REAL TIME HUMAN DETECTION

A Mini Project report submitted to

Jawaharlal Nehru Technological University, Hyderabad

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

Submitted By

RAVULA DEERAJ REDDY	**19641A0502**
GAJA SAI CHANDU	**19641A0503**
BASANI MANOJ KUMAR	**19641A0526**
MOHAMMED MUZAKKIR BIN RAHMAN	**19641A0504**

Under the guidance of

Mrs. A. SWETHA

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VAAGDEVI COLLEGE OF ENGINNERING

(UGC Autonomous, Accredited by NBA, Accredited by NAAC with "A")

Bollikunta, Khila Warangal (Mandal), Warangal Urban-506 005 (T.S)

VAAGDEVI COLLEGE OF ENGINEERING

(UGC Autonomous, Accredited by NBA, Accredited by NAAC with "A")

Bollikunta, Khila Warangal (Mandal), Warangal Urban-506 005 (T.S)

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the MINI PROJECT Phase Report entitled **"REAL TIME HUMAN DETECTION"** is being submitted **by R.DEERAJ REDDY(19641A0502),G.SAI CHANDU(19641A0503), B.MANOJ KUMAR(19641A0526), MOHAMMED MUZAKKI BIN RAHMAN(19641A0504**) in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** to **Vaagdevi College of Engineering, Bollikunta** during the academic year **2022-2023.**

Project Guide
Mrs.A.SWETHA
(Assistant Professor)

Head of the Department
Dr.N.Sathyavathi
(Professor)

External Examiner

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr. K.PRAKASH**, Principal, Vaagdevi College of Engineering for making us available all the required assistance and for his support and inspiration to carry out this MINI Project in the institute.

We extend our heartfelt thanks to **Dr.N.SATHYAVATHI**, Head of the Department of CSE, Vaagdevi College of Engineering for providing us necessary infrastructure and thereby giving us freedom to carry out the MINI Project

We express heartfelt thanks to the guide, **A.SWETHA**, Assistant Professor, Department of CSE for his constant support and giving necessary guidance for completion of this MINI Project

Finally, we express our sincere thanks and gratitude to our family members, friends for their encouragement and outpouring their knowledge and experiencing throughout thesis.

RAVULA DEERAJ REDDY	**(19641A0502)**
GAJA SAI CHANDU	**(19641A0503)**
BASANI MANOJ KUMAR	**(19641A0526)**
MOHAMMED MUZAKKIR BIN RAHMAN	**(19641A0504)**

ABSTRACT

- In this python project, we are going to build the Human Detection and Counting System through Webcam or you can give your own video or images. This is an intermediate level deep learning project on computer vision, which helped us to grab the very basics of AI, ML and DL.
- The system will tell how many people are present in the screen at any given time that is it can give real time analysis.
- Our project will be using basics of DL (majorly), through which will be defining different sort of functions to capture the human body in real time and then following the capture we will be counting the number of humans detected.
- We will be dividing the given image, video or live webcam image to the model as an input which will then pass through some custom made functions which created through libraries like OpenCV, NumPy. The functions will be dividing the input into different frames to analyze and show the output.

TABLE OF CONTENTS

1. INTRODUCTION

1.1. OVERVIEW

- Over the recent years, detecting human beings in a video scene of a surveillance system is attracting more attention due to its wide range of applications in abnormal event detection, human gait characterization, person counting in a dense crowd, person identification, gender classification, fall detection for elderly people, etc.
- The scenes obtained from a surveillance video are usually with low resolution.
- Most of the scenes captured by a static camera are with minimal change of background.
- Objects in the outdoor surveillance are often detected in far field. Most existing digital video surveillance systems rely on human observers for detecting specific activities in a real-time video scene.
- However ,there are limitations in the human capability to monitor simultaneous events in surveillance displays.
- Hence, human motion analysis in automated video surveillance has become one of the most active and attractive research topics in the area of computer vision and pattern recognition

1.2 OBJECTIVE

The main objective of human detection is to identify and locate one or more effective targets from still image or video data. It comprehensively includes a variety of important techniques, such as image processing, pattern recognition, artificial intelligence and machine learning.

