# CUSTOMIZABLE AI BASED GESTURE RECOGNITION SYSTEM

BY
SHAGUN (CS23TECH14013)

# OUR MAIN TOPICS TODAY

Introduction

Literature Survey (in tabular form )

Objective

Block Diagram

Algorithms

Tools and technologies Used

Implementation

Future scope

References

# INTRODUCTION TO HMI

- research in the design and the use of computer technology,focuses on the interfaces between people (users) and computers
- VUI are used for speech recognition and synthesizing systems, and the emerging multi-modal , GUI allow humans to engage with embodied character agents in a way that cannot be achieved with other interface paradigms
- has led to an increase in the quality of interaction and resulted in many new areas of research beyond

# Gesture

UI for providing real-time data to a computer in which instead of typing with keys or tapping on a touch screen, the device perceives and interprets movements as the primary source of data input.

Types :

(i) Static
(ii) Dynamic

# LITERATURE SURVEY

# Comparison

| Database And Year | Title | Methodology | Key Points | Use Cases |
|---|---|---|---|---|
| IEEE 2007 | A Real Time Hand Gesture Recognition Method | (i)Hand detection with adaboost<br>(ii) Adaptive hand segmentation is executed during detection and tracking with motion and color cues.<br>(iii) Scale space features detection is applied, hand gesture type is determined. | 1. Hand detection<br>2. Hand tracking<br>3. Hand Segmentation | |
| IEEE 2014 | Survey on Hand Gesture Recognition approach | (i)Glove based approach<br>(ii) 3D Model Based Approach | | Sign language, clinical surgery, Robot |
| IEEE | Gesture Recognition Using openCV | a)Capture the image through web cam or cam<br>b)Convert the image from RGB to gray Scale<br>c)Background is separated by | Short Comings:<br>1.High background noise must not be there.<br>2.Input from web cam has less pixel hence image is not sharp.<br>3.Good lighting is also must.<br>4.Many gesture recognition systems do not read motions accurately or optimally due to factors like insufficiently background light, high background noise etc.<br>5.Different users make gestures differently, causing difficulty in identifying motions. | |

# Our Objective

- Use gesture detection to embellish the extent of usability, better user experience
- Increase the ease of access
- **Personalizing the system in accordance with preferences of the user using a Dashboard**
- increased responsiveness of the system
- Help people having disability like bell's palsy

