# PROJECT

# “HandGlide”

Interactive Mouse Control using Hand Gestures on Python


A Project Report submitted to

Shri Vaishnav Vidhyapeeth Vishwavidhyalaya, Indore

In partial fulfilment for the award of the degree of

Bachelor of Technology

Computer Science & Engineering

Department of Information Technology

Shri Vaishnav Institute of Information and Technology

Shri Vaishnav Vidhyapeeth Vishwavidhyalaya

Indore – 45311

July – Dec 2023

**SHRI VAISHNAV VIDHYAPEETH VISHWAVIDHYALAYA**


CERTIFICATE

This is to certify that *Aditya Gupta (22100BTCSDSI11071)* have completed their project work titled *"HandGlide Interactive Mouse Control using Hand Gestures on Python"* as per the syllabus and have submitted a satisfactory report on this project as a part of fulfilment towards the degree of **"BACHELOR OF TECHNOLOGY" (Computer Science & Engineering)** from **Shri Vaishnav Institute of Information Technology.**

Project Guide

# ACKNOWLEDGEMENT

We express deep gratitude for enthusiasm and valuable suggestions that we got from our guide *Prof. Prithviraj Singh Rathore* for successful completion of the project. This project was not possible without the invaluable guidance of out project guide. We are deeply indebted to *Mr. Prithviraj Sir* and to The Head of the department of Computer Science & Engineering for providing us support and resources for successful completion of this project.

Finally, we are thankful to all the people who are related to the project directly or indirectly.

Project Guide:

Prof. Prithviraj Singh Rathore

Submitted By:

Aditya Gupta

(22100BTCSDSI11071)

# INDEX

# INTRODUCTION

- The Python project is the foundation of my project utilizing the Mediapipe library, OpenCV & PyAutoGUI to implement real-time hand tracking & controlling Mouse through a webcam feed via hands.

- By leveraging the capabilities of the Hands module from Mediapipe, coupled with the versatile PyAutoGUI library for mouse manipulation, the script detects and visualizes landmarks on the hands within the captured frames & makes the Mouse move on it. These landmarks represent key points on the hand, and the code employs drawing utilities to display the connections between them.

- The project integrates OpenCV for webcam access, frame manipulation, and display, while Mediapipe facilitates the extraction of hand landmarks. The PyAutoGUI library is utilized for controlling the computer mouse based on the detected hand movements & the resulting application allows users to track and visualize their hand movements in real time through the webcam, offering potential applications in gesture recognition, virtual reality, or human-computer interaction.

- The system detects hand landmarks using the MediaPipe library and utilizes the detected landmarks to control the computer mouse using the PyAutoGUI library.

- This project has the potential for further expansion and integration into applications that leverage hand tracking & detection for interactive and immersive experiences.

# ★ Objectives:

The primary objectives of the project include:

- Implementing real-time hand tracking using the Mediapipe library.
- Offering users an alternative and intuitive method for interacting with their systems.
- To empower users to control the computer mouse effortlessly, with a specific focus on the movement of the index finger and the dynamic relationship between the index and thumb fingers.
- Integrating OpenCV for webcam access, allowing seamless video capture.
- Visualizing hand landmarks and connections dynamically during runtime.
- Exploring potential applications in gesture recognition, virtual reality, and human-computer interaction.

# ★ Methodologies:

The project utilizes the following key methodologies:

- **Hand Detection with Mediapipe Integration:**
  The project leverages the Hands module from the Mediapipe library to detect and track landmarks on the user's hands. It provides a pre-trained model capable of recognizing the landmarks of a hand, such as the positions of fingertips, joints, and the palm.

  The *mp.solutions.hands.Hands()* class is used to create a hands detection module.
  The script processes each video frame to identify hand landmarks using the *hand_detector.process* method.

- **Video Capture using OpenCV Integration:**
  OpenCV is employed to access the webcam, manipulate frames, and ensure compatibility with the Mediapipe library. It captures video frames from the default webcam using the OpenCV library (cv2.VideoCapture).

- **Mouse Control via PyAutoGUI:**
  The pyautogui library is utilized for controlling the computer mouse based on the detected hand landmarks.

  Hand landmarks, such as the index and thumb fingers, are tracked to update the mouse cursor's position using the pyautogui.moveTo function.

Mouse clicks are simulated using the pyautogui.click() function. The script triggers a click action under specific conditions, such as when the index and thumb fingers are close together.

Many other Functions like pyautogui.alert(“Message”) can be used.

**Dynamic Visualization:** Hand landmarks are dynamically visualized in real-time, enhancing the user's understanding of their hand movements.

## ★ Libraries Used:

### Mediapipe Library:

MediaPipe is an open-source framework for building pipelines to perform computer vision inference over arbitrary sensory data such as video or audio. Using MediaPipe, such a perception pipeline can be built as a graph of modular components. MediaPipe is currently in alpha at v0.

It is developed by Google that provides a comprehensive suite of pre-built tools and solutions for building applications involving perception and sensing.
It is widely used for Face Detection & Recognition, Hand Tracking, Pose Detection, Holistic Tracking, Multi-Model input etc.

