LOVELY PROFESSIONAL UNIVERSITY

VECHILE SPEED DETECTION

A PROJECT REPORT

Submitted by

Mohit
ROLL NO. - RKM086A13
SECTION – KM086
REG. NO. - 11802594

INT-248 ADVANCE MACHINE LEARNING

SUBMITTED TO –

ANKITA WADHAWAN

BTECH CSE
School of Computer Science and Engineering

Git hub link- https://github.com/Mohit-Malik-M/Deep-learning.git

ABSTRACT

Road Safety is an integral part of modern roads. Therefore speed limits are given for a particular road depending on the quality of the road and the how prone the road is to accidents. Speed Cameras are set up at intervals of the road to catch speed-limit violators. Just like any other technology, speed cameras have progressed over the years.

This project focusses on making a speed camera without sensors and only with Image processing of videos by using OpenCV in Python. Objects would be tracked, and speeds of vehicles would be estimated.

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

Initially a Doppler radar was used to capture the speed of Vehicles on roads. Cameras were used to capture speeding vehicles. A Doppler radar uses the change in frequency of a passing vehicle to determine the speed of the vehicle. (This phenomenon is known as Doppler effect).

Later Traffic Radars started using sensors to detect speeds. At present, traffic radars integrate the use of sensors and image processing to catch traffic violators. The traffic violation need not be speeding, it can also be wrong/reserved lane detection, passing the red-light detection, tailgating, wrong side overtaking etc.

In the future, traffic radars may even detect seatbelts and texting while driving offences. Such work is currently under research and may come out soon.

2. RELEVANCE

Speeding Cameras are present in many highways in order to catch speeding vehicles and improve the safety on roads. Developed cities have more speed cameras and have more well monitored roads.

Cameras today not only catch speed, but also register if a vehicle goes on the wrong lane or crosses a red light.

If there are consequences to rash driving on roads (like road fines or license suspension), people would be more careful in following street rules. This in turn would reduce the number of accidents on the road.

3. OUTCOME

This project would be able to successfully determine the speed of a vehicle and save the vehicle picture in a separate folder. Due to video clarity issues, number-plate detection is beyond the scope for this project.

This project focusses on vehicle detection and tracking and speed estimation.

4. PROBLEM STATEMENT

The objective of this project is to create a traffic radar using Image Processing in Python by using OpenCV and TensorFlow.

When it comes to tracking the speed of vehicles on a segment of road, the vital steps of this projects is:

- Vehicle Detection
- Speed estimation
- Capturing vehicle picture

4.1 Vehicle Detection

As the background of the vehicles is stationary (as the speed camera is stationary) image subtraction is used to detect moving vehicle.

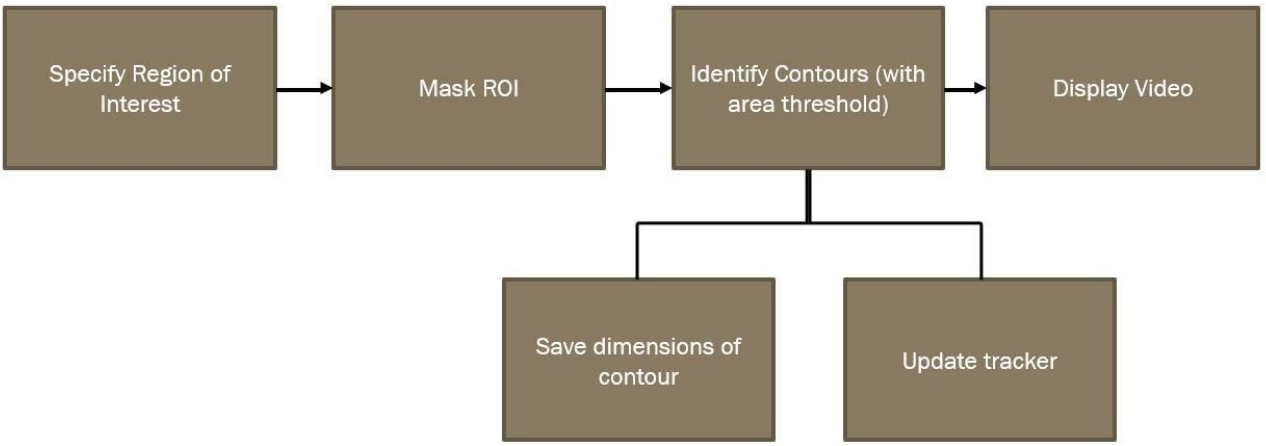

Figure 1 : Vehicle Detection Block Diagram

4.2 Speed Estimation

The speed of a vehicle can be estimated when a tracked vehicle covers a segment of road.