1.3 COMPUTER VISION

Computer vision is an interdisciplinary scientific field that deals with how computers can gain high-level understanding from digital images or videos. It is most widely used field of artificial intelligence (AI) that enables computers and systems to derive meaningful information from digital images, videos and other visual inputs. Different types of computer vision include image segmentation, object detection, facial recognition, edge detection, pattern detection, image classification, and feature matching.

- **Application of Computer Vision**

It has various different application that too in various fields. Some of them are listed below:

- Object Detection
- Screen Reader
- Intruder Detection
- Code and Character Reader
- Robotics

- Motion Analysis
- Image Restoration

There are many left to list as it is very wide topic and here in this project

- We have used one of the application i.e. **Object Detection.**

Detection in Computer Vision

Detection is a computer technology related to computer vision and image processing that deals with detecting instances of semantic objects of a certain class (such as humans, buildings,or cars) in digital images and videos.

For Detection process in computer vision, there are various methods and each one have different level of accuracy according to their advancement level, like is some methods that is invented in very early stage, they give more cases of false detection as compared to the advanced methods that had been discovered after that.

And here we have used the human as an entity which we are detecting our project and along with that, we are also counting humans through image, video and camera.

Human Detection

Human detection is the task of locating all instances of human beings present in an image, and it has been most widely accomplished by searching all locations in the image, at all possible scales, and comparing a small area at each location with known templates or patterns of people.

In this we can use various predefined methods and can detect the human in any image, video and can even get various factors like accuracy, each detections counting, etc.

Some common methods are : -

- **Using Haar Cascade Classifier:**

Here we make use of .xml file for human detection, and using that we detect the humans in real time videos and images

- **Using HOG(Histogram of Oriented Gradients)**

Here we make used of predefined functions and with that we detect, and this case gives some how better accuracy as compared to Harr Cascade Classifier.

- **Using Tensorflow**

TensorFlow is an open-source API from Google, which is widely used for solving machine learning tasks that involve Deep Neural Networks. And again this method gives even better accuracy than above two methods.

- Here we have implemented the application using the **HOG(Histogram of Oriented Gradients)** and got almost the better accuracy.
- We have implemented the project for three different cases:
- ✓ Image
- ✓ Video
- ✓ Camera

2.LITERATURE SURVEY

2.1 EXISTING SYSYTEM

- Most existing digital video surveillance systems rely on human observers for detecting specific activities in a real-time video scene.
- However,there are limitations in the human capability to monitor simultaneous events in surveillance displays.

2.2 PROPOSED SYSTEM

An intelligent system detects and captures motion information of moving targets for accurate object classification. The classified object is being tracked for high-level analysis. In this study, we focus on detecting humans and do not consider recognition of their complex activities

2.3 PREREQUISITIES

The project in Python requires you to have basic knowledge of python programming and the OpenCV library. We will be needing following libraries:

- OpenCV: A strong library used for machine learning
- Imutils: To Image Processing
- Numpy: Used for Scientific Computing. Image is stored in a numpy array
- Argparse: Used to give input in command line

