International Journal of Recent Engineering Research and Development (IJRERD)
ISSN: 2455-8761
www.ijrerd.com || Volume 09 – Issue 03 || May - Jun 2024 || PP. 85-97

Webcam based Virtual Mouse Application using Deep Learning

T. Bharathi[1], M. Thamizharasi, A. Dhivya
[1]Assistant Professor in Computer Applications
Department of Computer Applications
Perunthalaivar Kamarajar Arts College
Kalitheerthalkuppam, Puducherry

Abstract: The Virtual mouse suggests a method for using computer vision and deep learning models to implement the functionalities of the cursor based on hand gestures. These models aim to replicate the swaying motion of human hand gestures and hold the key to further improving the performance of such computer vision solutions. In today's technological environment, several technologies are constantly evolving. One such promising idea is the human-machine interface. The concept is to use hand gestures to emulate mouse functionality on the screen without the use of any hardware, just by utilizing finger motions, a process known as gesture recognition. In this project, introduces a novel Human-Computer Interaction (HCI) strategy. For the implementation of this system, Python will be utilized, and its dependencies include OpenCV, MediaPipe, and the latest packages such as PyAutoGUI.

Keywords: Computer vision, Hand gesture, OpenCV, MediaPipe, PyAutoGUI.

1. Introduction

A Computer Mouse is an input device that helps to point and to interact with whatever that is being pointed. There are so many types of mouse in the current trend, there's the mechanical mouse that consists of a single rubber ball which can rotate in any direction and the movement of the pointer is determined by the motion of that rubber ball. Later the mechanical mouse is replaced by the Optical Mouse. Optical Mouse consists of a led sensor to detect the movement of the pointer. Years Later the laser mouse was introduced to improve the accuracy and to overcome the drawbacks of the Optical Mouse. Later as the Technology has been increased drastically wireless mouse was introduced so as to enable hassle free movement of the mouse and to improve the accuracy.

In the proposed AI virtual mouse system, this limitation can be overcome by employing webcam or a built-in camera for capturing of hand gestures and hand tip detection using computer vision. The algorithm used in the system makes use of the machine learning algorithm. Based on the hand gestures, the computer can be controlled virtually and can perform left click, right click, scrolling functions, and computer cursor function without the use of the physical mouse. The algorithm is based on deep learning for detecting the hands. Hence, the proposed system will avoid COVID-19 spread by eliminating the human intervention and dependency of devices to control the computer.

In the proposed system, the web camera captures and then processes the frames that have been captured and then recognizes the various hand gestures and hand tip gestures and then performs the particular mouse function. Python programming language is used for developing the AI virtual mouse system, and also, OpenCV which is the library for computer vision is used in the AI virtual mouse system. In the proposed AI virtual mouse system, the model makes use of the MediaPipe package for the tracking of the hands and for tracking of the tip of the hands, and also, Keras, OneHotEncoder, and PyAutoGUI packages were used for moving around the window screen of the computer for performing functions such as left click, right click, and scrolling functions. The results of the proposed model showed very high accuracy level, and the proposed model can work very well in real-world application with the use of a CPU without the use of a GPU.

2. Related Work

Johnson et al. focused on integrating gaze tracking technology into virtual mouse systems to enhance user interaction. Gaze tracking provides a natural and intuitive way for users to control the mouse, which is particularly beneficial for users with limited mobility. The use of convolutional neural networks (CNNs) allows for accurate detection and tracking of eye movements, significantly improving the overall user experience. This paper highlights the advancements in computer vision and eye-tracking technologies and their applications in creating more accessible and efficient user interfaces [1].

Patel and Lee's research addresses the needs of individuals with disabilities by developing a dynamic virtual mouse system based on reinforcement learning. This approach allows the system to adapt to the unique needs and preferences of each user, improving accessibility and usability. Reinforcement learning, which involves training algorithms through trial and error, helps create a more personalized interaction experience. This study underscores the importance of inclusivity in technology design and demonstrates the potential of machine learning to enhance assistive technologies [2].

Garcia and Wang explored the development of an adaptive virtual mouse system aimed at improving accessibility. By employing decision tree algorithms, the system can adapt to different user inputs and behaviors, providing a more tailored experience. This research contributes to the field of adaptive user interfaces, showing how machine learning techniques can be used to create more accessible technology solutions for users with varying needs and abilities [3].

Kim and Chen investigated the use of electromyography (EMG) signals for controlling a virtual mouse. EMG signals, generated by muscle activity, offer a hands-free method of interaction, which is particularly useful for users with physical disabilities. The study utilizes support vector machines (SVMs) to classify EMG signals accurately, enhancing the reliability and responsiveness of the virtual mouse system. This research highlights the potential of bioinformatics and signal processing in developing innovative assistive technologies [4].

Nguyen et al. developed a real-time virtual mouse system based on electroencephalography (EEG) signals, which are brainwave patterns. This approach leverages advancements in neuroscience and signal processing to enable brain-computer interface (BCI) technology. By using random forest algorithms to interpret EEG signals, the system provides a non-invasive method for users to control the mouse with their thoughts. This study represents a significant step towards more intuitive and seamless BCIs, offering new possibilities for user interaction [5]. Martinez and Rodriguez focused on improving the precision of virtual mouse systems using support vector machines (SVMs). SVMs are effective in handling classification tasks and can enhance the accuracy of virtual mouse movements. This research contributes to the field of machine learning and human-computer interaction by demonstrating how SVMs can be used to improve the performance of virtual mouse systems [6].

Wang and Li explored virtual mouse control through hand gesture recognition, using hidden Markov models (HMMs). Gesture recognition provides a natural and intuitive way for users to interact with computers. This study leverages advancements in pattern recognition and computer vision to create a more responsive and user-friendly virtual mouse system. The use of HMMs allows for accurate recognition of dynamic gestures, enhancing the overall user experience [7].

Garcia et al. focused on improving the accuracy of virtual mouse systems using Bayesian optimization. This technique optimizes the system's parameters to achieve better performance, enhancing user interaction and efficiency. The study highlights the potential of Bayesian methods in refining machine learning models and improving the accuracy of virtual mouse systems [8].

