VEHICLE SPEED DETECTION

A Project Work

Submitted in the partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE (AI&ML)

Submitted by:

TUSHAR CHAUHAN

SAKSHAM BAHUGUNA

MOHIT MELWANI

University Roll Number :

18BCS6162

18BCS6164

18BCS6209

Under the Supervision of:

YOGIRAJ BHALE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
APEX INSTITUE OF TECHNOLOGY
CHANDIGARH UNIVERSITY, GHARUAN, MOHALI - 140413,
PUNJAB

APRIL,2021

DECLARATION

I, **'TUSHAR CHAUHAN, SAKSHAM BAHUGUNA, MOHIT MELWANI**' student of **'Bachelor of Engineering in Computer Science(AI&Ml)**, **session: 2021.** Department of Computer Science and Engineering, Apex Institute of Technology, Chandigarh University, Punjab, hereby declare that the work presented in this Project Work entitled '**VEHICLE SPEED DETECTION'** is the outcome of our own bona fide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics. It contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

TUSHAR CHAUHAN

Candidate UID: 18BCS6162

Date: April 25, 2021

Place: Mohali

ACKNOWLEDGEMENT

I have taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and organizations. I would like to extend my sincere thanks to all of them.
I take this opportunity to express my profound gratitude and deep regards to my teacher Prof. Neha Sharma (Chandigarh University, Gharaun, Punjab) for their exemplary guidance, monitoring and constant encouragement throughout the course of this project. The blessing, help and guidance given by them time to time shall carry me a long way in the journey of life on which I am about to embark.
I would also like to thank my parents and friends who helped me complete this project within the deadline.

ABSTRACT

The daily life of people encounters more problems as the population continuously increases in urban area, and road traffic becomes more congested because of high demand and less level of road capacity and infrastructure. Since the effects of these problems are significant in daily life, it is important to seek efficient solutions to reduce them.

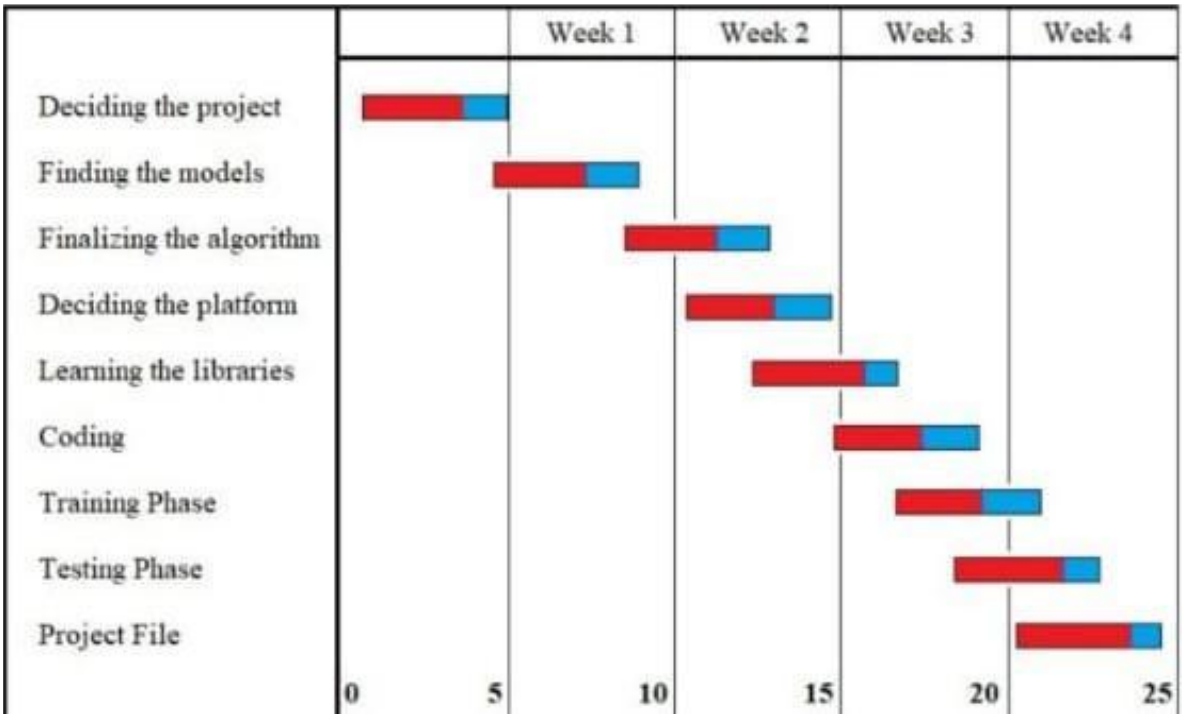
Road accidents have been very common in the present world with the prime cause being the careless driving. The necessity to check this has been very essential and different methods have been used so far. However with the advancement in the technology, different governing bodies are demanding some sort of computerized technology to control this problem of over speed driving. Vehicle speed detection is very important for observing speed limitation law and it also demonstrates traffic conditions. Traditionally, vehicle speed detection or surveillance was obtained using radar technology, particularly, radar detector and radar gun.