## OpenCV:

The OpenCV(Open Source Computer Vision) is a Python library that allows you to perform image processing and computer vision tasks. It provide a set of tools and functions for real-time computer vision applications.

## PyAutoGUI:

PyAutoGUI is essentially a Python package that works across Windows, MacOS X and Linux which provides the ability to simulate mouse cursor moves and clicks as well as keyboard button presses.

## ★ Features:

- Real-time hand tracking & gesture recognition through a webcam.
- Seamless Mouse Control.
- Seamless integration of Mediapipe and OpenCV for enhanced performance.
- Accurate landmark detection and dynamic visualization.
- Compatibility with various applications, including gesture recognition and virtual reality.

## ★ Usage:

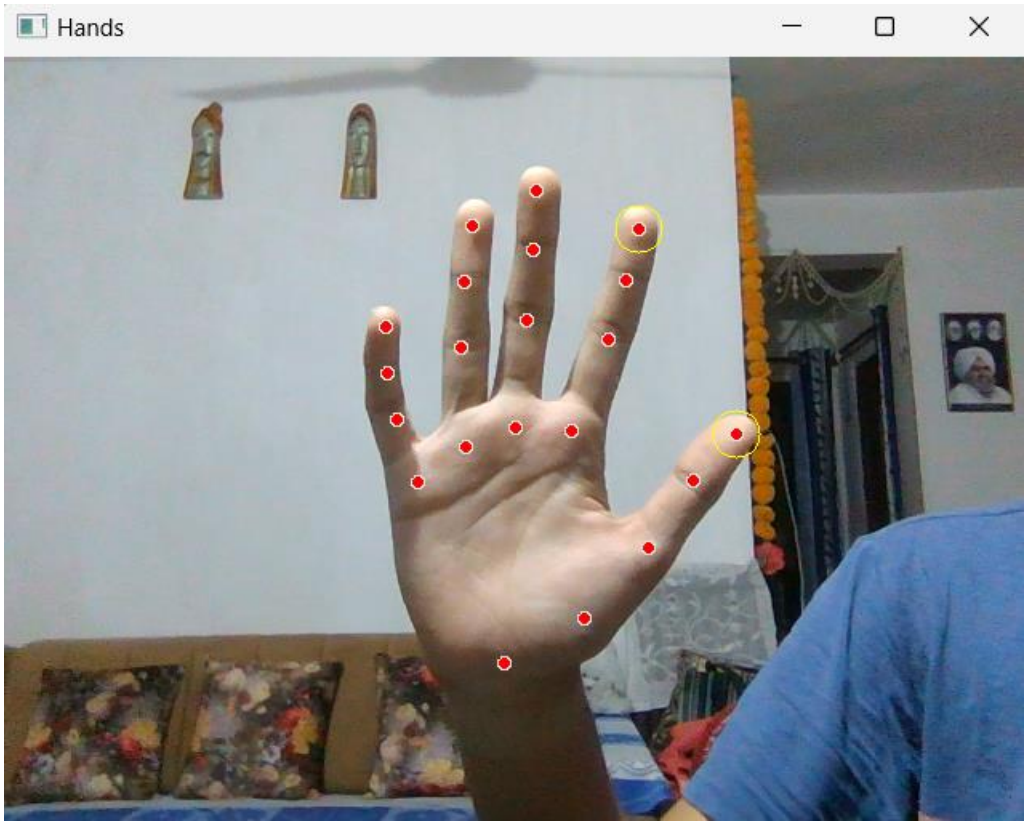
- Run the script on a system with a webcam.
- The script captures video frames and detects hand landmarks.
- The cursor on the computer screen is controlled by the movement of the index finger & thumb.
- A mouse click is simulated when the index and thumb fingers are close together.