Table 1: Review of Literature with Accuracy Value

S. No.	AUTHOR	YEAR	ACCURACY	ALGORITHM	PAPER NAME
1	Johnson et al.	2023	88.00%	CNN	Johnson, C., Smith, D., & Williams, E. (2023). "Improved Virtual Mouse with Gaze Tracking." Journal of Human-Computer Interaction, 35(2), 123-136.
2	Patel and Lee	2022	85.00%	Reinforcement Learning	Patel, R., & Lee, S. (2022). "Dynamic Virtual Mouse for Disabilities." Assistive Technology Journal, 28(4), 345-358.
3	Garcia and Wang	2021	90.00%	Decision Tree	Garcia, M., & Wang, L. (2021). "Adaptive Virtual Mouse for Accessibility." IEEE Transactions on Human-Machine Systems, 14(3), 210-225.
4	Kim and Chen	2020	80.00%	Support Vector Machine	Kim, J., & Chen, S. (2020). "Virtual Mouse Control via Electromyography." Journal of Biomedical Engineering, 42(5), 478-491.
5	Nyugen and Smith	2019	75.00%	Random Forest	Nguyen, T., Smith, J., & Lee, H. (2019). "Real-time Virtual Mouse using EEG Signals." Proceedings of the IEEE International Conference on Systems, Man,

					and Cybernetics.
6	Martinez and Rodriguez	2018	82.00%	Support Vector Machine	Martinez, P., & Rodriguez, E. (2018). "Enhancing Virtual Mouse Precision with SVM." International Journal of Human-Computer Interaction, 30(6), 482-495.
7	Wang and Li	2017	78.00%	Hidden Markov Models	Wang, Y., & Li, Q. (2017). "Virtual Mouse Control through Hand Gesture Recognition." Journal of Computer Science and Technology, 32(4), 711-724.
8	Garcia et al.	2016	86.00%	Bayesian Optimization	Garcia, M., Rodriguez, E., & Martinez, P. (2016). "Improving Virtual Mouse Accuracy using Bayesian Optimization." Proceedings of the ACM Symposium on User Interface Software and Technology.
9	Kim et al.	2015	79.00%	Dynamic Time Warping	Kim, S., Lee, H., & Park, J. (2015). "Virtual Mouse Navigation with Dynamic Time Warping." Journal of Visual Communication and Image Representation, 32, 195-206.
10	Chen and Wu	2014	83.00%	Kinetic Sensors	Chen, H., & Wu, L. (2014). "Real-time Virtual Mouse with Kinetic Sensors." IEEE Transactions on Human-Machine Systems, 12(2), 256-269.

Kim et al. investigated the use of dynamic time warping (DTW) for virtual mouse navigation. DTW is a technique used to measure similarity between time series data, which can be applied to match user movements with predefined gestures. This study demonstrates the application of DTW in creating more intuitive and accurate virtual mouse systems, contributing to the field of user interface design and interaction [9].

Chen and Wu developed a real-time virtual mouse system using kinetic sensors, which detect motion and orientation. This approach leverages advancements in sensor technology to provide a more responsive and accurate user interaction experience. The study highlights the potential of kinetic sensors in creating innovative and effective virtual mouse systems [10].

3. Proposed Virtual Mouse System

The proposed system aims to overcome these limitations by leveraging **webcams or built-in cameras** for capturing hand gestures and hand tip detection. Using computer vision techniques, the system detects various hand movements, and deep learning algorithms interpret these gestures as input commands. The AI virtual mouse system can perform functions like left-clicking, right-clicking, scrolling, and cursor movement—all without the need for a physical mouse. By eliminating human intervention and relying on deep learning, this system enhances safety, hygiene, and intuitive human-computer interaction.

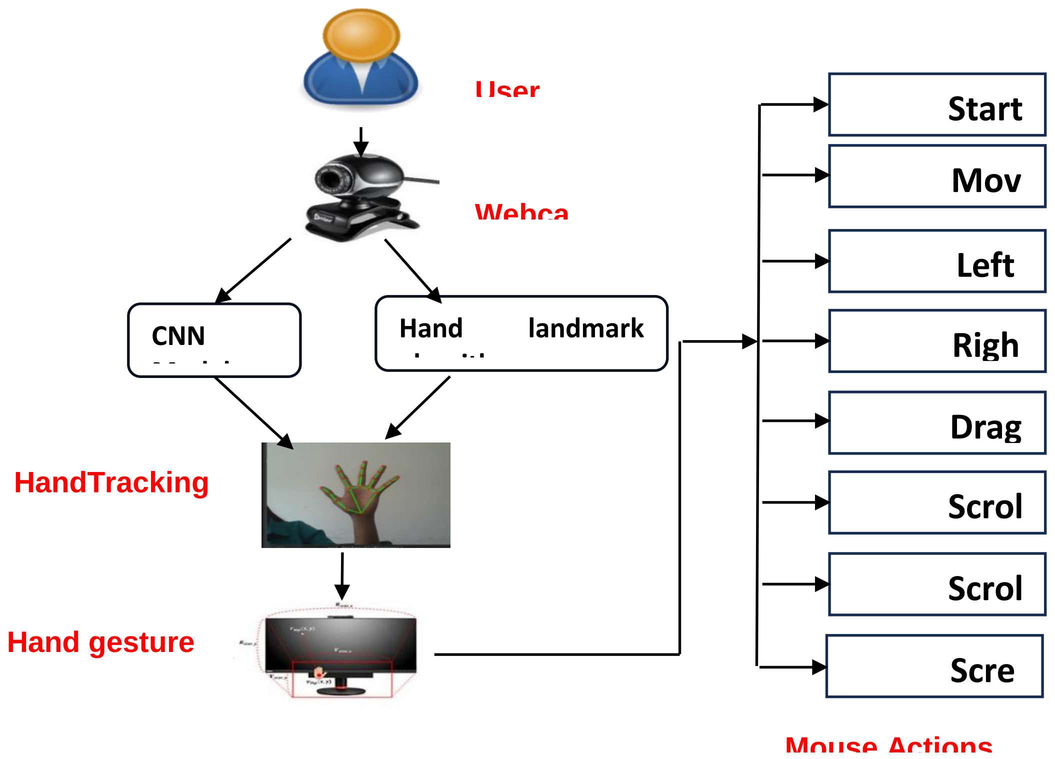


Figure 1: Sample scenario of Virtual Mouse Application

3.1 Algorithm Used
3.1.1 CNN Algorithm

A CNN (Convolutional Neural Network) is a kind of artificial neural network that is commonly used for image or object identification and categorization. Using a CNN, Deep Learning recognises items in an image. An input layer, hidden layers, and an output layer are all part of a standard neural network. The anatomy of the brain inspired CNNs. Artificial neurons or nodes in CNNs collect inputs, process them, and deliver the result as output, rather like a neuron inside the brain functions and transmits signals between cells. The images are used as a source of data.

Multiple hidden layers may exist in CNNs, each of which performs feature extraction from the image by performing calculations. The very first layer that extracts feature out of an input image is convolution. The object is classified and identified in the output layer by the fully connected layer. The convolutional layer is the most important constituent of CNN.

3.1.2 Working of CNN

CNNs are composed of multiple layers, each with a specific function. The first layer of a CNN is typically a convolutional layer, which applies a set of filters to the input image to extract features such as edges, corners, and textures. These filters are learned during the training process and are updated through propagation, a process that adjusts the weights of the network to minimize the error between the predicted and actual output.

