e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:05/Issue:02/February-2023 Impact Factor- 7.868 www.irjmets.com

VEHICLE SPEED DETECTION USING OPEN CV IN PYTHON

Prof. N. K. Budhnerm[*1], Komal Rangam[*2], Shriya Shinde[*3], Sara Sayyed[*4], Zaid Tamboli[*5]

[*1,2,3,4,5]Department of Computer Engineering, AISSMS College, RB Motilal Kennedy Rd, Near R.T.O, Railway Officers Colony, Sangamvadi, Pune,Maharashtra 411001

ABSTRACT

In accordance with information from a video source that has been recorded, this project estimates the speed of a vehicle. The number of accidents rises as the number of vehicles does as well. Thus, it's crucial to set speed limits for cars in specific zones or locations.Thus, a better method of determining the speed of moving vehicles is required. The vehicles' video streaming might be used for this instead of spending money on pricy sensors like radarsIn addition to placing significant strain on road capacity, the steadily growing number of on-road cars has made traffic management challenging and given rise to issues including congestion, crashes, and air pollution, among others. Our daily lives are significantly impacted by these issues. To lessen the impact, a robust, healthy, and effective traffic management system is required. In addition to these issues with vehicle traffic, it is also possible to study statistical factors that can help with highway management, such as the typical number of vehicles on the road at any given moment and the level of congestion

I. INTRODUCTION

In the recent years we can see there is a vast increase in the number of vehicles all around the globe. Along with the increase in number of vehicles increases the number of accidents. Therefore, it is important to limit the speed of the vehicles at certain zones or areas. Radar speed measurement tools are commonly used for this purpose which can be inaccurate in certain cases such as in sensing smaller vehicles with weaker echoes. Also, it is difficult for these tools to detect vehicles changing in speeds too often or fast. Therefore, there is a need for a better technique to detect the speed of the moving vehicles. Instead than spending a fortune on expensive sensors like radars, this may be accomplished by leveraging the vehicle's video streaming. As an input, the moving vehicle's video stream is provided, after which it is passed through a filter to determine its speed.

II. METHODOLOGY

The below shown figure (fig 1) demonstrates the block diagram of our vehicle speed detection system.

The block diagram below explains that firstly, a video is given as input to the system. The given input video is at first preprocessed according to the requirements. From the processed video sample, the vehicle is detected using the filters. This vehicle is then tracked and analyzed in order to find its speed.

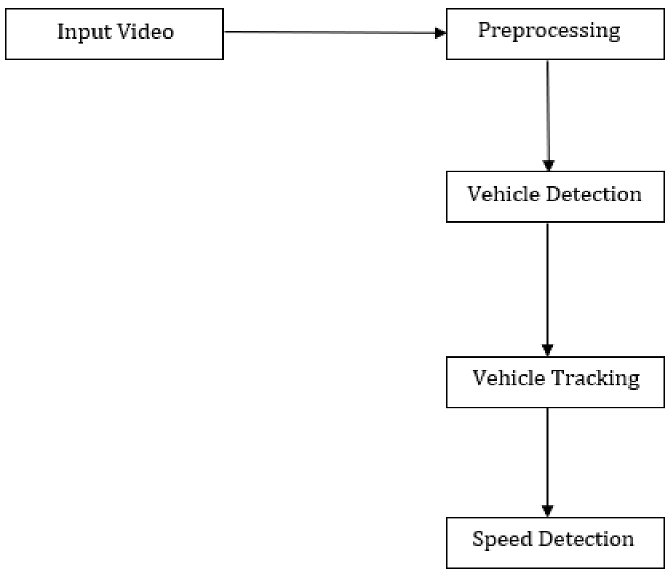


Fig. 1: Block diagram of vehicle speed detection system

III. INPUT VIDEO

Our primary requirement is to use a camera to capture a live stream of a moving car. We utilize OpenCV for this purpose. Grayscale conversion is done on the camera-captured footage before further processing. For capturing a live stream video, a Video Capture object is constructed. Either the device report or the title of a video record can be the topic of discussion. The video will be clever, and if the price is really expensive, the video will be mediocre (Well, that is the course by which you can demonstrate accounts in moderate movement). In most situations, 25 milliseconds will be adequate.