Traffic congestion is the main problem faced by big cities, such as Jakarta. One approach to reduce congestion levels is to improve traffic management that regulates and controls the number of vehicles. To evaluate the impact of traffic management before direct implementation on the highway, traffic modeling can be carried out. Parameters in modeling traffic must be determined from a calibration process where the vehicle is accurately measured for its position and speed. This study aims to propose an efficient calibration procedure with accurate results, based on recorded vehicle movement in perspective view. First, the road image is projected using the Direct Linear Transformation (DLT) method, then the vehicle position is detected using the Background Subtraction and tracked using Mixture of Gaussian (MoG) to determine the vehicle speed. Finally, we develop a prototype of Automated Traffic Flow Monitoring based on Python programming. In the experiment results, the accuracy of vehicle position detection is evaluated based on the Euclidean distance. The average difference between the results of position detection with ground-truth is 12.07 pixels with a camera angle 40 °. The percentage of speed measurement accuracy using the DLT projection method is 96.14%.

Table of Contents

LIST OF FIGURES

Gantt chart


Week 1
Week 2
Week 3
Week 4
Deciding the project
Finding the models
Finalizing the algorithm
Deciding the platform
Learning the libraries
Coding
Training Phase
Testing Phase
Project File
0
5
10
15
20
25

1.INTRODUCTION

PROBLEM DEFINITION

Road accidents have been very common in the present world with the prime cause being the careless driving. The necessity to check this has been very essential and different methods have been used so far. Vehicle speed detection is very important for observing speed limitation law and it also demonstrates traffic conditions. Video and image processing has been used for traffic surveillance, analysis and monitoring of traffic conditions in many cities and urban areas.Our project estimates the real-time vehicle speed on roads from any real-time stream or recorded video.

WHOM IT MAY HELP?

This simple but efficient approach of Vehicle Speed Detector can be applied in the places where traffic rules are not followed in a strict manner. Authorities may install this on the roads and keep check on the vehicle drivers.

PROJECT OVERVIEW/SPECIFICATIONS

This project Vehicle Speed Detector aims at detecting the vehicle and finding the accurate speed of the vehicle under a given input in the form of a video. This project uses OpenCV, Deep learning and computer vision. The input is the video or mp4 file of any road or highway. After the computational work done at the backend in the code, the output is the objects traced by red rectangles with the speed over box.

HARDWARE SPECIFICATION

1: Camera (High resolution)
2: CPU – i3 and above.
3: Min Ram 8gb
4: NVIDIA GPU
5: Screen

1.5 SOFTWARE SPECIFICATION

1: Windows 10 and above.
2: Anaconda Jupyter Notebook.
3: Python 3

Libraries:

- OpenCV
- math
- time
- numpy
- flask
- os

2.LITERATURE REVIEW

2.1 EXISTING SYSTEM

Traditionally, vehicle speed detection or surveillance was obtained using radar technology, particularly, radar detector and radar gun. The radar system operation is known as Doppler shift phenomenon. The basic concept about this system is Doppler shift that happens when the created sound is reflected off a moving vehicle and the frequency of the returned sound is slightly changed. This method, with spatial equations and equipments, obtains the speed of a moving vehicle. However, this method still has several disadvantages such as the cosine error that happens when the direction of the radar gun is not on the direct path of the incoming vehicle. In addition, the cost of equipment is one of the important reasons, and also shading (radar wave reflection from two different vehicles with distinctive heights), and radio interference (error caused by the existence of similar frequency of the radio waves on which a transmission is broadcasted) are two other influential factors that cause errors for speed detection and finally, the fact that radar sensor can track only one car at any time is another limitation of this method.