3.DESIGN OF THE PROJECT

3.1 FLOWCHART

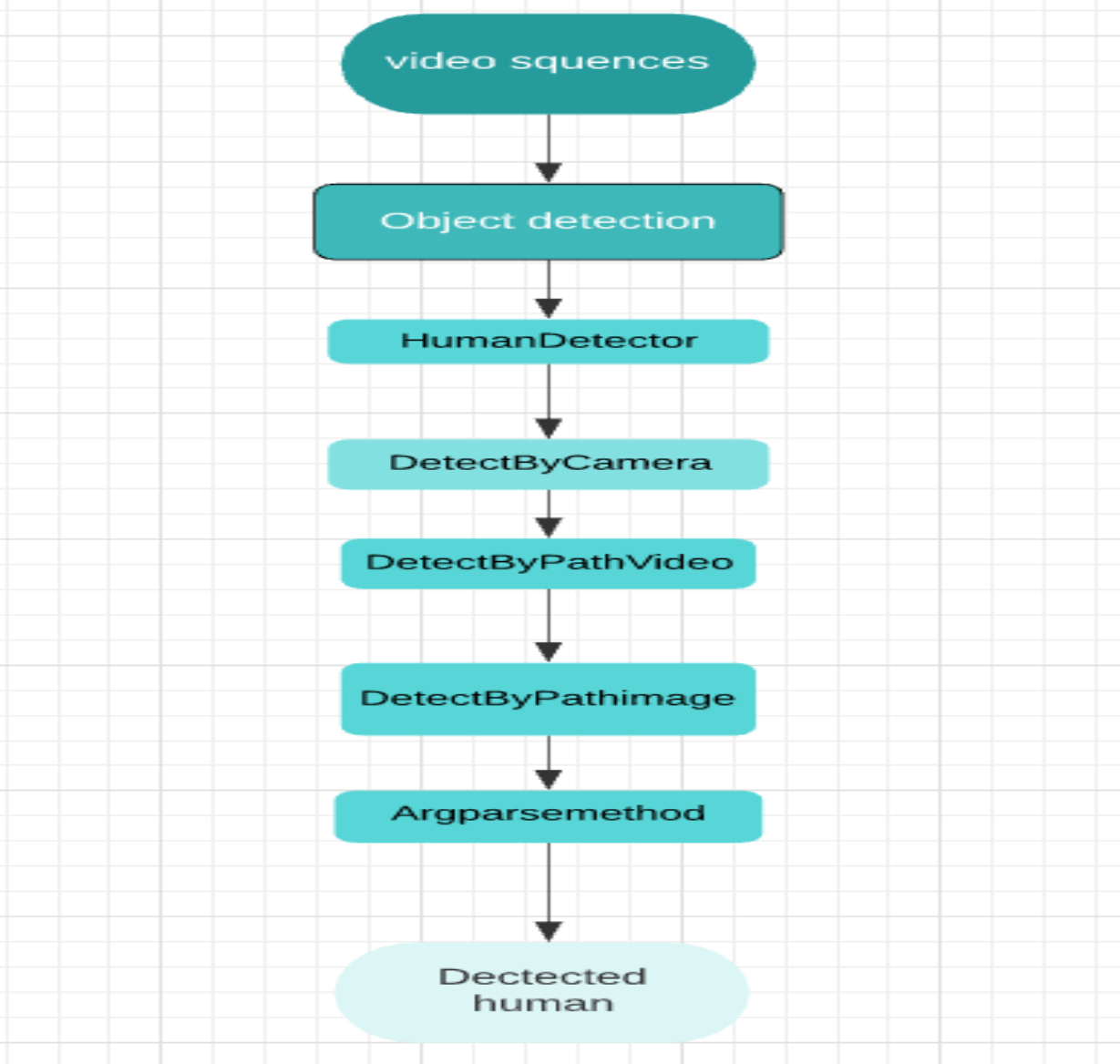


Fig1. Flow chart

3.2 HARDWARE AND SOFTWARE REQUIREMENTS OF THE PROJECT

SOFTWARE REQUIREMENTS :

- Operating Systems : Windows or Macos or Linux
- Code Editor : Visual studio code
- Languages : Python
- **Python :** Python is a language that has been decoded. Python, as per Guido van Rossum, has a chart theory that enhancements code decipherability, as well as a sentence structure that permits programming engineers to communicate ideas in less lines of code by using basic whitespace. It gives structures that connect with few individuals. Contains a novel changed association. Supports a few ideal models, planned, important, helpful, and a huge and intensive.

