iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 11 **Issue:** III **Month of publication:** March 2023

DOI: https://doi.org/10.22214/ijraset.2023.49314

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 11 Issue III Mar 2023- Available at www.ijraset.com

Wireless Cursor Controller: Hand Gestures for Computer Navigation

Om Mapari[1], Mahesh Shinde[2], Shubham Shigwan[3], Tushar Ambatkar[4]
[1, 2, 3, 4]*Information Technology, Pune Institute of Computer Technology, Pune, India*

Abstract: ***This research paper presents a wireless cursor controller system that uses hand gestures to navigate and control a computer's cursor without the need for a traditional mouse. The system employs a fixed position, inexpensive webcam with a high-quality recording feature installed on the top of a computer monitor or a fixed laptop camera. It uses Media-Pipe package for hand tracking and OpenCV for recording motions using computer vision. The system is capable of recognizing hand gestures to perform actions such as left-click, right-click, drag and drop, scroll, and adjust volume or brightness. The proposed system is expected to provide a simple and natural way of communicating for people with hand problems, as well as for those who want to interact with computers more efficiently.***

Keywords: OpenCV, Human-Computer Interaction (HCI), Media Pipe, Python, Gesture

I. INTRODUCTION

In the current era, technological advancements have led to the development of virtual reality devices that have become a part of our daily life. However, these devices still require physical input devices, such as traditional mice, which can sometimes be cumbersome to use. This research paper proposes a wireless cursor controller system that eliminates the need for a traditional mouse by utilizing hand gestures to control the computer's cursor. The system uses computer vision technology and a camera to track hand movements and perform mouse-related tasks. This system is expected to improve the interaction between humans and computers by providing a more natural way of communication that mimics human gestures. Furthermore, the system is expected to be particularly useful for people with hand problems, as they can use hand gestures to control the computer's mouse functionalities. The proposed system's implementation involves using the Python programming language and various libraries such as OpenCV, Media-Pipe, Autoy, Pyput, and PyAutoGUI. The rest of the paper will discuss the system's design, implementation, and evaluation, as well as its potential impact on the field of human-computer interaction.

II. LITERATURE SURVEY

Virtual mouse technology has become increasingly popular over the years and is being widely used in various applications such as gaming, virtual reality, and human-computer interaction. In this literature survey, we will be discussing various studies related to virtual mouse technology

[1]"Virtual cursor Implementation using OpenCV" by Kollipara Sai et al. (2019) proposed detecting fingers using color variations in the image. After recognizing the color of the hand, it highlights the hand using green color, but multiple detections are not possible. [2]"Virtual Mouse using Hand Gesture" by Abhilash S S et al.(2018) proposed detecting fingers using color variations in the image and applied preprocessing on the image to improve the accuracy of finger detection. [3]"A Vision-Based Application For Virtual Mouse Interface Using Hand Gesture" by Sankha Sarkar et al.(2018) keeps track of fingers and tells us which finger is up and which one is down. When we fold the finger or put the finger down, the lines are also down, and the corresponding graph line is down. [4]"Real-time virtual mouse system using RGB-D images and fingertip detection" by Dinh-SonTran et al. (2020) proposed hand identification and segmentation with a Microsoft Kinect Sensor version 2 depth picture, K-cosine Corner Detection for fingertip, border tracing algorithm tracking, and locking the target individual. However, the accuracy of the screen degrades as the number of persons increases. [5]"Video-Based Hand Gesture Detection System Using Machine Learning" by Manjunath R Kounte et al. (2020) used CNN and Machine Learning.

For faster calculations of neural networks, they employed the Nvidia Jetson Nano kit. It is a project focused on speed and efficiency, with a hardware-efficient dynamic gesture detection system.

Virtual mouse technology has a wide range of applications, and these studies provide valuable insights for further research and development in this field.

III. PROPOSED METHODOLOGY

A. Importing The Required Libraries

The necessary libraries for the project, including OpenCV, PyAutoGui, and MediaPipe, are imported into the Python program after being properly installed on the computer system. These are the only libraries needed for the project.

