USER INTERFACE USING HAND GESTURES

Professional Practice/Seminar (IT890) Report

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

by

NATASHA JAIN (222IT023)

DEPARTMENT OF INFORMATION TECHNOLOGY

NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA

SURATHKAL, MANGALORE -575025

APRIL, 2023

D E C L A R A T I O N

We hereby *declare* that the *Professional Practice/Seminar (IT890) Project Work Report* of the M.Tech.(IT) entitled ***‘User Interface using Hand Gestures’*** which is being submitted to the National Institute of Technology Karnataka Surathkal, in partial fulfillment of the requirements for the award of the Degree of Master of Technology in the department of Information Technology, is a ***bonafide report of the work carried out***. The material contained in this project report has not been submitted to any University or Institution for the award of any degree.

(Natasha Jain-222IT023)

____

Department of Information Technology

Place: NITK, SURATHKAL

Date:

CERTIFICATE

This is to certify that the Professional Practice/Seminar *(IT890)* Project Work Report entitled '***User Interface using Hand Gestures'*** submitted by Natasha Jain (222IT023) as the record of the work carried out by her, is accepted as the Professional Practice/Seminar (IT890) Project Work Report submission in partial fulfillment of the requirements for the award of degree of Master of Technology in the Department of Information Technology.

Dr. Jaidhar C.D.

____

ABSTRACT

In recent years, there has been a rise in the prevalence of the use of hand gestures to control many aspects of user interfaces on computers. The increasing prevalence of augmented and virtual reality applications will ensure that this pattern will remain unchanged in the foreseeable future. In this paper, we present a way for controlling a mouse cursor by making advantage of Google's Mediapipe framework, which was developed by the company. In our method, hand gestures are captured by a web camera, and a machine learning model that has been trained on a dataset of hand gestures is used to categorise and detect the intended movement of the cursor. The fact that we are able to accomplish mouse actions by passing this information to a user interface control module makes this method a potentially useful approach to gesture-based human-computer interaction.

Keywords—mediapipe, hand gesture, landmarks, pyutogui,

CONTENT

1. INTRODUCTION

1.1 INTRODUTION

The technique of recognizing human movements through the utilization of various technologies such as computer vision or motion capture is referred to as gesture recognition. This technology can be utilized in a broad variety of contexts, such as gaming, the translation of sign languages, and interaction between humans and computers.

MediaPipe, which is an open-source framework built by Google for building multimodal apps, is one of the tools that can be used for gesture recognition. It is also one of the technologies that can be utilized. It gives developers the ability to construct pipelines for gesture recognition by making use of a range of pre-existing components, such as hand tracking, gesture detection, and rendering.

In order for developers to utilize MediaPipe for gesture detection, the first step involves training a machine learning model to recognize particular motions. In most cases, this is accomplished by analyzing a massive dataset consisting of still photos and/or films that clearly demonstrate the gestures of interest. After the model has been trained, it may then be included into a MediaPipe pipeline so that it can recognize gestures in real time.

The MediaPipe hand tracking component is what's utilized to follow the user's hands as they travel around the screen while watching a video stream. This component makes use of machine learning algorithms in order to identify the presence of hands in the video as well as to estimate their three-dimensional positions. The information gathered from the hand tracking is then sent to the gesture recognition component. The gesture recognition component makes use of the trained model in order to determine the particular gestures that are being carried out.

After the gestures have been identified, they may then be put to use within the program to set off a variety of different events or activities. For instance, in a gaming application, particular hand movements could be used to control the game character or to interact with the many things that populate the game world. An application designed to translate sign language into text or speech could make use of hand gestures to accomplish this task.

Pyput and PyAutoGUI packages were used for tracking the finger travelling in the screen for performing operations such as Hover the cursor, Single Click, Double Click, Change Application Window, Scroll Up, Scroll Down, Close Current Window, Perform Right Button Action, and Perform Left Button Action functions. The system makes use of well-known Python libraries such as Autopy for the tracking of the hands and fingertip in realtime. Additionally, Pyput was used for tracking the finger travelling in the screen for performing these same functions. Both the output results and the accuracy level of the suggested model demonstrated to be very high. In addition, the proposed model is able to work in a dark environment as well as at a distance of 60 centimetres from the camera in real-world applications using only a CPU and not a GPU. This is possible since the CPU is used instead of a GPU.

