PROFESSIONAL TRAINING REPORT

at

Sathyabama Institute of Science and Technology

(Deemed to be University)

Submitted in partial fulfillment of the requirements for the award

of Bachelor of Engineering Degree in

Computer Science and Engineering

By

A.VIGNESWARI

Reg No: 37110837

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF COMPUTING

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited with Grade "A" by NAAC

JEPPIAAR NAGAR, RAJIV GANDHI SALAI,CHENNAI-600119

AUGUST 2020

SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)
Accredited with Grade "A" by NAAC
JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI - 600 119
www.sathyabama.ac.in

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **VIGNESWARI.A (Reg. No. 37110837)** and who carried out the project entitled "**DETECTING SOCIAL DISTANCE BETWEEN PEOPLE USING OPENCV WITH PYTHON**" under our supervision from April 2020 to June 2020.

Internal Guide

Ms. Lakshmi priya. K, M.E., Ph.D.

Head of the Department

Dr.S.vigneshwari ,M.E., Ph.D.

Submitted for Viva voce Examination held on___________________

Internal Examiner **External Examiner**

DECLARATION

I **VIGNESWARI.A** (Reg No:**37110837**) hereby declare that the Project Report entitled "**DETECTING SOCIAL DISTANCE BETWEEN PEOPLE USING OPENCV WITH PYTHON "** done by me under the guidance of **Mrs.Lakshmi priya.K,M.E,(ph.d)**, is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

DATE:

PLACE: **SIGNATURE OF THE CANDIDATE**

ACKNOWLEDGEMENT

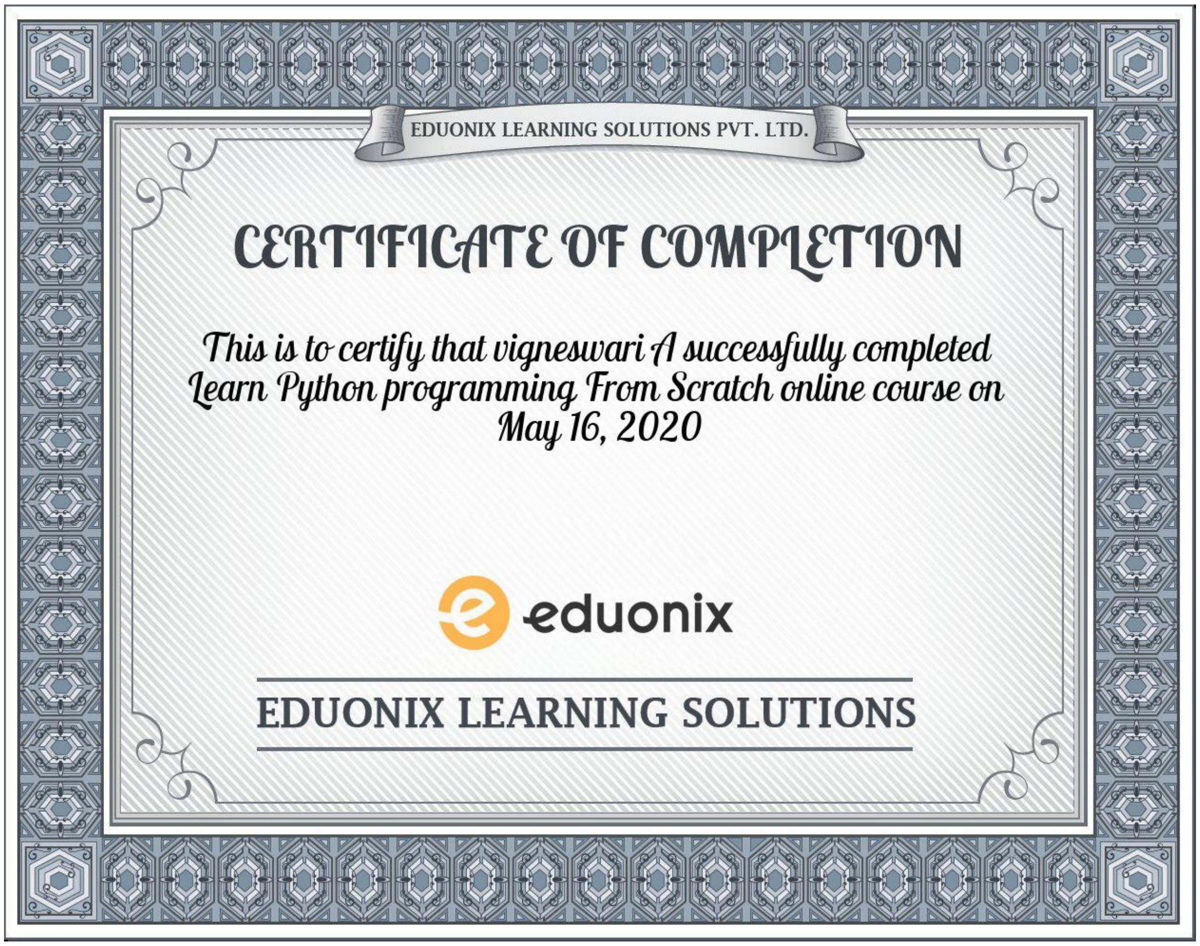
We are pleased to acknowledge our sincere thanks to Board of management of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. We are grateful to them.

