PONJESLY COLLEGE OF ENGINEERING

NAGERCOIL – 629 003

(Affiliated to Anna University, Chennai)

NM1093 – OPEN CV

Name :

Year/ Sem :

Reg.No :

PONJESLY COLLEGE OF ENGINEERING

NAGERCOIL – 629 003

(Affiliated to Anna University Chennai)

Section …… Class No. …… Reg. No. ……

Certified that this is the bonafide record of the work done by …… in the Department of Information Technology, during the Academic Year ……

Head of the Dept. **Staff Member in Charge**

Submitted for the Anna University Practical Examination, held at PONJESLY COLLEGE OF ENGINEERING, Nagercoil on ……

Internal Examiner **External Examiner**

GESTURE RECOGNITION SYSTEM

Problem statement:

In many human-computer interaction (HCI) systems, the ability to accurately recognize hand gestures plays a vital role in enhancing accessibility, automation, and immersive user experiences. Traditional input devices like keyboards, mice, and touchscreens limit the natural way humans interact with technology. For users with physical disabilities or for hands-free applications such as robotics control, AR/VR interfaces, or sign language interpretation, gesture recognition systems offer an intuitive alternative. However, reliably detecting hand gestures in static images or videos requires precise landmark detection, interpretation of finger positions, and robust classification methods. This problem is compounded by variability in hand shapes, orientations, lighting conditions, and camera quality. Thus, the core challenge lies in developing a system that can recognize hand gestures from static input images accurately and consistently, using minimal resources, and without requiring extensive training data or custom hardware.

Existing gesture recognition systems often depend on large datasets or expensive hardware like depth cameras, motion sensors, or gloves. These approaches, while accurate, are not always scalable or feasible for common applications. There is a growing need for accessible, low-cost, and efficient methods that can work using simple RGB cameras and pre-trained models. Another issue arises in detecting gestures under inconsistent backgrounds or lighting, where traditional pixel-based or contour-based methods fail. Therefore, using a reliable and robust landmark-based approach such as MediaPipe offers a significant advantage. The goal is to bridge the gap between technical feasibility and real-world usability by creating a system that is easy to implement, platform-independent, and responsive enough to classify hand gestures in various settings with minimal computation.

Abstract:

Hand gesture recognition is a vital component in developing natural human-computer interaction (HCI) systems. This project introduces a lightweight and efficient static gesture recognition system that utilizes MediaPipe and OpenCV to detect and interpret hand gestures from images. MediaPipe's hand tracking solution is employed to extract 21 precise hand landmarks, which are then analyzed to determine the number and position of extended fingers. Using simple geometric rules, the system classifies gestures such as "Fist," "Thumbs Up," "Two Fingers Up," and "Open Hand" based on the spatial relationship between key landmarks like the finger tips and joints. The method avoids the complexity of deep learning models while still maintaining high accuracy and speed.

The proposed solution is ideal for real-time applications due to its minimal computational requirements. It can run on basic hardware without the need for GPU acceleration or large-scale training datasets. This makes it highly suitable for deployment on embedded devices, IoT systems, or mobile platforms. Its versatility allows it to serve a wide range of use cases including gesture-based control systems, accessibility tools for individuals with disabilities, interactive gaming interfaces, contactless communication devices, and smart home automation. Moreover, since it relies on visual input alone, it can be implemented in environments where physical buttons or touchscreens may be impractical.

In addition, the system's modularity enables easy extension to more complex gestures or real-time video input. Developers can integrate this solution with other AI modules for applications such as sign language interpretation, robotic control, or virtual/augmented reality interactions. Since MediaPipe handles most of the pre-processing and landmark detection, users can focus on designing intuitive gesture-based interfaces without worrying about low-level vision processing.

The system also supports easy customization, allowing developers to define new gesture classes based on additional landmark rules. Its low-latency performance ensures real-time responsiveness, which is critical for interactive applications. By eliminating the need for large training datasets, this approach significantly lowers development time and cost while maintaining practical accuracy.

In summary, this gesture recognition system showcases the power of combining open-source tools with smart geometric analysis to create effective, fast, and scalable computer vision solutions for modern HCI needs.

Introduction:

Human gestures are among the most intuitive forms of communication. As computers become more integrated into daily life, making their interfaces more natural and accessible is a key concern. Gesture recognition is one such area of research in human-computer interaction (HCI) that allows machines to interpret human gestures, particularly hand movements, as commands or inputs. The objective of this project is to detect and classify hand gestures using static images through landmark detection and geometrical analysis.

