HAND GESTURE DETECTION USING ARTIFICIAL INTELLIGANCE.

TEAM MEMBERS:

- SAI TEJA BALUSU (**TEAM LEAD**)
- RAKESH NATH DHULIPALLA
- JAI SAI MALAKALAPALLI

ABSTRACT:

This project is based on sign language recognition which includes image processing. Computer vision using CNN (Convolutional Neural Network) will capture image or live video footage through the camera and process the hand gestures using image processing techniques i.e., packages like keras and TensorFlow which will process the image and convert them into understandable language. Priorly we will be testing and training the dataset for immediate response from the live footage or image captured. Helpful for disabled people and we can also help people in public surveillance.

INTRODUCTION:

Now a days threats were increasing day by day where there is also advancement in the security and security technologies. Artificial intelligence is in everything we use in our daily lifestyle like from Amazon (A to Z) till apple voice assistant Siri. We are upgrading in technologies day by day in order to make our life easy and secure.

CNN is a type of Artificial neural network which is used for image recognition and processing i.e., specifically design for processing the pixel data. It is used for facial recognition as we using now in our project.

So, the project which we have taken is based on hand gesture recognition which is a helpful project for disabled people in the society.

RELATED WORK:

- https://data-flair.training/blogs/sign-language-recognition-python-ml-opencv/
- https://techvidvan.com/tutorials/hand-gesture-recognition-tensorflow-opencv/
- https://gogul.dev/software/hand-gesture-recognition-p1
- https://aihubprojects.com/hand-detection-gesture-recognition-opencv-python/

PROBLEM SPECIFICATION:

Our main motive is to help people who were in a bad situations at public places i.e., they can't even call for a rescue. So, if we can implement this in public surveillance cameras which can recognise the hand gestures who were in a threat, we can immediately rescue them by notifying cops.

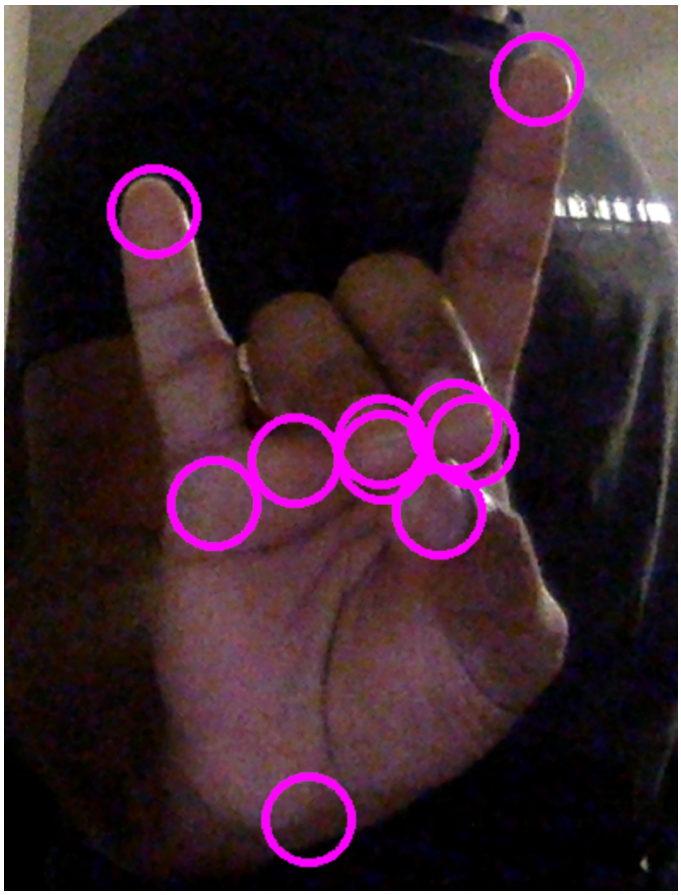
DATASET:

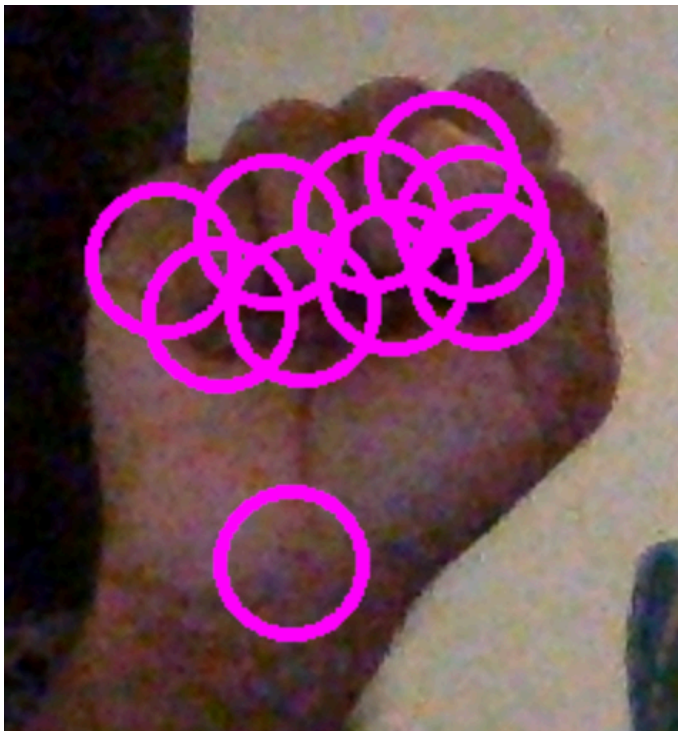
Instead of taking previous dataset we used live dataset method to take live datasets from raw video feed which helps us to process real world gestures.

```
# creating the set of data for training.
for i in range(1,numGesture+1,1):
   prompt = 'Name of Gesture #'+str(i)+' '
   name=input(prompt)
   gestureNames.append(name)
print(gestureNames)

while True:
    ignore, frame = cam.read()
    frame=cv2.resize(frame,(width,height))
    handData=findHands.Marks(frame)
    # reading the data for trainig.
    if train==True:
        if handData!=[]:
            print('Please gesture ',gestureNames[trainCount],': Press "t" when ready')
            if cv2.waitKey(1) & 0xff==ord('t'):
                print("1")
                knownGesture=findDistances(handData[0])
                knownGestures.append(knownGesture)
                trainCount = trainCount+1
                if trainCount==numGesture:
                    train=False
```

DATA FROM LIVE FEED:





PROBLEM ANALYSIS:

Basically, in public areas we have lot of surveillance cameras when people get into bad situations, they were unable to call for a help so for this we can use the surveillance cameras to analyse the people's hand gestures and notify police for an immediate rescue.

DESIGN AND MILESTONES:

We are creating test and train data sets from raw computer video feed which is having thousands of hand gesture images to predict. We will analyse this image feed using image processing techniques like opencv and mediapipe to convert the analysed image into understandable language.

PROPOSED METHOD:

We are using CNN (Convolutional Neural Network), opencv and mediapipe for Implementastion.

