**DAY:20 MEDIAPIPE AND POSE ESTIMATION**

**Step 1: Install Required Libraries**

Install the libraries needed to:

* Access your webcam (OpenCV)
* Detect human body poses (MediaPipe)

Open terminal and run:

pip install opencv-python mediapipe

**Step 2: Capture Video Input (Webcam)**

Use OpenCV to access and display your webcam feed.

import cv2

# Open the default webcam (device 0)

cap = cv2.VideoCapture(0)

while cap.isOpened():

ret, frame = cap.read()

if not ret:

break

# Display the live frame in a window

cv2.imshow('Webcam Feed', frame)

# Press 'q' to close the window

if cv2.waitKey(1) & 0xFF == ord('q'):

break

# Release resources

cap.release()

cv2.destroyAllWindows()

**Step 3: Use MediaPipe to Detect Body Landmarks**

Use MediaPipe to detect body landmarks (like shoulders, elbows, knees, etc.) on each frame from your webcam.

import cv2

import mediapipe as mp

# Initialize MediaPipe Pose and drawing utils

mp\_pose = mp.solutions.pose

pose = mp\_pose.Pose()

mp\_drawing = mp.solutions.drawing\_utils

# Open webcam

cap = cv2.VideoCapture(0)

while cap.isOpened():

ret, frame = cap.read()

if not ret:

break

# Convert BGR to RGB (MediaPipe uses RGB)

image\_rgb = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

# Process the image and get pose landmarks

results = pose.process(image\_rgb)

# Draw landmarks on the original frame

if results.pose\_landmarks:

mp\_drawing.draw\_landmarks(

frame, results.pose\_landmarks, mp\_pose.POSE\_CONNECTIONS)

# Display the frame

cv2.imshow('Pose Detection', frame)

# Press 'q' to exit

if cv2.waitKey(1) & 0xFF == ord('q'):

break

cap.release()

cv2.destroyAllWindows()

**Step 4: Extract Landmark Coordinates**

In this step, extract the X, Y, Z coordinates of each body part (landmark) detected by MediaPipe.

import cv2

import mediapipe as mp

mp\_pose = mp.solutions.pose

pose = mp\_pose.Pose()

mp\_drawing = mp.solutions.drawing\_utils

cap = cv2.VideoCapture(0)

while cap.isOpened():

ret, frame = cap.read()

if not ret:

break

# Convert image to RGB

image\_rgb = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

# Detect landmarks

results = pose.process(image\_rgb)

# Draw landmarks

if results.pose\_landmarks:

mp\_drawing.draw\_landmarks(

frame, results.pose\_landmarks, mp\_pose.POSE\_CONNECTIONS)

# Extract landmark coordinates

landmarks = results.pose\_landmarks.landmark

for idx, lm in enumerate(landmarks):

h, w, \_ = frame.shape # get image height & width

cx, cy = int(lm.x \* w), int(lm.y \* h)

print(f"Landmark {idx}: x={cx}, y={cy}, z={lm.z:.4f}, visibility={lm.visibility:.2f}")

# Optionally draw a circle on each point

cv2.circle(frame, (cx, cy), 5, (255, 0, 0), cv2.FILLED)

# Show the frame

cv2.imshow('Landmark Coordinates', frame)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

cap.release()

cv2.destroyAllWindows()

**Step 5: Calculate Angles Between Joints**

* Extract (x, y) for 3 landmarks: point A, B, C
* Calculate the angle at point B (joint)
* Use this to compare with expected pose angles

import cv2

import mediapipe as mp

import math

# Function to calculate angle between 3 points

def calculate\_angle(a, b, c):

a = [a.x, a.y]

b = [b.x, b.y]

c = [c.x, c.y]

ba = [a[0] - b[0], a[1] - b[1]]

bc = [c[0] - b[0], c[1] - b[1]]

dot\_product = ba[0]\*bc[0] + ba[1]\*bc[1]

magnitude = math.sqrt(ba[0]\*\*2 + ba[1]\*\*2) \* math.sqrt(bc[0]\*\*2 + bc[1]\*\*2)

if magnitude == 0:

return 0

angle\_rad = math.acos(dot\_product / magnitude)

angle\_deg = math.degrees(angle\_rad)

return round(angle\_deg, 2)

# Initialize MediaPipe

mp\_pose = mp.solutions.pose

pose = mp\_pose.Pose()

mp\_drawing = mp.solutions.drawing\_utils

# Start webcam

cap = cv2.VideoCapture(0)

while cap.isOpened():

ret, frame = cap.read()

if not ret:

break

# Convert frame to RGB

image\_rgb = cv2.cvtColor(frame, cv2.COLOR\_BGR2RGB)

results = pose.process(image\_rgb)

if results.pose\_landmarks:

landmarks = results.pose\_landmarks.landmark

mp\_drawing.draw\_landmarks(frame, results.pose\_landmarks, mp\_pose.POSE\_CONNECTIONS)

# Example: Calculate left elbow angle

shoulder = landmarks[mp\_pose.PoseLandmark.LEFT\_SHOULDER.value]

elbow = landmarks[mp\_pose.PoseLandmark.LEFT\_ELBOW.value]

wrist = landmarks[mp\_pose.PoseLandmark.LEFT\_WRIST.value]

angle = calculate\_angle(shoulder, elbow, wrist)

# Show angle on the frame

cx, cy = int(elbow.x \* frame.shape[1]), int(elbow.y \* frame.shape[0])

cv2.putText(frame, f'{angle}°', (cx, cy), cv2.FONT\_HERSHEY\_SIMPLEX, 0.9, (0, 255, 0), 2)

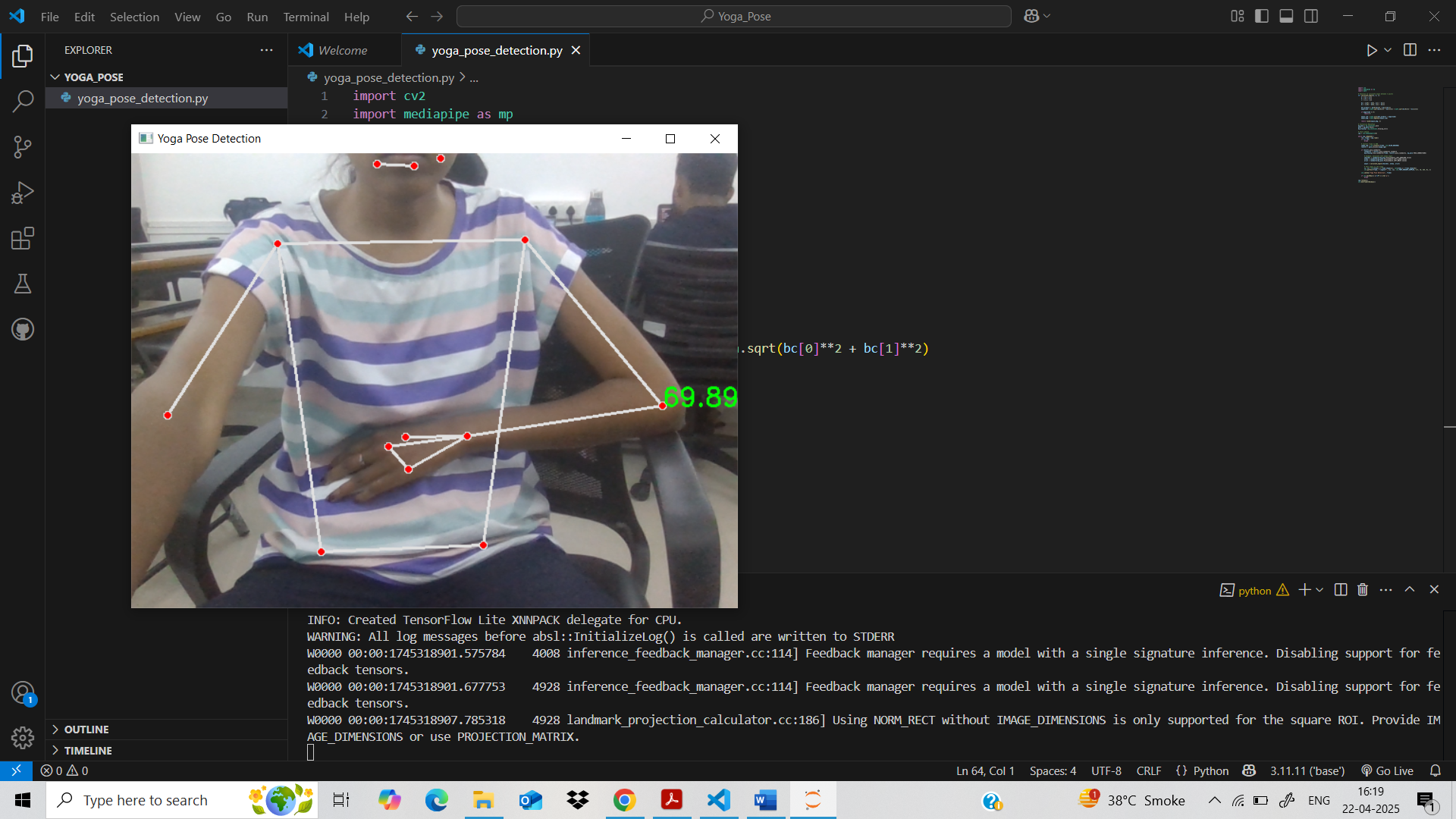
cv2.imshow('Yoga Pose Detection', frame)

if cv2.waitKey(1) & 0xFF == ord('q'):

break

cap.release()

cv2.destroyAllWindows()



**Step 6: Define a Yoga Pose Based on Angles**

Let’s focus on two poses mainly,

* Tadasana
* Vrikshasana