Traditionally, gesture recognition systems relied on either hardware-based sensors (like gloves) or large datasets to train deep learning models. These approaches are often expensive, computationally heavy, or impractical for simple applications. With the emergence of tools like MediaPipe, it's now possible to accurately detect hand landmarks using real-time models without the overhead of training and maintaining a model from scratch.

In this project, MediaPipe's hand tracking solution is used in static image mode to detect a single hand in an image. Once detected, the relative positions of 21 landmarks

(fingertips, joints, wrist) are analyzed to determine how many fingers are extended. For instance, the thumb's tip position compared to its inner joint (IP) helps decide whether the thumb is up or folded. Similarly, each finger is checked by comparing the vertical (y) coordinates of its tip and pip (proximal interphalangeal joint).

The system classifies gestures into commonly known categories such as "Fist", "Thumbs Up", "Two Fingers Up", "Open Hand", and other intermediate combinations. It then overlays the detected gesture name on the original image and shows hand skeletons using OpenCV.

This type of gesture detection has real-world applications in accessibility solutions, sign language interpretation, remote control of smart devices, and virtual interactions in AR/VR. Moreover, this static gesture analysis can be easily extended to dynamic video input for real-time interaction.

This project emphasizes the practicality and ease of building gesture recognition tools using freely available and pre-trained tools like MediaPipe, eliminating the need for complex training procedures or deep learning expertise. It opens the door for students, developers, and researchers to explore gesture-based systems without steep learning curves or costly implementations.