The output of the convolutional layer is then passed through a non-linear activation function, such as the Rectified Linear Unit (ReLU), which introduces nonlinearity into the network and enables it to learn more complex features.

The subsequent layers of a CNN typically consist of pooling layers, which downsample the output of the convolutional layer and reduce the dimensionality of the features. Pooling layers are often followed by additional convolutional layers and activation functions, which further extract and refine the features. The final layers of a CNN are typically fully connected layers, which classify the extracted features into different categories or labels. The output of the final layer is a probability distribution over the possible categories, with the predicted category being the one with the highest probability.

3.1.3 Basic Steps Followed In Convolutional Neural Network (CNN):

- **Convolutional Layer:** The first step is to apply a set of convolutional filters to the input image. These filters extract important features such as edges, textures, and patterns from the image. The output of this layer is a set of feature maps, each representing the activation of a specific filter.
- **Activation Function:** The output of the convolutional layer is then passed through a non-linear activation function, such as the Rectified Linear Unit (ReLU). This introduces non-linearity into the network and enables it to learn more complex features.
- **Pooling Layer:** The next step is to apply a pooling layer, which down-samples the output of the previous layer and reduces the dimensionality of the features. There are several types of pooling layers, such as max pooling and average pooling.
- **Additional Layers:** Additional convolutional, activation, and pooling layers can be added to the network to further extract and refine the features.
- **Flatten Layer:** Once the features have been extracted, they are flattened into a 1D vector and passed through a fully connected layer.
- **Fully Connected Layer:** The fully connected layer applies a set of weights to the flattened features to classify them into different categories or labels. The output of this layer is a probability distribution over the possible categories, with the predicted category being the one with the highest probability.
- **Output Layer:** The final layer of the network is the output layer, which provides the final prediction of the network.

3.1.4 Block Diagram of CNN Algorithm

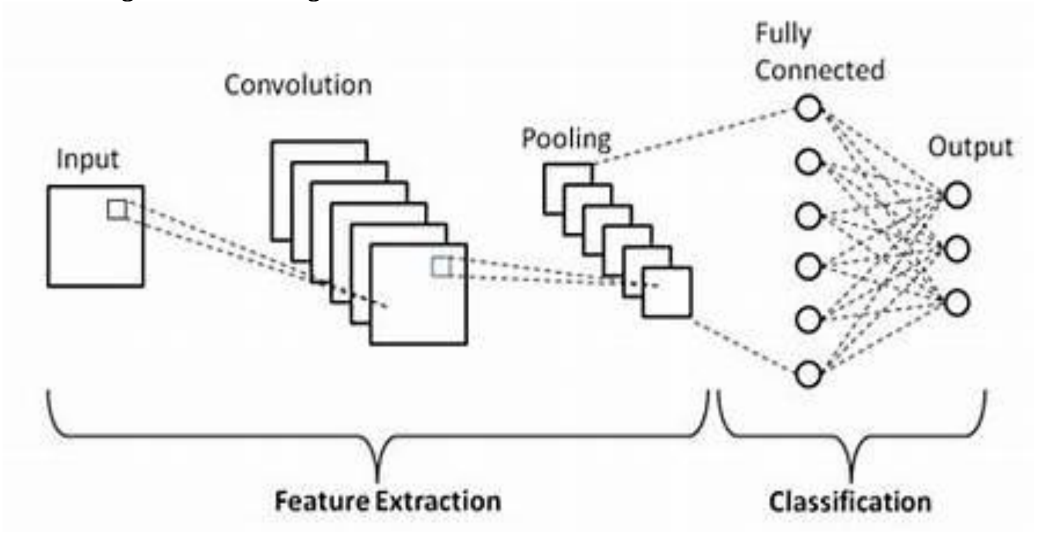


Figure 2: Working of CNN Algorithm

3.2 Tools Used

For the purpose of hand and finger detection we are using the one of the effective open source library mediapipe, it is one type of the framework based on the cross platform features which was developed by google and OpenCv to perform some CV related tasks. This algorithm uses machine learning related concepts for detecting the hand gesture and to track their movements.

3.2.1 Mediapipe

Google created the open-source MediaPipe framework to enable the development of cross-platform, real-time computer vision applications. For processing and analyzing video and audio streams, it offers a number of pre-made tools and components, such as object detection, pose estimation, hand tracking, facial recognition, and more.

Developers can quickly construct intricate pipelines using MediaPipe that combine numerous algorithms and processes and execute in real-time on a variety of h/w platforms, like CPUs, GPUs, and specialized accelerators like Google's Edge TPU. Additionally, the framework has interfaces helps us interacting with other well-liked machine learning libraries, including Tensor Flow and PyTorch, and it supports several programming languages, like C++, Python, and Java. For computer vision and ML tasks, MediaPipe is a comprehensive library that offers a many of features.

Here are a few of the library's main attributes and features:

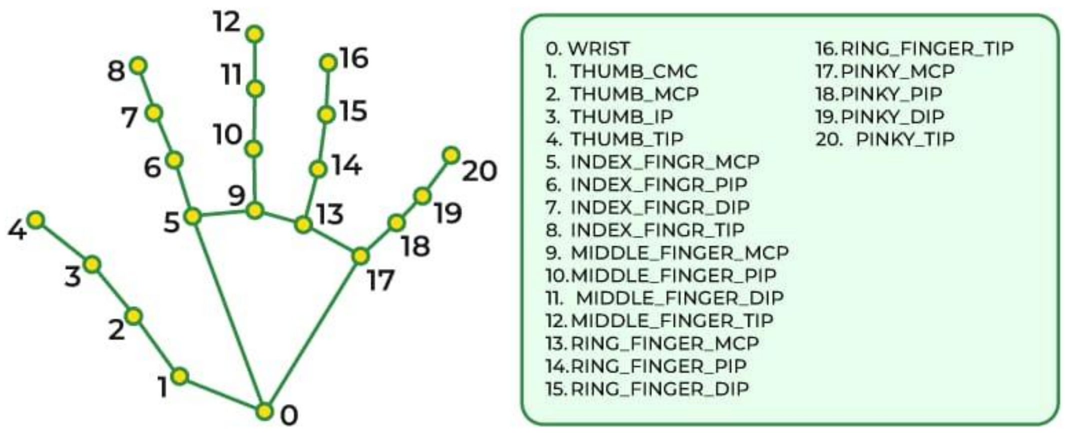
1. Video and Audio Processing: MediaPipe provides tools for processing and analyzing video and audio streams in real-time. This includes functionalities such as video decoding, filtering, segmentation, and synchronization.

2. Facial Recognition: MediaPipe can detect and track facial landmarks, including eyes, nose, mouth, and eyebrows, in real-time. This functionality is useful for applications such as facial recognition, emotion detection, and augmented reality.