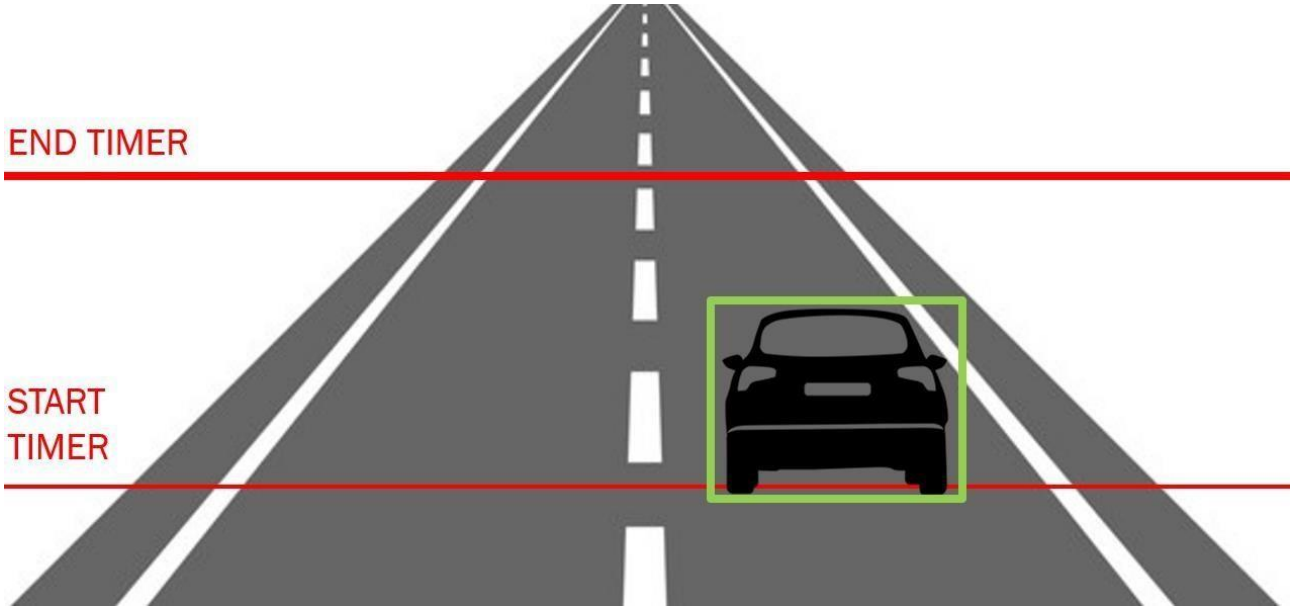

Figure 2 : Speed Timer Diagram

4.3 Capturing Vehicle Image

Based on Contour detection, the image of a particular ID is saved to a folder along with the speed. Picture is saved as soon as the speed is estimated.

5. LITERATURE SURVEY

Below are some of the literature works related to the topic. These include vehicle detection and speed estimation.

Title	Authors	Year of Publication	Dataset Considered	Methodology Proposed	pros/cons	Future work possible

Determining vehicle speed based on video using convolutional Neural Network	Alexander Grents, Vitalii Varkentin, Nikolay Goryaev	2020	750 images from over 52,000 objects. The images were obtained from traffic cameras of Intersvyaz company in the city of Chelyabinsk. Categories: cars, buses, trolleybuses, trucks, minibuses, ambulances, firetrucks, vans, uncategorized	Mask RCNN architecture is used for vehicle detection. Masking was used on the road video so as to recognise vehicle accessible regions. A rectangular region is selected, and speed of a vehicle crossing the region is calculated with the help of equations.	Pros It is a straightforward method. Cons The accuracy of this radar depends of the video camera resolution and frame rate.	Vehicle going on the wrong lane/hardshoulder can be caught by these speed radars.

Title	Authors	Year of Publication	Dataset Considered	Methodology Proposed	pros/cons	Future work possible

Detection of Vehicle Position and Speed using Camera Calibration and Image Projection Methods	Alexander A S Gunawan, Deasy Aprilia Tanjung, Fergyanto E. Gunawan	2019	Dataset not mentioned	Direct Linear Transformation is used for estimating the length of a segment of road. Vehicle position is detected using Background Subtraction and speed is determined by Mixture of Gaussian (MoG) Camera Calibration is used to calculate target position of vehicle in 3D space. were used.	Pros High Accuracy (96% claimed) Length of segment of road is calculated through image. No need for manual measurements. Cons Best result is when camera is on top of the road. Does not support multi-vehicle traffic	Multi veicle detection can be used. Tailgating can be monitored.