2.2 PROPOSED SYSTEM

Our model/project used open cv, deep learning and computer vision for the detection of vehicle and calculating the speed of vehicle.
As to find the speed we have to know the distance and time. And as the distance of road is not provided in the video so average mode is used for the calculation of speed where average of some initial vehicles speed with respect to coordinates is taken and converted into real world speeds using a defined parameter which depends on the deviation from calculated average speed. Average mode samples a certain number of vehicles to find there average speed on screen (in pixels). Subsequent cars are compared to the average, and their speeds are reported as percent differences from the average. This mode is useful when you don't know the distance of the road in the video, so it can be applied to almost any road. It's important to note that speed is calculated once a vehicle passes the light blue line
Final speed is provided by dividing the difference between initial and final coordinates by time until vehicle crossed the light blue line optimization note: processing time is compensated with real time hence the time used for processing does not affect the actual time, vehicle has taken.
Object detection is applied to detect only vehicle from the input video..

3.PROBLEM FORMULATION

Road accidents have been very common in the present world with the prime cause being the careless driving. The necessity to check this has been very essential and different methods have been used so far.
Speeding is one of the biggest traffic violations. It endangers everyone on the road. According to NHTSA [1], in year 2017 speeding killed 9717 people in the USA. Some of the consequences of over speeding are loss of vehicle control, increased stopping distance, economic losses, increased fuel consumption and loss of lives. As crash speeds get very high, airbags and seat belts may not work as well to protect the passengers from the collision. Over speeding costs of billions of dollars to the country's economy [1]. For example, Orland park, Illinois with population of 60000, police issued 26,821 citations. Out of those 4732 were for speeding. Law enforcement officers spend a significant amount of time in catching over speed violators. Also, a very small fraction of violators is caught by the existing system.
This project estimates the real-time vehicle speed on roads from any real-time stream or recorded video.
To find a car's speed, we need to know how its moving from frame to frame. We can already detect cars on any given frame, but we need a kind of permanence to detect as the move in the video. This is a rather long process, but in general we compare the current detections to the previous detections, and based on manually set parameters, we determine whether or not the new detections are valid movement
As the distance of road is not provided in the video so average mode is used for the calculation of speed where average of some initial vehicles speed with respect to coordinates is taken and converted into real world speeds using a defined parameter which depends on the deviation from calculated average speed. Average mode samples a certain number of vehicles to find there average speed on screen (in pixels). Subsequent cars are compared to the average, and their speeds are reported as percent differences from the average. This mode is useful when you don't know the distance of the road in the video, so it can be applied to almost any road. It's important to note that speed is calculated once a vehicle passes the light blue line .

Final speed is provided by dividing the difference between initial and final coordinates by time until vehicle crossed the light blue line Optimization note: processing time is compensated with real time hence the time used for processing does not affect the actual time, vehicle has taken.

4. RESEARCH OBJECTIVES

The research objectives for the Vehicle Speed Detector can be defined as:

4.1: General Objective:

Vehicle Speed Detector aims at providing accurate results of the calculated speed of vehicle from given input stream.

4.2: Specific Objectives:

The model is based on basically three steps that are:

*Object detection (vehicles)

