19AIE303 SIGNAL AND IMAGE PROCESSING
S5 B.Tech CSE(AI)
Project Report
Group 17

Submitted by:

NITHIN SYLESH	AM.EN.U4AIE19044
A VENKATESWARA RAO	AM.EN.U4AIE9014
RV.PANKAJAVALLI	AM.EN.U4AIE19050

December 2021

Department of Computer science and Engineering
Amrita Vishwa Vidyapeetham
Amritapuri Campus
Kollam 690525
Kerala

Detection of Hand and Fingers and Recognise counting using fingers

I. ABSTRACT

Pointing out by hand for originating some gestures is highly useful in terms of human computer interactions. Therefore, people can easily do some tasks by drawing a gesture in the air using their hands in front of a computer camera which translates these gestures to a speech or text to be understood by other people. Part of hand gesture recognition is counting by hand. This project proposes a technique detecting the hands raised and hand gesture numerals which are from 0 to 10 that are pointed out by people to be understood by a computer.

In this work, we present a real-time method for hand gesture recognition using an open source tool called media-pipe and opencv.MediaPipe Hands is a high-fidelity hand and finger tracking solution. It employs machine learning (ML) Pipeline to infer 21 3D landmarks of a hand from just a single frame.Pipeline consists of multiple models working together: A palm detection model that operates on the full image and returns an oriented hand bounding box. A hand landmark model that operates on the cropped image region defined by the palm detector and returns high-fidelity 3D hand keypoints.

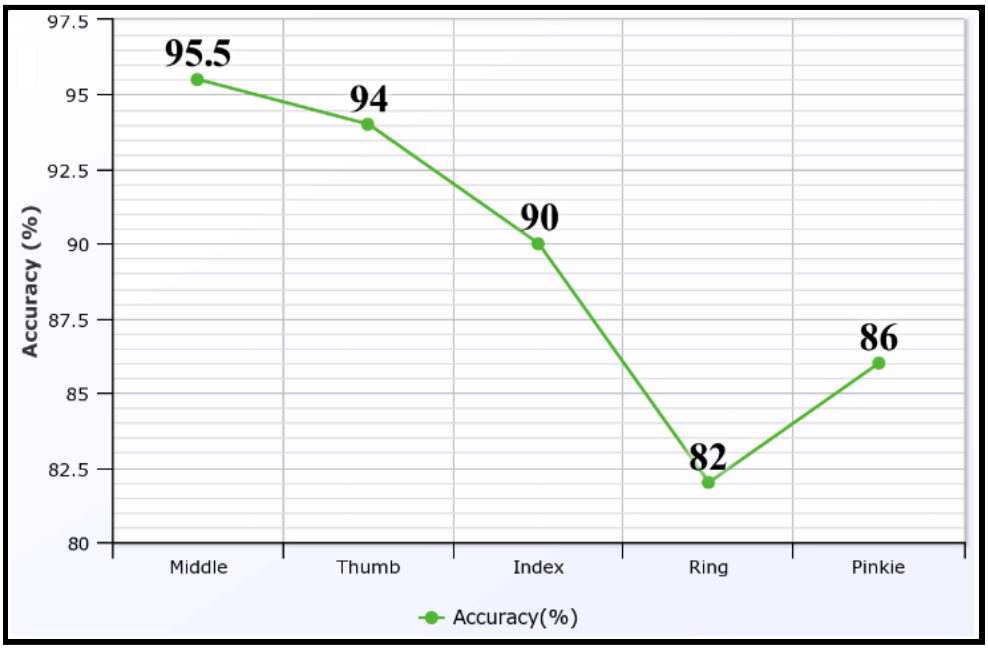
The implementation has been done in python.The output obtained is a real time detection of hands along with the recognition of gesture numericals and prints the numerical