CODE SNIPPETS:

```
import cv2
print(cv2.__version__)
import numpy as np
import time

class mpHands:
    import mediapipe as mp
```

```
class mpHands:
    import mediapipe as mp
    # default constructor method to self execute
    def __init__(self,maxHands=1,tol1=.5,tol2=.5):
        self.hands=self.mp.solutions.hands.Hands(False,maxHands,tol1,tol2)
    # marking the hands to calculate the distances.
    def Marks(self,frame):
        myHands=[]
        frameRGB=cv2.cvtColor(frame,cv2.COLOR_BGR2RGB)
        results=self.hands.process(frameRGB)
        if results.multi_hand_landmarks != None:
            for handLandMarks in results.multi_hand_landmarks:
                myHand=[]
                for landMark in handLandMarks.landmark:
                    myHand.append((int(landMark.x*width),int(landMark.y*height)))
                myHands.append(myHand)
        return myHands
```

```
# finding the distances between the fingers and key points.
def findDistances(handData):
    distMatrix=np.zeros([len(handData),len(handData)],dtype='float')
    for row in range(0,len(handData)):
        for column in range(0,len(handData)):
            distMatrix[row][column]=((handData[row][0]-handData[column][0])**2+(handData[row][1]-handData[column][1])**2) **(1./2.)
    return distMatrix
```

```
# trying to minimise the error.-
def findError(gestureMatrix,unknownMatrix,keyPoints):
    error=0
    for row in keyPoints:
        for column in keyPoints:
            error=error+abs(gestureMatrix[row][column]-unknownMatrix[row][column])
    return error
# finding the distances trying to calculating the gesture by probability.
def findGesture(unknownGesture,knownGestures,keyPoints,gestureNames,tol):
    errorArray = []
    for i in range(0,len(gestureNames),1):
        error=findError(knownGestures[i],unknownGesture,keyPoints)
        errorArray.append(error)
    errorMin = errorArray[0]
    minIndex=0
    for i in range(0,len(errorArray),1):
        if errorArray[i]<errorMin:
            errorMin=errorArray[i]
            minIndex = i
    if errorMin < tol:
        gesture =gestureNames[minIndex]
    if errorMin >= tol:
        gesture="Unknown"
    return gesture

width=1280
height=720
cam=cv2.VideoCapture (0)
cam.set(cv2.CAP_PROP_FRAME_WIDTH, width)
cam.set(cv2.CAP_PROP_FRAME_HEIGHT,height)
cam.set(cv2.CAP_PROP_FPS, 30)
cam.set(cv2.CAP_PROP_FOURCC,cv2.VideoWriter_fourcc(*'MJPG'))
findHands=mpHands(1)
time.sleep(5)
```

```
# important parts of the hands
keyPoints=[0,4,5,9,13,17,8,12,16,20]
train=True
tol = 1500 # If the error is less than the tolerance, then we'll say we haveamatch, otherwise we don't.
trainCount = 0
numGesture = int(input('How many gestures do you want to train on? '))
gestureNames = []
knownGestures = []
```

```
# creating the set of data for training.
for i in range(1,numGesture+1,1):
    prompt = 'Name of Gesture #'+str(i)+' '
    name=input(prompt)
    gestureNames.append(name)
print(gestureNames)

while True:
    ignore, frame = cam.read()
    frame=cv2.resize(frame,(width,height))
    handData=findHands.Marks(frame)
    # reading the data for trainig.
    if train==True:
        if handData!=[]:
            print('Please gesture ',gestureNames[trainCount],': Press "t" when ready')
            if cv2.waitKey(1) & 0xff==ord('t'):
                print("1")
                knownGesture=findDistances(handData[0])
                knownGestures.append(knownGesture)
                trainCount = trainCount+1
                if trainCount==numGesture:
                    train=False
    #working model of the gesture detection .
    if train == False:
        if handData!=[]:
            unknownGesture=findDistances(handData[0])
            myGesture = findGesture(unknownGesture, knownGestures, keyPoints, gestureNames, tol)
            #error=findError(knownGesture, unknownGesture, keyPoints)
            # writing text onto the frame
            cv2.putText(frame, myGesture,(100,175),cv2.FONT_HERSHEY_SIMPLEX, 3, (255,0,0),8)
    for hand in handData:
        for ind in keyPoints:
            cv2.circle(frame,hand[ind],25,(255,0,255),3)
    cv2.imshow('my WEBcam', frame)
    cv2.moveWindow('my WEBcam',0,0)
    if cv2.waitKey(1) & 0xff ==ord('q'):
        break
cam.release()
```

IMPLEMENTATION:

I. CREATING THE DATASET:

Primarily in increment 1 we have taken a dataset from internet for testing. In the final implementation we are creating our own dataset. For that we will be taking a live feed form a cam and will be detected the hand from frame to frame in region of interest and will be directed to the datasets of test and train for future use.

II. CONVOLUTIONAL NEURAL NETWORK WILL BE TRAINED ON DATASET THAT IS CAPTURED:

We need to be use image data generator of keras to load the data of test and train set from the directory, then plot the loaded dataset images using plot image's function. Now we design the Convolutional Neural Network with some hyper parameters depending on some trial and error.