1) *OpenCV:* OpenCV (Open-Source Computer Vision Library) is an open-source computer vision and machine learning software library that is designed to help developers create computer vision applications quickly and efficiently. It is widely used for real-time image processing, object detection, and video analysis. In this project, OpenCV is used for the real-time capturing of hands using a camera.
2) *Media Pipe:* MediaPipe is a framework developed by Google that offers ready-to-use, customizable, and scalable machine learning (ML) solutions for different applications. It provides a cross-platform framework for building multimodal applied ML pipelines. In this project, MediaPipe is used for obtaining hand landmarks from the captured frames.
3) *Py Auto Gui:* PyAutoGui is a Python library that can be used for GUI automation tasks, such as controlling the mouse and keyboard, taking screenshots, and generating keyboard and mouse events. In this project, PyAutoGui is used to simulate mouse and keyboard events based on the hand landmarks obtained from MediaPipe.

B. Capturing the frames and Processing

OpenCV is a computer vision library that is utilized for real-time capturing of hand movements using a camera. The function cv2.VideoCapture(0) is utilized for capturing frames/images from the camera, which are subsequently stored. This creates a VideoCapture object that reads the input video frame by frame.

C. Assigning Hand Landmark Model

In this project, the MediaPipe Hand Landmark Model is used for obtaining the landmarks of the hands from the captured video frames. This model provides a set of pre-trained neural networks that can be used to recognize the hand landmarks in real-time. the model is capable of detecting 21 hand landmarks, including the fingertips, joints, and palm centre. To use the MediaPipe Hand Landmark Model, the python program loads the pre-trained model using the MediaPipe framework. This framework provides an easy-to-use API for loading and running the model. Once the model is loaded, the program captures the video frames from the camera and passes them through the model for landmark detection. The model returns the landmarks for each hand in the frame, which are then used for controlling the mouse movements. The classify_hands() method is an essential part of the hand gesture recognition system. This method takes as input the hand landmarks obtained using the MediaPipe library and then uses them to classify the hand gesture. First, the method checks if there are any hands in the current frame. If there are multiple hands, the method selects the one with the highest confidence score. The confidence score indicates the probability that the detected hand belongs to the right or left hand. The method uses this score to determine whether the hand belongs to the left or right side. After determining the side of the hand, the method uses two separate HandRecog objects, namely handmajor and handminor, to classify the gesture. The handmajor object is used to identify major gestures such as open hand, closed hand, and thumbs up, while the handminor object is used to identify minor gestures such as finger movements and subtle hand movements.

D. Defining and Recognizing Gesture (Gesture Recognition Algorithm)

For checking the gesture, we need to check all figure state means they are open or not i.e finger is bended or straight because the gesture are defined on the finger state. Set all five fingers by computing ratio of distance between fingertip, middle knuckle, base knuckle. See below landmark points for clarification. The set_finger_state function in the GestureRecognition Algorithm is used to determine the current state of each finger in a given hand by computing the ratio of the distance between the fingertip, middle knuckle, and base knuckle. The function sets the finger attribute of the HandRecog object to an integer representation of the current finger state.

TABLE I
GESTURE BINARY ENCODINGS

Finger	Binary Code (state)	Number (Enum)
Pinkie Finger	00001	1
Ring Finger	00010	2
Middle Finger	00100	4

Index Finger	01000	8
Thumb	10000	16

Using this chart, one can convert between binary and decimal representations of numbers, or associate numbers with specific fingers on a hand.

The method then loops through the points list and computes the distance between the fingertip and middle knuckle (dis1) and the distance between the middle knuckle and base knuckle (dis2). If dis2 is zero (which should never happen). Otherwise, ratio is set to dist1 / dist2, rounded to one decimal place.

TABLE III
HAND LANDMARK POINTS TO BE CONSIDER FOR CALCULATING DIS1 AND DIS2

Finger	Dis1 [From, To]	Dis2 [From, To]
Thumb	---	---
Index Finger	[8, 5]	[5, 0]
Middle Finger	[12, 9]	[9, 0]
Ring Finger	[16, 13]	[13, 0]
Pinkie Finger	[20, 17]	[17, 0]

The finger attribute is then left-shifted one bit to make room for the next finger state. If ratio is greater than 0.5, it means that the finger is up, so the least significant bit of finger is set to 1 by performing a bitwise OR operation with 1.

At the end of the loop, the finger attribute represents the binary state of all the fingers, with the thumb as the least significant bit and the pinkie finger as the most significant bit. The method does not return anything but sets the finger attribute of the instance.