# Methodology

1) User registeristration on web app

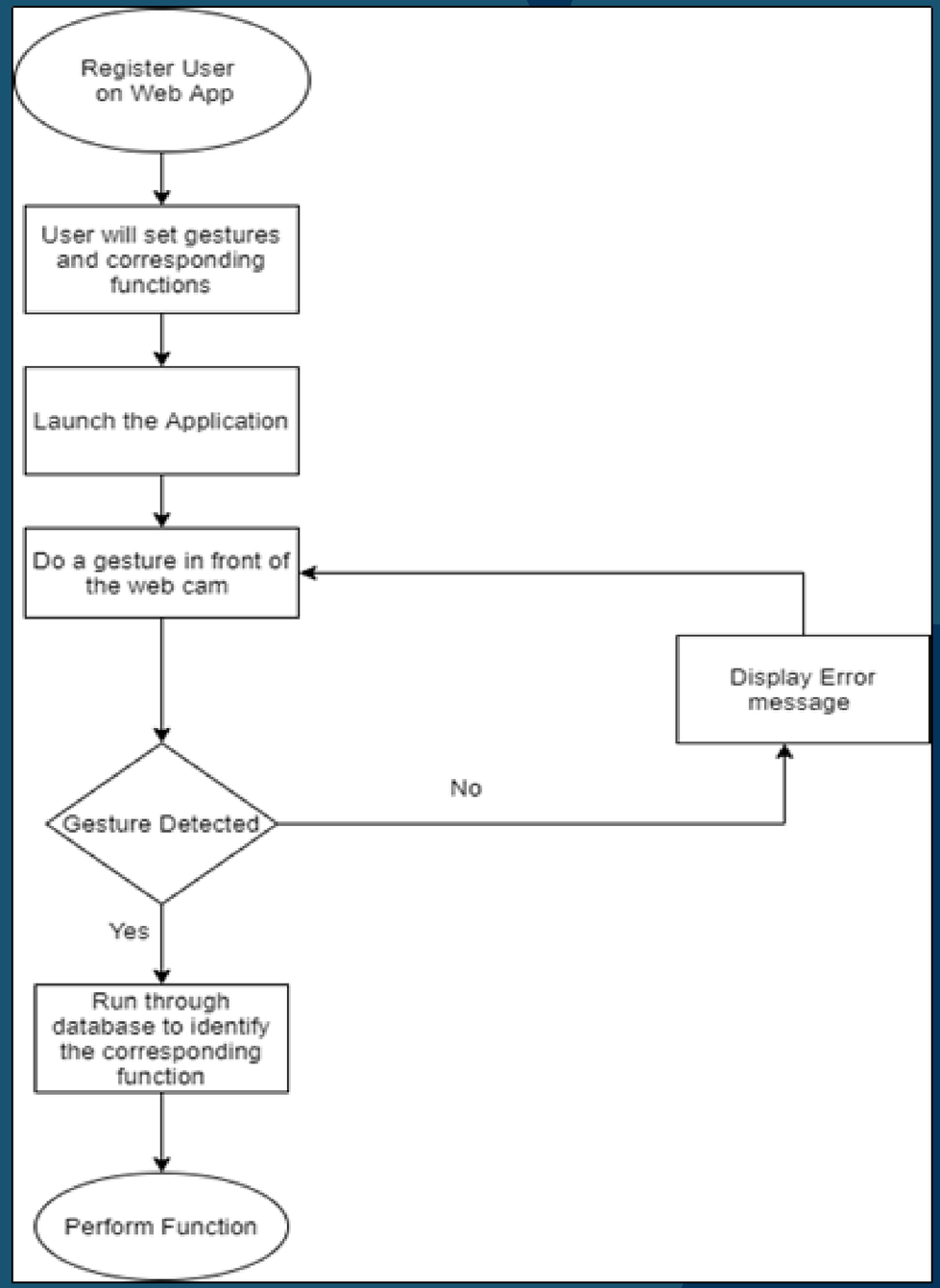
2) User will set gestures and corresponding functions