III. GESTURE PREDICTION:

We will be creating a square box for detection of Roi and calculating the average likewise in the dataset. This is done for identification of focal point of object. Now we detect the gesture representation with the test image. We use keras. models. load _ models to load prior saved data and Roi can be grubbed the threshold image which predicts hand as input.

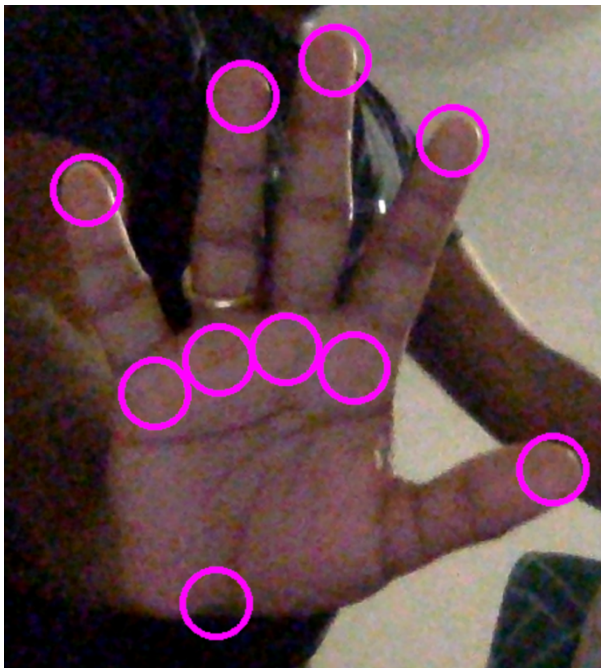
DATA PROCESSING:

We take raw web can video feed from the computer and uses opencv for computer vison and identify all the hand and then we use medipipe to locate the key values of the hand as every value might not be needed to assess the data. On a regular hand there can be more than 20 key values but, we considering only the important key values.

EXPLANATION:

```
Downloads — -zsh — 80×24
Last login: Tue May  3 21:46:15 on ttys000
(base) rakeshdhulipalla@Rakeshs-MBP ~ % cd downloads
(base) rakeshdhulipalla@Rakeshs-MBP downloads % python3 handGuster.py
4.5.5
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.
How many gestures do you want to train on? 1
Name of Gesture #1 1
['1']
Please gesture  1 : Press "t" when ready
```

Here we are training with only one gesture for explanation purpose but, in reality we can use a lot.



