cv2 library reads each image file in BGR format. We have implemented only one types of color conversions out of total 150+ conversions available in OpenCV, i.e., BGR to RGB.

## Application of Algorithm

Each frame captured is processed through the input algorithms and the required functions which have been called, draw the polylines over the fingers as well as draw the circles over finger tips indicating that our detection algorithm is working. It stores the position of finger in first frame, stores it, stores position of second frame, stores it too and then compares both values and commands the system to move the cursor correspondingly. On the detection of multiple fingers, it carries out functions defined on the successful detection of that specific number of fingers.

## Output frames

All the processes involved in the algorithm are done with OpenCV, whereas mouse callback functions lie within the pyautogui library. The result frame showing the finger position and with all the polylines and contours is displayed on the user’s screen with the help of imshow() function from cv2 library. The while loop which we created for the frame capturing is the one providing output frames. We can break the loop with the press of a specific key with the help of waitKey() function in the cv2 library. It will also result in the destroyAllWindows() function which will kill the open window and also kill the camera capture with the help of release() function.

## Mouse Movement

When the first frame is captured, the coordinates of the fingers are stored in the repository. After capturing the second frame, this process is repeated again and the difference between two coordinates is converted into signals which then moves the mouse cursor accordingly. We are converting the detected coordinate from camera resolution to the actual screen resolution. All this process is done in loop and the capturing or frames and change in coordinates is done with the help of the given algorithms. So we just have to start the loop and it will perform the rest of functions for us with just our hand movements.

## Clicking and other functions

Movement of the cursor is done quite well. After that comes the functions which play crucial role in controlling like left click and right click. It is done when we depict a set of gestures which we entered in our algorithm. Specified function will perform when a certain fixed combination of fingers is detected in the camera frames. We can even toggle various other functions before running the program.

# Results and Discussions

Our motive was to create a project that is feasible as well as efficient. Our project features multiple functions that have the added benefit of being user-defined. Every function that is present in the code solves real-life issues faced by users.