II. OBJECTIVES

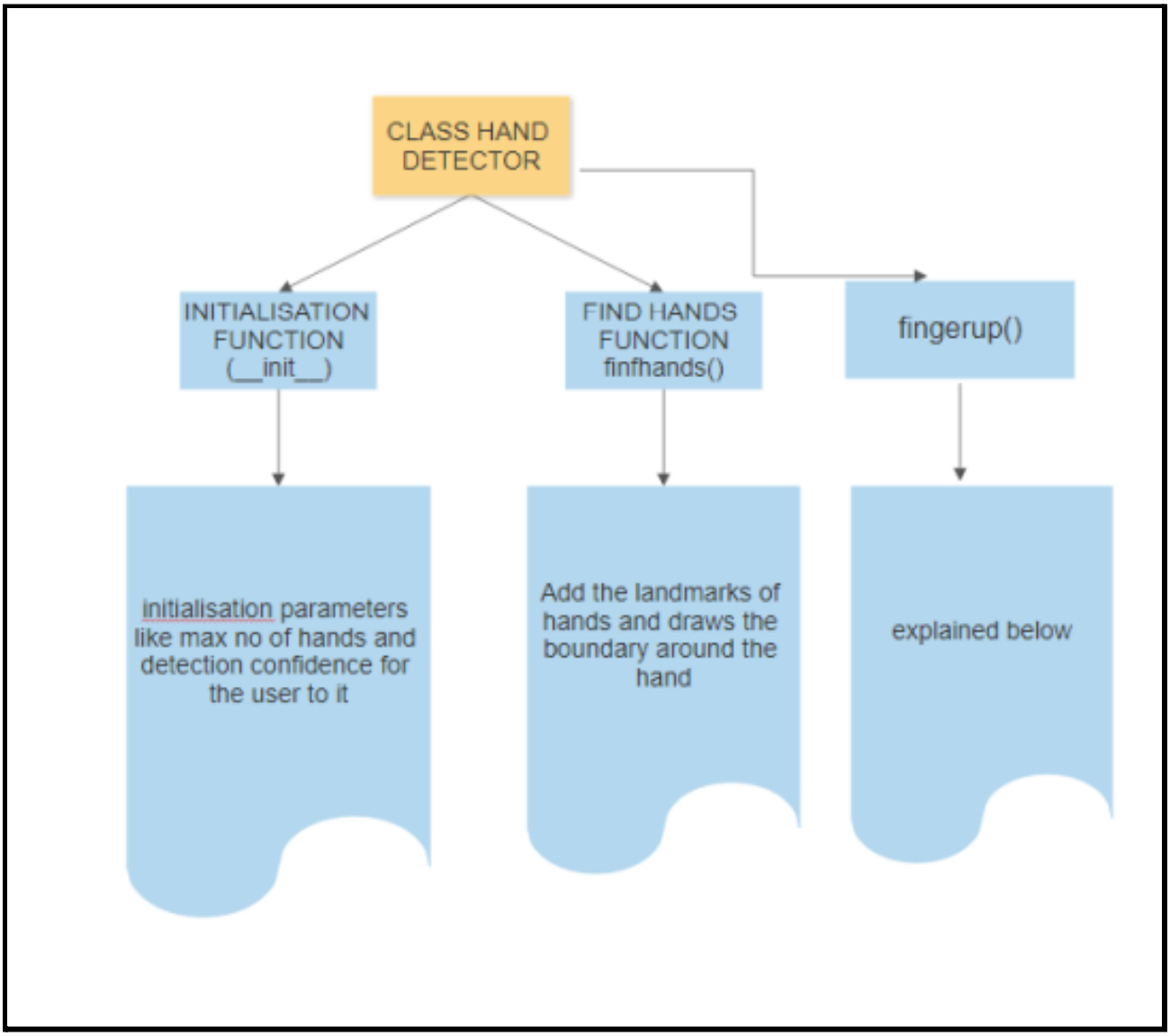
1. Build the hand detector
2. Perform Hand Landmarks Detections
3. Build the fingers counter
4. Print the detected finger count
5. Visualize the hand detection
6. Detects numbers from 0-10