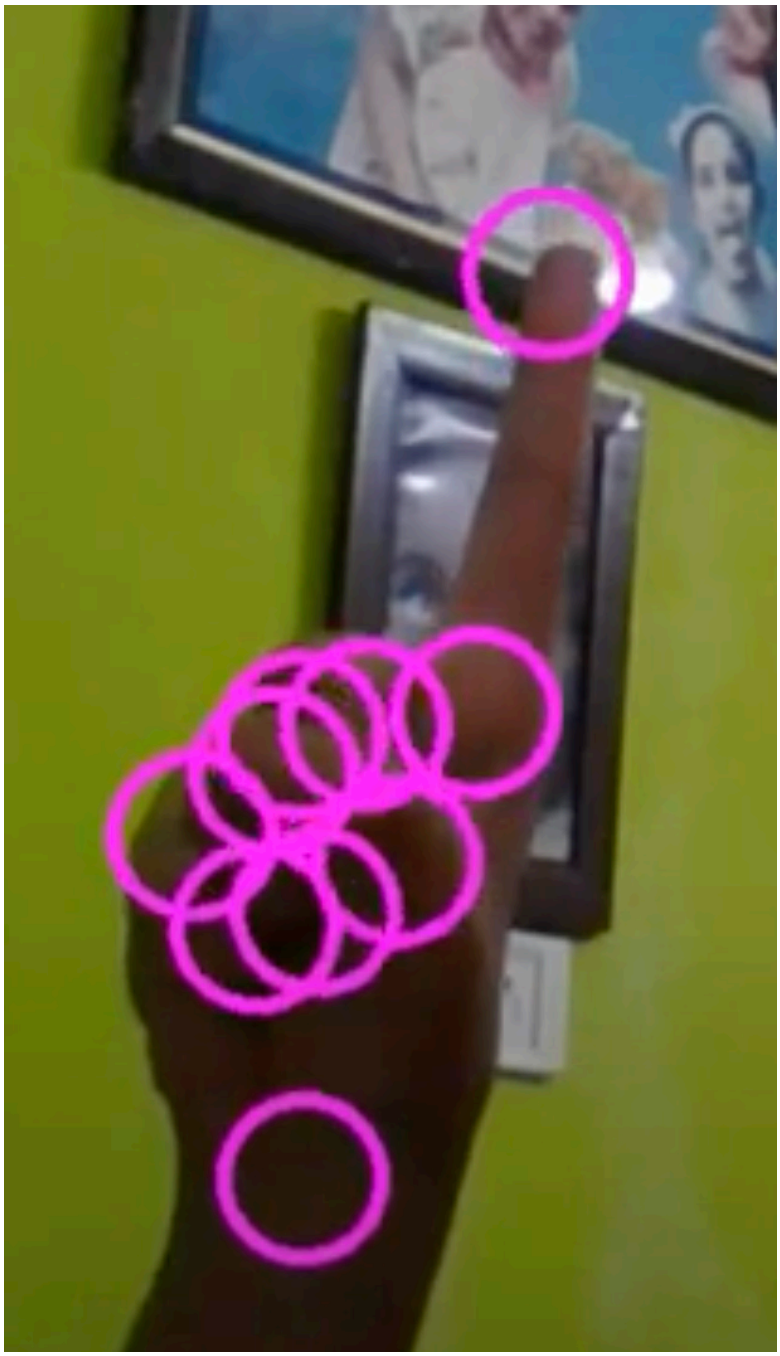
This is a picture taken when we are training the data. So you can observe the pink circles i.e. the key points of the hand that we use to recognise the future gestures.

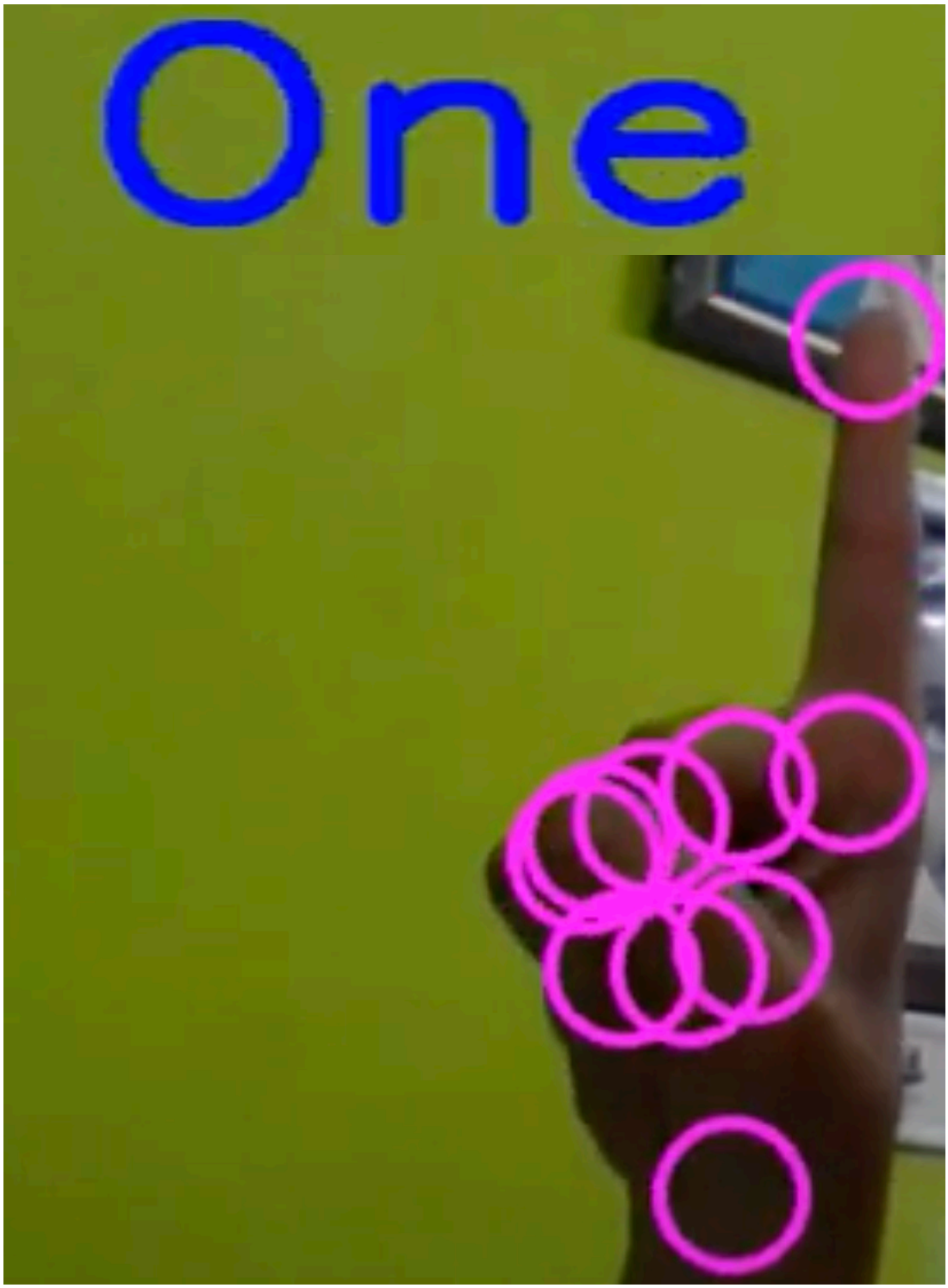
OUTPUTS:

- INPUT IN TERMINAL:

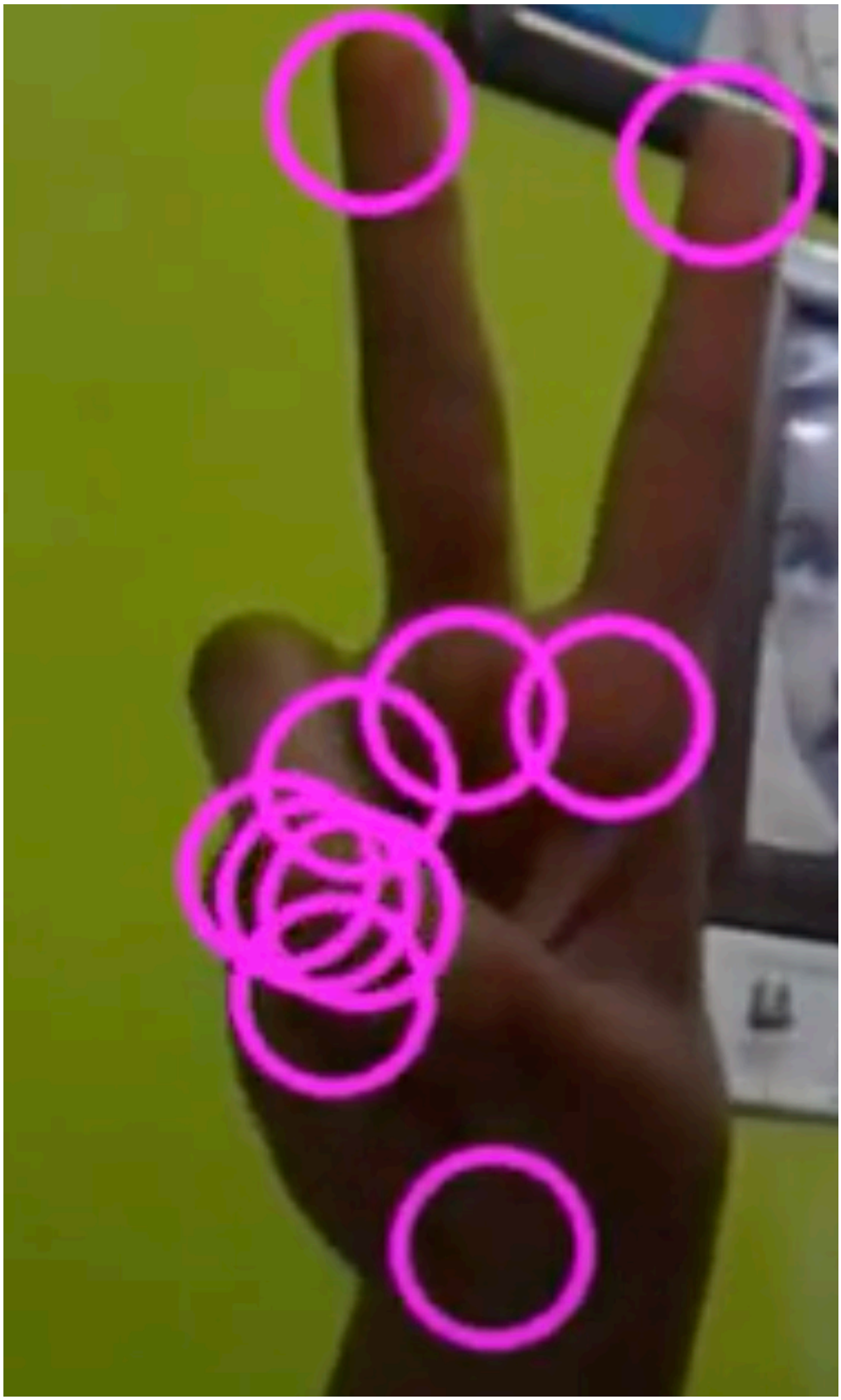
```
Name of Gesture #1 One
Name of Gesture #2 Two
Name of Gesture #3 Three
Name of Gesture #4 Four
Name of Gesture #5 Five
Name of Gesture #6 Power to the people
Name of Gesture #7 Go forth and prosper
Name of Gesture #8 All Ok
Name of Gesture #9 guns up
['One', 'Two', 'Three', 'Four', 'Five', 'Power to the people', 'Go forth and prosper', 'All Ok', 'guns up']
```