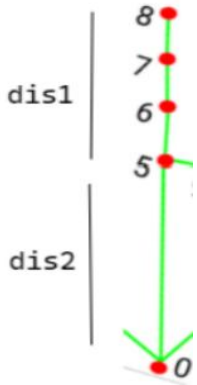


Fig. 1 Index Finger Landmarks

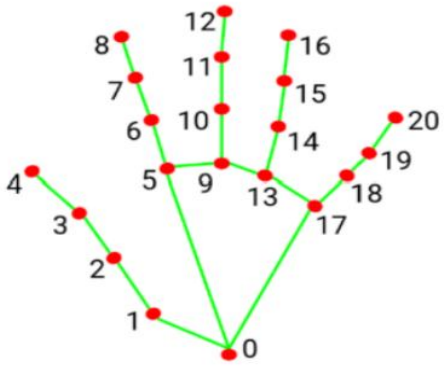


Fig. 2 Hand Landmarks and their Indices

dis1 = distance between fingertip and middle knuckle.
dis2 = distance between middle knuckle and base knuckle
$distance^2 = (x_2 - x_1)^2 + (y_2 - y_1)^2$
ratio = dist1/dist2
Now we are maintaining a binary number to preserve the state of finger i.e. which is up and which is down (bend). {00000}
if ratio > 0.5 : # if that finger is up
self.finger = self.finger | 1 # setting current finger bit

The get_gesture method is used to determine the gesture being made by the hand in the current frame, based on the finger state computed by the set_finger_state method.

1) The method first checks if the hand_result is None, which means that no hand is detected in the current frame, and returns the Gest.PALM enum value in this case.
2) Next, it checks if the finger state matches either Gest.LAST3 or Gest.LAST4, which correspond to the last three or last four fingers being extended. If the distance between the tip of the thumb and the tip of the index or middle finger is less than 0.05, the method checks the hand_label value to determine whether the gesture is a minor or major pinch, and returns the appropriate Gest.PINCH_MINOR or Gest.PINCH_MAJOR enum value.

3) If the finger state matches Gest.FIRST2, which corresponds to the first two fingers being extended, the method checks the ratio of the distance between the tip of the thumb and the tip of the index finger to the distance between the tip of the thumb and the tip of the middle finger. If this ratio is greater than 1.7, the method returns the Gest.V_GEST enum value. Otherwise, if the change in the z-coordinate of the index finger tip and the middle finger tip is less than 0.1, the method returns Gest.TWO_FINGER_CLOSED, indicating that the thumb and index finger are closed together. Otherwise, it returns Gest.MID, indicating that the gesture is somewhere in between the Gest.TWO_FINGER_CLOSED and Gest.V_GEST gestures.

4) Finally, the method updates the frame_count, ori_gesture, and prev_gesture attributes based on the current gesture. If the current gesture is the same as the previous gesture, the frame_count is incremented. Otherwise, the frame_count is reset to 0, and the prev_gesture is updated to the current gesture. If the frame_count reaches 5 or more, the ori_gesture is set to the current gesture, and returned. Otherwise, the previous original gesture is returned.

E. Controlling the System through Gesture (For Smooth Scrolling)

In our research, we aim to implement various mouse functionalities, such as drag, left-click, right-click, and moving the mouse pointer, based on different gestures detected.

1) *Cursor Movement:* For instance, in the case of cursor movement, we detect a V gesture, where both INDEX and MID fingers are up, and position the mouse cursor at the midpoint of the V gesture. To stabilize the mouse pointer, we convert the coordinates of the midpoint of the V gesture, obtained from the mediapipe library, into screen coordinates by multiplying them with the screen size. We then calculate the distance between the previous mouse coordinates and the current mouse coordinates on the screen using the formula:

$$\text{distance} = (x - x\_old)^2 + (y - y\_old)^2$$

If the distance is less than 25 pixels, we do not move the mouse coordinates, and the current position of the mouse remains the same.

x, y = x_old, y_old

However, if the distance is less than 900 pixels and greater than 25 pixels, we move the mouse coordinate in the direction of the mouse by a ratio of 0.07.

x, y = x_old + (x- x_old) * 0.07, y_old + (y-y_old) * 0.07

If the distance is greater than 900 pixels, we move the mouse coordinate by a ratio of 2.1.

x, y = x_old + (x- x_old) *2.1, y_old + (y-y_old) * 2.1

2) *Left Click:* we perform a left-click using the pyautogui library when the previous gesture detected was a V gesture, and the current gesture is a MID gesture.