3) Launch the application

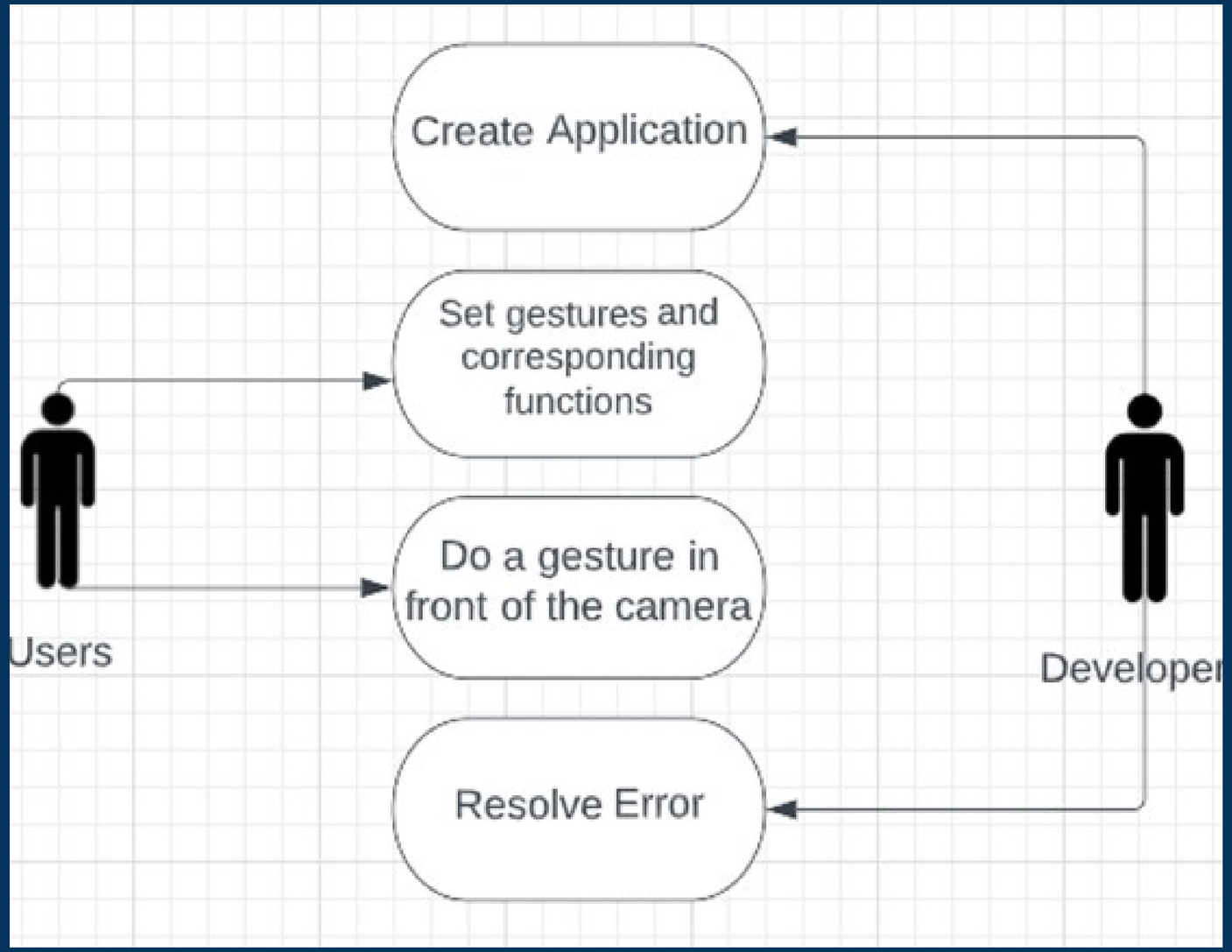
4) Gestures will be detected

) Detected gesture will be run through the user atabase to identify the corresponding function

6) nction will be performed using different libraries and functions



# Use Case Diagram



# Platforms used in the project

PYTHON

Version : 3.10.0

# Python Modules Used

## HAARCASCADE

Presentations are communication tools that can be used as lectures.

## SELENIUM

For Browser Automation

## PYAUTOGUI

For system functions, controlling mouse, keyboard actions

## OPEN CV

For taking input image or video

## TIME

For using time delay at some points

## Media Pipe

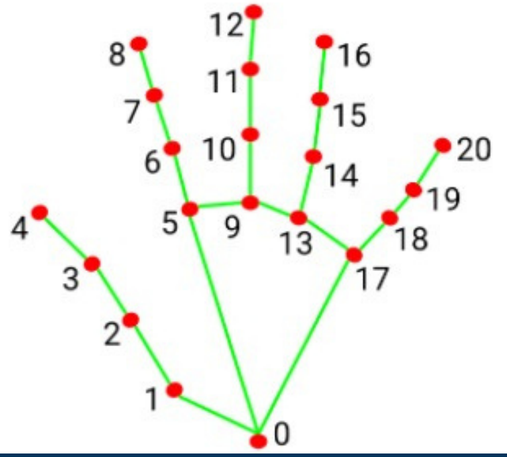
## SKLEARN

For PCA

and more

# Implementation

In the step, we will create a function `detectHandsLandmarks()` that will take an image/frame as input and will perform the landmarks detection on the hands in the image/frame using the solution provided by Mediapipe and will get **twenty-one 3D landmarks** for each hand in the image. The function will display or return the results depending upon the passed arguments.