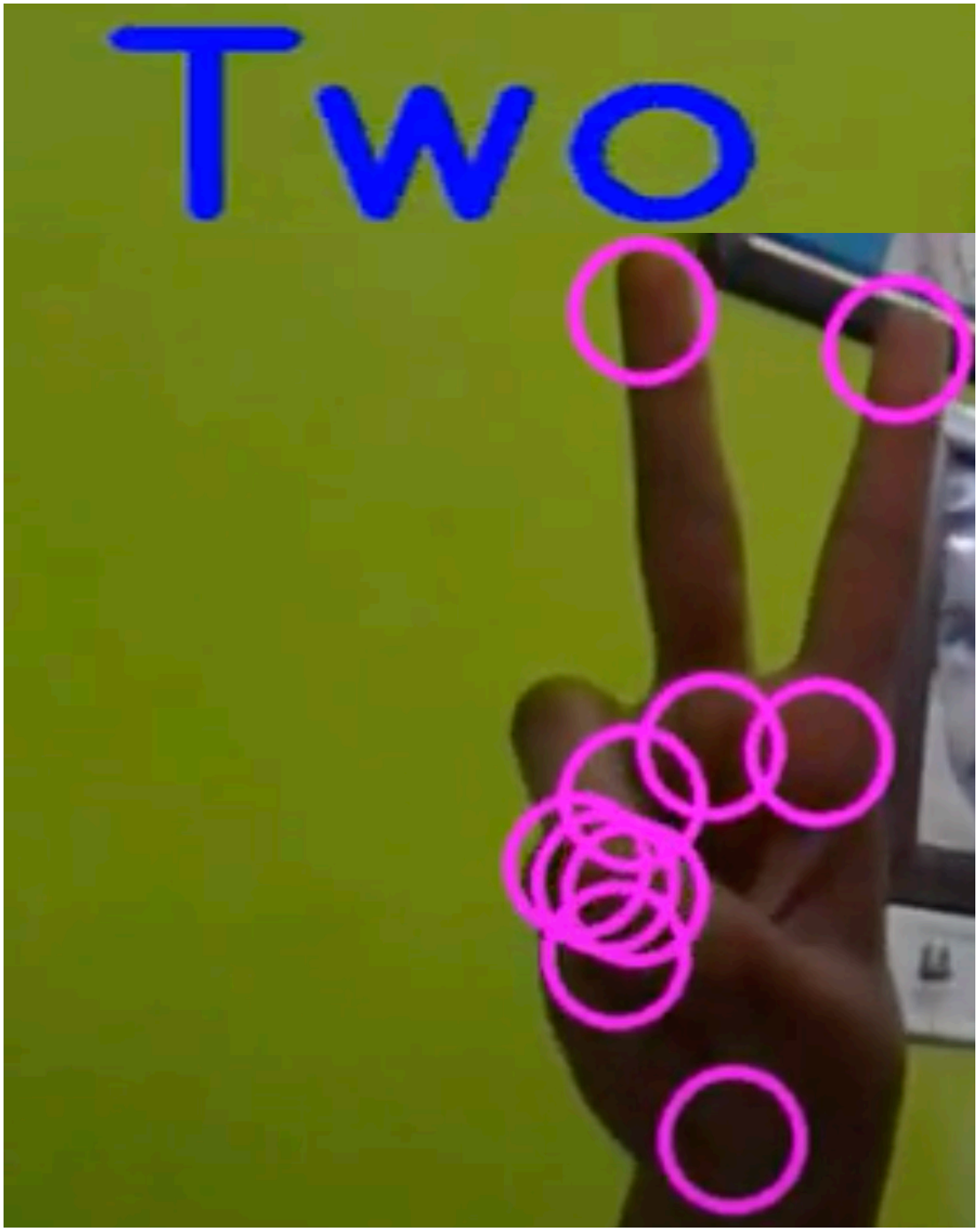
- FOR 1:




One

- FOR 2:




Two

EXPERIMENTAL SETTINGS:

We are trying to implement the gesture detection and converting them into sentences which will help in the implementation of the gesture to the fullest.

VALIDATION METHODS:

It is based on the training data that we used to tarin the model. Here the training data is given using opencv and mediapipe using love video feed.

LIMITATIONS:

- The model might get confused due to hand gestures which are similar such as ok and number 3.
- Sometimes we will be getting the wrong predictions due to lack of knowledge in hand gestures.
- If the Hand crosses the square box of ROI the hand cannot be recognised.

FUTURE WORK:

We can increase the training and testing data for better probable results.

PROJECT MANAGEMENT :

Here we are using opencv and mediapipe where they are used to solve computer vision problem and solve the computer vision problems which helps to take video input, create, train and test the working model.

IMPLEMENTATION STATUS REPORT.

- **WORK COMPLETED:**
  We have successfully able to create hand gesture recognition system which will recognise the hand gives the output of the recognized sign.

- **DESCRIPTION:**
  We are trying to capture the hand from entire frame and trying to determine the key areas of the hand such as figure tips, then calculate the distances between the key areas to determine the gesture from the trained dataset.

- **RESPONSIBILITY:**

  I. Sai Teja Balusu ----------processing of data.
  II. Rakesh Nath Dhulipalla ----validating output.
  III. Jai Sai Malakalapalli ---------training data.

- **CONTRIBUTIONS:**

  I. Sai Teja Balusu ----------35/100
  II. Rakesh Nath Dhulipalla ---- 35/100
  III. Jai Sai Malakalapalli ---------30/100

- **ISSUES/CONCERNS:**

  I. The main issue is to train the data properly as different individuals has different types of gestures for the same gesture which will be difficult to identify.
  II. Due to the small training dataset and limited gestures, it is not highly applicable in every aspect of life.
  III. While making our own dataset using video recorder it should be properly captured in order to train the data to recognise.

REFERENCES:

- https://aihubprojects.com/hand-detection-gesture-recognition-opencv-python/
- https://github.com/samurkubraa/Hand-Signs-Classification
- https://github.com/Aryan7Mohan/Hand_Sign_Recognition

GITHUB LINK:

- https://github.com/saitejabalusu3/AI_PROJECT_GROUP_09_HANDSIGN_RECOUGNITION