- **Open CV :** OpenCV is the huge open-source library for the computer vision, machine learning, and image processing and now it plays a major role in real-time operation which is very important in today's systems. By using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When it integrated with various libraries, such as NumPy, python is capable of processing the OpenCV array structure for analysis. To Identify image pattern and its various features we use vector space and perform mathematical operations on these features.

- **HOG**(Histogram oriented gradient) **:** The histogram of oriented gradients (HOG) is a feature descriptor used in computer vision and image processing for the purpose of object detection. The technique counts occurrences of gradient orientation in localized portions of an image

HARD WARE REQUIREMENTS :

- Processor : Intelcore i3,i5,i7,apple silicon m1,etc..
- Ram : 4GB
- Hard Disk : 100GB
- File system ; 32-bit or 62-bit

4 . IMPLEMENTAION

4.1 List of Modules

i. Human Detector method

ii. Detect By Camera method.

iii. Detect By Path Video method.

iv. Detect By Path image method

v. Arg parse method.

i. Human Detector Method

In this deep learning project, we can take images also. So our method will check if a path is given then search for the video or image in the given path and operate. Otherwise, it will open the web Cam.

ii. Detect By Camera method

It returns a check which is True if this was able to read a frame otherwise False. Now, For each Frame, we have call detect method. Then we write the frame in our output file.

iii. Detect By Path Video method

This method is very similar to the previous method except we will give a path to the Video. First, we check if the video on the provided path is found or not**.**

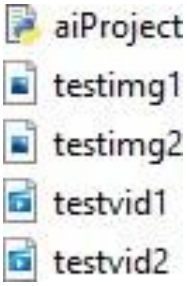
iv. Detect By Path image method.

This method is used if a person needs to be detected from an image.

v. Arg parse method.

The function arg parse simply parses and returns as a dictionary the arguments passed through your terminal to our script.

4.2 Project Directory :



4.3 Libraries used in our project:

Cv2, Imutils, Numpy, Argparse

```
import cv2
import imutils
import numpy as np
import argparse
```

5. CREATING AND TESTING

Creating a model using HOG Descriptor with Support Vector Machine to Detect Humans:

```
HOGCV = cv2.HOGDescriptor()
HOGCV.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())
```

In the below code we give the dimensions for the box to detect the person. So in general we will detect the person in the frame. And show it one after another that it looks like a video.That is exactly what our Detect() method will do. It will take a frame to detect a person in it.Make a box around a person and show the frame.. and return the frame with person bounded by a green box.

```
def detect(frame):
    bounding_box_cordinates, weights =  HOGCV.detectMultiScale(frame, winStride = (4, 4), padding = (8, 8), scale = 1.03)

    person = 1
    for x,y,w,h in bounding_box_cordinates:
        cv2.rectangle(frame, (x,y), (x+w,y+h), (0,255,0), 2)
        cv2.putText(frame, f'person {person}', (x,y), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,255), 1)
        person += 1

    cv2.putText(frame, 'Status : Detecting ', (20,20), cv2.FONT_HERSHEY_DUPLEX, 0.8, (255,0,0), 2)
    cv2.putText(frame, f'Total Persons : {person-1}', (30,50), cv2.FONT_HERSHEY_DUPLEX, 0.8, (255,0,0), 2)
    cv2.imshow('output', frame)

    return frame
```

HumanDetector() method:

There are two ways of getting Video.

1.Web Camera (It will open our system camera)

2.Path of file stored(We will give the path for video and image)

```
def humanDetector(args):
    image_path = args["image"]
    video_path = args['video']
    if str(args["camera"]) == 'true' : camera = True
    else : camera = False

    writer = None
    if args['output'] is not None and image_path is None:
        writer = cv2.VideoWriter(args['output'],cv2.VideoWriter_fourcc(*'MJPG'), 10, (600,600))

    if camera:
        print('[INFO] Opening Web Cam.')
        detectByCamera(writer)
    elif video_path is not None:
        print('[INFO] Opening Video from path.')
        detectByPathVideo(video_path, writer)
    elif image_path is not None:
        print('[INFO] Opening Image from path.')
        detectByPathImage(image_path, args['output'])
```