We convey our thanks to **Dr.T.Sasikala, M.E.,Ph.D. ,Dean ,School of computing and Dr. S. Vigneshwari, M.E., Ph.D., Head of the Department, Department of Computer Science and Engineering** for providing us the necessary support and details at the right time during the progressive reviews.

We would like to express our sincere and deep sense of gratitude to our Project Guide **Mrs.Lakshmi priya.K M.E.,Ph.D**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

We wish to express our thanks to all Teaching and Non-teaching staff members of the Department of **COMPUTER SCIENCE AND ENGINEERING** who were helpful in many ways for the completion of the project.

TRAINING CERTIFICATE



EDUONIX LEARNING SOLUTIONS PVT. LTD.

CERTIFICATE OF COMPLETION

This is to certify that vigneswari A successfully completed Learn Python programming From Scratch online course on May 16, 2020

eduonix

EDUONIX LEARNING SOLUTIONS

ABSTRACT

This project is about detecting social distance between people in a recorded video using the technique computer vision.computer vision is an interdiscliplinary scientific field. That deals with how computers can be made to gain high-level understanding from digital image and videos.From the perspective of engineering,it seeks to automate tasks that the human visual system can do.This project is build using python and some other packages of opencv like numpy,cv2,matplotlib,imutils,itertools by using this technique we can detect motion of a human and calculating a distance between them to determine whether they are maintaining a social distance between them if they are violating the distance it will indicate us by drawing rectangle red contour over the person else it draw green contour over them.By using this technique we would be able to find people who have not followed social distancing and the one who have followed and we could pick out the people who had contact with the affected one.

TABLE OF CONTENT

S APPENDIX

LIST OF FIGURES

CHAPTER – 1

INTRODUCTION

1.1 GENERAL

Real Time Human Motion Detection and Tracking.This document describes a real-time system for human detection,tracking and motion analysis from a saved video and the various process on that video Generally the motion detection is the process of detecting a change in the position of an object relative to its surroundings or a changes in the surroundings relative to the object eg (**fig 1.1)**. Motion detection can be Achieved by either mechanical or electronic methods.

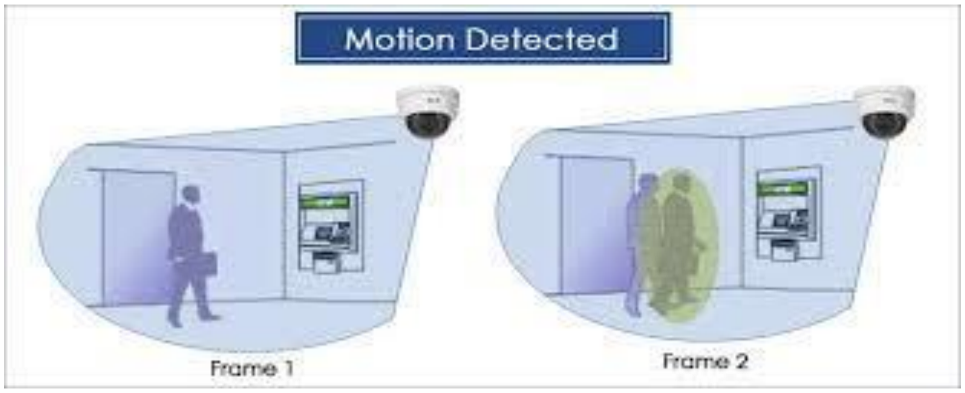


Fig 1.1 MOTION DETECTION

1.2 OUTLINE OF THE PROJECT

My project is to detect a human motion and to check whether they are maintaining a social distance between them if they are not maintaining social distance it will indicating us by drawing a red rectangle contour over them from a recorded video. Human motion analysis is receiving increasing attention from computer vision researchers.Motion and analysis of human body parts involves the low level segmentation of the human body into segments connected by joints and recovers the 3D structure of the human body using its 2D projection over a sequences of images.

CHAPTER - 2
AIM AND SCOPE

2.1 AIM

To detect whether the humans are adhering the social distance by calculating the distance between them while in movement by drawing different color of contour over the human.

2.2 PROBLEM STATEMENT

Our project to detect a human motion and to confirm whether they are in social distancing by drawing contour.If they are maintaining social distancing it will indicates us by green colour rectangular contour else red colour contour as **(fig 2.1)** will be drawn on a playing video which is extracted from a saved file.

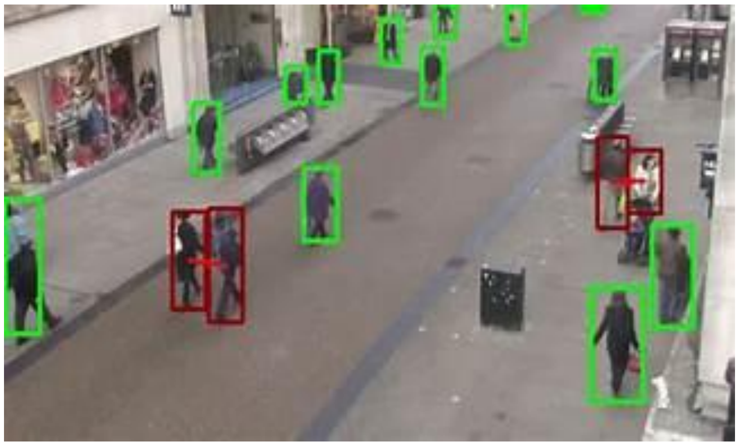


Fig 2.1 social distance detection

2.3 LITERATURE REVIEW

My project is build by using a python as a flatform and opencv as a main source and some packages of opencv like cv2, numpy and matplotlib.