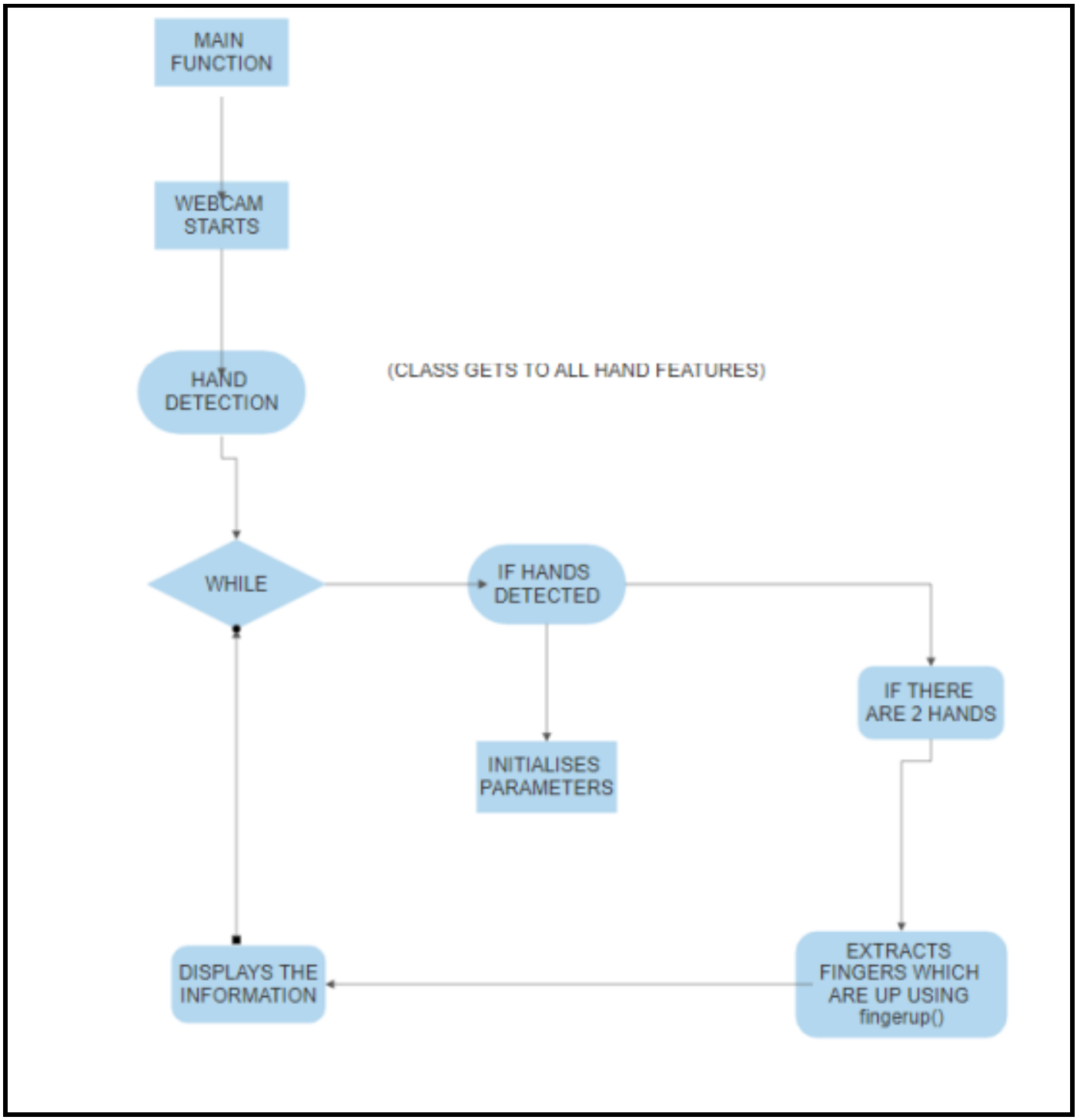
III. ASSUMPTIONS

- We assume that whenever the hand comes into view, it shows fingers of both the hands as with either one of them the finger count code won't run , unless the user shows two hands .
- Secondly we assume that the hand is shown from the bottom of the image such that fingers lie on the relatively upper part of the image with respect to the rest of the hand.
- Does not record multi-digit numbers
- Does not record numbers from one hand, it needs both the hands
- Background should not blend into the fingers as the HSV values which are used to detect the fingers by the algorithm will detects foriegn objects.
- Here is a graph of detecting accuracy :



IV. SYSTEM ARCHITECTURE




MAIN FUNCTION
WEBCAM STARTS
HAND DETECTION
(CLASS GETS TO ALL HAND FEATURES)
WHILE
IF HANDS DETECTED
INITIALISES PARAMETERS
IF THERE ARE 2 HANDS
EXTRACTS FINGERS WHICH ARE UP USING fingerup()
DISPLAYS THE INFORMATION

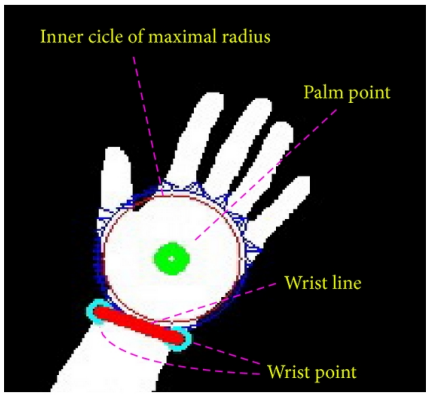
Taken from :media pipe

Core algorithm :