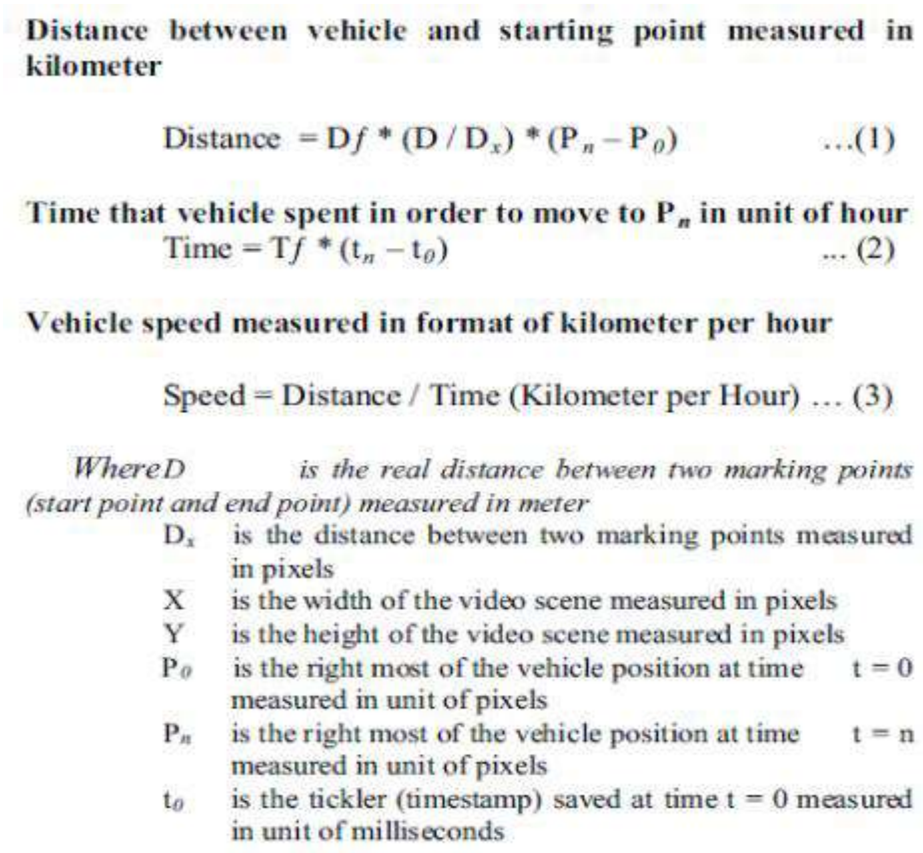
IV. PREPROCESSING

The goal of preprocessing is to enhance the visual data in an image or video.

The quantity of subcomponents that use different enhancement or correction characteristics on an input image The subcomponents control the rectified image when one or more preprocessing settings are enabled.

V. OBJECT/VEHICLE DETECTION

The object detection in our approach hinges on a flexible foundation subtraction technique known as the Gaussian blend model. After this model framework has collected every pixel, the DBSCAN (Density-based spatial social affair of organizations with turmoil) gathering technique is used to display portions of the frontal area foci.



Distance between vehicle and starting point measured in kilometer

$$\text{Distance} = Df * (D / D_x) * (P_n - P_0) \qquad \ldots(1)$$

Time that vehicle spent in order to move to P_n in unit of hour

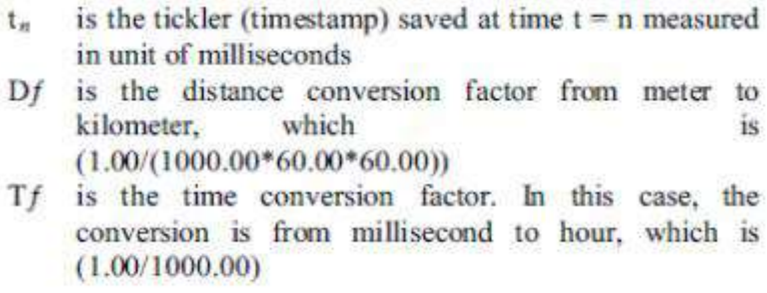
$$\text{Time} = Tf * (t_n - t_0) \qquad \ldots(2)$$

Vehicle speed measured in format of kilometer per hour

$$\text{Speed} = \text{Distance} / \text{Time (Kilometer per Hour)} \ldots (3)$$

Where D is the real distance between two marking points (start point and end point) measured in meter