*Create the bounding box outside the vehicle

*Computing the speed of the vehicle

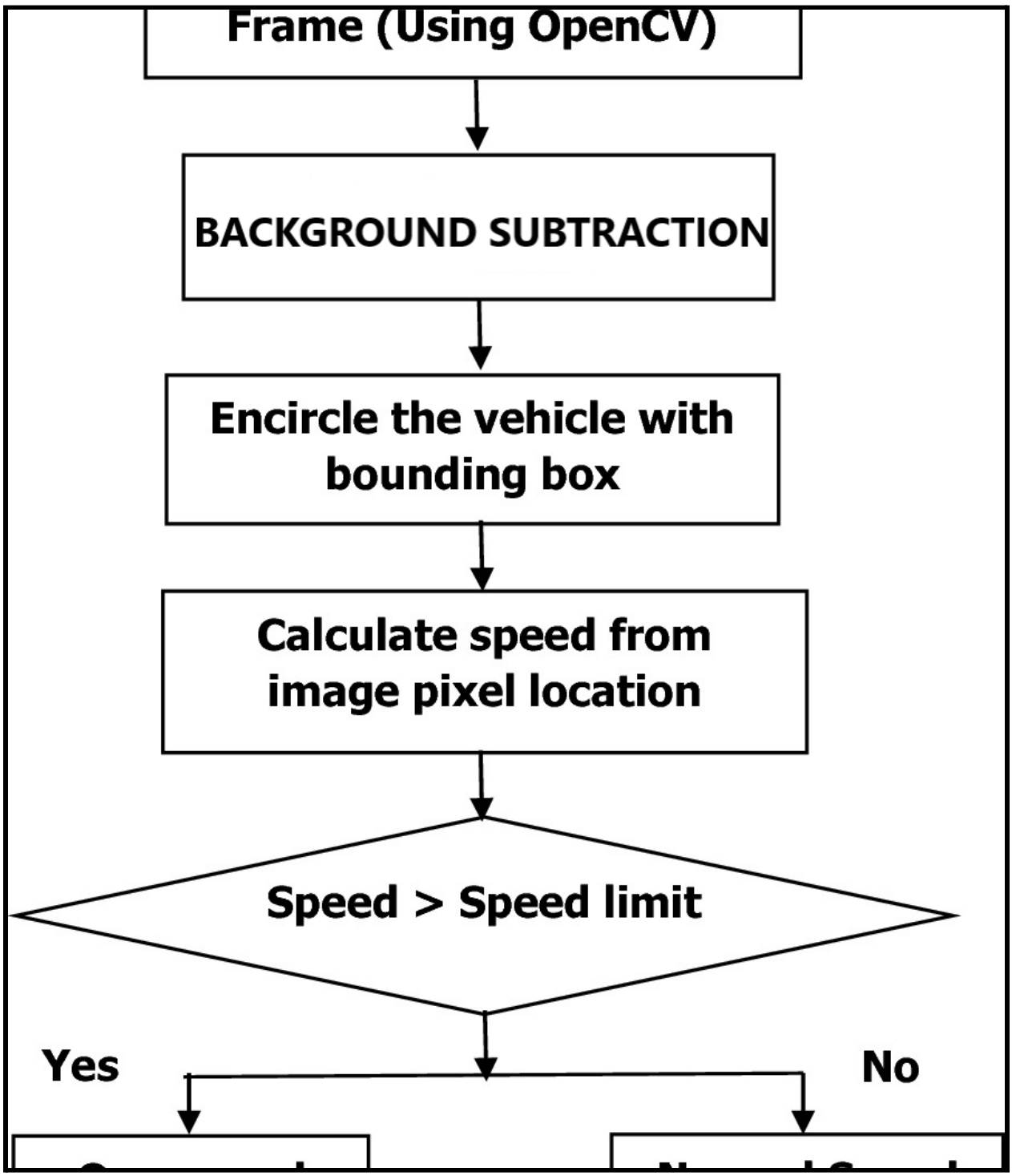
4.3: Immediate Objectives:

We have proposed this model as this model is the need of the hour. It helps in calculating the accurate speed of vehicles and also helps authorities to keep check on careless drivers or on the people who are not obeying laws

5.METHODOLOGY

The following methodology will be followed to achieve the objectives defined for proposed research work:
The model/project used OpenCV, deep learning and computer vision for the detection of vehicles and calculate it’s speed. Object detection is applied to detect vehicles from the either input video or real time streams.
To find a car's speed, we need to know how its moving from frame to frame. We can already detect cars on any given frame, but we need a kind of permanence to detect as the move in the video. This is a rather long process, but in general we compare the current detections to the previous detections, and based on manually set parameters, we determine whether or not the new detections are valid movement
As the distance of road is not provided in the video so average mode is used for the calculation of speed where average of some initial vehicles speed with respect to coordinates is taken and converted into real world speeds using a defined parameter which depends on the deviation from calculated average speed. Average mode samples a certain number of vehicles to find there average speed on screen (in pixels). Subsequent cars are compared to the average, and their speeds are reported as percent differences from the average. This mode is useful when you don't know the distance of the road in the video, so it can be applied to almost any road. It's important to note that speed is calculated once a vehicle passes the light blue line 12

Final speed is provided by dividing the difference between initial and final coordinates by time until vehicle crossed the light blue line Optimization note: processing time is compensated with real time hence the time used for processing does not affect the actual time, vehicle has taken.