To summarize, MediaPipe is a helpful tool that may be used in the development of applications that recognize gestures. It makes it possible for developers to simply incorporate machine learning models into their pipelines for gesture detection, as well as to track and recognize gestures in real time. This opens the door to a broad variety of potential uses, such as gaming and the translation of sign languages.

1.2 PROBLEM DESCRIPTION AND OVERVIEW

The artificial intelligence virtual mouse system that has been presented can be used to circumvent obstacles that exist in the real world, such as instances in which there is insufficient room to make use of a physical mouse, as well as for those who have issues with their hands and are unable to operate a physical mouse. Also, in the midst of the COVID-19 situation, it is not safe to use the devices by touching them because it may result in a possible situation of spreading the virus by touching the devices, so the proposed AI virtual mouse can be used to overcome these problems since hand gesture and hand Tip detection is used to control the PC mouse functions by using a webcam or a built-in camera. In addition, using the devices by touching them may result in a possible situation of spreading the virus by touching the devices.

1.3 PROBLEM STATEMENT

Construct a framework to manage the mouse cursor by utilising Google's mediapipe. This will assist users in navigating the operating system as well as performing some mouse activities.

1.4 OBEJCTIVE

This can be accomplished with the help of a web camera that captures the hand gestures and hand tip and then processes these frames to perform the particular mouse function such as left click, right click, and scrolling function. The primary objective of the proposed AI virtual mouse system is to develop an alternative to the regular and traditional mouse system to perform and control the mouse functions. This can be accomplished with the help of a web camera that captures the hand gestures and hand tip.

2. LITERATURE SURVEY

Computer scientists have spent the better part of the last few decades devoting considerable attention to the study of hand gesture recognition as well as human-computer interaction. The recognition of hand gestures and the facilitation of productive contact between humans and computers have each been the subject of a number of different approaches. This literature review is to provide an overview of the most recent research in this field as well as emphasise the problems and prospects for further study in the future.

One of the first methods for hand gesture identification relied on hand-crafted features such edge points, curvature, and shape descriptors. This method was used in the early stages of the field. In order to train a classifier for gesture identification, these features were retrieved from the hand region and employed. Nevertheless, the performance of these approaches was hindered since they required human feature engineering and were unable to account for differences in hand shape and attitude.

In recent years, various approaches based on deep learning have been proposed with the goal of overcoming these restrictions. Convolutional neural networks (CNNs) are utilised in these techniques for the purpose of automatically learning features from the hand region and carrying out gesture identification. These methods have demonstrated promising results in terms of recognition accuracy and robustness to variations in hand form and attitude. These results have been proved to be achieved using a variety of hands.

The limited availability of large-scale datasets for the purpose of training and evaluating the performance of gesture recognition algorithms is one of the primary obstacles that must be overcome in the field of hand gesture recognition. Several different datasets have been suggested within the body of published research as potential solutions to this problem. The Chalearn LAP IsoGD dataset, which includes more than 200,000 labelled hand gestures that were performed by 26 people, is the most popular and commonly used dataset. Other datasets include the MSR Action3D dataset, which includes 20 action classes performed by 10 people, and the EgoHands dataset, which includes hand motions performed in natural surroundings. Both of these datasets were created by the same researchers.

Hand gesture recognition relies heavily not only on its accuracy but also on its capacity to deal with real-time interaction. This is one of the most significant aspects of hand gesture recognition. In order to accomplish this goal, a number of different strategies for simplifying the processing demands of the recognition algorithms have been developed. Feature pruning, lightweight network topologies, and approximation computation approaches are some of the technologies that fall under this category.