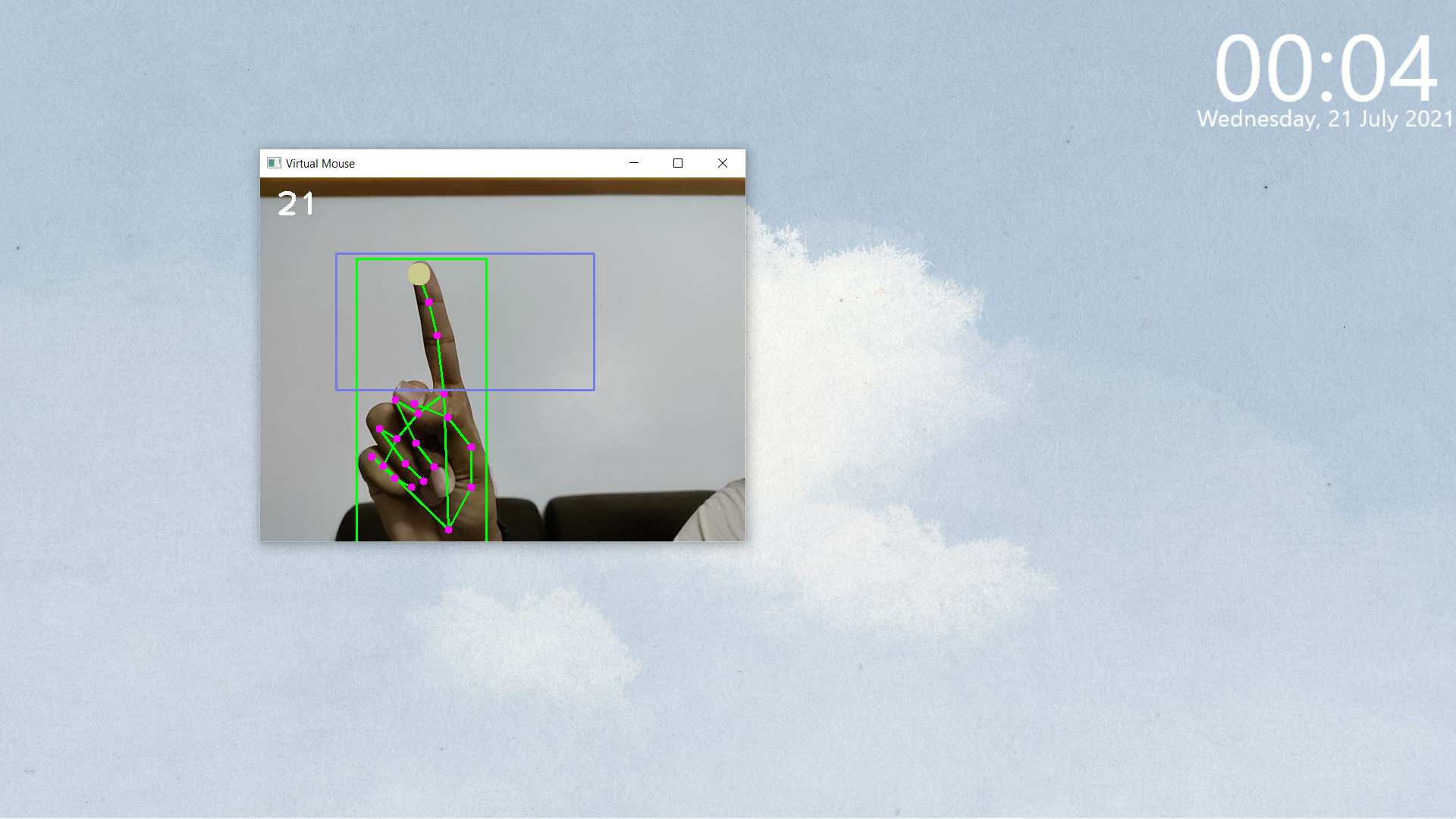
The proposed system controls the functions of the mouse pointer with the hand tracking module and performs traditional mouse functions such as left-click, right-click, cursor movement, and various other add-on functions.

We designed a hand-tracking module that aids us in tracking the movement of the fingers, including their orientation. It also detects the number of fingers pointing at the webcam. This particular part of the code helps us to lay a baseline for adding necessary functions.