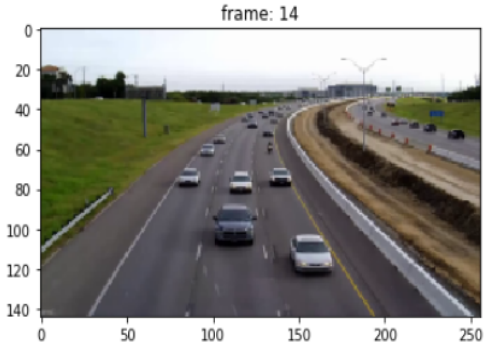

Frame (Using OpenCV)
BACKGROUND SUBTRACTION
Encircle the vehicle with bounding box
Calculate speed from image pixel location
Speed > Speed limit
Yes
No

1. Import Video Frames

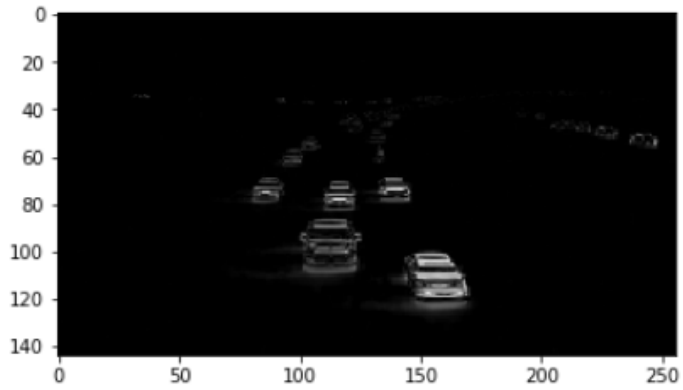
- Open the Video file or camera using **cv2.VideoCapture()**
- Read frame by frame
- Save each frame using **cv2.imwrite()**
- Release the Video Capture and destroy all windows

2. Data Exploration

Let's display two consecutive frames:



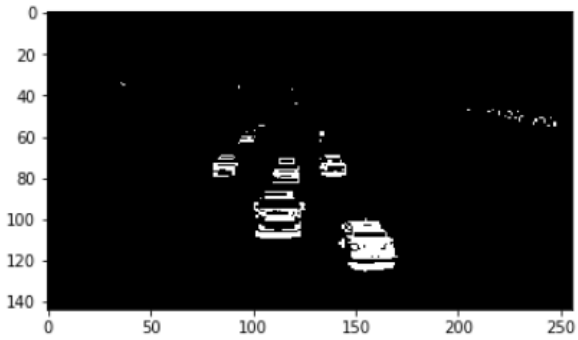
It is hard to find any difference in these two frames, isn't it? As discussed earlier, taking the difference of the pixel values of two consecutive frames will help us observe the moving objects. So, let's us covert the frames into grayscale.



Now we can clearly see the moving objects in the 13th and 14th frames. Everything else that was not moving has been subtracted out.

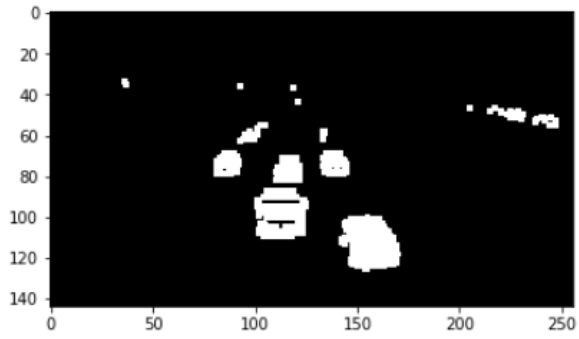
3. Image Pre-processing

Let's see what happens after applying thresholding to the above image:



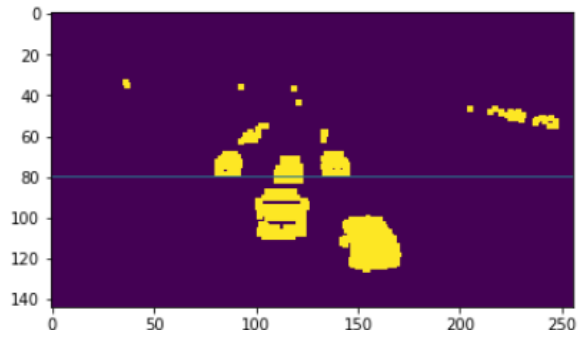
Now, the moving objects (vehicles) look more promising and most of the noise (undesired white regions) are gone. However, the highlighted regions are a bit fragmented. So, we can apply image dilation over this image:

The moving objects have more solid highlighted regions. Hopefully, the number of contours for every object in the frame will not be more than three.