0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

jupyter Minor Project Team 61-Anju Arya Last Checkpoint: 5 minutes ago (autosaved)

File Edit View Insert Cell Kernel Help Trusted Python 3

Run Code

```
def countFingers(image, results, draw=True, display=True):
    # Get the height and width of the input image.
    height, width, _ = image.shape

    # Create a copy of the input image to write the count of fingers on.
    output_image = image.copy()

    # Initialize a dictionary to store the count of fingers of both hands.
    count = {'RIGHT': 0, 'LEFT': 0}

    # Store the indexes of the tips landmarks of each finger of a hand in a list.
    fingers_tips_ids = [mp_hands.HandLandmark.INDEX_FINGER_TIP, mp_hands.HandLandmark.MIDDLE_FINGER_TIP,
                        mp_hands.HandLandmark.RING_FINGER_TIP, mp_hands.HandLandmark.PINKY_TIP]

    # Initialize a dictionary to store the status (i.e., True for open and False for close) of each finger of both hands.
    fingers_statuses = {'RIGHT_THUMB': False, 'RIGHT_INDEX': False, 'RIGHT_MIDDLE': False, 'RIGHT_RING': False,
                        'RIGHT_PINKY': False, 'LEFT_THUMB': False, 'LEFT_INDEX': False, 'LEFT_MIDDLE': False,
                        'LEFT_RING': False, 'LEFT_PINKY': False}
```

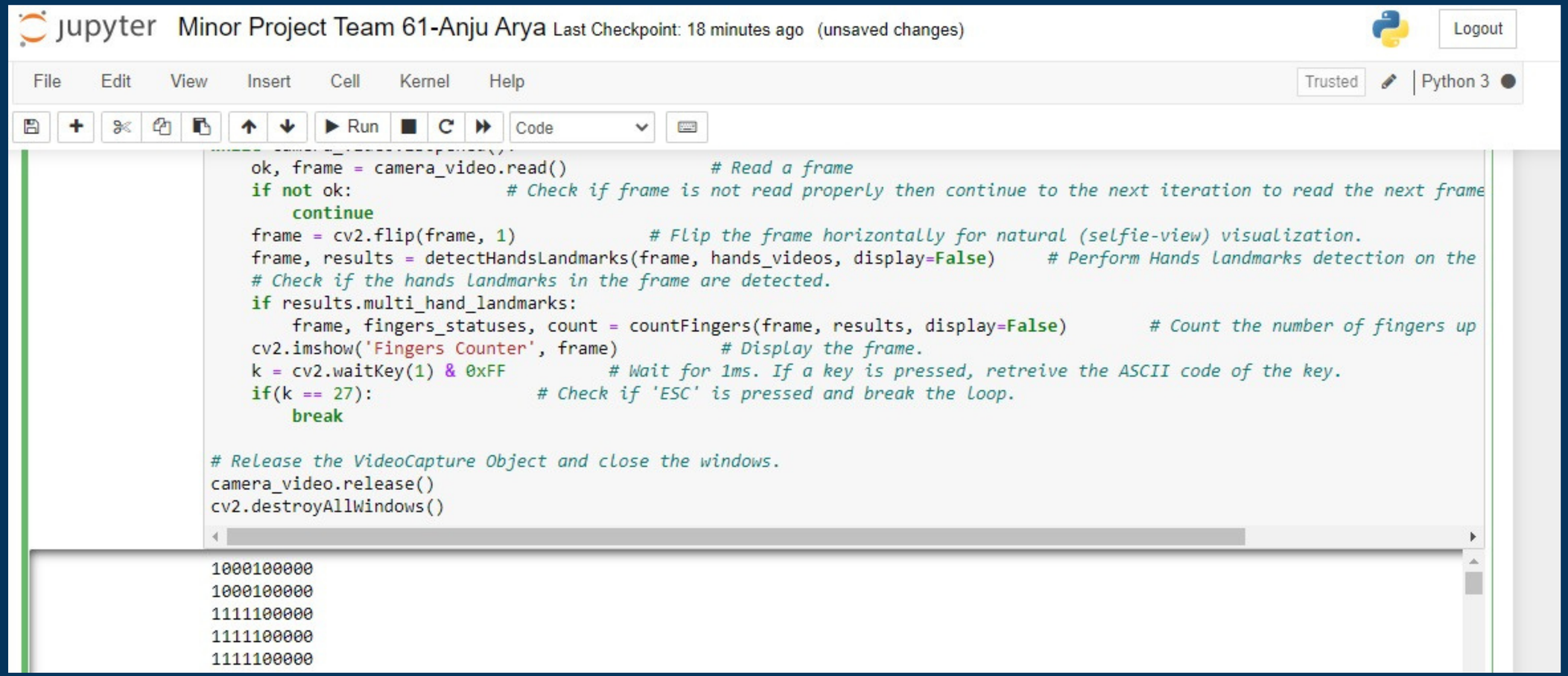
jupyter Minor Project Team 61-Anju Arya Last Checkpoint: 4 minutes ago (autosaved)