Detect By Camera() method :

```
def detectByCamera(writer):
    video = cv2.VideoCapture(0)
    print('Detecting people...')

    while True:
        check, frame = video.read()

        frame = detect(frame)
        if writer is not None:
            writer.write(frame)

        key = cv2.waitKey(1)
        if key == ord('q'):
            break

    video.release()
    cv2.destroyAllWindows()
def detectByPathVideo(path, writer):
```

Detect By Path Video() method:

Here we have to give the path for video.mp4

```
def detectByPathVideo(path, writer):

    video = cv2.VideoCapture(path)
    check, frame = video.read()
    if check == False:
        print('Video Not Found. Please Enter a Valid Path (Full path of Video Should be Provided).')
        return

    print('Detecting people...')
    while video.isOpened():
        #check is True if reading was successful
        check, frame =  video.read()

        if check:
            frame = imutils.resize(frame , width=min(800,frame.shape[1]))
            frame = detect(frame)

            if writer is not None:
                writer.write(frame)

            key = cv2.waitKey(1)
            if key== ord('q'):
                break
        else:
            break
    video.release()
    cv2.destroyAllWindows()
```

Detect By Path image() method:

Image path is given

```
def detectByPathImage(path, output_path):
    image = cv2.imread(path)

    image = imutils.resize(image, width = min(800, image.shape[1]))

    result_image = detect(image)

    if output_path is not None:
        cv2.imwrite(output_path, result_image)

    cv2.waitKey(0)
    cv2.destroyAllWindows()
```

Arg parse() method :

The function arg parse() simply parses and returns as a dictionary the arguments passed through your terminal to our script.

- There will be Three arguments within the Parser:
- **Image**: The path to the image file inside your system(-i)
- **Video:** The path to the Video file inside your system (-v)
- **Camera:** A variable that if set to 'true' will call the camera Detect() method.(-c true)

```
def argsParser():
    arg_parse = argparse.ArgumentParser()
    arg_parse.add_argument("-v", "--video", default=None, help="path to Video File ")
    arg_parse.add_argument("-i", "--image", default=None, help="path to Image File ")
    arg_parse.add_argument("-c", "--camera", default=False, help="if u want to use camera type true")
    arg_parse.add_argument("-o", "--output", type=str, help="path to output file")
    args = vars(arg_parse.parse_args())

    return args
```

Main Function:

```
if __name__ == "__main__":
    HOGCV = cv2.HOGDescriptor()
    HOGCV.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())

    args = argsParser()
    humanDetector(args)
```