PYTHON is an interpreted, high level, general purpose programming language.we can use python for developing desktop GUI application, websites and web application by taking care of common programming tasks. python is one of the best languages to implemt computer vision.

OPENCV is a python wrapper for the opencv c++ implementation. The library is cross platform and free for use under the opensource BSD license. opencv is capable of image analysis and processing but have more like capture from cam.This means it is great at taking frames out of video or taking in two frames from a stereoscopic camera and run algorithim to extract information.

IMPORTED PACKAGE :

CV2 opencv is implemented using cv2 and numpy. To load, display, save an image.It is easier for user to understand opencv python than cv2 and makes it easier to find the packages with search engines.cv2(old interface in old opencv version was named as cv)is the name that opencv developers chode when they created the binding generators.

NUMBY numpy is a library for the python programming languages, adding support for large,multidimensional arrays and matrices,along with a large collection of high level mathematical function to operate on these arrays.All the opencv arrays structure are converted into and from numpy arrays.so whatever operation you can do in numpy ,you can compine it with opencv,which increase number of weapons in your arsenal.

IMUTILS are a series of convenience functions to make basic image processing functions such as translation,rotation,resizing,skeletonization and displaying matplotlib images easier with opencv .

ITERTOOLS python itertools is a module that provides various functions that work on iterators to produce complex itertors.This module works as a fast,memory-efficient tool that is used either by themselves or in combination to form iterator algebra.

SCIPY is a free and open-source python library used for scientific computing and technical computing. scipy builds on the numpy array object and is part of the numby stack which includes tools like matplotlib,pandas and sympy and an expanding set of scientific computing libraries.It is used in various scientific task like linear algebra,integration and signal processing.

INBUILD METHODS

VIDEOCAPTURE() this method graps the next frame from video file or camera and the return true(non-zero) in the case of success..that is,we call videocapture ::grap() for each camera and after that call the videocapture::retrieve() to decode and get frame from each camera.

IMREAD() used to avoid data structure and function name conflicts with others libraries,**opencv** has its own namespace: cv..Then create a mat object that will store the data of the loaded image.Mat image;now we call the **imread** function which loads the image name specified by the first argument(argv[1]).use the function **cv2.imread()** to read an image.the image should be in the working directory or a full path or image should be given.

IMSHOW(I) displays the grayscale image I in a figure.**imshow** uses the default display range for the image data type and optimize figure,axes and image object properties for image display..**imshow(**RGB) displays the truecolor image RGB in a figure. **imshow(**BW) displays the binary image BW in a figure.use the function **cv2.imshow()** to display an image in a window.The window automatically fits to the image size.

blobFromImage( ) creates 4-dimensional blob from image optionally resizes and crops image from center,subtract mean values,scales values by scale factor,swap blue and red channels.

Cv2.putText( ) it is an opencv python library it binding designed to solve computer vision problems.it is used to draw a text string on any image.It takes image on which text is to be drawn ,text string to be drawn,coordinates and font style as a parameter.

orderedDict( ) is a dict subclass that maintains the items insertion order.when we iterate over an orderedDict,items are returned in the order they were inserted.A regular dictionary doesn’t track the insertion order.so when iterating over it,items are returned in an arbitrary order.

Non_maximum_suppression( ) is a key post-processing step in many computer vison applications.In the context of object detection,it is used to transform a smooth response map that triggers many imprices object windows hypotheses in ,ideally, a single bounding box for each detected object (i.e) **fig 2,2**.The most common approach for NMS for object detection is greedy approach.

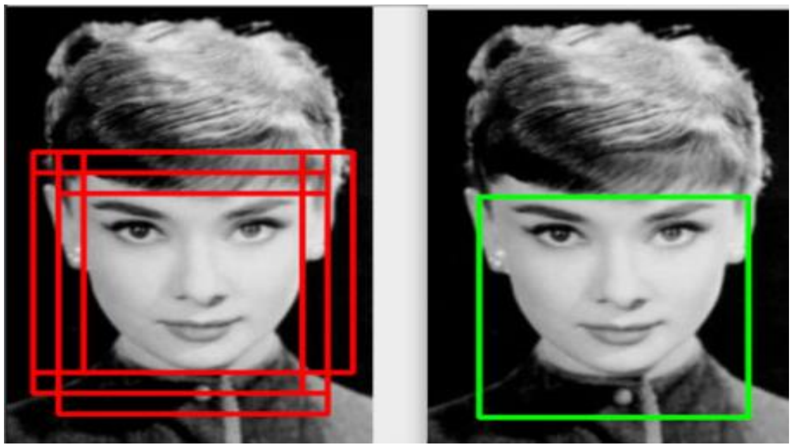


Fig 2.2 non_maximum _suppression image

WAITKEY( ) is a keyboard binding function in opencv it is used to introduce a delay of n milliseconds while rendering images to windows. Its argument is the time in milliseconds for any keyboard event .if you press any key in that time,the program continues. if 0 is passed, it waits indefinitely for a key stroke.It can also be set to detect specific key strokes like, if key a is pressed etc.