Title	Authors	Year of Publication	Dataset Considered	Methodology Proposed	pros/cons	Future work possible
Vehicle speed measurement model for video-based systems	Saleh Javadi, Mattias Dahl, Mats I. Pettersson	2019	Dataset not mentioned	Intrusion detection is used when an object crosses a virtual line and enters a region of interest. The detection distance is directly proportional to the vehicle's speed and inversely proportional to the camera's frame rate.	Pros It is a simple idea to implement. Cons Does not classify vehicles Best result is only when camera is on top of the road	Tailgating can be monitored. Driving on the wrong lane can be detected

Title	Authors	Year of Publication	Dataset Considered	Methodology Proposed	pros/cons	Future work possible

An Efficient Approach for Detection and SpeedEstimation of Moving Vehicles	Tarun Kumar and Dharmendar	2016	Dataset not mentioned	Only road segment is taken	Pros It is a simple idea to implement. Cons Low frame rates leads to higher errors. Best result is only when camera is on top of the road	Tailgating can be monitored. Driving on the wrong lane can be detected

Title	Authors	Year of Publication	Dataset Considered	Methodology Proposed	pros/cons	Future work possible
An Efficient Approach for Detection and SpeedEstimation of Moving Vehicles	Tarun Kumar and Dharmendar	2016	Dataset not mentioned	Only road segment is taken	Pros It is a simple idea to implement. Cons Low frame rates leads to higher errors. Best result is only when camera is on top of the road	Tailgating can be monitored. Driving on the wrong lane can be detected

6. PROJECT MODEL

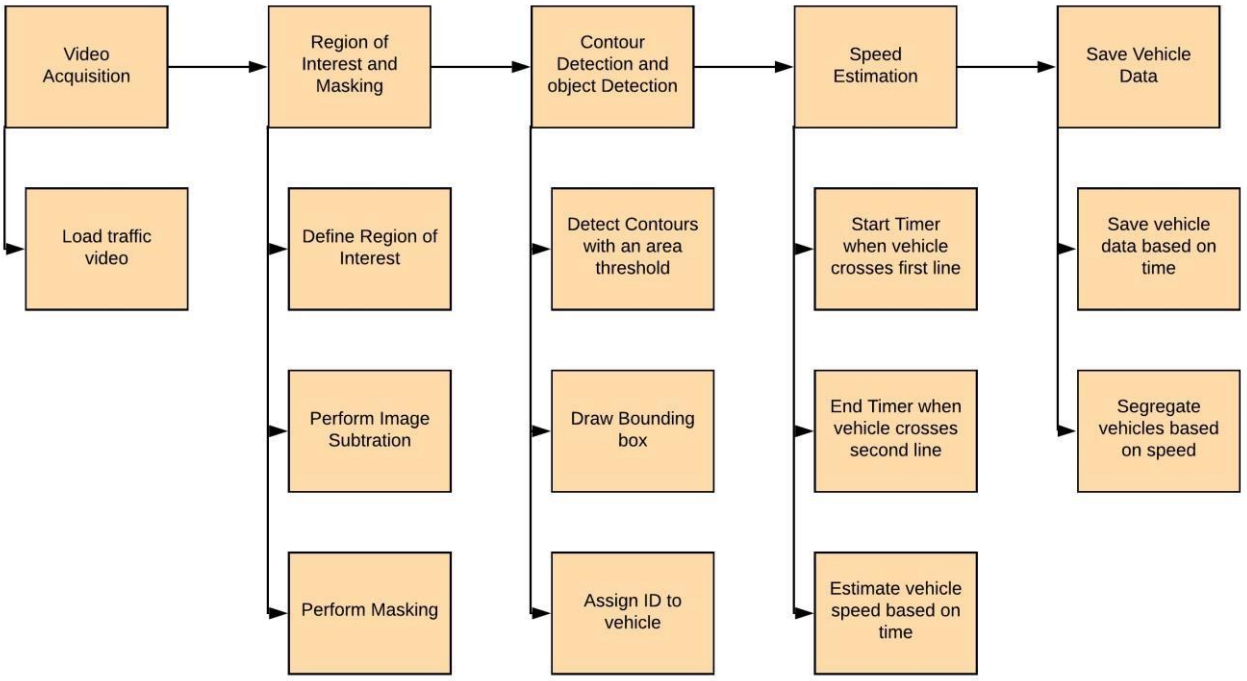

Figure 3 : Project Model Block Diagram

6.1 Video Acquisition

The video used in this project is a street view in Abu Dhabi. The number plates of the vehicle in the video are however not clearly visible.