File Edit View Insert Cell Kernel Help Trusted Python 3

Run Code

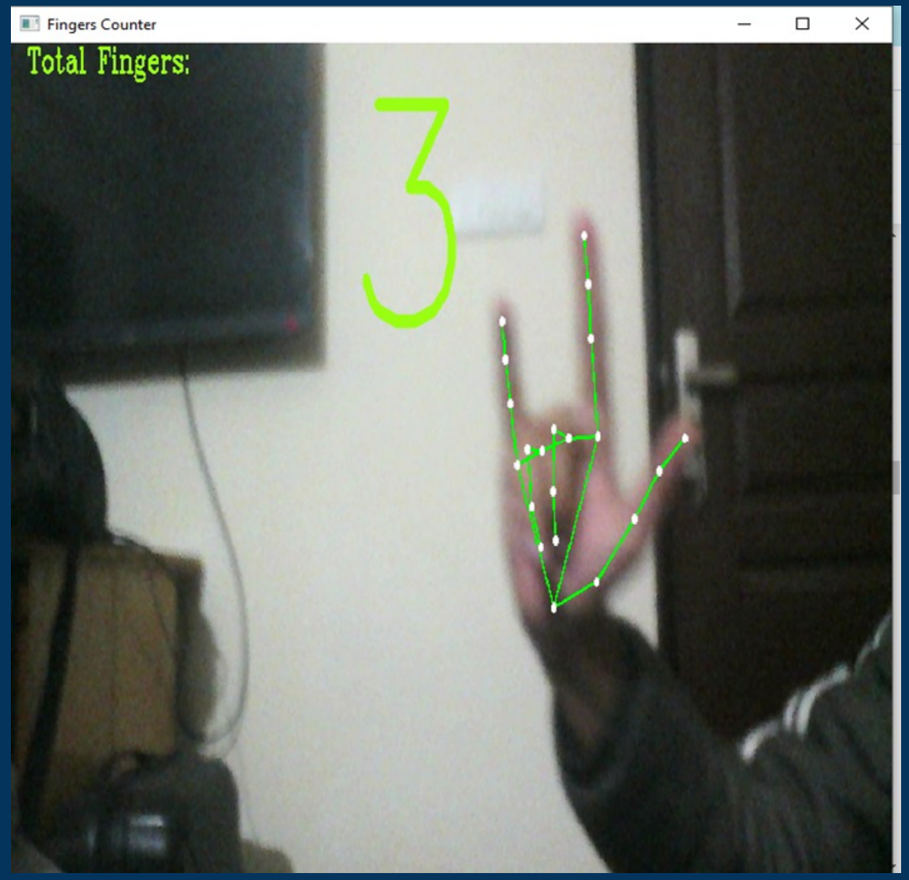
In [ ]:

```
mp_hands = mp.solutions.hands
hands = mp_hands.Hands(static_image_mode=True, max_num_hands=2, min_detection_confidence=0.5)
hands_videos = mp_hands.Hands(static_image_mode=False, max_num_hands=2, min_detection_confidence=0.5)
mp_drawing = mp.solutions.drawing_utils

def detectHandsLandmarks(image, hands, draw=True, display = True):
    # Create a copy of the input image to draw landmarks on.
    output_image = image.copy()

    # Convert the image from BGR into RGB format.
    imgRGB = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)

    # Perform the Hands Landmarks Detection.
    results = hands.process(imgRGB)

    # Check if landmarks are found and are specified to be drawn.
    if results.multi_hand_landmarks and draw:

        # Iterate over the found hands.
        for hand_landmarks in results.multi_hand_landmarks:

            # Draw the hand landmarks on the copy of the input image.
            mp_drawing.draw_landmarks(image = output_image, landmark_list = hand_landmarks,
                                      connections = mp_hands.HAND_CONNECTIONS,
                                      landmark_drawing_spec=mp_drawing.DrawingSpec(color=(255,255,255),
                                                                                   thickness=2, circle_radius=2),
                                      connection_drawing_spec=mp_drawing.DrawingSpec(color=(0,255,0),
                                                                                     thickness=2, circle_radius=2))
```

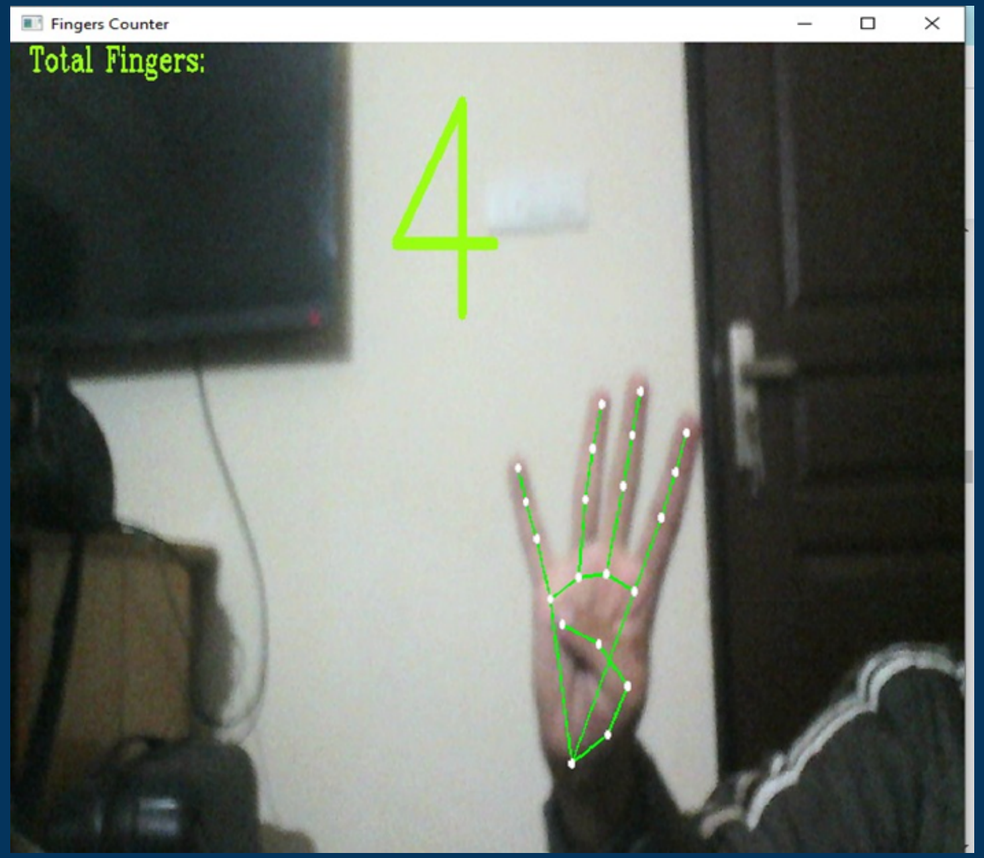

jupyter Minor Project Team 61-Anju Arya Last Checkpoint: 18 minutes ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Help
Trusted
Python 3
Run
Code
ok, frame = camera_video.read() # Read a frame
if not ok: # Check if frame is not read properly then continue to the next iteration to read the next frame
continue
frame = cv2.flip(frame, 1) # Flip the frame horizontally for natural (selfie-view) visualization.
frame, results = detectHandsLandmarks(frame, hands_videos, display=False) # Perform Hands landmarks detection on the
# Check if the hands landmarks in the frame are detected.
if results.multi_hand_landmarks:
frame, fingers_statuses, count = countFingers(frame, results, display=False) # Count the number of fingers up
cv2.imshow('Fingers Counter', frame) # Display the frame.