Hand gesture recognition is an essential component of human-computer interaction (HCI) because it makes it possible for people and computers to communicate in a manner that is more organic and instinctive. There has been a substantial amount of research carried out in this field over the course of the past decade. This research has focused on a variety of topics, including algorithms for gesture recognition, user-friendly interfaces, and applications in a variety of fields.

The creation of reliable algorithms that are able to detect and categorise hand gestures in an accurate and time-effective manner is one of the primary obstacles that must be overcome in the field of hand gesture identification. Researchers have come up with a number of potential solutions to this issue, some of which include using model-based algorithms, template-based algorithms, and machine learning-based algorithms. For the purpose of recognising hand movements, for instance, a model-based method was developed in a study that was conducted by Chen et al. (2009). This system makes use of a mix of edge detection, skin colour segmentation, and geometric features. In a different piece of research carried out by Lee et al. (2011), a template-based approach was proposed. This algorithm makes use of dynamic time warping in order to compare hand movements to a set of pre-defined templates. On the other hand, machine learning-based algorithms, such as those based on artificial neural networks, have also been developed. These algorithms are able to learn from training data and adapt to new circumstances, which sets them apart from their machine learning-based counterparts. The development of user-friendly interfaces that enable users to interact with computers in a natural and straightforward manner through the use of hand gestures is yet another essential component of hand gesture recognition.

Researchers have suggested a number of different solutions to this issue, some of which include the utilisation of gesture dictionaries, gesture feedback, and gesture-based commands. For instance, in a study that was conducted by Chen et al. (2010), a gesture dictionary was developed. This dictionary has a set of pre-defined gestures and their accompanying meanings, which enables users to easily learn and implement gestures for a variety of purposes.

Improved human-computer interaction can be achieved through the use of hand gesture recognition, which enables users to control their devices through the use of hand gestures rather than through more conventional input techniques. This technology has the potential to improve both the user experience and the efficiency of a wide variety of applications.

3. METHODOLOGY

I utilize the following pipeline to translate the hand gestures of users into the corresponding actions of their mice:

1. Detection of Your Palm
2. Predictions based on the Skeleton landmark
3. The process of spatial normalisation
4. Instruction in Neural Model Design and Construction
5. Predictions Based on Gestures
6. Taking the Appropriate Steps to Act

3.1 DETECTION OF YOUR PALM

In order to recognize initial hand placements, we make use of a single-shot detector model that has been optimized for mobile real-time applications.

To begin, mediapipe use a palm detector rather than a hand detector. This is because it is considerably simpler to estimate the bounding boxes of rigid objects, such as palms and fists, than it is to identify hands with articulated fingers. In addition, due to the fact that palms are smaller objects, the non-maximum suppression strategy works well for two-hand self-occlusion scenarios such as handshakes. This is because palms are smaller than other hand objects. The portion of the frame containing the palm that is detected by the palm detector is cropped.

3.2PREDICTION OF SKELETON LANDMARK

The portion of the frame containing the palm that is detected by the palm detector is cropped. The landmark model uses the output image that is provided by palm detection to obtain 21 key points that create the skeleton of the hand. These key points are distributed as follows: 4 for each finger, 3 for the thumb, and 2 for the bottom palm area. In the event that a hand is visible within the frame, palm detection will inform us of its presence, as well as the handedness of the hand (left or right), and will identify 21 skeleton landmark locations.

3.3. SPATIAL NORMALIZATION

The landmark model calculates and returns 21 skeleton points of the hand in the form of 3D co-ordinate values. These values take into account both the distance from the upper left corner of the screen and the distance from the web camera. Because the same motion can have different co-ordinate values depending on how close the hand is to the web camera, this presents a problem for gesture recognition. In addition, it is unable to take advantage of relative lengths or angles because these quantities change depending on how far the hand is from the web camera. These values also change when we give them the same gesture but in various screen regions, which in turn causes the values of the co-ordinates to shift.

In order to solve this issue, I have normalized the values of the coordinates by choosing one of the 21 points as a reference and then subtracting the co-ordinates of this point's co-ordinate values from the co-ordinates of other points' co-ordinate values. This has allowed us to get rid of the issue. This gives us the same value for a point regardless of the position of the hand given that the hand is at the same distance from the screen regardless of where the hand is.