3) *Right Click:* Similarly, we perform a right-click using the pyautogui library when the previous gesture detected was a V gesture, and the current gesture is an INDEX gesture.

F. Flow diagram of proposed Wireless Cursor Controller

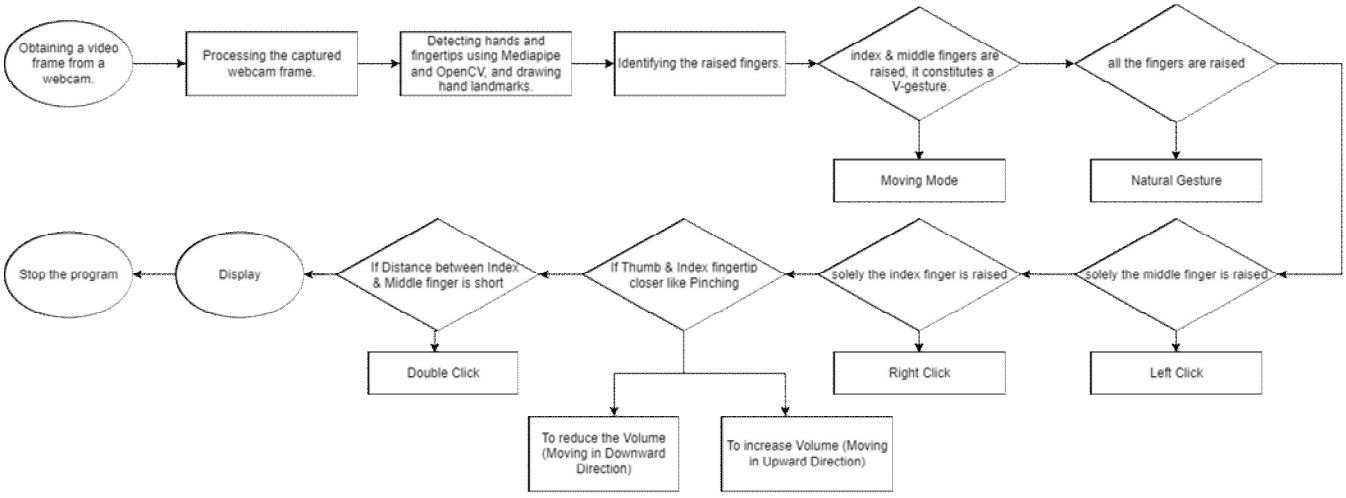


Fig. 3 Flow diagram of proposed Wireless Cursor Controller

IV. EXPERIMENTAL RESULTS

The proposed system presented in this research paper is capable of performing mouse functionality using finger-based gestures. Hand detection and recognition are performed using the OpenCV library, while the Mediapipe library is used to obtain a 21-landmark point model. The system is able to detect gestures through the use of mathematical logic and distance formula. The various mouse controls are executed through different gestures, as follows:

1) *Mouse Cursor Moving Around The Computer Screen:* To move the mouse cursor around the screen, the index finger with ID = 1 and middle finger with ID = 2 must be raised to form a 'V' shape.
2) *Mouse Left Button click:* To perform a left mouse button click, the index finger with ID = 1 and middle finger with ID = 2 must be raised, with the middle finger slightly bent down.
3) *Mouse Right Button click:* To perform a right mouse button click, the index finger with ID = 1 and middle finger with ID = 2 must be raised, with the index finger slightly bent down.
4) *Mouse Double click:* To perform a double click, the index finger with ID = 1 and thumb with ID = 0 must be raised and brought together.
5) *Drag and Drop:* To perform drag and drop, an open fist is used to select contents, while a fist movement is used to drag the selected content and an open fist is used to drop the content to a specific location on the computer.
6) *Scrolling on Computer Screen:* To scroll on a computer screen, dynamic gestures are used for horizontal and vertical scroll. If the index finger with ID = 1 and thumb with ID = 0 form a pinch gesture, scrolling is performed on the screen. The speed of the scroll is proportional to the distance moved by the pinch gesture from the start point. Vertical and horizontal scrolls are controlled by vertical and horizontal pinch movements, respectively.
7) *Brightness Control:* To control the brightness, dynamic gestures are used. If the index finger with ID = 1 and thumb with ID = 0 form a pinch gesture, the brightness controlling cursor moves. The rate of increase/decrease of brightness is proportional to the distance moved by the pinch gesture from the start point.
8) *Volume Control:* To control the volume, dynamic gestures are used. If the index finger with ID = 1 and thumb with ID = 0 form a pinch gesture, the volume controlling cursor moves. The rate of increase/decrease of volume is proportional to the distance moved by the pinch gesture from the start point