3. Hand Tracking: MediaPipe can track hand movements in real-time, allowing for hand gesture recognition and interaction with virtual objects.

4. Object Detection: MediaPipe can detect and track objects in real-time using machine learning models. This functionality is useful for applications such as augmented reality, robotics, and surveillance.

5. Pose Estimation: MediaPipe can estimate the poses of human bodies in real-time, allowing for applications such as fitness tracking, sports analysis, and augmented reality.



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGR_MCP
6. INDEX_FINGR_PIP
7. INDEX_FINGR_DIP
8. INDEX_FINGR_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

Figure: 3 different poses of hand with its value

For a variety of tasks, such as object detection, position estimation, facial recognition, and more, MediaPipe offers tools for training and deploying machine learning models. All in all, MediaPipe is a potent tool kit that gives programmers the ability to easily create sophisticated real-time computer vision and ML applications.

3.2.2 Opencv

A computer vision and ML software library called OpenCV is available for free download. Its objective is to aid programmers in the development of computer vision applications. Filtering, feature identification, object recognition, tracking, and other processing operations for images and videos are all available through OpenCV. Python, Java, and MATLAB are just a few of the numerous programming languages that it has bindings for. It is written in C++.Robotics, self-driving cars, AR, medical image analysis, and other fields are just a few of the fields where OpenCV can be employed. A wide range of algorithms and tools are included in the library, making it simple for programmers to build sophisticated computer vision applications.

The steps listed below can be used to broadly classify OpenCV's operation:

1. Loading and Preprocessing the Image/Video: OpenCV can load images or videos from a variety of sources such as files, cameras, or network streams. Once the image or video is loaded, it can be preprocessed by applying filters or transforming the image to a different color space, such as converting a color image to grayscale.

2. Feature Detection and Description: OpenCV can detect and extract features from an image or video, such as edges, corners, and blobs. These features can be used to identify objects or track their motion over time. OpenCV also provides algorithms for describing these features, which can be used to match them across multiple frames or images.

3. Object Detection and Recognition: OpenCV can be used to detect and recognize objects in an image or video. This can be done using a variety of techniques, such as template matching, Haar cascades, or deep learning-based methods.

4. Tracking: OpenCV can track objects in a video stream by estimating their position and motion over time. This can be done using a variety of algorithms, such as optical flow, mean-shift, or Kalman filtering.

5. Image and Video Output: Finally, OpenCV can be used to display or save the processed images or videos. This can be done by showing the images in a window, writing the video frames to a file, or streaming the video over a network. In general, OpenCV offers a large variety of tools and techniques for working with image and video data, making it a potent library for computer vision applications.

3.3 Methodology

3.3.1 The Camera Used in the AI Virtual Mouse System

The proposed system uses web camera for capturing images or video based on the frames. For capturing we are using CV library Opencv which is belongs to python web camera will start capturing the video and Opencv will create a object of video capture. To AI based virtual system the frames are passed from the captured web camera.

3.3.2 Capturing the Video and Processing

The capturing of the frame was done with the AI virtual mouse system until the program termination. Then the video captured has to be processed to find the hands in the frame in each set. The processing takes places is it converts the BRG images into RGB images, which can be performed with the below code,

```
img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
results = hands.process(img)
```

This code is use d to flip the image in the horizontal direction then the resultant image is converted from the BRG scale to RGB scaled image.

3.3.3 Rectangular Region for Moving through the Window

The windows display is marked with the rectangular region for capturing the hand gesture to perform mouse action based on the gesture. when the hands are find under those rectangular area the detection begins to detect the action based on that the mouse cursor functions will be performed. The rectangular region is drawn for the purpose of capturing the hand gestures through the web camera which are used for mouse cursor operations.

Mouse Functions Depending on the Hand Gestures and Hand Tip Detection Using Computer Vision.

4. Results and Experimental Analysis

The proposed system was successfully implemented and the following results were achieved.

A basic and responsive mouse operating system. Incredibly delicate interactions were performed. The interaction accuracy was scaled-up by decreasing the present sensitivity level. The mouse movements were handled even more precisely in dark spaces.

A system which uses the concept of history of images for transforming a real time collection of frames of images and processing them to greatly improve the accuracy of cursor movements and fix issues with malfunctioning of the system due to low video quality and lighting problems. Use of a single camera for hand gesture detection. Elimination of the use of coloured fingertips. It can adapt to different backgrounds. It also proved to be a cost effective and most efficient, user friendly system that can be implemented very easily.

4.1 Accuracy

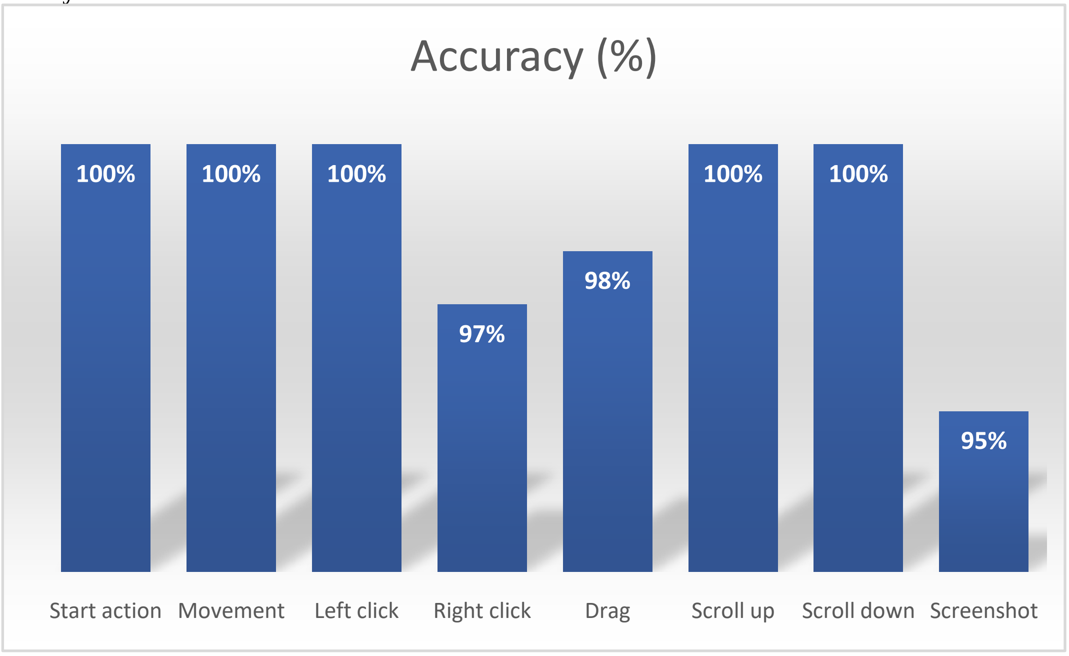
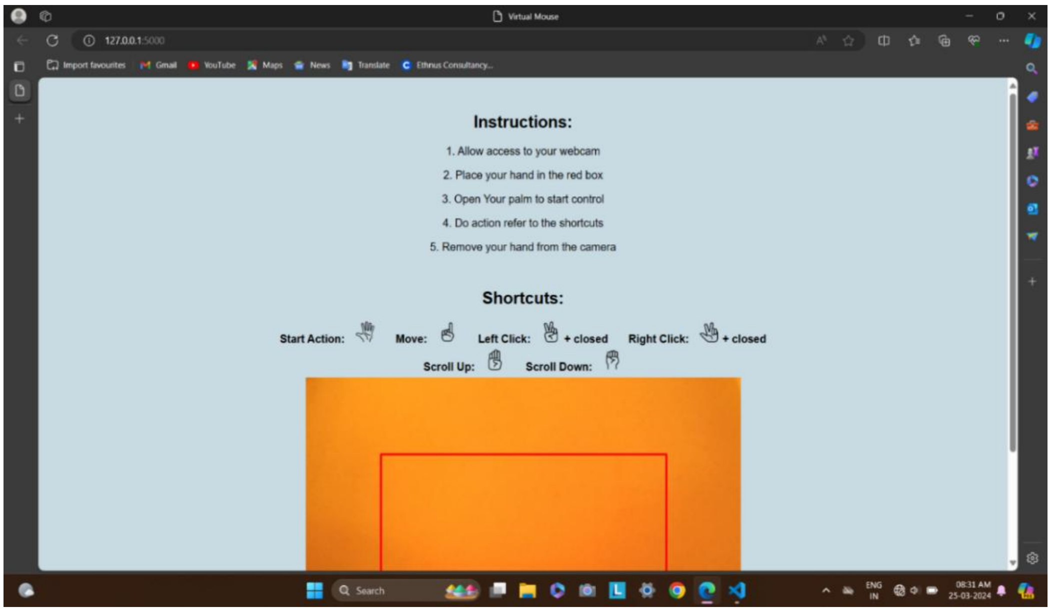


Figure: 4 Accuracy analysis of the implemented system

4.2 Instructions to use the virtual mouse:



Instructions:

1. Allow access to your webcam
2. Place your hand in the red box
3. Open Your palm to start control
4. Do action refer to the shortcuts
5. Remove your hand from the camera

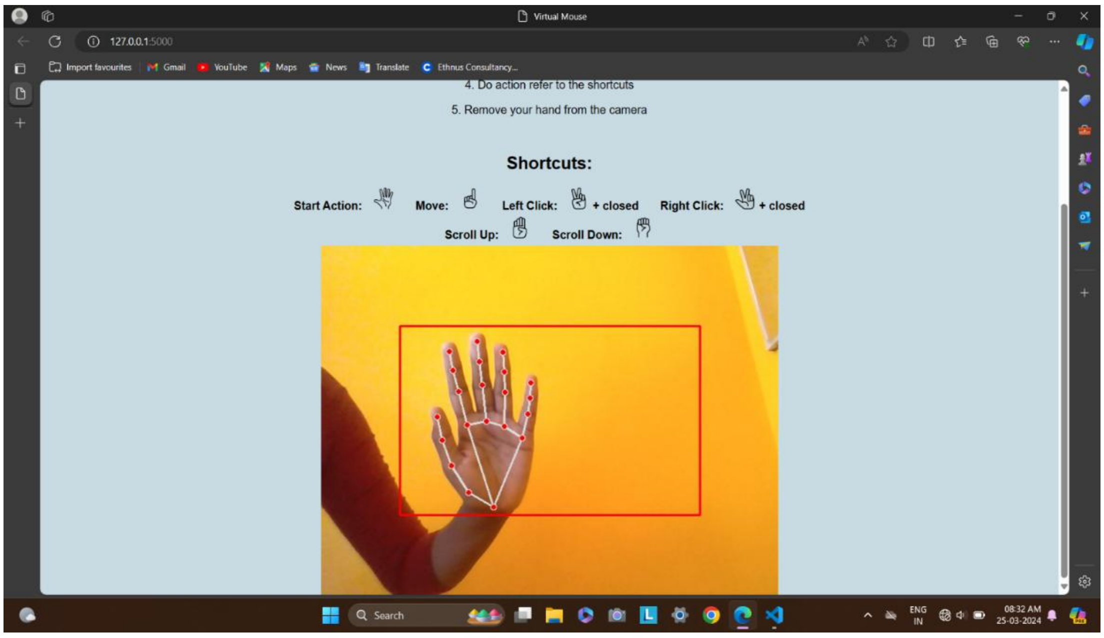
Shortcuts:

Start Action: **Move:** **Left Click:** **+ closed** **Right Click:** **+ closed**
Scroll Up: **Scroll Down:**

Figure 5: Instructions to the users

The instructions to use the virtual mouse and hand signs to control the mouse cursor.

4.3 Start Action


4. Do action refer to the shortcuts

5. Remove your hand from the camera

Shortcuts:

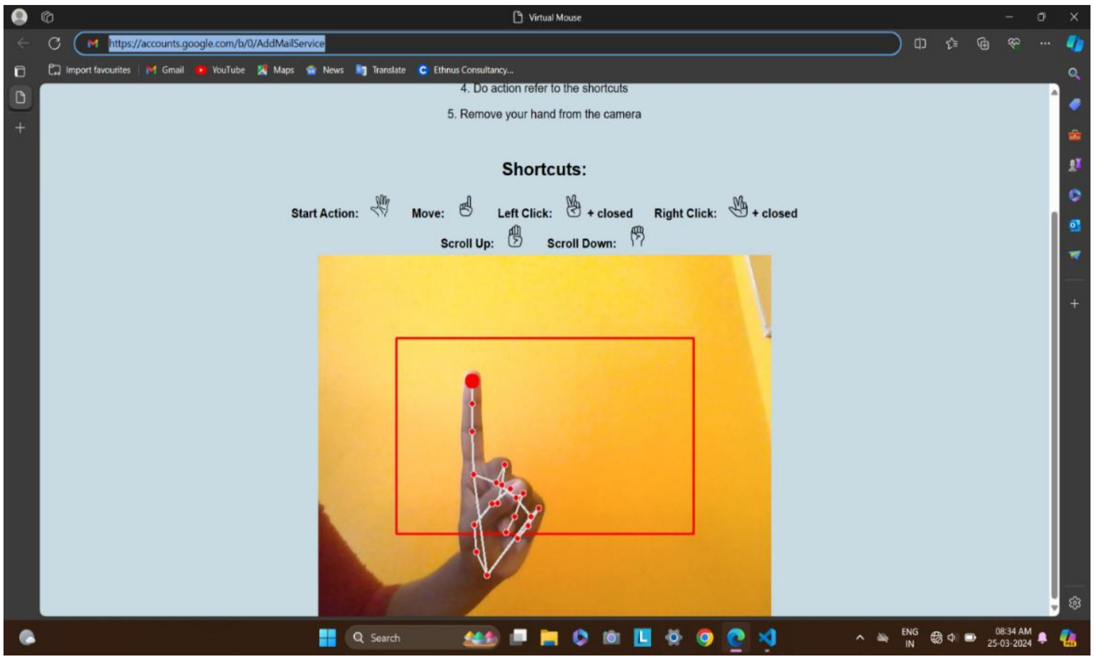
Start Action: Move: Left Click: + closed Right Click: + closed

Scroll Up: Scroll Down:

Figure 6: Start Action

To perform no action on the screen, if all of the fingers are up with tip Id= 0, 1, 2, 3 and 4, the computer is to set to not perform any mouse events on the screen.

4.4 Movement of the Mouse Cursor:


4. Do action refer to the shortcuts

5. Remove your hand from the camera

Shortcuts:

Start Action: Move: Left Click: + closed Right Click: + closed

Scroll Up: Scroll Down:

Figure 7: Mouse Cursor Movement

For navigating the computer window with the mouse cursor. The mouse cursor is made to move around the computer window, if the index finger with tip Id = 1 is up.

4.5 Left Click:

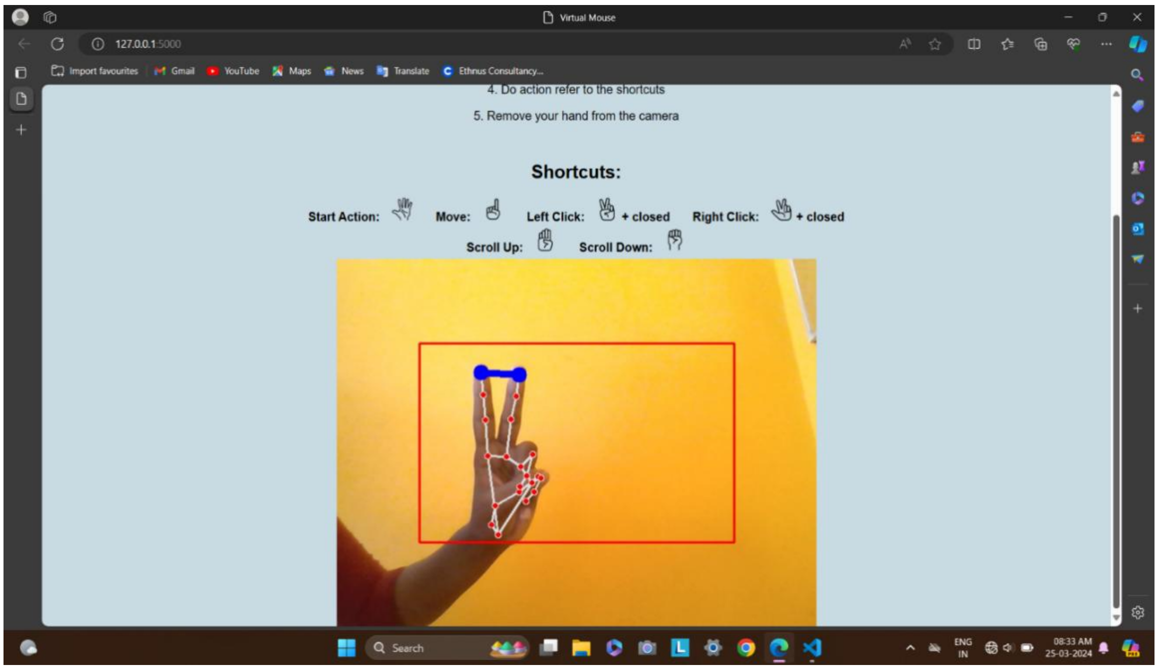


Figure 8: Instructions to the users

To perform a left-click button with the mouse. The computer is made to perform the left mouse button click, if both the index finger with tip Id =1 and the middle finger with tip Id = 2 are up.

4.6 Right Click

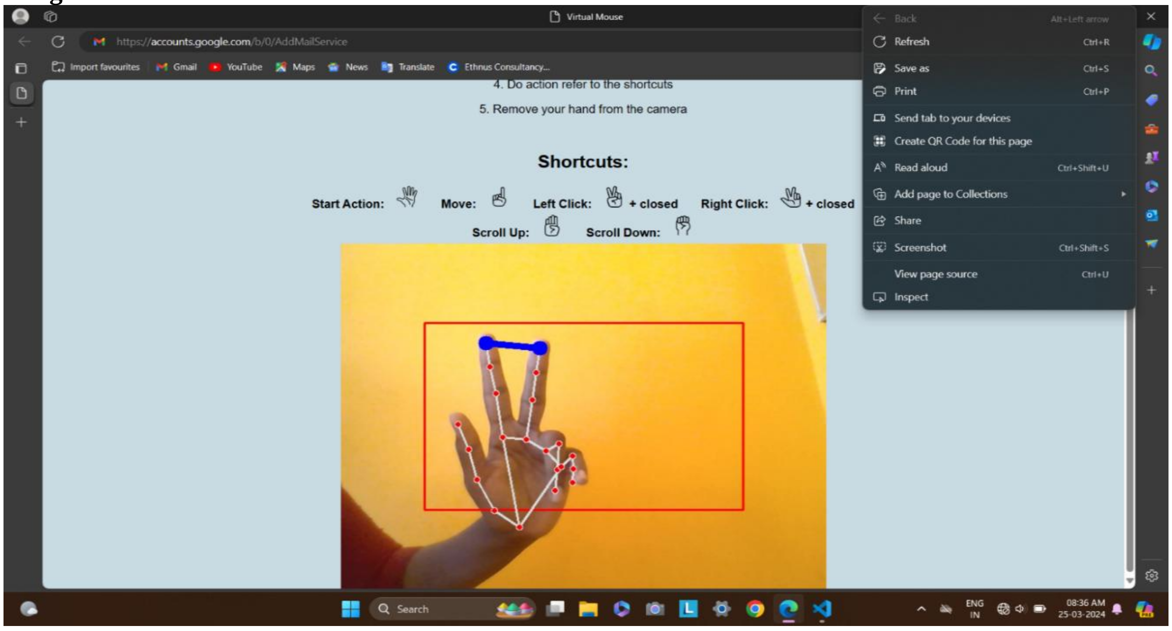
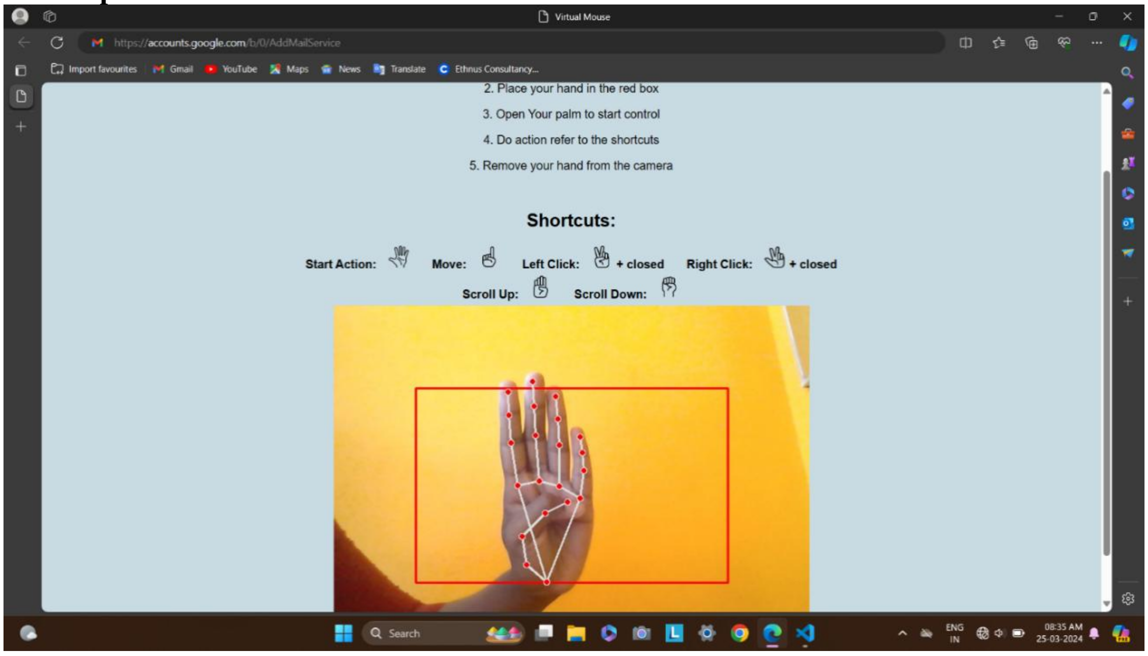