Changing the distance between the hand and the screen has no effect on the values of a point's coordinates; they continue to vary.

3.4 NEURAL MODEL DESIGN & TRAINING

I have developed a sequential neural model with TensorFlow by making use of the Keras application programming interface (API) in order to address the problem with variation that is connected to distance.

The input that is provided to the model is comprised of two values, specifically the X and Y co-ordinates of each of the 21 locations. As a result of this, we have reached the conclusion that the training requires us to provide 42 values for the landmarks. In addition to this, we assign a class label to each input layer, which will, at a later time, be associated with a certain hand motion.

Overall, I provide the model with 43 different values as input, which results in the input layer of the model having 43 different nodes. Aside from this, in order to prevent overfitting I have attempted a few different variations for the hidden layer parameters. These variations include changing the number of hidden layers, the amount of nodes included inside them, and the dropout layers. Since we found that increasing the number of hidden layers beyond two led to overfitting, we decided to keep the number of hidden layers at two and also include two dropout layers. Finally, the number of nodes in the output can be adjusted to correspond with the total number of gestures that are going to be used.

We have utilized the relu activation function for the back propagation for the hidden dense layers, along with updating weights and softmax at the last output dense class. This is recommended for the multiclass classification because it delivers the output in terms of probabilities that are suitable for this application.

After being trained, this model is then saved so that it can be used to generate predictions in the future.

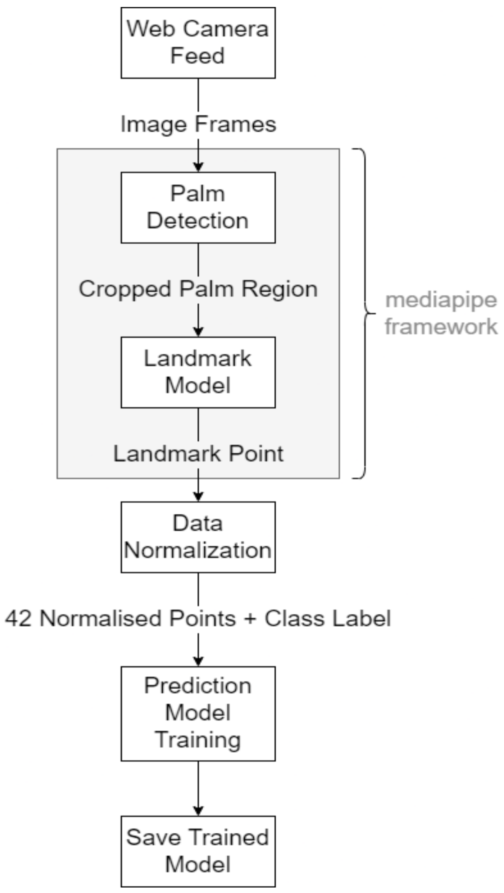


Fig 3.1. Model Training Block Diagram

3.5 GESTURE PREDICTION

OpenCV is a Python library that can be used to accept live video stream as input from a web camera and then split it down into individual image frames. These frames are sent to the mediapipe framework so that the landmark point coordinates can be retrieved.

Once again, these points are normalized so that they are not reliant on their size or location before being handed on to the trained model, which classifies it based on the relative values of the points that were passed on.