D_x	is the distance between two marking points measured in pixels
X	is the width of the video scene measured in pixels
Y	is the height of the video scene measured in pixels
P_0	is the right most of the vehicle position at time $t = 0$ measured in unit of pixels
P_n	is the right most of the vehicle position at time $t = n$ measured in unit of pixels
t_0	is the tickler (timestamp) saved at time $t = 0$ measured in unit of milliseconds

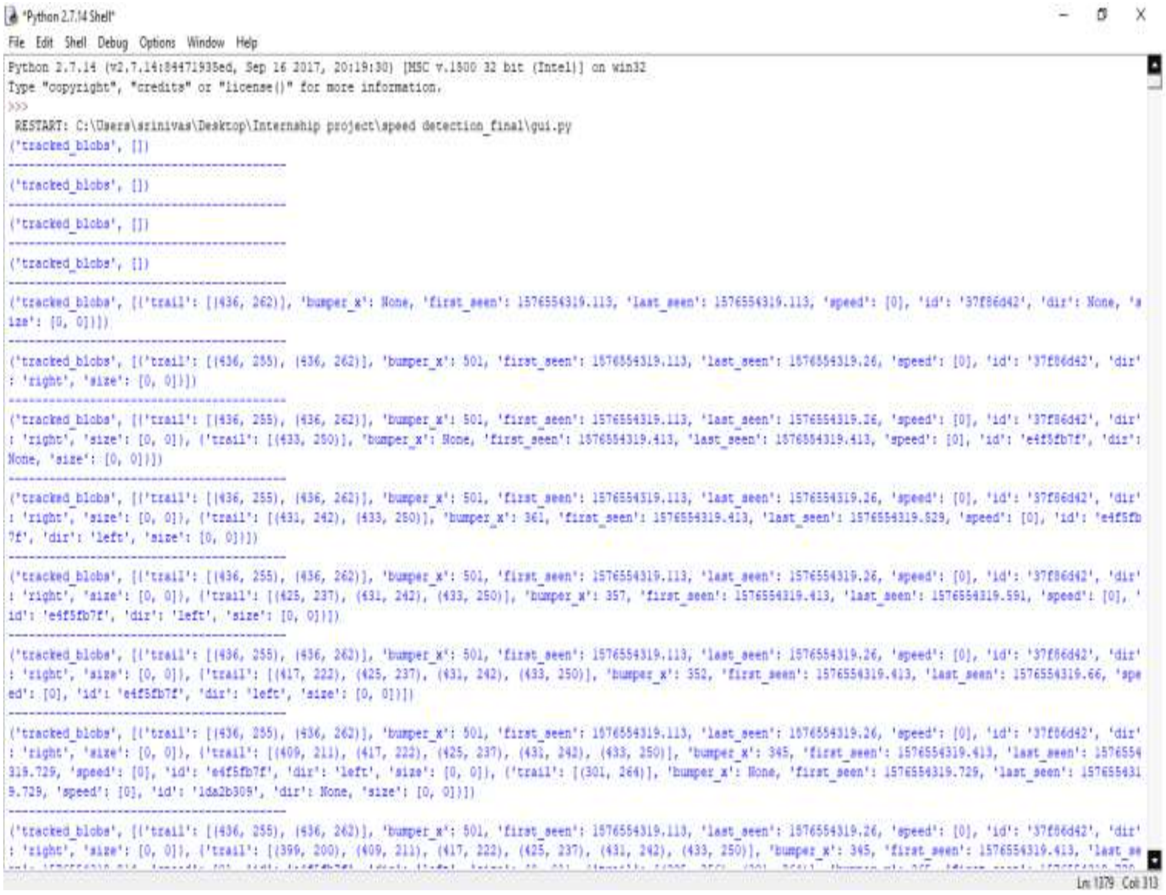


t_n	is the tickler (timestamp) saved at time $t = n$ measured in unit of milliseconds
Df	is the distance conversion factor from meter to kilometer, which is (1.00/(1000.00*60.00*60.00))
Tf	is the time conversion factor. In this case, the conversion is from millisecond to hour, which is (1.00/1000.00)

e-ISSN: 2582-5208
International Research Journal of Modernization in Engineering Technology and Science
(Peer-Reviewed, Open Access, Fully Refereed International Journal)
Volume:05/Issue:02/February-2023 Impact Factor- 7.868 www.irjmets.com