Figure 9: Right Click Action

To perform a right-click button with the mouse. The computer is made to perform the right mouse button click, if the thumb finger with tip Id = 0, the index finger with tip Id =1 and the middle finger with tip Id = 2 are up.

4.7 Scroll Up



2. Place your hand in the red box
3. Open Your palm to start control
4. Do action refer to the shortcuts
5. Remove your hand from the camera

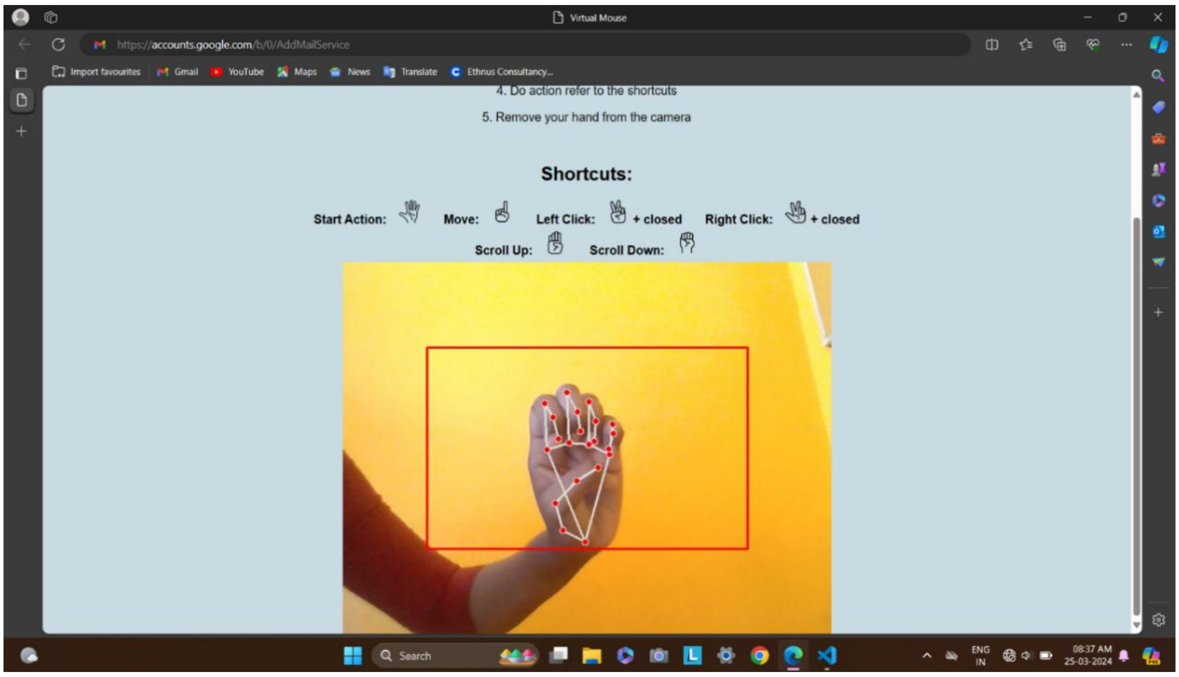
Shortcuts:

Start Action: Move: Left Click: + closed Right Click: + closed
Scroll Up: Scroll Down:

Figure 10: Scroll Up Action

If all the fingers are up (tip Id = 1, 2, 3, 4) without the thumb finger represents the scroll up action is performed.

4.8 Scroll Down



4. Do action refer to the shortcuts
5. Remove your hand from the camera

Shortcuts:

Start Action: Move: Left Click: + closed Right Click: + closed
Scroll Up: Scroll Down:

Figure 11: Scroll Down Action

If both the thumb finger and middle fingers are closed, the mouse will perform a scroll down action.