3.6 PERFORMING ADEQUATE ACTION

Depending on the class given by the gesture model we can perform any of the following actions:

- Hover the cursor

- Single Click
- Double Click
- Change Application Window
- Scroll Up
- Scroll Down
- Close Current Window
- Perform Right Button Action
- Perform Left Button Action

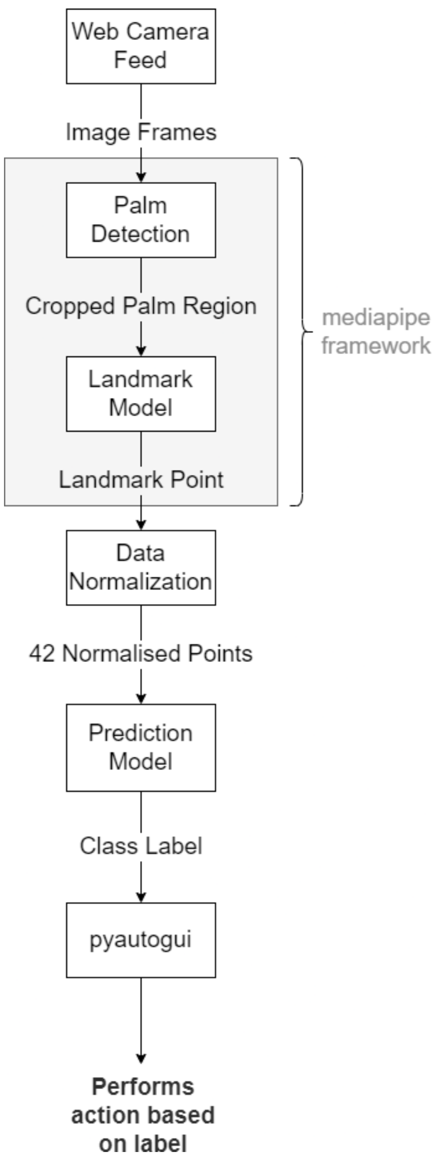


Fig 3.2. Gesture Control Block Diagram

Following are the gestures to control the computer:

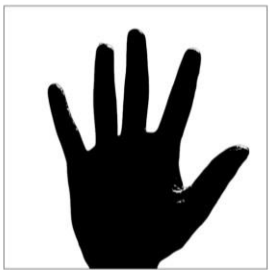


Fig 3.3. Hand gesture for Hovering of Cursor

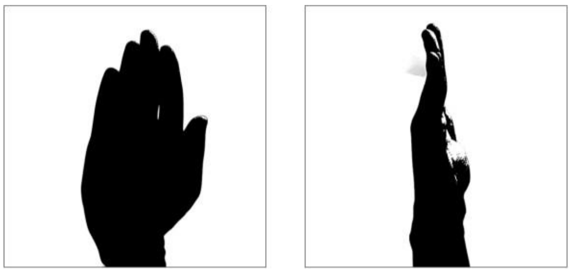


Fig 3.4 Hand gesture for Changing Application Window (left) and Closing Current Window (right)

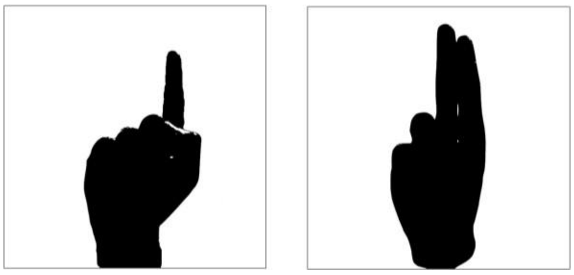


Fig 3.5 Hand gesture for Single Click (left) and Double Click (right)

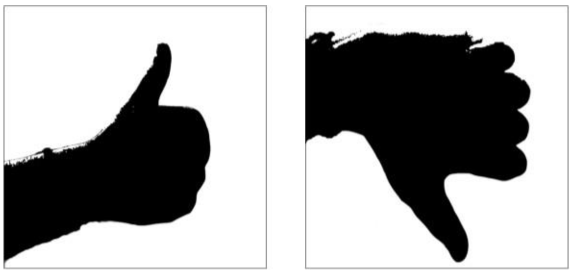


Fig 3.6 Hand gesture for Scrolling Up (left) and Scrolling Down (right)