DESTROYALLWINDOWS( ) simply destroy all the windows we created.If you want to destroy any specific windows, use the function cv2.destroyAllwindows() where you pass the exact window name as the argument.

2.4 SCOPE

The initial goal of computer vision was to enable machines to see the the visual world and interpret it the way a human would, but AII has advanced computer vision beyond human vision and now machines can be see things humans can’t, like air quality and temperature.**fig 2.3** gives the brief details about computer vision in which computervision are used.

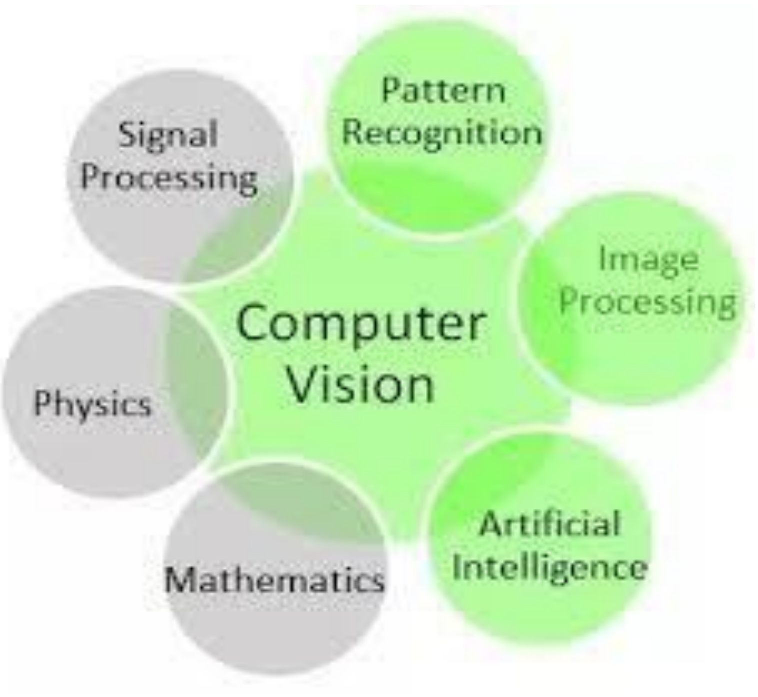


FIG 2.3 OVERVIEW OF COMPUTERVISION

CHAPTER - 3
ALGORITHIM USED

3.1 GENERAL

The first the video will load from the file in case of recorded else the video will capture using camera then we detect a motion by object tracking procedure.Among the many algorithims centroid tracking algorithim is best and efficient. Then to give a clear noise removel view we use blobfrom image function which works based on the greedy approach.

3.2 ALGORITHIM

OBJECT TRACKING : object tracking can be broken into three different sections:initial object detection,assigning IDs and tracking the object across frames. The most commonly used object tracking framework is opencv.

Initially an set of object detection is created.This is typically done by set of bounding box coordinated and using them as inputs for the network.After this,a unique ID is created for each of these initial object detections.

Next,the objects are continually detected as the frames advance and the objects move,and the unique IDs are maintained.Because unique IDs are assigned to the objects as **fig (3.1)**,many different objects can be tracked throughout the video.

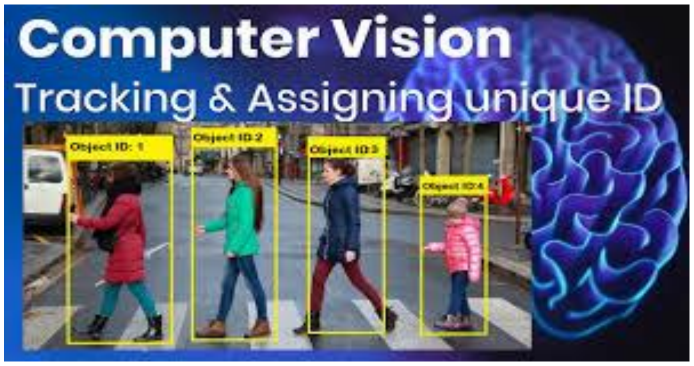


FIG 3.1 ASSIGNING UNIQUE ID

The best object tracking algorithim will only have a single object detection phase,and as a result,the runining time of the algorithim should be improved.A video object tracking algorithim should also be able to handle instances where an object moves outside the boundaries of the frame,as well as be able to pick an object up again if it has “lost” it.

CENTROID TRACKING It is the one of the most common methods of implementing video object tracking is using the opencv framework.centroid tracking in opencv operates by determining the Euclidean distance between already known/labeled object centroids,and new object centroids over the subsequent frames of a video.

First,an object detector is used to create bounding boxes,and once these bounding boxes are accepted the centroids for the objects can be computed.Recurrent CNNs are common network choices used to create the bounding boxes.

After the initial bounding boxes have been made,the Euclidean distance between the existing objects and the new objects can be calculated. If the initial and new bounding boxes can associated with one another,the coordinates of the existing object are updated to the new bounding box location.while a given object will potentially move between frames,the distance between a given object’s centroid and object’s surrounding it should be greater than the distance between the object’s centroid at one frame and the object’s centroid at the next frame.