CODE :

```
 import cv2
import imutils
import numpy as np
import argparse

HOGCV = cv2.HOGDescriptor()
HOGCV.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())

def detect (frame):
   bounding_box_cordinates, weights = HOGCV.detectMultiScale(frame, winStride = (4, 4), padding = (8, 8), scale = 1.03)

   person = 1
   for x,y,w,h in bounding_box_cordinates:
      cv2.rectangle(frame, (x,y), (x+w,y+h), (0,255,0), 2)
      cv2.putText(frame, f'person {person}', (x,y), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0,0,255), 1)
      person += 1

   cv2.putText(frame, 'Status : Detecting ', (20,20), cv2.FONT_HERSHEY_DUPLEX, 0.8, (255, 0, 0), 2)
   cv2.putText(frame, f'Total Persons : {person-1}', (30,50), cv2.FONT_HERSHEY_DUPLEX, 0.8, (255,0,0), 2)
   cv2.imshow('output', frame)
   return frame
```

```
def humanDetector(args):
    image_path = args["image"]
    video_path = args['video']
    if str(args["camera"]) == 'true': camera = True
    else: camera = False

    writer = None
    if args['output'] is not None and image_path is None:
        writer = cv2.VideoWriter(args['output'],cv2.VideoWriter_fourcc(*'MJPG'), 10, (600, 600))
    if camera:
        print('[INFO] Opening Web Cam.')
        detectByCamera(writer)
    elif video_path is not None:
        print(' [INFO] Opening Video from path.')
        detectByPathVideo(video_path, writer)
    elif image_path is not None:
        print('[INFO] Opening Image from path.')
        detectByPathImage(image_path, args['output'])

def detectByCamera(writer):
    video = cv2.VideoCapture(0)
    print('Detecting people...')
    while True:
        check, frame = video.read()

        frame = detect(frame)
```

```
        if writer is not None:
            writer.write(frame)

        key = cv2.waitKey(1)
        if key == ord('q'):
            break
    video.release()
    cv2.destroyAllWindows()

def detectByPathVideo(path, writer):

    video = cv2.VideoCapture(path)
    check, frame = video.read()
    if check == False:
        print('Video Not Found. Please Enter a Valid Path (Full path of Video Should be
Provided).')
        return

    print('Detecting people...')
    while video.isOpened():
        #check is True if reading was successful
        check, frame = video.read()
        if check:
            frame = imutils.resize(frame, width=min(800,frame.shape [1]))
            frame = detect(frame)

            if writer is not None:
```

```
            writer.write(frame)
        key = cv2.waitKey(1)
        if key == ord('q'):
            break
    else:
        break
video.release()
cv2.destroyAllWindows()

def detectByPathImage(path, output_path):

    image = cv2.imread(path)
    image = imutils.resize(image, width = min(800, image. shape[1]))
    result_image = detect(image)
    if output_path is not None:
        cv2.imwrite(output_path, result_image)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

def argsParser():
    arg_parse = argparse.ArgumentParser()
    arg_parse.add_argument("-v", "--video", default=None, help="path to Video File ")
    arg_parse.add_argument("-i", "--image", default=None, help="path to Image File ")
    arg_parse.add_argument("-c", "--camera", default=False, help="if u want to use camera type
true ")
    arg_parse.add_argument("-o", "--output", type=str, help="path to output file")
    args = vars(arg_parse.parse_args())
```

```
    return args

if __name__ == "__main__":
    HOGCV = cv2.HOGDescriptor()
    HOGCV.setSVMDetector(cv2.HOGDescriptor_getDefaultPeopleDetector())

    args = argsParser()
    humanDetector(args)
```

5. RESULTS

Output and Performance Analysis

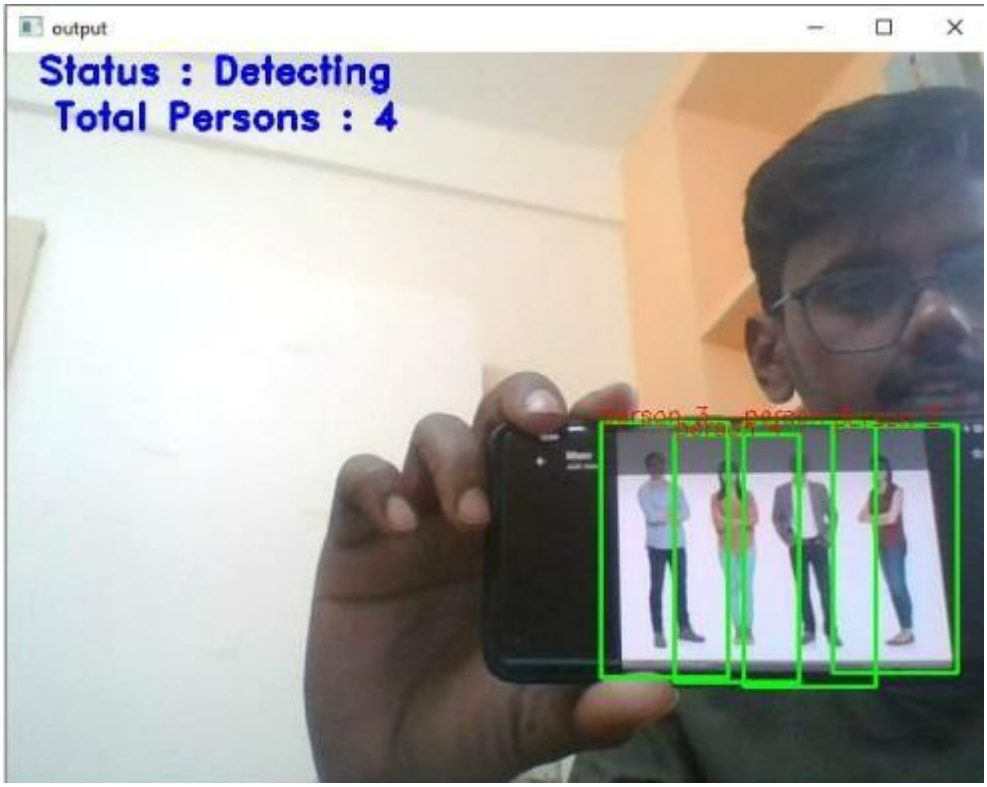
1. While Using Camera:

```
C:\Users\admin>cd Desktop

C:\Users\admin\Desktop>cd AIproject

C:\Users\admin\Desktop\AIproject>python aiproject.py -c true
[INFO] Opening Web Cam.
```