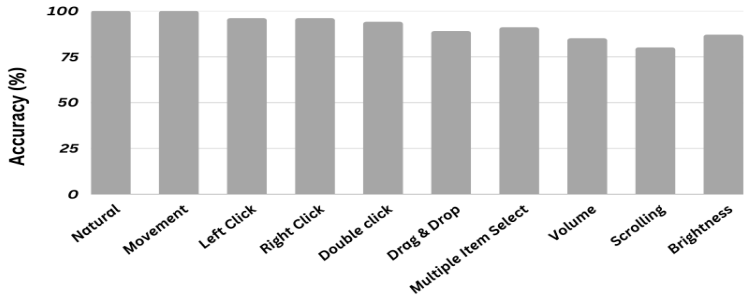


Fig. 4 Experimental Analysis of WWC functions

The experimental results demonstrate the feasibility and accuracy of the proposed system in performing various mouse controls through finger-based gestures.

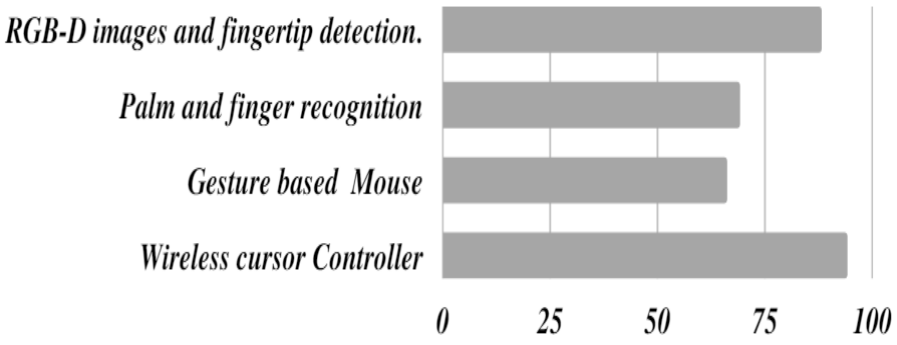


Fig. 5 Accuracy Graph of Different Available Mouse System

V. CONCLUSIONS

In this research project, we have proposed an AI virtual mouse system that can control the mouse cursor functions using hand gestures instead of a physical mouse. Our proposed system outperformed existing models in terms of accuracy, making it suitable for real-world applications. Moreover, it has the potential to reduce the spread of COVID-19 since it can be used without physical contact. However, some limitations were observed in right-click functions and selecting text using drag-and-drop. Future research should focus on overcoming these limitations by improving finger-tip detection algorithms and adding new gestures to interact with other smart systems.

VI. FUTURE SCOPE

Our proposed AI virtual mouse system has opened up new avenues for research and development in the field of human-computer interaction. In the future, we can explore the integration of other body, hand, and facial key-points to perform various tasks. The inclusion of machine learning algorithms such as OpenPose can enrich the tracking system's accuracy and improve its functionality. Additionally, the proposed system can be extended to operate other computer peripherals such as the keyboard and assistive devices to help people with physical disabilities. With further advancements, the proposed system could become a mainstream device for interfacing with computers, providing a more intuitive and natural way of interacting with machines.