This is a two step process, furthermore to count fingers its a three step process:
Firstly, the mediapipe module- a machine learning model- is used to get features of hands the two models involved in it are :

Palm Detection Model:

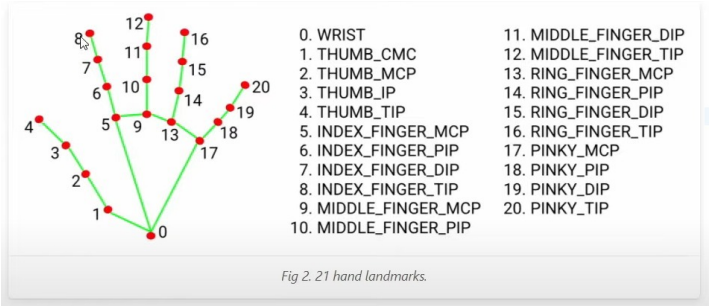
30, 000 + images of hands have trained this model to get features of palms. Palms are primarily detected by segmenting the hand and then getting maximum radius from the center of the palm to the edge of the palm and the wrist on the other side.



Next step is to mask the detected palm to get only the fingers.

Hand Landmark Model :

After getting the image without the palm a horizontal line next to the thumb is divided into four to get the regions ro identify each finger :



THIS WAS A LIFE SAVER : [Real-Time Hand Gesture Recognition Using Finger Segmentation]

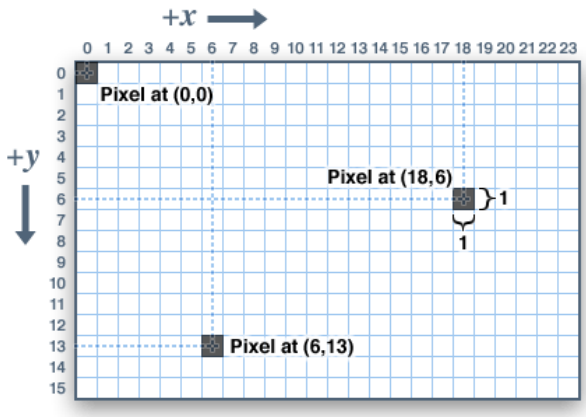
NOW as we have obtained fingers and their landmarks our next step is to count the fingers which are up :

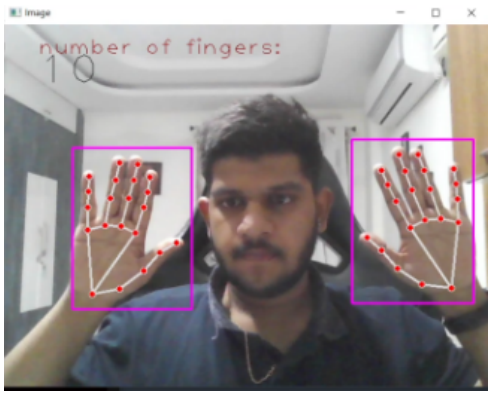
```
def fingersUp(self,myHand):      # main principle is that for each finger let the
    """                                                  #https://google.github.io/media
    Finds how many fingers are open and returns in a list.
    Considers left and right hands separately
    :return: List of which fingers are up
    """
    myHandType =myHand["type"]
    myLmList = myHand["lmList"]
    if self.results.multi_hand_landmarks:
        fingers = []
        # Thumb
        if myHandType == "Right":
            if myLmList[self.tipIds[0]][0] > myLmList[self.tipIds[0] - 1][0]:


                fingers.append(1)       #if thumb is up appends 1 to the fingers li
            else:
                fingers.append(0)       #if thumb is down appends 0 to the fingers
        else:
            if myLmList[self.tipIds[0]][0] < myLmList[self.tipIds[0] - 1][0]:
                                                                                    # t
                fingers.append(1)
            else:
                fingers.append(0)

        # 4 Fingers
        for id in range(1, 5):
            if myLmList[self.tipIds[id]][1] < myLmList[self.tipIds[id] - 2][1]:   #
                fingers.append(1)
            else:
                fingers.append(0)
    return fingers
```

This is the core implementation :
It follows a basic law:
For the ease of explanation :





Imagine two hands posted up on this coordinate system now coming to four fingers other than the thumb it follows the principle that the tip coordinates will be lesser from the coordinates of Lower part of the same finger mentioned with another lesser landmark number [-2 per say] and for the thumb as the left thumb when closed will appear to the right on the screen and when closed the coordinates increase from before and then only it is closed otherwise it is open , vice vera for the right thumb .