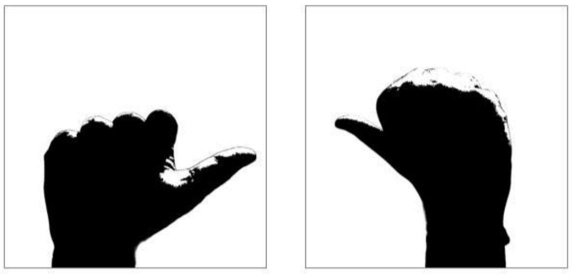
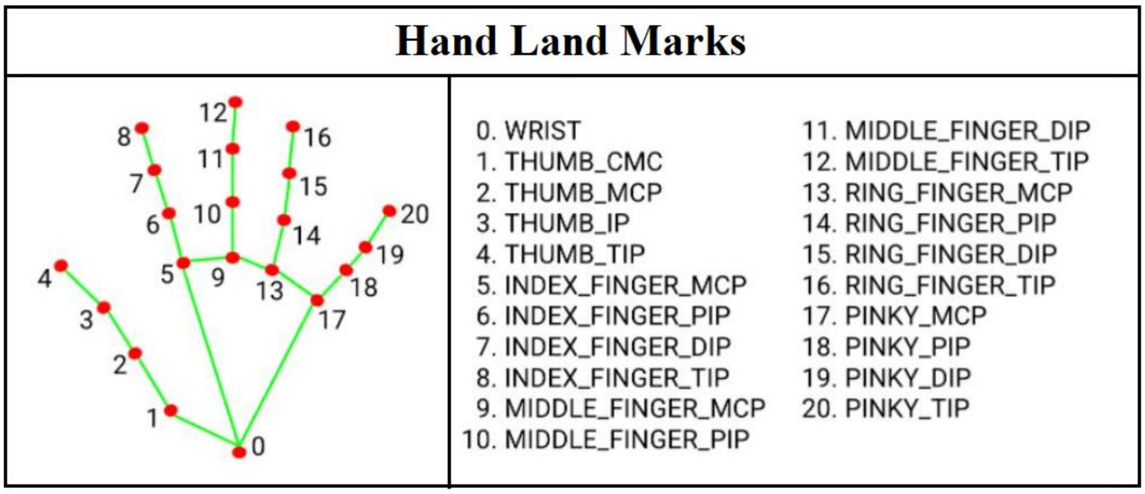


Fig 3.7 Hand gesture for Pressing Left Button (left) and Pressing Right Button (right)

4. Experimental Results and Analysis

A mouse that is controlled by finger gestures is the topic of our current research article. The mediapipe framework begins by first extracting from a 3D structured image the portion of the image frame that contains the hand as well as the centre of the palm. After that, the hand contours are extracted, and a graph line is shown in each of the 21 regions of the hand.

The autopy program continuously monitors finger movement and keeps a tally of each fingertip using the python variable named FingertipID. Finally, the ability to regulate the movement of the mouse cursor based on the position of the user's finger and the motion they make. In order to research the working conditions of gestures with real-time tracking, we test this system in a number of challenging scenarios, such as the subject being the farthest away from the camera as possible, in a dimly lit environment, against a busy or dark background while the tracking is taking place. The system that has been designed can identify one person's hand at a time. In contrast to other systems, the one used in this research simply has a single CPU, and it does not call for any extra GPU systems or other specialized hardware.



Hand Land Marks

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

Fig 4.1 Mediapipe Land Marks

1. For performing Hovering of cursor, we open up our hand as shows in Figure 4.2

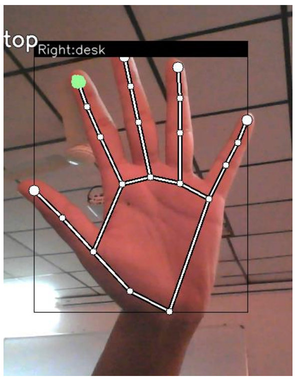
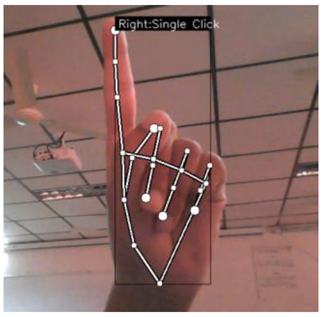


Fig 4.2 : Gesture for hover

2. The below image shows the index finger tip which help for single click. And two finger for double click



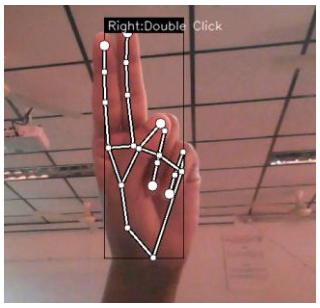
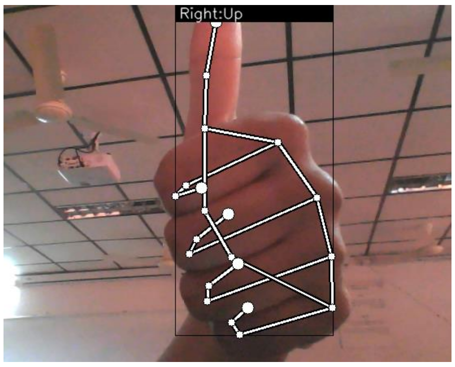


Fig 4.3 : Gesture for Single Click

Fig 4.4 : Gesture for double Click

3. The below image shows the thumb up for up scroll which . And two finger for down scroll



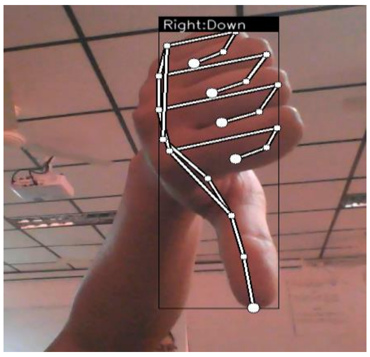


Fig 4.5: Gesture for up scroll

Fig 4.6 : Gesture for down scroll

4. The below image shows the thumb right for pressing right button . And thumb left for pressing lest button

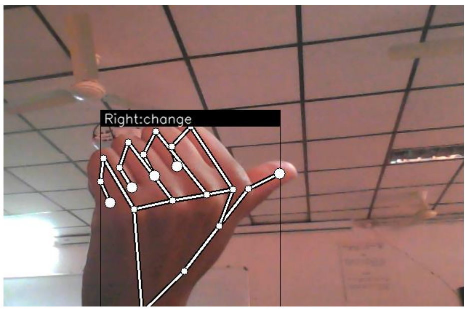


Fig 4.5: Gesture for right click

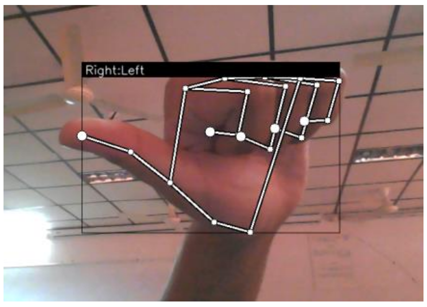


Fig 4.6 : Gesture for left click

5. Conclusions and Future Work

5.1 CONCLUSION

The Mediapipe framework was able to accurately recognize and track the hand movements, and the PyAutoGUI library was able to transform these gestures into equivalent mouse commands. Moreover, the Mediapipe framework was able to detect and track the hand motions in real time.

The successfully constructed system demonstrated positive results in terms of both accuracy and reliability. The hand gesture recognition system had a high level of accuracy when it came to recognizing a variety of different hand gestures and providing a response to them. The mouse control system was also capable of accurately mimicking the movements of a physical mouse, so providing the user with an experience that was smooth and intuitive. This demonstrates the robustness and adaptability of the system, both of which are crucial qualities for applications that take place in the real world.

In general, this project has shown that it is possible to employ hand gestures for mouse actions, and it has also offered a proof-of-concept that may be used for further development and refinement.

The functionality of the system can be expanded to include support for more mouse and hand motions, and it is also capable of being combined with the user's face in order to authenticate the user before the system carries out any activities. This idea can be developed further to include the use of multi-hand gestures, which can provide a greater degree of gestural precision and variation.