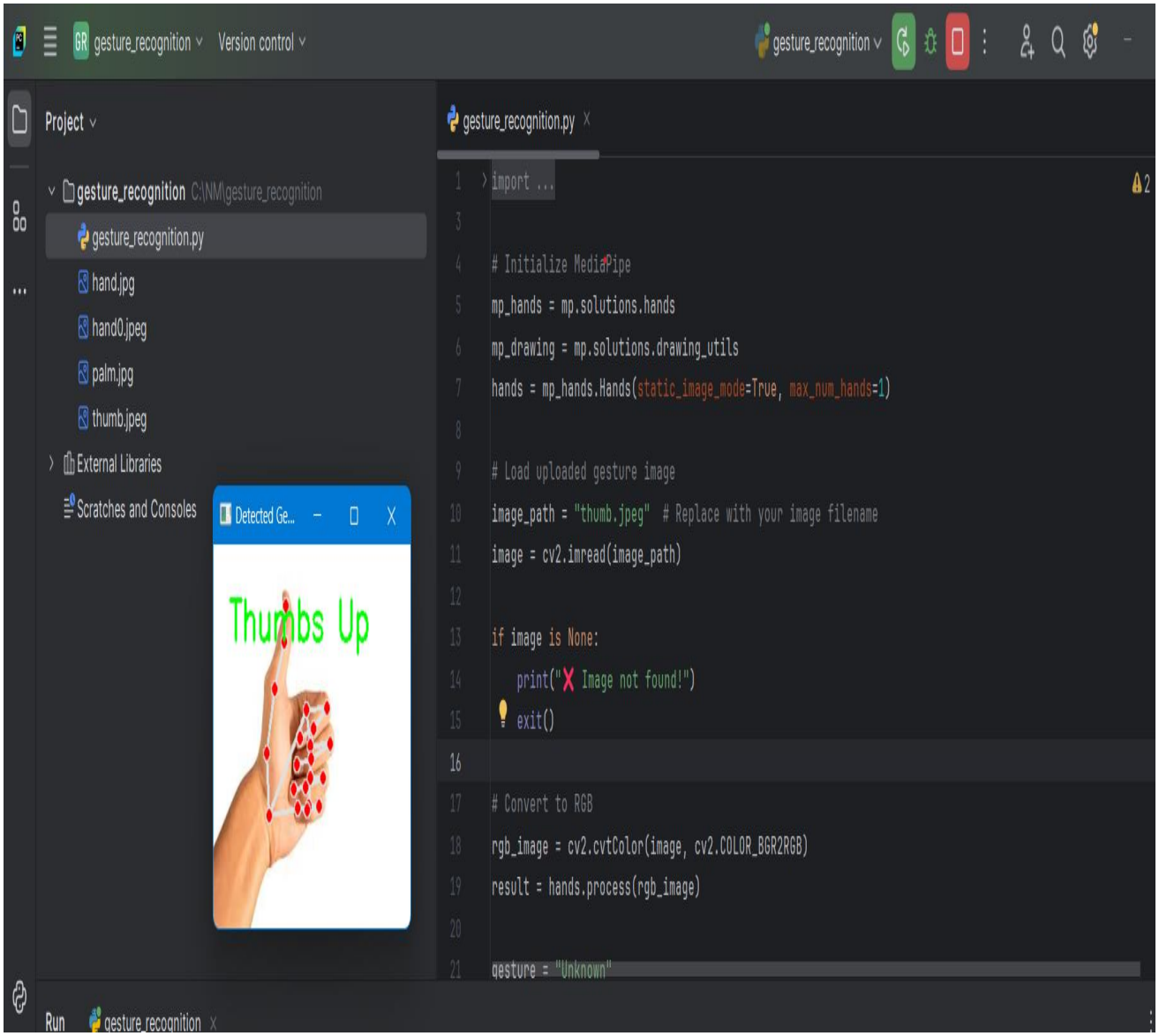
Program:

```
import cv2
import mediapipe as mp

# Initialize MediaPipe
mp_hands = mp.solutions.hands
mp_drawing = mp.solutions.drawing_utils
hands = mp_hands.Hands(static_image_mode=True, max_num_hands=1)

# Load uploaded gesture image
image_path = "thumb.jpeg"  # Replace with your image filename
image = cv2.imread(image_path)

if image is None:
    print("❌ Image not found!")
    exit()

# Convert to RGB
rgb_image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
result = hands.process(rgb_image)

gesture = "Unknown"

if result.multi_hand_landmarks:
    hand_landmarks = result.multi_hand_landmarks[0]

    # Get landmark coordinates
    landmarks = hand_landmarks.landmark
```

```
    fingers_up = 0

    # Thumb: Check if tip is to the right of IP (for right hand)
    if landmarks[mp_hands.HandLandmark.THUMB_TIP].x >
landmarks[mp_hands.HandLandmark.THUMB_IP].x:
        fingers_up += 1

    # Other 4 fingers
    tips = [mp_hands.HandLandmark.INDEX_FINGER_TIP,
            mp_hands.HandLandmark.MIDDLE_FINGER_TIP,
            mp_hands.HandLandmark.RING_FINGER_TIP,
            mp_hands.HandLandmark.PINKY_TIP]

    pips = [mp_hands.HandLandmark.INDEX_FINGER_PIP,
            mp_hands.HandLandmark.MIDDLE_FINGER_PIP,
            mp_hands.HandLandmark.RING_FINGER_PIP,
            mp_hands.HandLandmark.PINKY_PIP]

    for tip, pip in zip(tips, pips):
        if landmarks[tip].y < landmarks[pip].y:
            fingers_up += 1

    # Gesture classification
    if fingers_up == 0:
        gesture = "Fist"
    elif fingers_up == 1:
        gesture = "Thumbs Up"
    elif fingers_up == 2:
```

```
        gesture = "Two Fingers Up"
    elif fingers_up == 5:
        gesture = "Open Hand"
    else:
        gesture = f"{fingers_up} Fingers Up"

    # Draw hand landmarks
    mp_drawing.draw_landmarks(image, hand_landmarks,
mp_hands.HAND_CONNECTIONS)
else:
    gesture = "No hand detected"

# Show result
cv2.putText(image, gesture, (20, 50), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,
255, 0), 2)
cv2.imshow("Detected Gesture", image)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output:


gesture_recognition
Version control
gesture_recognition
Project
gesture_recognition C:\NM\gesture_recognition
gesture_recognition.py
hand.jpg
hand0.jpeg
palm.jpg
thumb.jpeg
External Libraries
Scratches and Consoles
Detected Ge...
Thumbs Up
gesture_recognition.py
import ...
# Initialize MediaPipe
mp_hands = mp.solutions.hands
mp_drawing = mp.solutions.drawing_utils
hands = mp_hands.Hands(static_image_mode=True, max_num_hands=1)
# Load uploaded gesture image
image_path = "thumb.jpeg" # Replace with your image filename
image = cv2.imread(image_path)
if image is None:
print("X Image not found!")
exit()
# Convert to RGB
rgb_image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
result = hands.process(rgb_image)
gesture = "Unknown"
Run
gesture_recognition

Conclusion:

In conclusion, this project successfully implements a static image-based hand gesture recognition system using MediaPipe and OpenCV in Python. By analyzing the relative positions of detected hand landmarks, the system effectively identifies gestures such as a fist, thumbs up, two fingers up, or an open hand. The use of MediaPipe's pre-trained models ensures high accuracy and reliability, even in diverse conditions, without the need for custom datasets or deep learning models. The project demonstrates how computer vision and gesture analysis can be integrated seamlessly to create intelligent, user-friendly interfaces. This system serves as a foundational tool for building more advanced applications in the realms of sign language translation, touchless controls, robotics, and accessibility technology. Its lightweight nature, simplicity of implementation, and strong performance make it ideal for both educational and real-world applications.