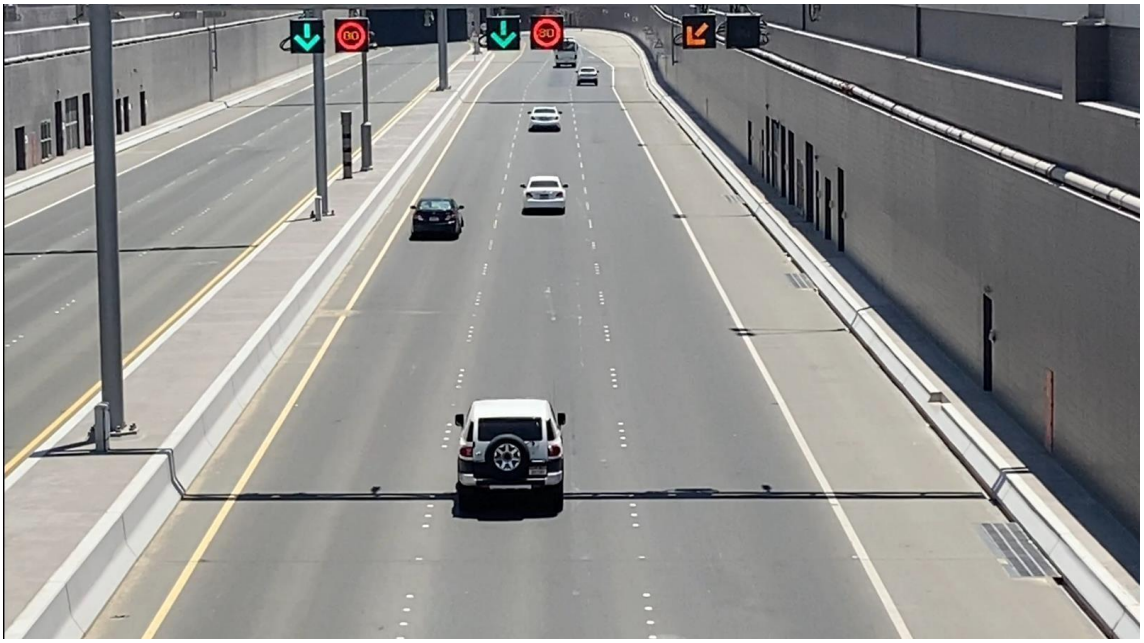

Figure 4 : Sample Video Screenshot

6.2 Region of Interest and Masking

Region of Interest (ROI) takes a smaller portion of the original video. On this ROI, Image subtraction is performed to detect a moving vehicle. (Image Subtraction helps find the difference between two

frames). Masking is performed to make the moving vehicles appear white and the rest of the image black.

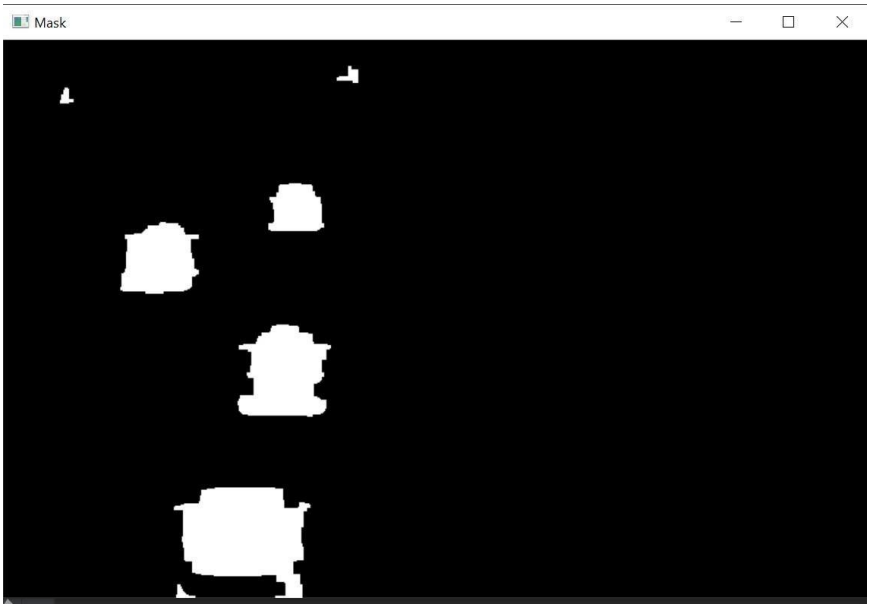

Figure 5 : Masked Image

6.3 Contour Detection and Object Tracking

Based on the area threshold of number of pixels, the contours are detected. The threshold is used to avoid detecting contours of smaller moving objects that are not vehicles. The object is tracked based on the distance between two contours between frames. An ID is assigned to each contour.