However, we are not going to use the entire frame to detect moving vehicles. We will first select a zone, and if a vehicle moves into that zone, then only it will be detected.

So, let me show you the zone that we will be working with:



The area below the horizontal line y = 80 is our vehicle detection zone. We will detect any movement that happens in this zone only. You can create your own detection zone if you want to play around with the concept.

Now let's find the contours in the detection zone of the above frame:

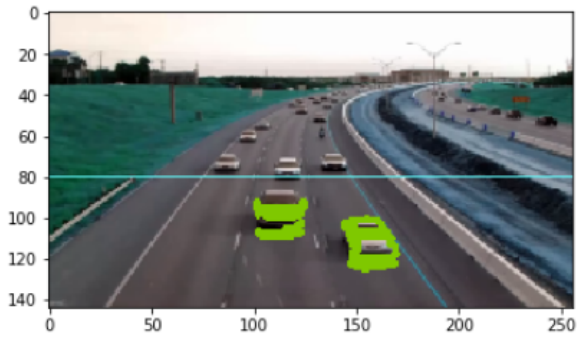
```
# find contours


contours, hierarchy =
cv2.findContours(thresh.copy(),cv2.RETR_TREE,cv2.CHAIN_APPROX_NONE)
```

The code above finds all the contours in the entire image and keeps them in the variable '*contours'*. Since we have to find only those contours that are present in the detection zone, we will apply a couple of checks on the discovered contours.

The first check is whether the top-left y-coordinate of the contour should be >= 80 (I am including one more check, x-coordinate <= 200). The other check is that the area of the contour should be >= 25. You can find the contour area with the help of the *cv2.contourArea( )* function.

Next, let's plot the contours along with the original frame:



6.RESULTS AND DISCUSSION

Intelligent Transportation System is one of the important components in the development of smart cities. Detection of vehicle speed on the highway is supporting the management of traffic engineering. The purpose of this study is to detect the speed of the moving vehicles using digital image processing. Our approach is as follows: The inputs are a sequence of frames, frame rate (fps) and ROI. The steps are following: First we separate foreground and background using Gaussian Mixture Model (GMM) in each frames. Then in each frame, we calculate the location of object and its centroid. Next we determine the speed by computing the movement of centroid in sequence of frames. In the calculation of speed, we only consider frames when the centroid is inside the predefined region of interest (ROI). Finally we transform the pixel displacement into a time unit of km/hour. Validation of the system is done by comparing the speed calculated manually and obtained by the system. The results of software testing can detect the speed of vehicles with the highest accuracy is 96.17% and the lowest accuracy is 77.41%. And the detection results of testing by using real video footage on the road is included with real speed of the vehicle.

7.References

1.E. Atkociunas, R. Blake, A. Juozapavicius, M. Kazimianec, Image Processing in Road Traffic Analysis, Nonlinear Analysis: Modelling and Control, 2005, Vol. 10, No. 4, 315-33
Google Scholar

2.Arash Gholami Rad, Abbas Dehghani and Mohamed Rehan Karim, "Vehicle speed detection in video image sequences using CVS method", at International Journal of the Physical Sciences Vol. 5(17), pp. 2555-2563, 18 December, 2010, ISSN 1992-1950 © 2010 Academic Journals
Google Scholar

3.H. P. Moravec. Towards Automatic Visual Obstacle Avoidance. Proc. 5th International Joint Conference on Artificial Intelligence, pp. 584, 1977
Google Scholar

4.Z. Zheng, H. Wang and EKTeoh. Analysis of Gray Level Corner Detection. Pattern Recognition Letters, Vol. 20, pp. 149-162, 1999
CrossRef Google Scholar

5.Rafael C. Gonzalez and Richard E. Woods "Digital Image Processing" Prentice Hall, 2001
Google Scholar

6.D. A. Forsyth, J. Ponce. "Computer Vision. A Modern Approach", Prentice Hall, 2003
Google Scholar

7.Open source computer vision library - reference manual, INTEL Corporation, 1999-2001.
Google Scholar

8.News Article: http://articles.timesofindia.indiatimes.com/2012-06-27/chandigarh/32439930-1-road-accidents-nakas-internalroads