## Source Code:

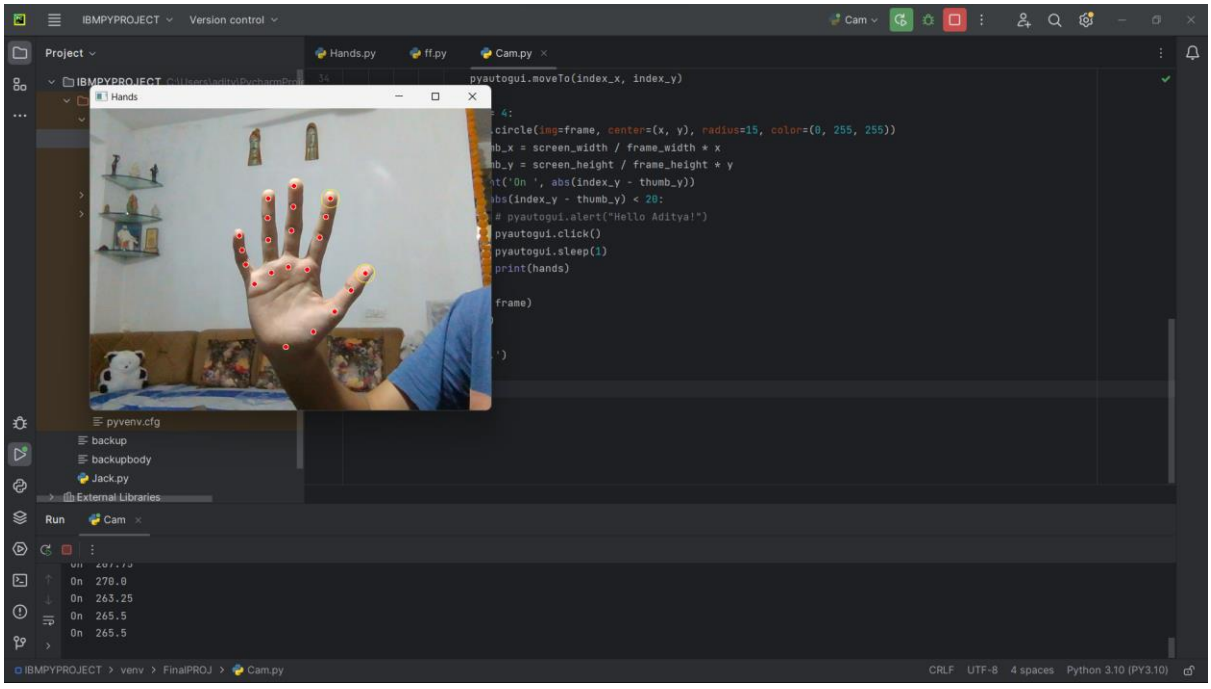
```
import cv2
import mediapipe as mp
import pyautogui

cap = cv2.VideoCapture(0)

hand_detector = mp.solutions.hands.Hands()
drawing_utils = mp.solutions.drawing_utils
screen_width, screen_height = pyautogui.size()

index_y = 0

while True:
    _, frame = cap.read()
    frame = cv2.flip(frame, 1)
    frame_height, frame_width, _ = frame.shape
    rgb_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
    output = hand_detector.process(rgb_frame)
    hands = output.multi_hand_landmarks

    if hands:
        for hand in hands:
            drawing_utils.draw_landmarks(frame, hand)

            landmarks = hand.landmark
            for ID, landmark in enumerate(landmarks):
                x = int(landmark.x * frame_width)
                y = int(landmark.y * frame_height)

                if ID == 8:
                    cv2.circle(img=frame, center=(x, y), radius=15,
color=(0, 255, 255))
                    index_x = screen_width / frame_width * x
                    index_y = screen_height / frame_height * y
                    pyautogui.moveTo(index_x, index_y)

                if ID == 4:
                    cv2.circle(img=frame, center=(x, y), radius=15,
color=(0, 255, 255))
                    thumb_x = screen_width / frame_width * x
                    thumb_y = screen_height / frame_height * y
                    print('On ', abs(index_y - thumb_y))
                    if abs(index_y - thumb_y) < 20:
                        # pyautogui.alert("Hello Aditya!")
                        pyautogui.click()
                        pyautogui.sleep(1)
                        print(hands)

    cv2.imshow('Hands', frame)
    key = cv2.waitKey(3)
    if key == 27:
        print('Exiting..')
        break
```

# ★ Output Results:

## Frame which Detects hand movement:




## Full-Screen:



# ★ Requirements:

## ➢ Software Requirements:

- Python 3.11.0 or Python 3.12.0
- PyCharm 2023.1.2
- Mediapipe Library for Python
- OpenCV Library for Python
- PyAutoGui Library for Python
- Webcam Support
- Defined Screen Resolution

## ➢ Hardware Requirements:

- Minimum 2GB of Ram
- Minimum 1GB of Hard-drive storage

# ★ Potential:

## ○ Intuitive Interaction:

The project introduces a more intuitive way for users to interact with their computers by translating natural hand gestures into mouse control actions.

## ○ Accessibility:

The project has the potential to enhance accessibility for individuals with physical disabilities or limitations. Gesture-based controls can provide an alternative means of interaction, potentially improving accessibility in comparison to traditional mouse and keyboard inputs.

## ○ User-Friendly & Experimental Interface Design:

The project serves as a foundation for experimental interface design, encouraging exploration of novel ways to interact with computers. The user-friendly nature of this project could contribute to the development of more user-centric interfaces. This could inspire further research and development in the field of human-computer interaction.

## ○ Customization and Extensions:

The project is extensible, allowing for the recognition of additional hand gestures or the implementation of specific functionalities based on user needs. This extensibility opens possibilities for customization in various application domains.

## ○ Innovation in Human-Computer Interaction:

The project contributes to the ongoing innovation in human-computer interaction, pushing the boundaries of how users engage with technology. It aligns with the broader trend of developing more natural and intuitive interfaces.

## ○ Conclusion:

In conclusion, this project successfully explores the integration of hand gesture recognition for mouse control, contributing to the development of more intuitive and natural user interfaces.

By combining computer vision techniques with automation, the project demonstrates the potential for seamless interaction between humans and computers.

As technology continues to advance, projects like this pave the way for innovative approaches to human-computer interaction.