Output:

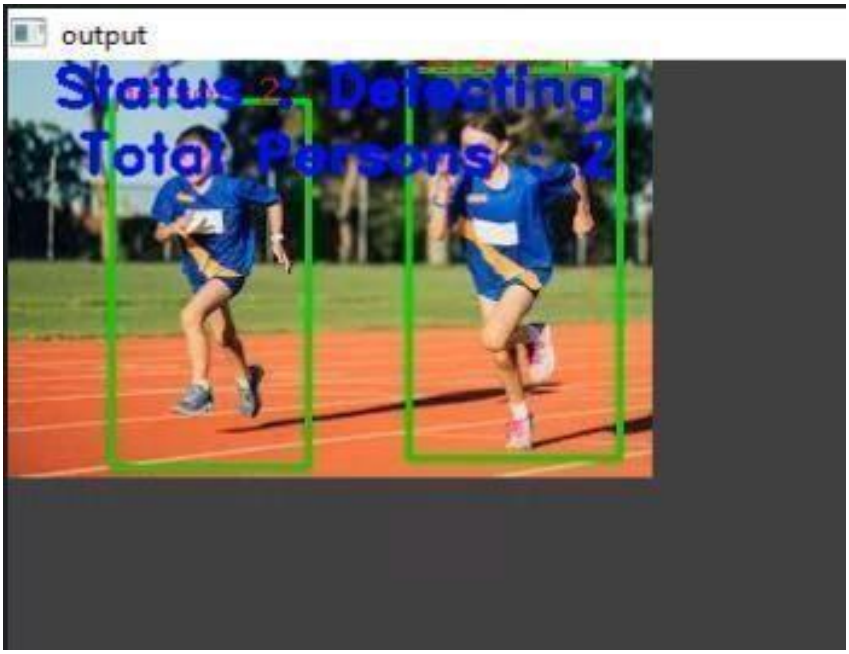


3 . While using Image:

- **Giving path to the image 1:**

```
C:\Users\admin\Desktop\AIproject>python aiproject.py -i C:\Users\admin\Desktop\AIproject\testimg1.jpeg
[INFO] Opening Image from path.
```

Output :



❖ **Giving path to the image 2 :**

```
C:\Users\admin\Desktop\AIproject>python aiproject.py -i C:\Users\admin\Desktop\AIproject\testimg2.jpeg
[INFO] Opening Image from path.
```

Output :

4.While Using Video :