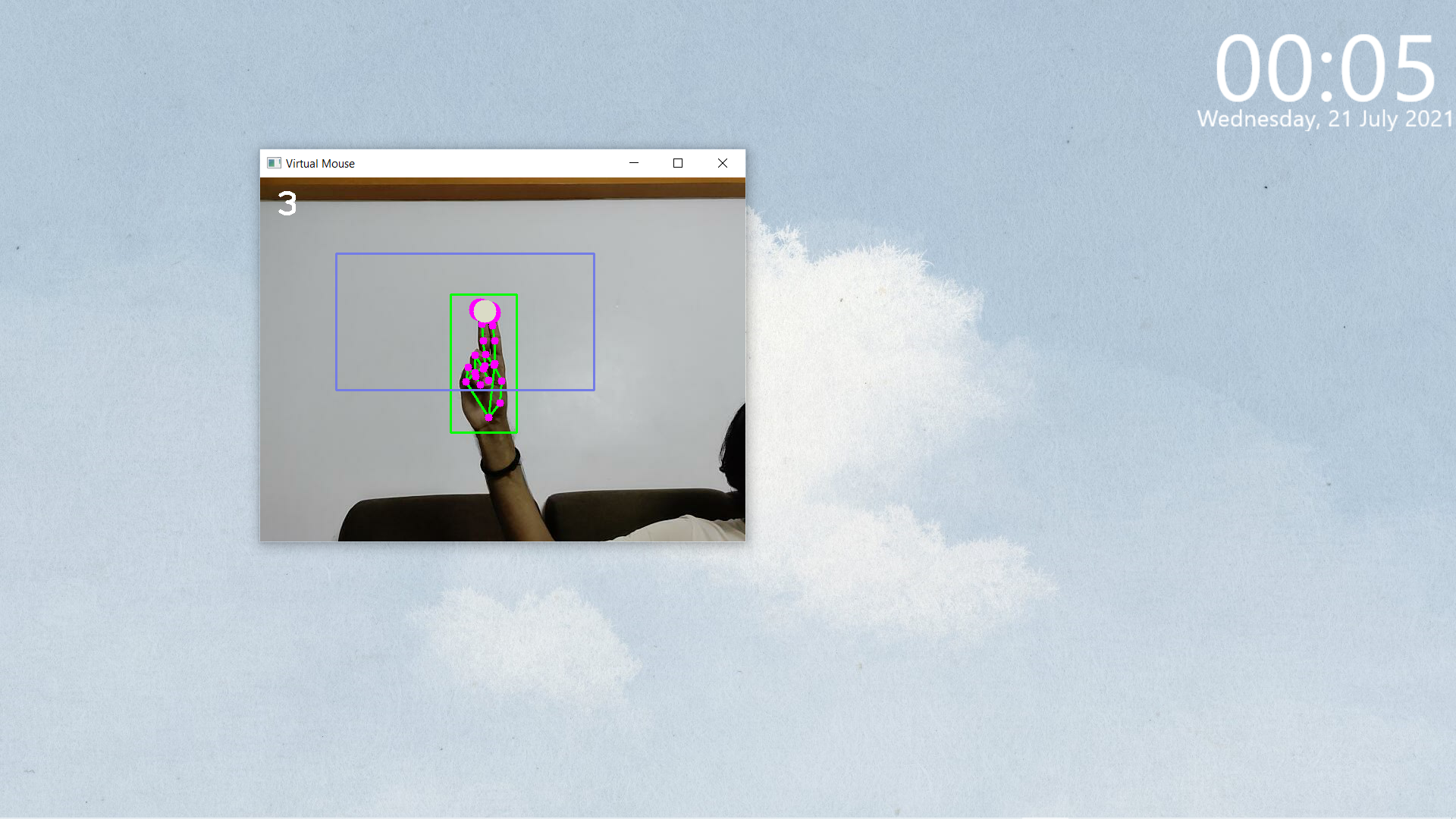
The second part of our code was designed to include the main functions when moving a single finger around, the cursor moves along, when the number of fingers is two, the left click function is initiated. When the finger count is 3, right-click function is performed.

There’s a lot of more functions which we can add but lack of good camera system might lead us into various problems and errors as well as less system stability which is the only thing holding us back. We can also add functions like screenshot, scrolling and zooming to make it more feature rich.

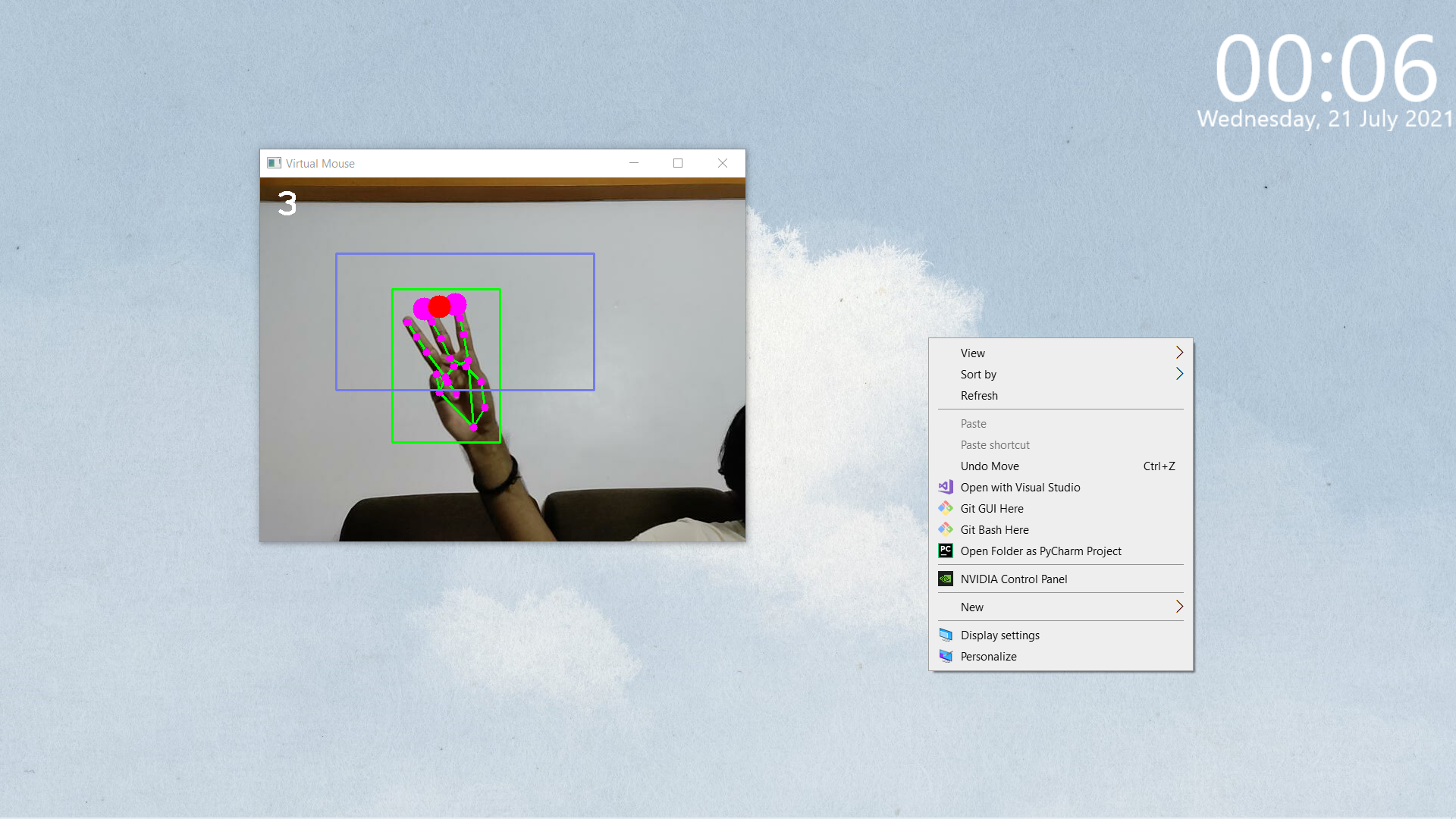
It can be very well seen our project that hand detection is done properly and the mouse callback functions work properly without any issues. Any average PC can easily get upto 50 frames per second and it’s operation looks to be working smooth without any major issues.



**Fig -2.** Index finger mouse movement



**Fig -3.** Two fingers for left click



**Fig -4.** Three finger for right click

# Limitations

Detection of objects or say human hand requires proper lighting conditions with an average quality camera. Having a very low fps camera or a bad resolution camera will surely affect the efficiency and accuracy of the project. Detection of frames and proper color is necessary for which a good camera is required. Also, if an object similar to hand appears in the background, it may affect the results and can cause abrupt clicks or uncontrolled mouse movements. However, most of the camera’s which we use on our common laptops and desktops are more than enough for the well-functioning of the program. Presence of multiple hands might startle the system as well, as there’s no specific way as such like priority order for the mouse to decide the precedence.

# Future Scope

Our project successfully performs various functions, yet there are multiple things we wish to include further in our project. First off, we want to promote this product as a full-fledged package. Upon installing the package, the program will automatically install all the necessary modules required to successfully run the program. All of these modules will be installed in a so-called virtual package.