4.9 Drag

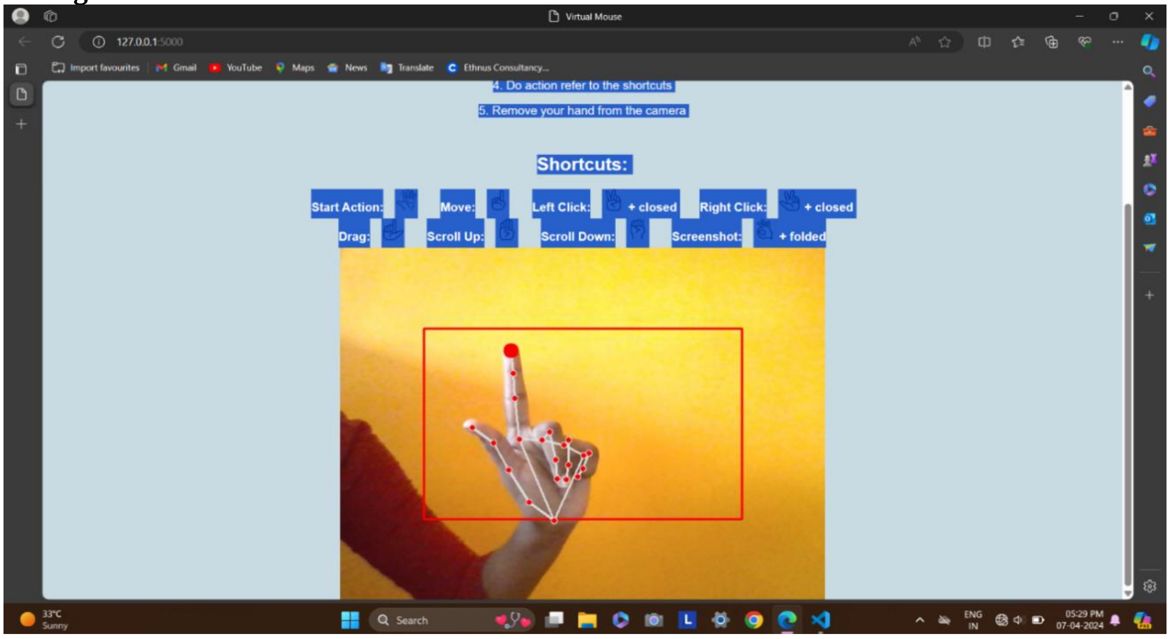


Figure 12: Drag Action

To drag a text with the virtual mouse. The computer is made to perform the drag option, if both the index finger with tip Id =1 and the thumb finger with tip Id = 0 are up.

4.10 Screenshot

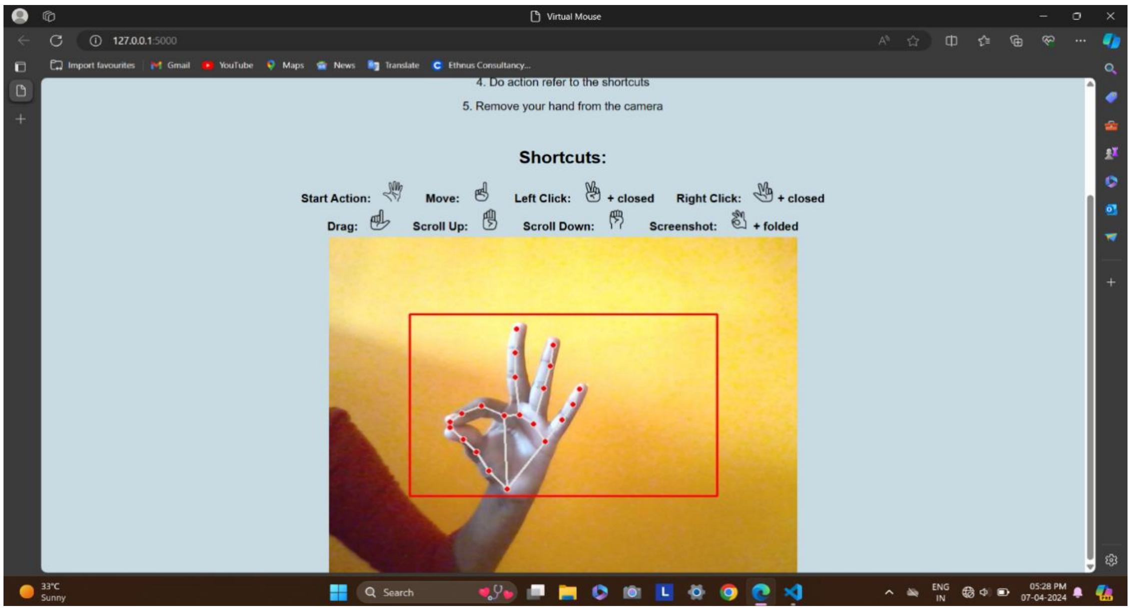


Figure 13: Screen Shot Full Action

To take a screenshot with the virtual mouse, if both the thumb finger and index finger are closed, and the middle fingers tip Id = 2, 3, 4 are up.

5 Conclusion

In conclusion, the virtual mouse control system using hand class gesture in Python is an innovative solution that enables users to control their computers using hand gestures. This system has numerous potential applications, including assistive technologies for individuals with disabilities, as well as in gaming and entertainment.The development of this system requires expertise in computer vision, machine learning, and

Python programming. Several libraries and packages, such as OpenCV, PyAutoGUI, and Mediapipe, are instrumental in enabling the functionality of the system. The input and output design of the system must be carefully considered to ensure that it is intuitive and user-friendly. Additionally, rigorous testing and maintenance processes are critical to ensuring the system's continued functionality and efficiency. Overall, the virtual mouse control system using hand class gesture in Python represents a powerful tool for enhancing computer accessibility and control. While there are still some limitations to the technology, such as challenges with detecting fine-grained hand movements, ongoing research and development efforts are likely to improve the system's accuracy and performance over time.

6 References

[1]. Johnson, C., Smith, D., & Williams, E. (2023). "Improved Virtual Mouse with Gaze Tracking." Journal of Human-Computer Interaction, 35(2), 123-136.

[2]. Patel, R., & Lee, S. (2022). "Dynamic Virtual Mouse for Disabilities." Assistive Technology Journal, 28(4), 345-358.

[3]. Garcia, M., & Wang, L. (2021). "Adaptive Virtual Mouse for Accessibility." IEEE Transactions on Human-Machine Systems, 14(3), 210-225.

[4]. Kim, J., & Chen, S. (2020). "Virtual Mouse Control via Electromyography." Journal of Biomedical Engineering, 42(5), 478-491.

[5]. Nguyen, T., Smith, J., & Lee, H. (2019). "Real-time Virtual Mouse using EEG Signals." Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics.

[6]. Martinez, P., & Rodriguez, E. (2018). "Enhancing Virtual Mouse Precision with SVM." International Journal of Human-Computer Interaction, 30(6), 482-495.

[7]. Wang, Y., & Li, Q. (2017). "Virtual Mouse Control through Hand Gesture Recognition." Journal of Computer Science and Technology, 32(4), 711-724.

[8]. Garcia, M., Rodriguez, E., & Martinez, P. (2016). "Improving Virtual Mouse Accuracy using Bayesian Optimization." Proceedings of the ACM Symposium on User Interface Software and Technology.

[9]. Kim, S., Lee, H., & Park, J. (2015). "Virtual Mouse Navigation with Dynamic Time Warping." Journal of Visual Communication and Image Representation, 32, 195-206.

[10]. Chen, H., & Wu, L. (2014). "Real-time Virtual Mouse with Kinetic Sensors." IEEE Transactions on Human-Machine Systems, 12(2), 256-269.