V. INDIVIDUAL CONTRIBUTION

Nithin	AM.EN.U4AIE19044	Base code of hand Detection() modules implementation and foolproofing
VENKAT	AM.EN.U4AIE9014	finger up() detection including integration of both hands to display counts from 5-10 digits.
PANKAJA	AM.EN.U4AIE19050	Assisting with errors in code and literature review.

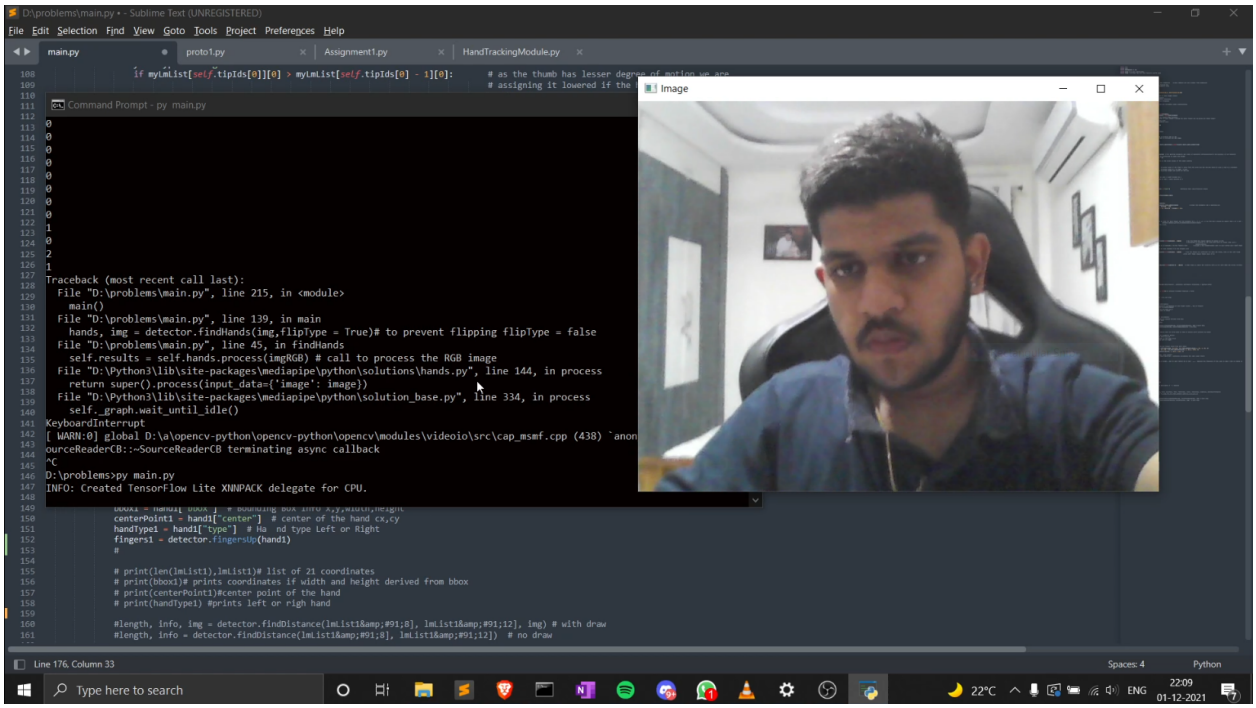
VI. INPUT

- **Video input**

No other way of input

VII. OUTPUT

With no hands :



With both hands :

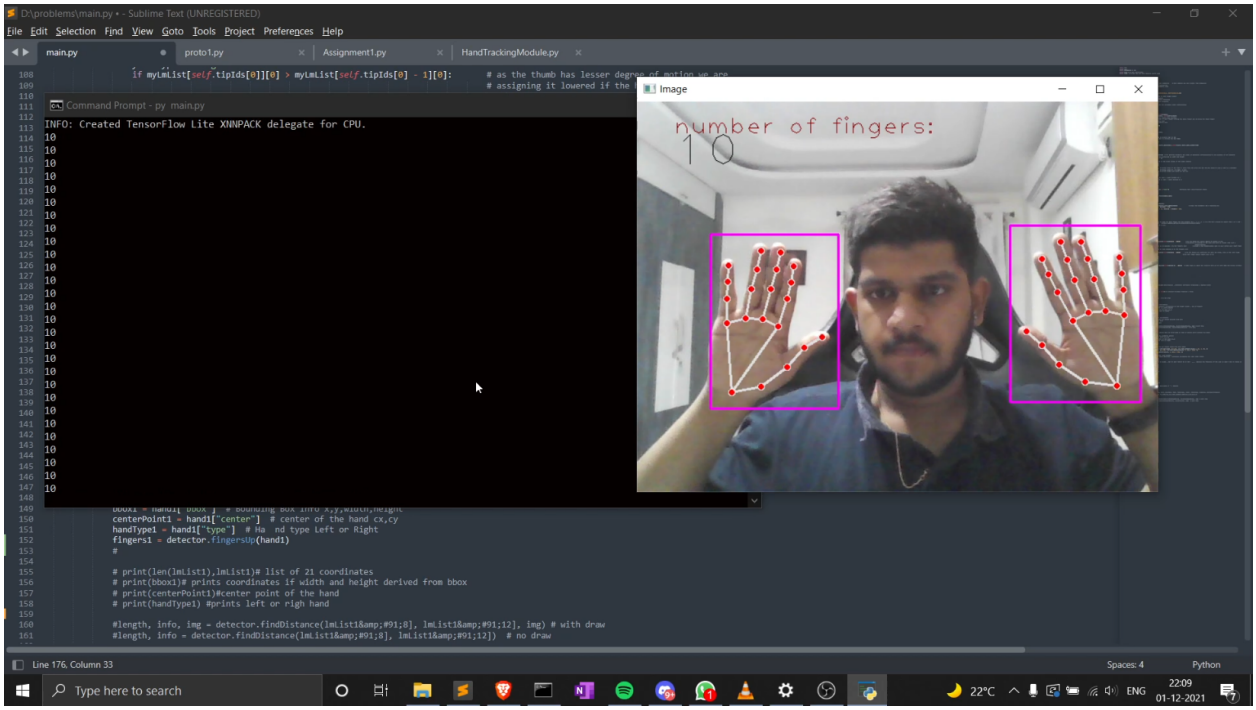


Image
number of fingers:
10
Command Prompt - py main.py
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.

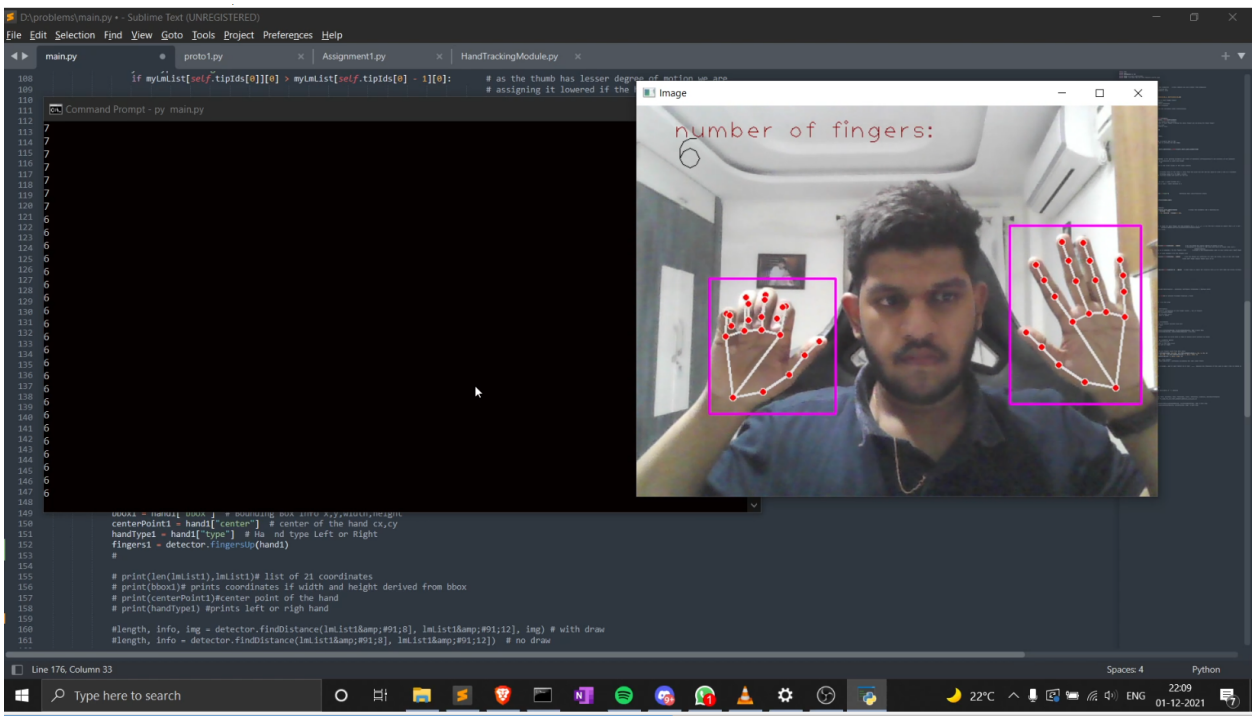


Image
number of fingers:
6
Command Prompt - py main.py

VIII. CONCLUSION

Counting numbers is important for numerous sectors of life. This research and implementation proved that counting by finger-hand gesture can be originated from people by using a computer system,through a menial gesture using their hands. Technically, an open source cross platform framework building multimodal applied machine learning pipelines pre tested with 30,000 hand images has been proposed so as to predict number based at the hand gesture.The framework has great amount of flexibility and thus may be used for numerous applications with accurate results.We successfully used the framework to count up to ten fingers.

Pivotal efforts were made to implement number detection from 5 - 10 and involves two hands, and the efforts did reap.Later discussions to make the model record multi-digit numbers was made , though unsuccessful we made vital progress . Firstly, we were able to implement multi-digit output but only for non repeating numbers , For example: 123456,76893.However, the same results couldn't translate into recording and showing recurring numbers , For example:66557883 , 999999 and so on. If non recurring digits had to be recorded then the user will have to mention the place of the digit he is inputting i.e tens , hundreds , thousands etc… then the code can be implemented respectively by assigning a respective data-type for each place and removing recurring digits to display. As this is menial for the user, for this project we have stuck to detecting numbers from 0 - 10 for this project.

The code was written in Python with Opencv and Mediapipe.The framework was tested on a machine featuring intel core i7 9th gen 2.40 GHz processor and 16 GB ram on windows 10 platform.

IX. REFERENCE

[1] S. S. Rautaray and A. Agrawal, "Vision based hand gesture recognition for human computer interaction: a survey," Artificial Intelligence Review, vol. 43, pp. 1-54, 2015.

[2]Facebook.OculusQuestHandTracking.
https://www.oculus.com/blog/oculusconnect-6-introducing-hand-tracking-onoculus-quest-facebook-horizon-and-more/. 1

[3] Liuhao Ge, Hui Liang, Junsong Yuan, and Daniel Thalmann. Robust 3d hand pose estimation in single depth images: from single-view cnn to multi-view cnns. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3593–3601, 2016. 1

[4] Liuhao Ge, Hui Liang, Junsong Yuan, and Daniel Thalmann. Robust 3d hand pose estimation from single depth images using multi-view cnns. IEEE Transactions on Image Processing, 27(9):4422–4436, 2018. 1

[5] Liuhao Ge, Zhou Ren, Yuncheng Li, Zehao Xue, Yingying Wang, Jianfei Cai, and Junsong Yuan. 3d hand shape and pose estimation from a single rgb image. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 10833–10842, 2019. 1

[6] Google. Tensorflow lite on GPU. https://www.tensorflow.org/lite/ performance/gpu advanced. 3

[7] Google. Tensorflow.js Handpose. https://blog.tensorflow.org/2020/03/face-and-hand-tracking-in-browser-withmediapipe-and-tensorflow js.html. 1

[8] Yury Kartynnik, Artsiom Ablavatski, Ivan Grishchenko, and Matthias Grundmann. Real-time facial surface geometry from monocular video on mobile gpus. CoRR, abs/1907.06724, 2019. 2

[9] Tsung-Yi Lin, Piotr Dollar, Ross B. Girshick, Kaiming He, ´ Bharath Hariharan, and Serge J. Belongie. Feature pyramid networks for object detection. CoRR, abs/1612.03144, 2016. 2

[10] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollar. Focal loss for dense object detection. ´ CoRR, abs/1708.02002, 2017.