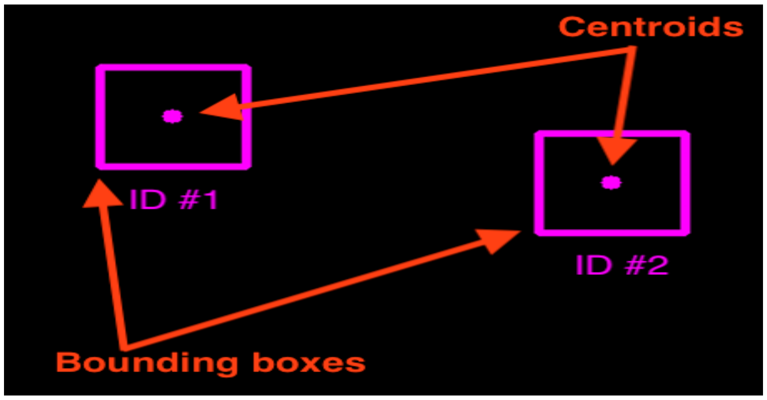


FIG 3.2 FINDING CENTROIDS

3.3 OVERVIEW

First extract the video from the file on which detection and tracking has to be made my passing video name if the all the files in same directory else the path in which the video file has been located as argument if we are going to perform detection in recorded video else if the detection is on live video then the camera number has to be passed as a argument instead of filename. After running the program the video will display in output frame with the people whose motions are detected and tracked and red colour and green colour rectangular contour over them and with then we can find whether the humans are mainintaing social distance or not with the help of contour.If they are not following the social distance it will indicate us by a rectangular red contour.

3.4 PROJECT DESCRIPTION

computer vision tasks include methods for acquiring, processing, analysing and understanding digital images, and extraction of high dimensional data from the real world in order to produce numerical or symbolic information
,e.g in the forms of decision. understanding in this context means the transformation of visual images into description of the world that make sense to thought processes and can elict.The project is build using python and opencv this python program will allow you to detect motion and also to check whether the publics are mainitaing a social distance between them or not.

After running the program the red and green colour rectangle contour on a moving public will found like **fig(4.1)** If the person are in close distance with other he /she will marked by a red contour if the same person are away from other then he will be marked by green contour as shown in the f**ig(4.2).**

3.5 WORK FLOW

The below flowchar gives the pictorial representation of the social distance detecting algorithim.

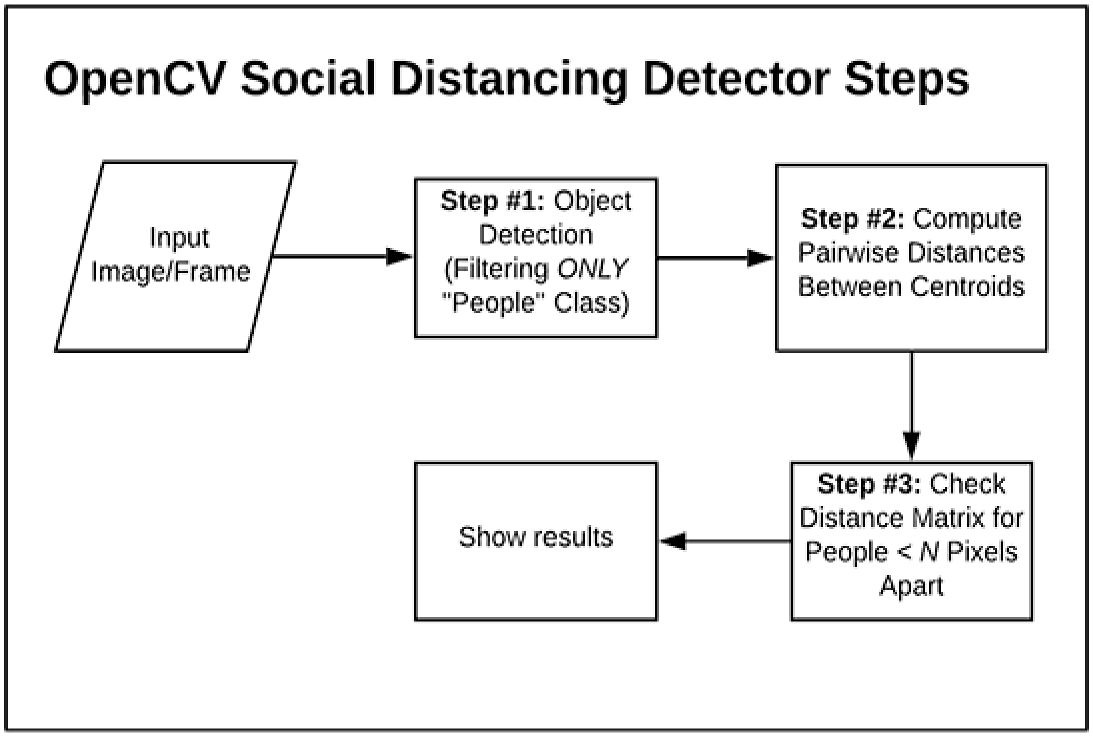

OpenCV Social Distancing Detector Steps
Input Image/Frame
Step #1: Object Detection (Filtering ONLY "People" Class)
Step #2: Compute Pairwise Distances Between Centroids
Step #3: Check Distance Matrix for People < N Pixels Apart
Show results

FIG 3.3 FLOWCHART FOR SOCIAL DISTANCE DETECTING

In our case result is if the distance between one people to other is not with in mention range red colour rectangular contour will be visible until he/she reached the mention the distance range else green colour contour will over them.we can also different colour.