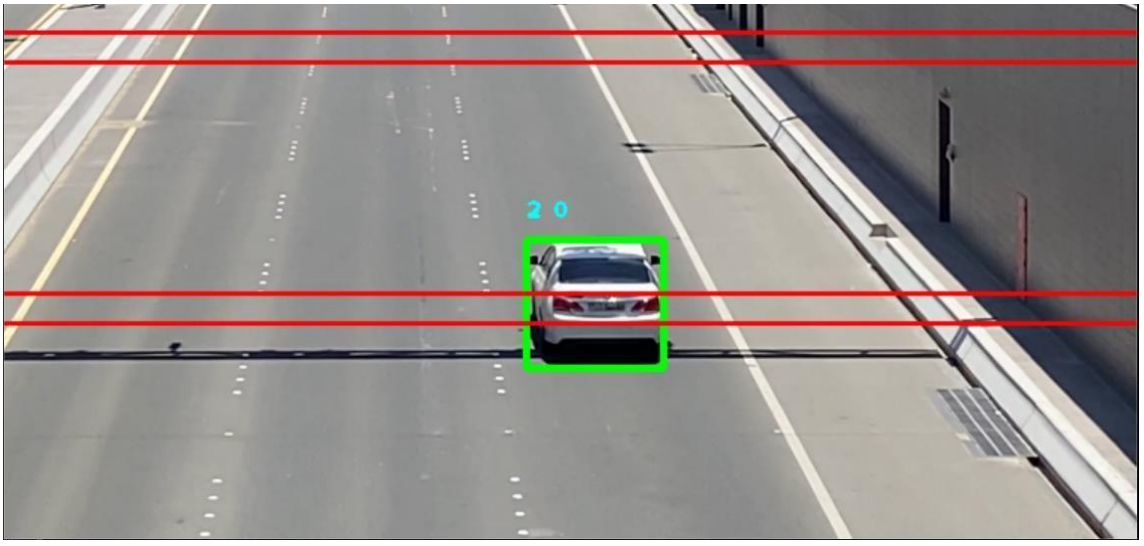

Figure 6 : Contour Detection

6.4 Speed Estimation

Time difference between the position of a vehicle is calculated and the speed is estimated based on a formula. The timer starts when the vehicle crosses the first line, and the timer ends when the vehicle

crosses the second line. The speed is displayed on top of the bounding box only when the vehicle crosses both the lines.

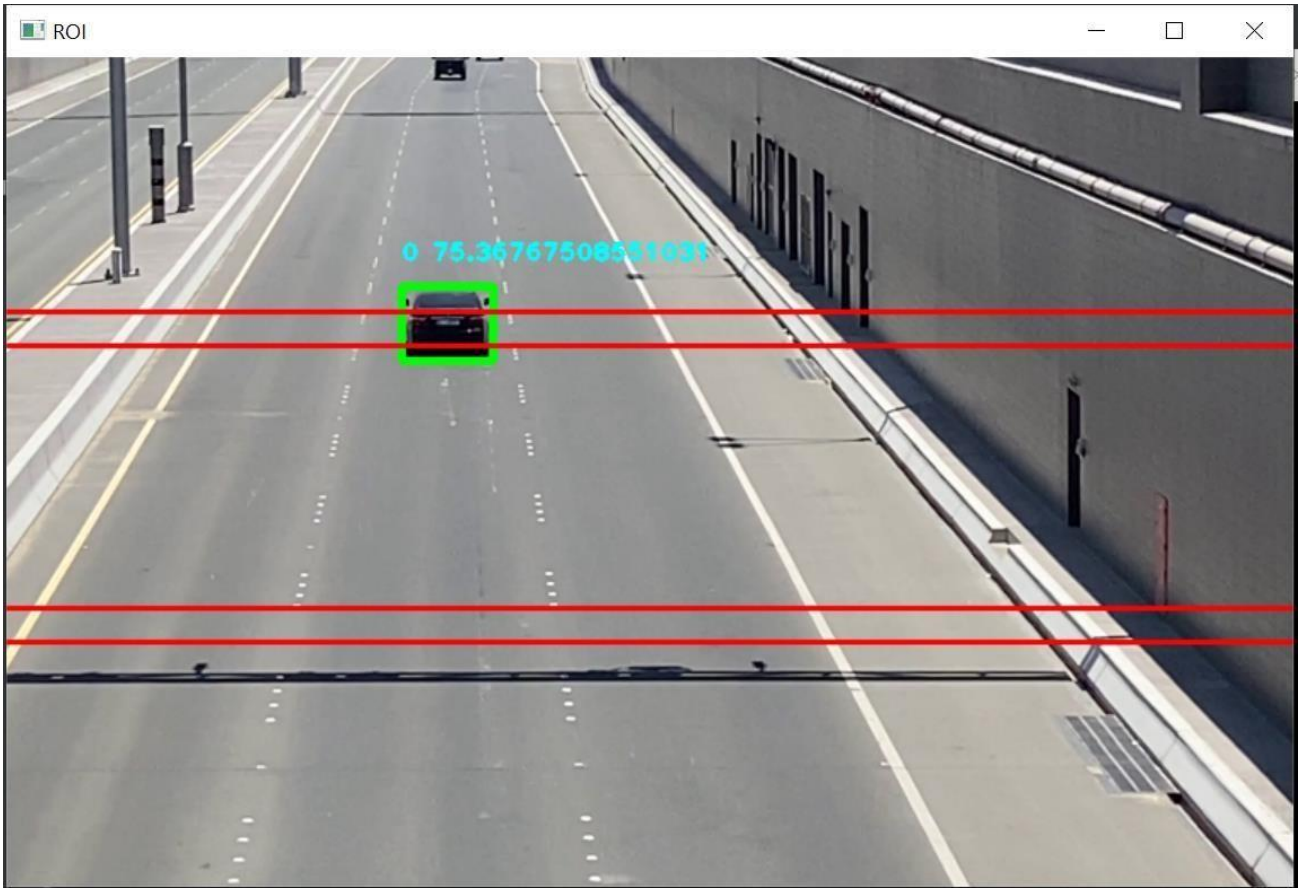

Figure 7 : Speed Estimation

6.5 Save Vehicle Data

The picture of the bounding box (the vehicle) is saved into a file along with the speed. Vehicles crossing the speed limit is segregated into a separate folder.