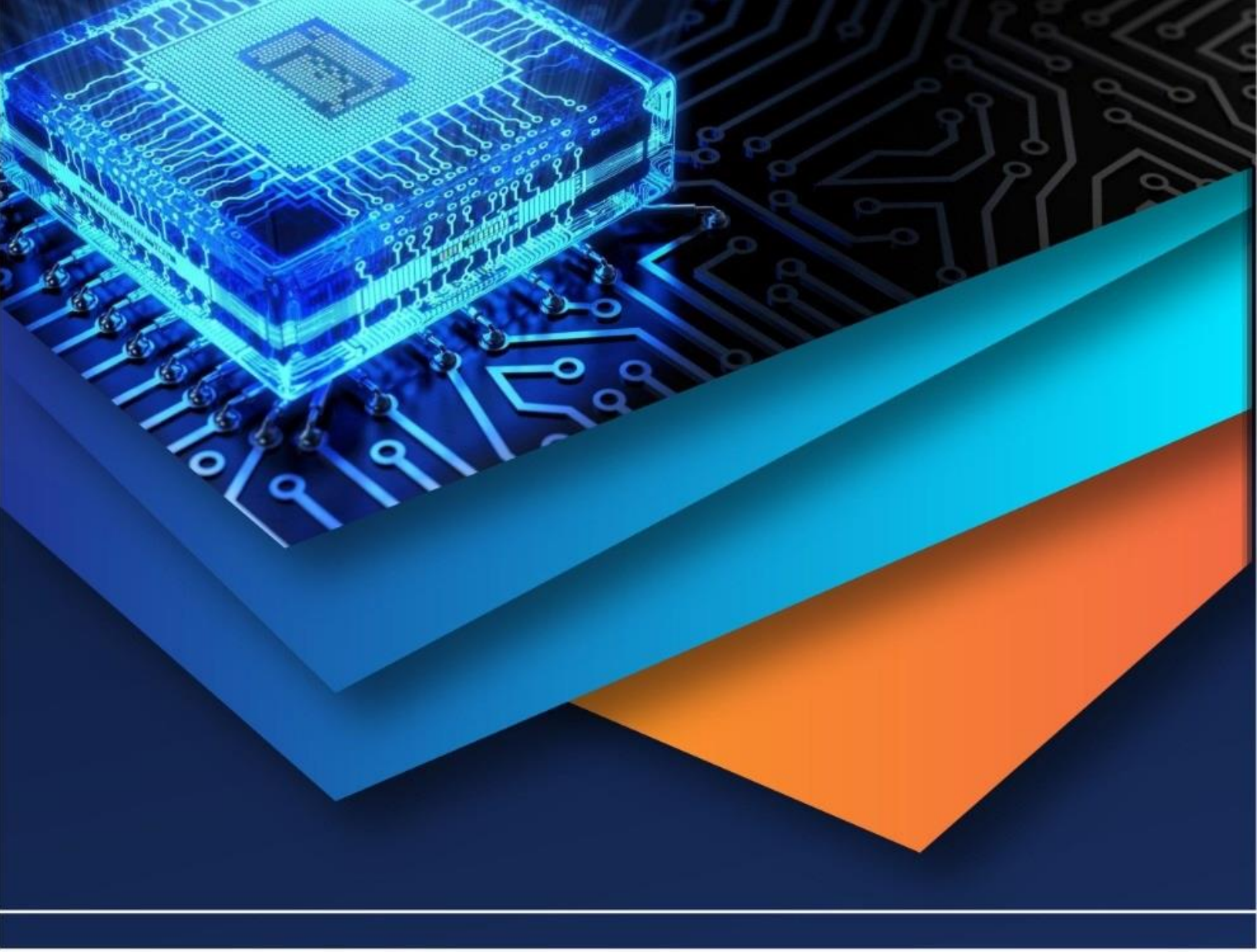
VII. ACKNOWLEDGMENT

We gratefully acknowledge the Pune Institute of Computer Technology for their support in making this research possible. Our sincere appreciation also goes to Prof. Manish Khodaskar sir for generously sharing his invaluable expertise and guidance throughout the research process. We extend our heartfelt thanks to the reviewers for their valuable feedback and insights.

.

REFERENCES

[1] Kollipara Sai Varun, I. Puneeth, Dr. T. Prem Jacob, “Virtual cursor Implementation using OpenCV”, IEEE Xplore (2019)
[2] Abhilash S S, Lisho Thomas, Naveen Wilson, Chaithanya C, “Virtual Mouse using Hand Gesture”, International Research Journal of Engineering and Technology (IRJET) (2018)
[3] Sankha Sarkar, Indrani Naskar, Sourav Sahoo, Sayan Ghosh,” A Vision Base Application for Virtual Mouse Interface Using Hand Gesture”, International Journal of Innovative Science and Research Technology (2021)
[4] Dinh-SonTran, HyungJeong Yang, Ngoc-Huynh Ho Soo-Hyun Kim & Guee Sang Lee,” Real-time virtual mouse system using RGB-D images and fingertip detection”, Springer (2020)
[5] Manjunath R Kounte, E Niveditha, A Sai Sudeshna, Kalaigar Afrose,” Video Based Hand Gesture Detection System Using Machine Learning”, International Journal of Advance Science and Technology (2020)



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY