University of Helwan

Faculty of Engineering

Computer and Systems Department

Course : Python (Summer Training 2023)

AI Virtual Mouse with Face Recognition

Edited by :

Mostafa Alaa Eldin Hassan Mohamed

Mohaned Mohamed Mostafa Rashed

Mohamed Waled Mohamed Atta

Usama Mohamed Mohamed Abdel-G

Mostafa Ehab Mohamed Mohamed

Eslam Mohamed Abdel-Rady Khalf

Instructor:

Eng. Ahmed Saeed

Oct 19, 2023

➔ Face Recognition (First Part):

This part of the code is responsible for recognizing a face in the webcam feed. It uses the face_recognition library along with OpenCV to accomplish this. Here's a breakdown of the key components:

- Importing necessary libraries:

  - mediapipe for hand tracking.
  - math for mathematical calculations.
  - autopy for controlling the mouse.
  - numpy for numerical operations.
  - cv2 for computer vision tasks.
  - face_recognition for face recognition.
  - cvzone (although it's not used in this part).

  Capturing the webcam feed with OpenCV (cv2.VideoCapture).

- Encoding a reference image:

  A reference image (a known face) is loaded using cv2.imread.

  The image is converted to RGB format (cv2.cvtColor).

  The reference image is then encoded using face_recognition.face_encodings and stored in encodeList.

- Running a loop to continuously process frames from the webcam:

  Reading a frame from the webcam.

  Converting the frame to RGB.

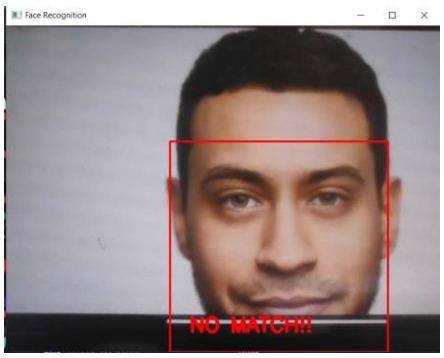
  Detecting faces in the frame and encoding them (face_recognition.face_locations and face_recognition.face_encodings).

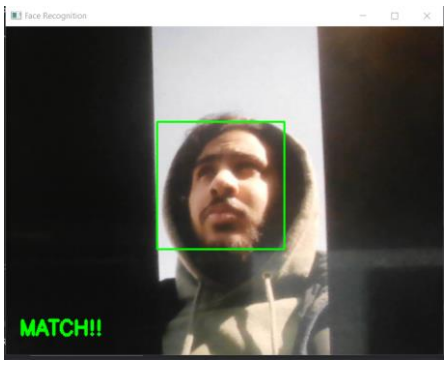
  Comparing the encoding of the current face with the reference encoding.

  If a match is found, it highlights the recognized face with a green rectangle and displays "MATCH!!" on the frame.

  If no match is found, it displays "NO MATCH!!".

  Exiting the loop if a match is found (matchTrue == 1).





➢ Controlling Mouse Using Hand Tracking (Second Part):

This part of the code is responsible for controlling the mouse pointer using hand tracking. It uses mediapipe for hand tracking and autopy to control the mouse. Here's an explanation of the key components:

Importing the required libraries, similar to the first part.

➢ Initializing the handDetector class:

This class is responsible for hand tracking and detection.

It uses the mediapipe library to track hands.

Capturing the webcam feed using OpenCV and setting the frame size.

➢ Running a loop to continuously process frames from the webcam:

Finding hand landmarks using the handDetector class.

Getting the position of specific landmarks on the hand (e.g., index finger, middle finger).

Checking which fingers are up and detecting gestures.

Converting the position of hand landmarks to screen coordinates.

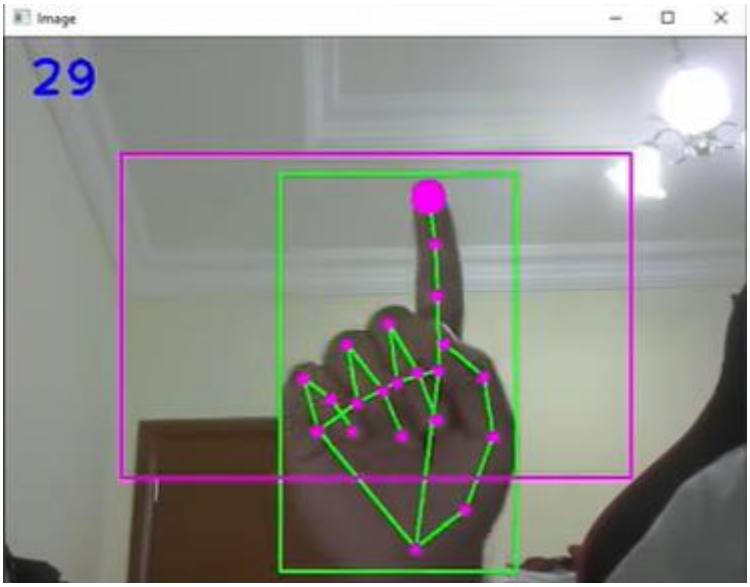
Smoothing the mouse movement for a more natural experience.

Moving the mouse pointer with autopy.mouse.move.

Enabling click functionality when both the index and middle fingers are up and they are close together.

Displaying the webcam feed with tracking information and mouse clicks.

Note: The code also has comments that provide additional context and explanations for various parts of the code.



➔ Summary:

The face recognition part is a prerequisite to enable mouse control using hand tracking. Once a known face is recognized, the code proceeds to enable hand tracking for controlling the mouse.