Figure 8: Saved Vehicle Pictures

6.6 Create Summary

The vehicle data is saved in a text file. The vehicles that exceeded the speed limit are pointed. A summary of number of vehicles and the speed violators are displayed.

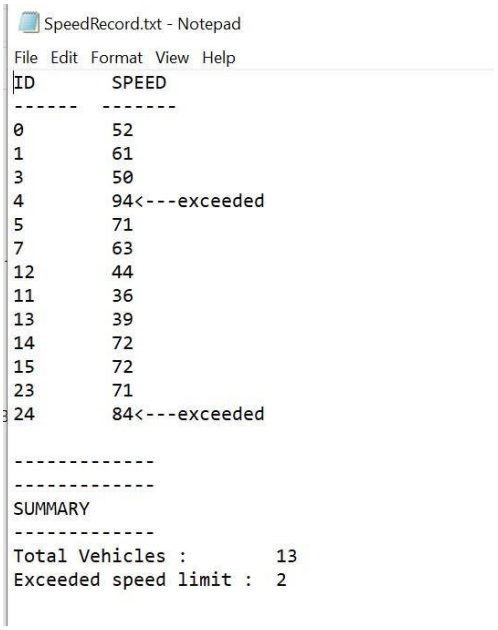

Figure 9 : Saved Summary

7. IMPLEMENTATION

7.1 Video Acquisition

```
cap = cv2.VideoCapture("Resources/traffic3.mp4")
```

7.2 Region of Interest and Masking

```
#KERNALS
kernalOp = np.ones((3,3),np.uint8) kernalOp2 =
np.ones((5,5),np.uint8) kernalCl = np.ones((11,11),np.uint8)
fgbg=cv2.createBackgroundSubtractorMOG2(detectShadows=True)
#MASKING
fgmask = fgbg.apply(roi)
ret, imBin = cv2.threshold(fgmask, 200, 255, cv2.THRESH_BINARY)
mask1 = cv2.morphologyEx(imBin, cv2.MORPH_OPEN, kernalOp) mask2 =
cv2.morphologyEx(mask1, cv2.MORPH_CLOSE, kernalCl)
```

7.3 Contour Detection

```
contours,_ =
cv2.findContours(mask2,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE) detections
= []   for cnt in contours:     #print(cnt)
    area = cv2.contourArea(cnt)     if
area > 1000:          x,y,w,h =
cv2.boundingRect(cnt)

cv2.rectangle(roi,(x,y),(x+w,y+h),(0,255,0),3)
detections.append([x,y,w,h])
```

7.4 Object Tracking

```
for rect in
objects_rect:     x, y,
w, h = rect     cx = (x
+ x + w) // 2     cy =
(y + y + h) // 2

    # Find out if that object was detected
already     same_object_detected = False      for
id, pt in self.center_points.items():
dist = math.hypot(cx - pt[0], cy - pt[1])
if dist < 150:
            self.center_points[id] = (cx, cy)
#print(self.center_points)
            objects_bbs_ids.append([x, y, w, h,
id])             same_object_detected = True
```

7.5 Speed Estimation

7.5.1 Timer Start and stop

```
if (y >= 325 and y <= 345):

    self.s1[id] = time.time()


if (y >= 150 and y <= 170):
    self.s2[id] = time.time()

    self.s[id] = self.s2[id] - self.s1[id]
```

7.5.2 Speed Formula

```
def getsp(self,id):
if (self.s[id]!=0):
        s =
439.8/self.s[id]     else:
s = 0     return s
```

7.6 Drawing Rectangles and displaying on the screen

```
for box_id in boxes_ids:
x,y,w,h,id = box_id

    cv2.putText(roi,str(id)+" "+str(tracker.getsp(id)),(x,y-15),
cv2.FONT_HERSHEY_PLAIN,1,(255,255,0),2)
    #print(tracker.getsp(id))      cv2.rectangle(roi, (x, y),
(x + w, y + h), (0, 255, 0), 3)
```

7.7 Drawing Reference Lines

```
cv2.line(roi, (0, 325), (960, 325), (0, 0, 255), 2) cv2.line(roi,
(0, 345), (960, 345), (0, 0, 255), 2)

cv2.line(roi, (0, 150), (960, 150), (0, 0, 255), 2) cv2.line(roi,
(0, 170), (960, 170), (0, 0, 255), 2)
```

7.8 Save Vehicle Images and speeds

```
def capture(self,img,x,y,h,w,sp,id):
    if(self.capf[id]==0):
        self.capf[id] = 1
        self.f[id]=0
        crop img = img[y-5:y + h+5, x-5:x + w+5]
        n = str(id)+" speed "+str(sp)
        file = 'D://TrafficRecord//' + n + '.jpg'
        cv2.imwrite(file, crop img)
        self.count += 1
        filet = open("D://TrafficRecord//SpeedRecord.txt", "a")
        if(sp>limit):
            file2 = 'D://TrafficRecord//exceeded//' + n + '.jpg'
            cv2.imwrite(file2, crop img)
            filet.write(str(id)+" \t "+str(sp)+"<---exceeded\n")
            self.exceeded+=1
        else:
            filet.write(str(id) + " \t " + str(sp) + "\n")
        filet.close()
```