k = cv2.waitKey(1) & 0xFF # Wait for 1ms. If a key is pressed, retreive the ASCII code of the key.
if(k == 27): # Check if 'ESC' is pressed and break the loop.
break
# Release the VideoCapture Object and close the windows.
camera_video.release()
cv2.destroyAllWindows()
1000100000
1000100000
1111100000
1111100000
1111100000

# OUTPUT

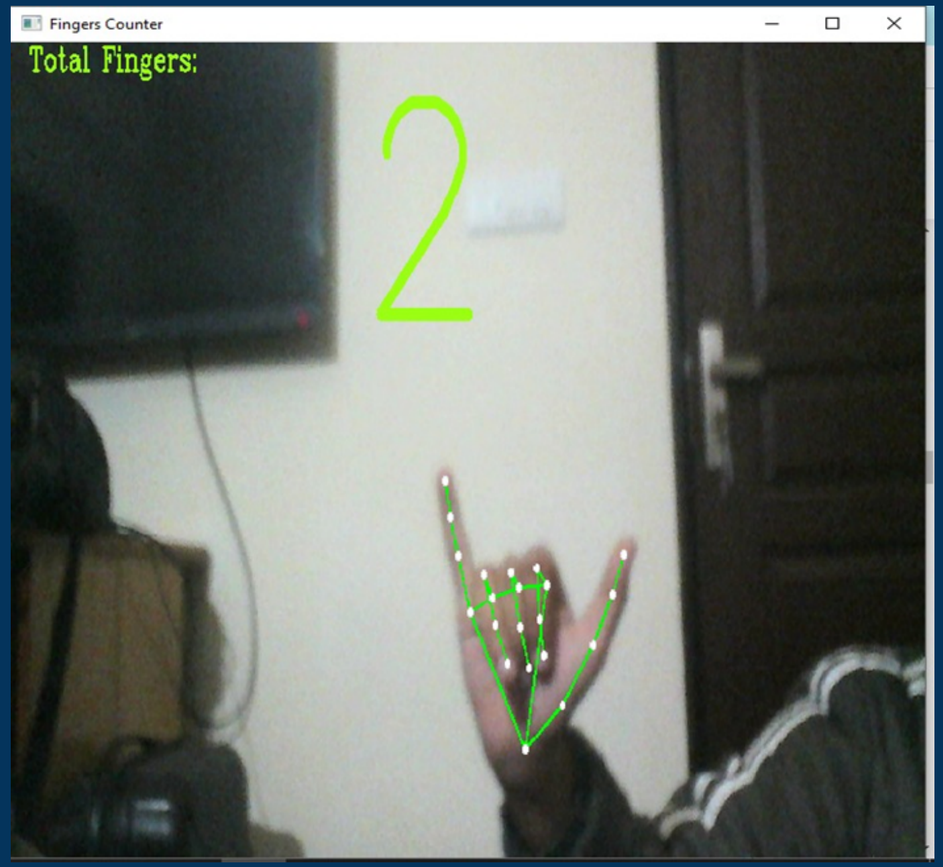

11111100000




1001100000




1111000000




1000100000

# References :


- https://ieeexplore.ieee.org/document/4284820
- https://www.researchgate.net/publication/221932086_Hand_Gesture_Modeling_and_Recognition_using_Geometric_Features_A_Review
- http://www.ijerd.com/paper/Conference-RTEECE/Ver-3/25-1725.pdf
- https://link.springer.com/article/10.1007/s11760-010-0176-6
- https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.659.8098&rep=rep1&type=pdf
- https://wenku.baidu.com/view/d569f90e4a7302768e9939a4.html

# THANK YOU