CHAPTER – 4

PERFORMANCE ANALYSIS

4.1 REQUIREMENTS

I. HARDWARE IMPLEMENTATION

1. Any operating system with minimum of 2GB RAM
2. Keyboard
3. Mouse
4. Moniter

II. SOFTWARE IMPLEMENTATION

1.. Any one browser installed in os
2. Python
3. Python packages
    I. Opencv
    II. Imutils
    III. Itertools

4.2 ROLE OF PYTHON

This python program will allow you to detect motion and also store the time interval of the motion. Requirement .Video can be treated as stack of pictures called frames. Analysis of all windows. Time record of movements. Python wrapper for the original opencv .It make use of numpy, which is a highly optimized library for numerical operation with a MATLAB-style syntax.

4.3 RESULTS AND DISCUSSION

The result of the project is the to find people who have not followed social distancing and the one who have followed by rectangle contour bounding .

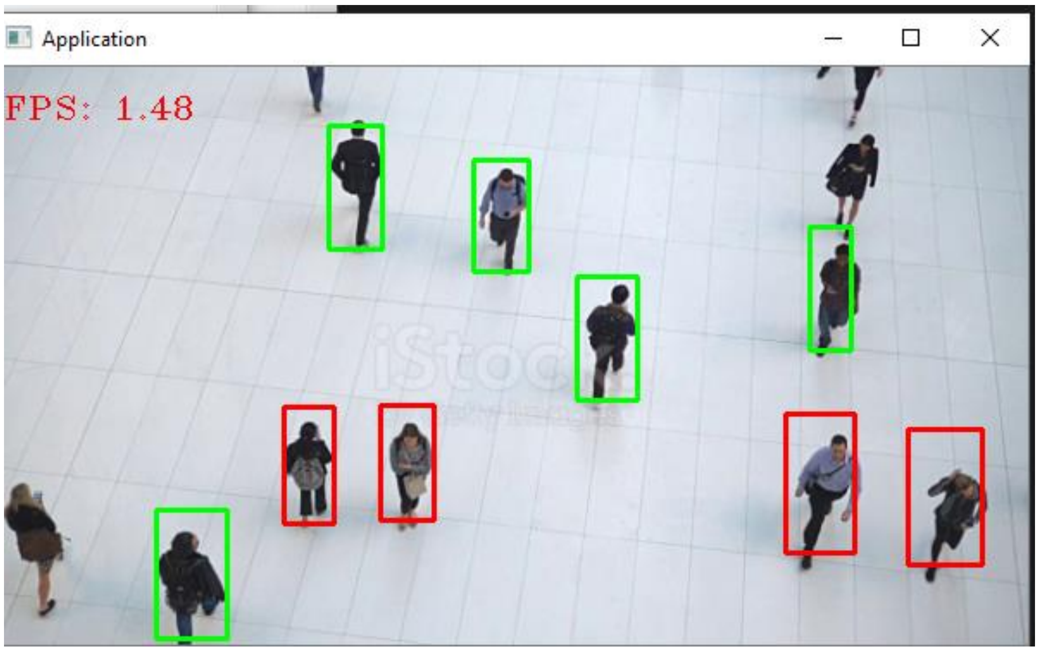


FIG 4.1 SOCIAL DISTANCE DETECTED

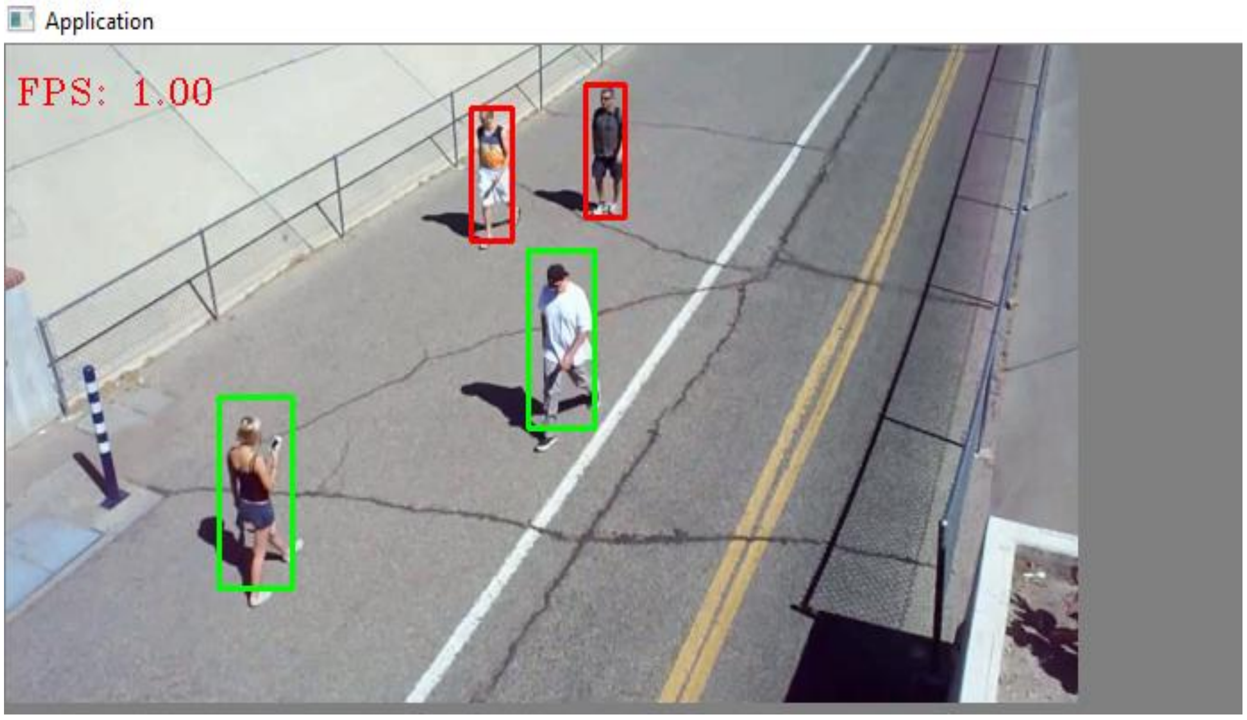


FIG 4.2 PERSON IN GREEN CONTOUR

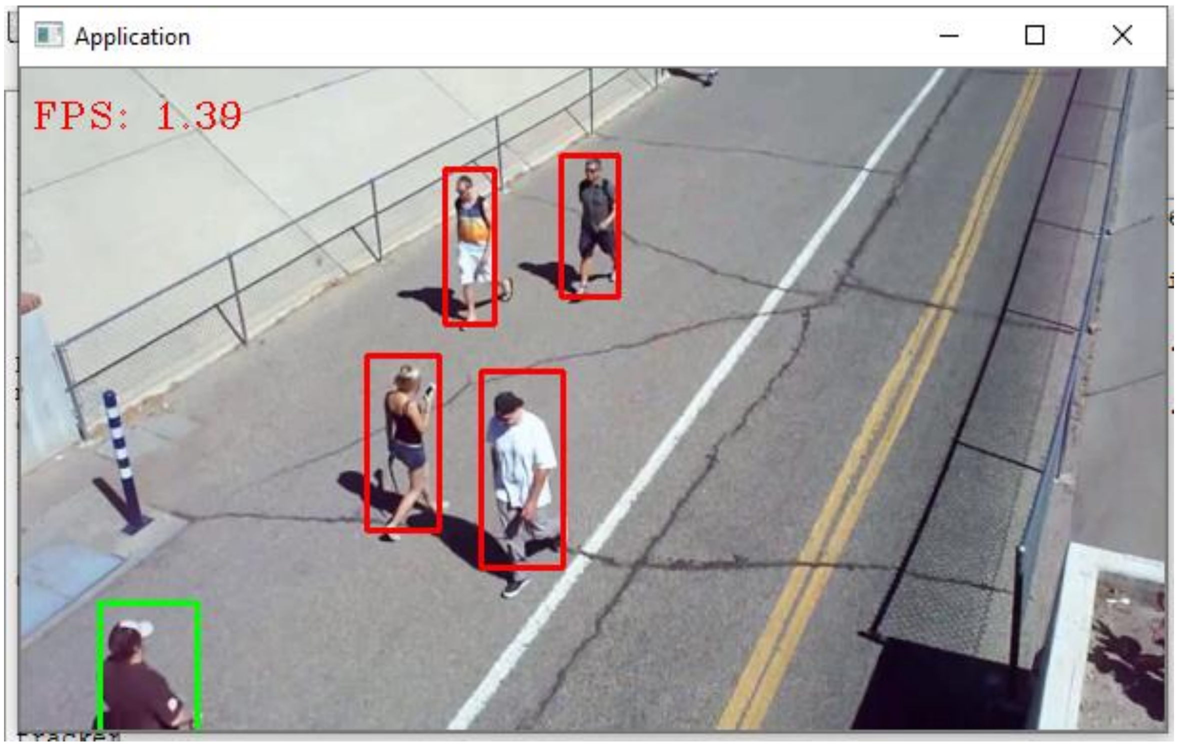


FIG 4.2 SAME PERSON WITH RED CONTOUR

CHAPTER 5
SUMMARY AND CONCLUSION

5.1 CONCLUSION

The technique is simple. It is easily understood by the users who has not need much more knowledge about technical field. By using this project we could easily pick out the people who had contact with the affected one and we can able to find the percentage of people who are following the government rules.

5.2 FUTURE WORK

This project can be further implemented by using DEEP LEARNING and MEACHINE LEARNING to give an alert message like notification to be a highly secure. The latest technology computer vision will use along with python with more advanced packages to work with real time application.