- **Giving path to video 1:**

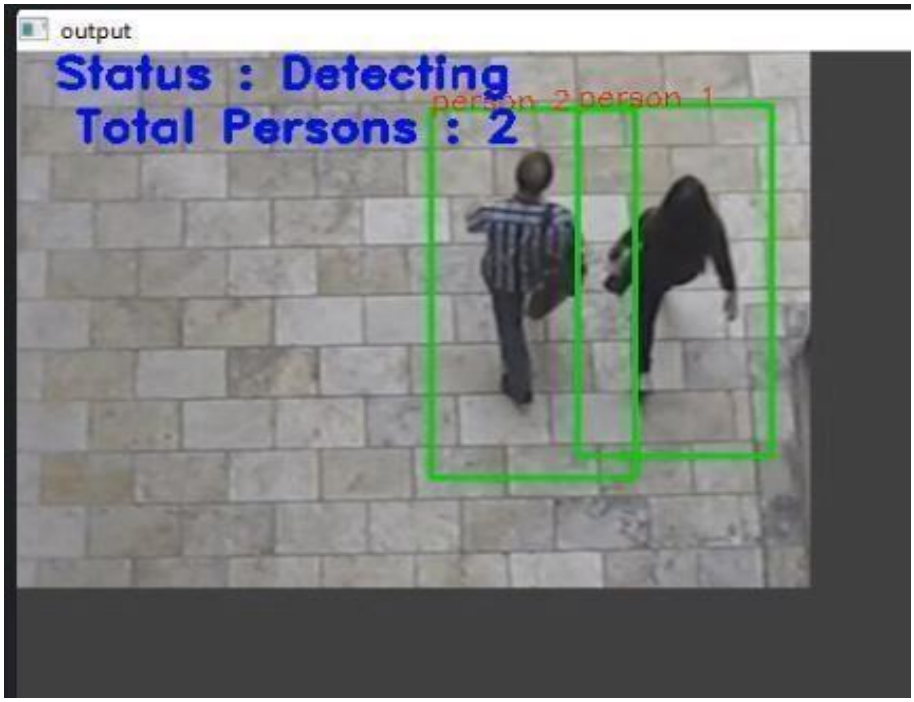
```
C:\Users\admin\Desktop\AIproject>python aiproject.py -v C:\Users\admin\Desktop\AIproject\testvid1.mp4
[INFO] Opening Video from path.
Detecting people...
```

Output at different instances:

❖ **Giving path to video 2 :**

```
C:\Users\admin\Desktop\AIproject>python aiproject.py -v C:\Users\admin\Desktop\AIproject\testvid2.mp4
[INFO] Opening Video from path.
Detecting people...
```

Output at differences instances:



CONCLUSION

We expect to build the Human Detection and Counting System through Webcam or we can give your own video or images. This is an intermediate level deep learning project on computer vision, which will help us to master the concepts. The system will tell how many people are present in the screen at any given time that is it can give real time analysis. We expect the system to be used in fields like proximity alert systems, intrusion alarm system, etc.

- This can be implemented in the commercial and rushed areas for having an accurate human data and for controlling the crowd.
- Now coming to the future scope of this project or application, since in this we are taking any image, video or with camera we are detecting humans and getting count of it, along with accuracy. So some of the future scope can be :
- This can be used in various malls and other areas, to analyse the maximum people count, and then providing some restrictions on number of people to have at a time at that place.
- This can replace various mental jobs, and this can be done more efficiently with machines.
- This will ultimately leads to some kind of crowd-ness control in some places or areas when implemented in that area.

REFERENCE

1. Face Detection and Recognition Using OpenCV

   https://ieeexplore.ieee.org/document/8974493

   Authors: Maliha Khan; Sudeshna Chakraborty; Rani Astya; Shaveta Khepra Publication year:2019

2. Face Recognition Implementation on Raspberrypi Using Opencv and Python

   https://papers.ssrn.com/sol3/papers.cfm? abstract_id=3557027

   Author: Manav Bansal Publication year: 2020

3. Facial Detection & Recognition Using Open CV library

   https://www.researchgate.net/publication/305703476_Facial_Detection_Recognition_Using_Open_CV_library

   Author: Manav Bansal, Sohan Garg