7.9 Create Summary

```
def    end(self):                                    file    =
open("D://TrafficRecord//SpeedRecord.txt",              "a")
file.write(" \ n ------------ \ n")      file.write(" ---------
---- \ n")      file.write ( "SUMMARY \ n")      file.write(" --
---------- \ n")            file.write("Total  Vehicles  : \
t"+str(self.count)+" \ n")     file.write("Exceeded speed limit
: \ t"+str(self.exceeded))      file.close()
```

8. OUTPUT SCREENSHOTS

8.1 Speed Detection

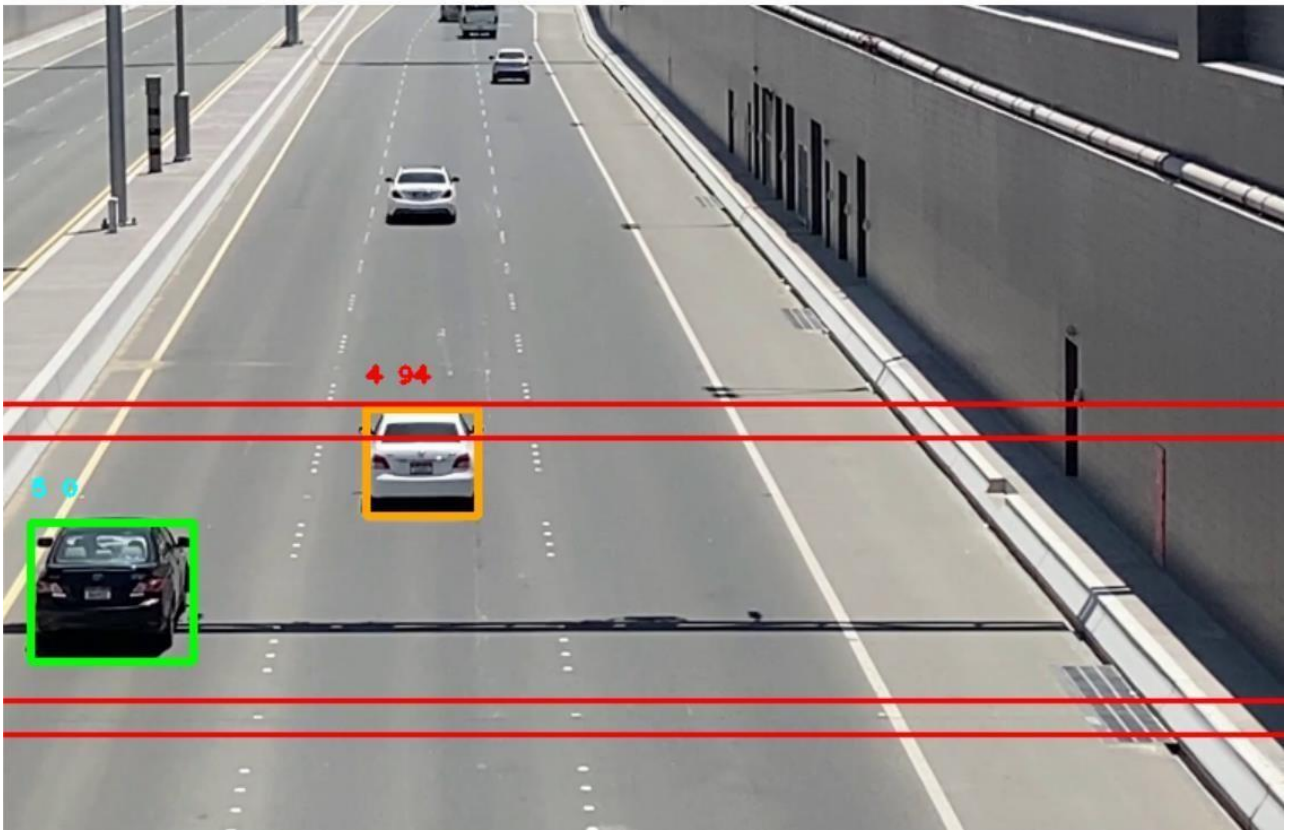

Figure 10 : Speed Radar Main Output

8.2 Saved Image Pictures

Figure 11: Saved Vehicle Images

8.3 Saved Vehicle Data

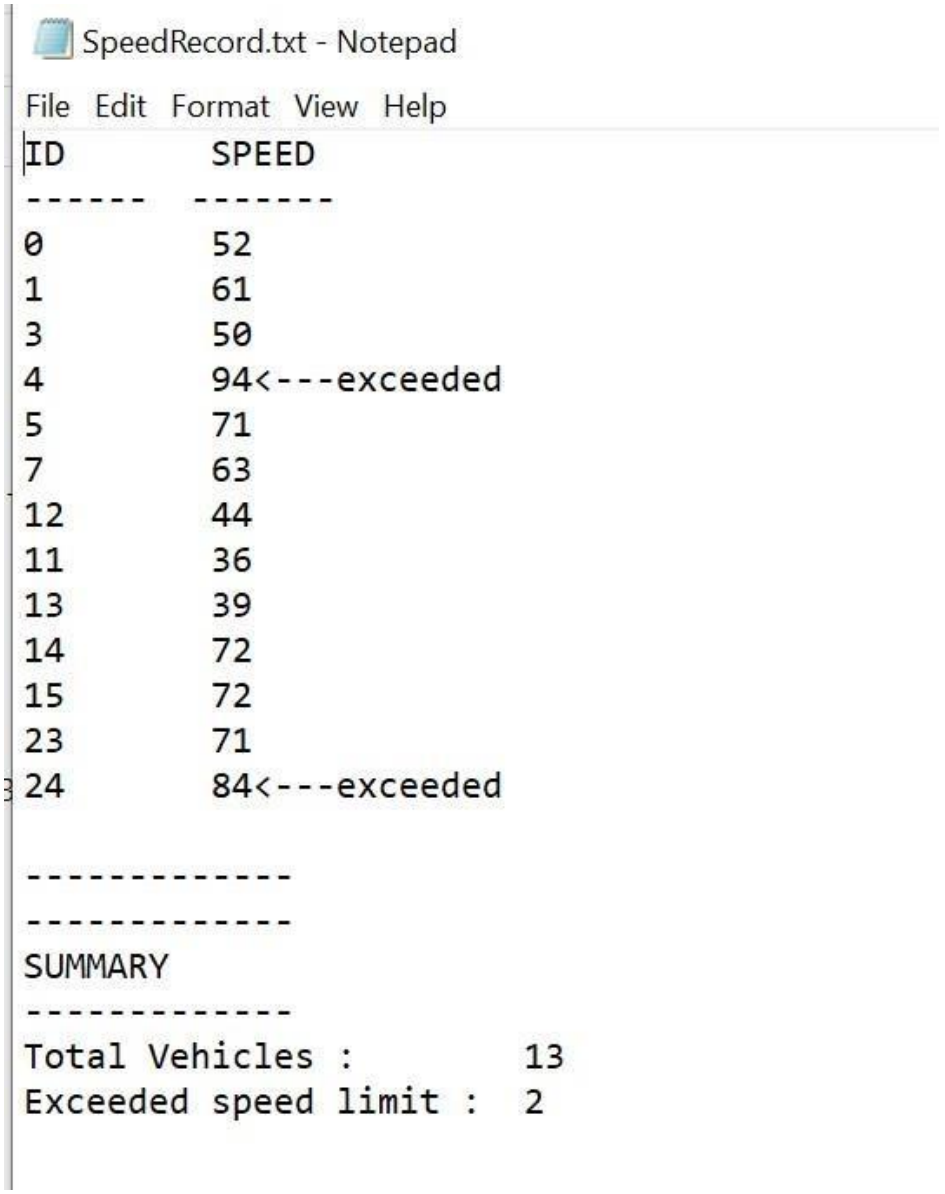

Figure 12 : Summary of Vehicles

9. RESULTS ANALYSIS

This project is successfully able to track vehicles and estimate their speed. Accuracy in detecting vehicles is 100% as long as there is no movement in the camera. Estimation of speed can have a difference of 0-2 km/hr depending of the program execution speed.

Multiple vehicles can be detected, and their speeds can be detected. However if two vehicles are moving extremely close to each other, it may be detected as a single object.

This project requires the camera to be as still as possible, as movement is used to distinguish vehicles from the background.

10.CONCLUSION

Road safety is an important factor for the police force. And as citizens it is our responsibility to follow rules and maintain safety on our roads. This project is able to estimate speed of vehicles and save vehicle data.

Future Improvements can be finding other violations like wrong lane detection and tailgating detection.

11.REFERENCES

[1] ScienceDirect – 2020 - Determining Vehicle Speeds Using Convolutional Neural Networks by Alexander Grents, Vitalii Varkentin*, Nikolay Goryaev

[2] ScienceDirect – 2020 - Detection of Vehicle Position and Speed using Camera Calibration and Image Projection Methods by Alexander A S Gunawana,* , Deasy Aprilia Tanjunga, Fergyanto E. Gunawan

[3] ScienceDirect – 2020 - An Efficient Approach for Detection and SpeedEstimation of Moving Vehicles By Tarun Kumar and Dharmender Singh Kushwaha

[4] YouTube – PySource youtube Channel

[5] ScienceDirect – 2020 - Vehicle speed measurement model for video-based systems by Saleh Javadi , Mattias Dahl, Mats I.Pettersson

[6] YouTube 25th March 2020 – OpenCV – Murtaza's Workshop