REFERENCES

[1] PYTHON :	http://www.imarticus.org
[2] COMPUTER VISION	http://www.realpython.com
[3] MOTION DETECTION	http://www.pyimagesearch.com
[4] OBJECT TRACKING	http://www.imageannotation.ai
[5] COMPUTER VISION	ComputerVision and Image Understanding - N.Paragios

APPENDIX

A. SOURCE CODE

```
import cv2
import datetime
import imutils
import numpy as np
from centroidtracker import CentroidTracker
from itertools import combinations
import math
protopath = "MobileNetSSD_deploy.prototxt"
modelpath = "MobileNetSSD_deploy.caffemodel"
detector = cv2.dnn.readNetFromCaffe(prototxt=protopath, caffeModel=modelpath)
# detector.setPreferableBackend(cv2.dnn.DNN_BACKEND_INFERENCE_ENGINE)
# detector.setPreferableTarget(cv2.dnn.DNN_TARGET_CPU)
CLASSES = ["background", "aeroplane", "bicycle", "bird", "boat",
            "bottle", "bus", "car", "cat", "chair", "cow", "diningtable",
            "dog", "horse", "motorbike", "person", "pottedplant", "sheep”]
tracker = CentroidTracker(maxDisappeared=40, maxDistance=50)
def non_max_suppression_fast(boxes, overlapThresh):
  try:
 if len(boxes) == 0:
             return []
 if boxes.dtype.kind == "i":
            boxes = boxes.astype("float")
        pick = [ ]
        x1 = boxes[:, 0]
        y1 = boxes[:, 1]
        x2 = boxes[:, 2]
        y2 = boxes[:, 3]
        area = (x2 - x1 + 1) * (y2 - y1 + 1)
        idxs = np.argsort(y2)
        while len(idxs) > 0:
            last = len(idxs) - 1
            i = idxs[last]
            pick.append(i)
            xx1 = np.maximum(x1[i], x1[idxs[:last]])
            yy1 = np.maximum(y1[i], y1[idxs[:last]])
            xx2 = np.minimum(x2[i], x2[idxs[:last]])
            yy2 = np.minimum(y2[i], y2[idxs[:last]])
            w = np.maximum(0, xx2 - xx1 + 1)
       h = np.maximum(0, yy2 - yy1 + 1)
       overlap = (w * h) / area[idxs[:last]]
       idxs = np.delete(idxs, np.concatenate(([last],
        np.where(overlap > overlapThresh)[0]))
       return boxes[pick].astype("int")
       except Exception as e:
       print("Exception occurred in non_max_suppression : {}".format(e))
```

```
def main():
    cap = cv2.VideoCapture('testvideo2.mp4')
    fps_start_time = datetime.datetime.now()
    fps = 0
    total_frames = 0
    while True:
        ret, frame = cap.read()
        frame = imutils.resize(frame, width=600)
        total_frames = total_frames + 1
        (H, W) = frame.shape[:2]
        blob = cv2.dnn.blobFromImage(frame, 0.007843, (W, H), 127.5)
        detector.setInput(blob)
        person_detections = detector.forward()
        rects = [ ]
        for i in np.arange(0, person_detections.shape[2]):
            confidence = person_detections[0, 0, i, 2]
            if confidence > 0.5:
                idx = int(person_detections[0, 0, i, 1])
                if CLASSES[idx] != "person":
                    continue
                person_box = person_detections[0, 0, i, 3:7] * np.array([W, H, W, H])
                (startX, startY, endX, endY) = person_box.astype("int")
                rects.append(person_box)
        boundingboxes = np.array(rects)
        boundingboxes = boundingboxes.astype(int)
        rects = non_max_suppression_fast(boundingboxes, 0.3)
        centroid_dict = dict()
        objects = tracker.update(rects)
        for (objectId, bbox) in objects.items():
            x1, y1, x2, y2 = bbox
            x1 = int(x1)
            y1 = int(y1)
            x2 = int(x2)
            y2 = int(y2)
            cX = int((x1 + x2) / 2.0)
            cY = int((y1 + y2) / 2.0)
            centroid_dict[objectId] = (cX, cY, x1, y1, x2, y2)
            red_zone_list = []
        for (id1, p1), (id2, p2) in combinations(centroid_dict.items(), 2):
            dx, dy = p1[0] - p2[0], p1[1] - p2[1]
            distance = math.sqrt(dx * dx + dy * dy)
            if distance < 75.0:
                if id1 not in red_zone_list:
                    red_zone_list.append(id1)
                if id2 not in red_zone_list:
         red_zone_list.append(id2)
        for id, box in centroid_dict.items():
            if id in red_zone_list:
                cv2.rectangle(frame, (box[2], box[3]), (box[4], box[5]), (0, 0, 255), 2)
            else:
                cv2.rectangle(frame, (box[2], box[3]), (box[4], box[5]), (0, 255, 0), 2)
        fps_end_time = datetime.datetime.now()
        time_diff = fps_end_time - fps_start_time
        if time_diff.seconds == 0:
           fps = 0.0
       else:
            fps = (total_frames / time_diff.seconds)
        fps_text = "FPS: {:.2f}".format(fps)
        cv2.putText(frame, fps_text, (5, 30), cv2.FONT_HERSHEY_COMPLEX_SMALL, 1, (0, 0,
255), 1)
        cv2.imshow("Application", frame)
        key = cv2.waitKey(1)
        if key == ord('q'):
            break
```

```
    cv2.destroyAllWindows()
main()
```