We also wish to make this program run on startup and get the program to automatically start working when the system is booted, we would no longer need to use a traditional mouse at all, not even to get the virtual mouse to run, thus making the program a fully functional virtual mouse.

All of these functions and add-ons will further enhance the working of the project and improve the user experience immensely. Right now, the project is still very crude. It needs further polishing to make the program accessible to every user of every generation. We wish to achieve this by including add-on functions that the user would like to use, thus putting the consumer in control of the project. AI tracking is the future of human-computer interaction, and our project is a mere stepping stone towards this marvelous technology.

# Conclusion

As we talked about this earlier, gesture recognition is seen to bring in a lot of technological advancements to enable the human-computer interactions to happen in a more natural way. It gives us a way to indirectly talk with the computer itself. It can be seen that this gesture recognition technology can be widely used in the field of augmented reality and computer games too. With the help of this technology, we can enable us to draw and paint pictures too.

We can design graphics, develop graphics and manipulate and edit them too. We have already seen motion tracking gaming systems where the hand movement is captured and it is then depicted in the game itself. This makes the games quite interactive and the addition of this technology makes it more attractive. We can improve the project by making it compatible with all systems and making it universally applicable to any system we want. We successfully showed the potential of the gesture recognition technology and what we can achieve with it’s implementations. Thus, we can simplify the interactions and can surely make the user interface more natural, interactive and fun with the gesture recognition.

Acknowledgment

It gives immense pleasure to express our deep sense of gratitude with sincere thanks and appreciation to our project guide Prof. Kiran More Sir for suggesting and supporting us to carry the project work. We would also like to thank Prof. Mahajan Sir, for his valuable guidance. We were able to complete this project successfully with the help of valuable guidance and constant encouragement of them.

References

1. Abhik Banerjee, Abhirup Ghosh, Koustuvmoni Bharadwaj,” Mouse Control using a Web Camera based on Color Detection”,IJCTT,vol.9, Mar 2014.
2. Angel, Neethu.P.S,”Real Time Static & Dynamic Hand Gesture Recognition”, International Journal of Scientific & Engineering Research Volume 4, Issue3, March-2013. H. Poor, *An Introduction to Signal Detection and Estimation*. New York: Springer-Verlag, 1985, ch. 4.
3. VIRTUAL MOUSE USING HAND GESTURE, Abhilash S , Lisho Thomas, Naveen Wilson, Chaithanya C,e-ISSN: 2395-0056, p-ISSN: 2395-0072
4. G. Saravanan, G. Yamuna, S.Nandhini(2016).Real time implementation ofRGBtoHSV/HSI/HSL and its reverse color space models. 2016 International Conference on Communication and Signal Processing (ICCSP). C. J. Kaufman, Rocky Mountain Research Lab., Boulder, CO, private communication, May 1995.
5. R. M. Prakash, T. Deepa, T. Gunasundari and N. Kasthuri, "Gesture recognition and fingertip detection for human computer interaction", *2017 International Conference on Innovations in Information Embedded and Communication Systems (ICIIECS)*, pp. 1-4, 2017.
6. R. S. Baştuğ, B. Yeşilkaya, M. Unay and A. Akan, "Virtual Mouse Control by Webcam for the Disabled", *2018 Medical Technologies National Congress (TIPTEKNO)*, pp. 1-4, 2018..
7. D. A. Barhate and K. P. Rane, "A Survey of Fingertip Character Identification in Open-Air Using Image Processing and HCI", *2018 3rd International Conference for Convergence in Technology (I2CT)*, pp. 1-4, 2018.
8. *Mingyu Chen, Ghassan AlRegib and Biing-Hwang Juang, "AirWriting Recognition-Part I: Modeling and Recognition of Characters Words and Connecting Motions", IEEE Transactions on Human-Machine Systems, vol. 46, no. 3, pp. 2168-2291, June 2016.*

1. [↑](#footnote-ref-1)