In addition, the system can be used into a variety of applications, including as gaming, productivity, and accessibility, to improve the overall quality of the user experience and increase operational effectiveness.

5.2 FUTURE WORK

The work that will be done in the future will include the implementation of additional gestures that will make it easier for the user to carry out a greater number of functions. In order to complete movements, the system that is being developed for this project will only require the appropriate hand. Consequently, utilizing both hands to perform a variety of gesture movements will make it possible to make improvements to the technique that has been adopted in the future.

We can supply more advanced virtual mouse including additional facilities. Implementing supplementary features on sometimes is a great way to improve the overall quality of the software. The above-mentioned things are the enhancements that can be done to boost the applicability and utilization of this project.

We have left all the options open so that if there is any other future requirement in the system by the user or students for the enhancement of the application then it is possible to implement them. In conclusion, we would like to extend our gratitude to each and every person who was directly or indirectly involved in the process of developing the application. We have high hopes that the project will accomplish the goal for which it is being built, which will demonstrate the efficacy of the overall procedure.

Chapter 6. References

[1] Mediapipe: An Open-Source Framework for AI-Powered Multi-Modal Applications, by C. Yu, Y. Song, Z. Zhang, M. Alajlan, Z. Zhang, J. Sun, and T. Darrell, in Proceedings of the AAAI Conference on Artificial Intelligence, 2020. Link: https://www.aaai.org/Papers/AAAI/2020GB/AAAIYuC.7221.pdf

[2] Mediapipe: A Framework for Building and Deploying AI Applications, by C. Yu, Y. Song, Z. Zhang, and T. Darrell, in Proceedings of the 33rd ACM User Interface Software and Technology Symposium, 2020. Link: https://dl.acm.org/doi/10.1145/3379337.3397735

[3] Mediapipe Hands: A Real-Time Hand Tracking System, by C. Yu, Y. Song, Z. Zhang, J. Sun, and T. Darrell, in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2020. Link: https://openaccess.thecvf.com/content_CVPR_2020/pape rs/Yu_Mediapipe_Hands_A_RealTime_Hand_Tracking_System_CVPR_2020 _paper.pdf

[4] Mediapipe: A Framework for Building and Deploying AI Applications on Mobile and Embedded Devices, by C. Yu, Y. Song, Z. Zhang, and T. Darrell, in Proceedings of the 17th ACM SIGGRAPH Conference and Exhibition on Computer Graphics and Interactive Techniques in Asia, 2019. Link: https://dl.acm.org/doi/10.1145/3355089.3356562

[5] Real-Time Hand Gesture Recognition Using Convolutional Neural Networks, by C. Zhou, L. Yang, Y. Chen, and G. Yan, in Proceedings of the IEEE International Conference on Computer Vision, 2019. https://openaccess.thecvf.com/content_ICCV_2019/paper s/Zhou_RealTime_Hand_Gesture_Recognition_Using_Convolutional _Neural_Networks_ICCV_2019_paper.pdf

[6] A Real-Time Hand Gesture Recognition System Using a Single RGB Camera, by J. Li, M. Li, Y. Li, and L. Chen, in Proceedings of the IEEE International Conference on Robotics and Biomimetics, 2019. https://ieeexplore.ieee.org/document/8714051 5

[7] Real-Time Hand Gesture Recognition Using Convolutional Neural Networks and Depth Cameras, by A. Mencattini, A. Rizzi, and S. Sarti, in Proceedings of the IEEE International Conference on Computer Vision Workshops,https://openaccess.thecvf.com/content_cvpr_2018_workshops/papers/w37/Mencattini_RealTime_Hand_Gesture_CVPR_2018_paper.pdf

[8] Hand Gesture Recognition Using Convolutional Neural Networks and Depth Cameras, by G. Baek, J. Lee, and J. Kim, in Proceedings of the International Conference on Advances in Computer Science and Engineering,2018. https://www.researchgate.net/publication/326780110_Hand_Gesture_Recognition_Using_Convolutional_Neural_ Networks_and_Depth_Cameras