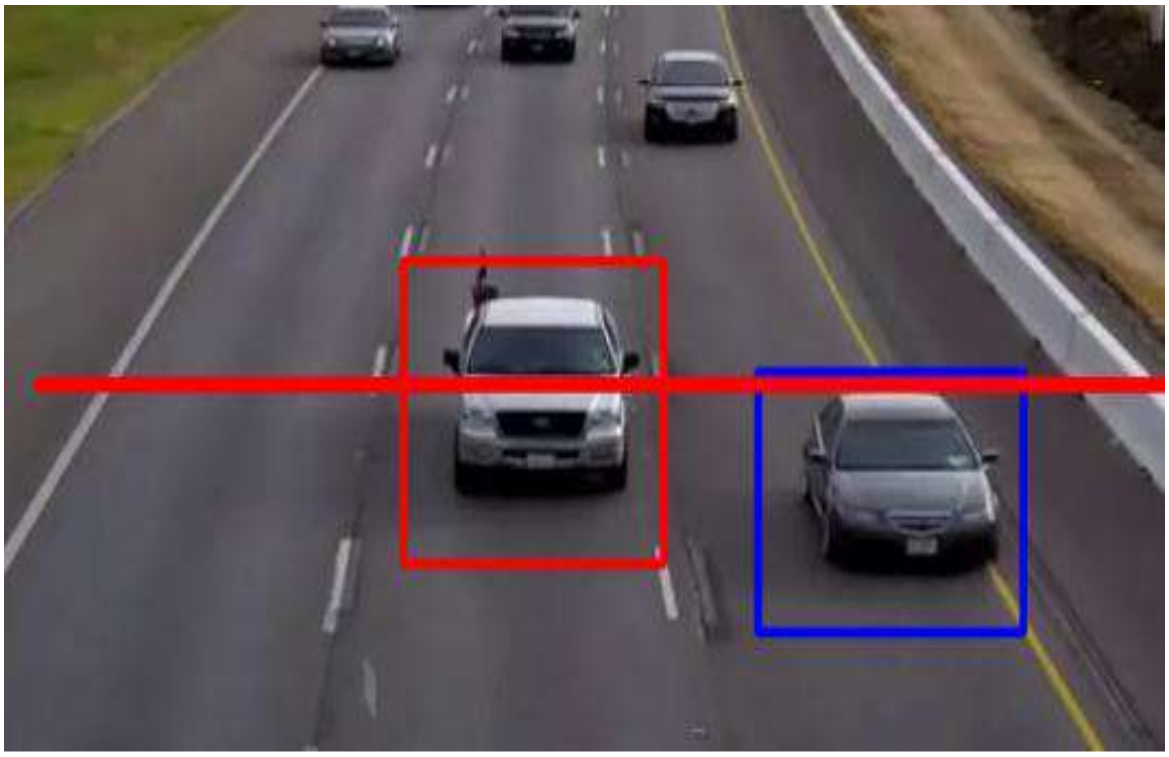
VI. RESULTS AND DISCUSSION

In this study, the vehicle's speed is determined using the optical stream Lucas Kanade system and the Kalman channel. The video is first provided to the system as input. The preprocessing of this raw video follows. The above techniques are used to examine this processed video.



```
Python 2.7.14 (v2.7.14:84471935ed, Sep 16 2017, 20:19:30) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
 RESTART: C:\Users\srinivas\Desktop\Internship project\speed detection_final\gui.py
('tracked_blobs', [])
----------------------------------------
('tracked_blobs', [])
----------------------------------------
('tracked_blobs', [])
----------------------------------------
('tracked_blobs', [])
----------------------------------------
('tracked_blobs', [{'trail': [(436, 262)], 'bumper_x': None, 'first_seen': 1576554319.113, 'last_seen': 1576554319.113, 'speed': [0], 'id': '37f86d42', 'dir': None, 's
ize': [0, 0]}])
----------------------------------------
('tracked_blobs', [{'trail': [(436, 255), (436, 262)], 'bumper_x': 501, 'first_seen': 1576554319.113, 'last_seen': 1576554319.26, 'speed': [0], 'id': '37f86d42', 'dir'
: 'right', 'size': [0, 0]}])
----------------------------------------
('tracked_blobs', [{'trail': [(436, 255), (436, 262)], 'bumper_x': 501, 'first_seen': 1576554319.113, 'last_seen': 1576554319.26, 'speed': [0], 'id': '37f86d42', 'dir'
: 'right', 'size': [0, 0]}, {'trail': [(433, 250)], 'bumper_x': None, 'first_seen': 1576554319.413, 'last_seen': 1576554319.413, 'speed': [0], 'id': 'e4f5fb7f', 'dir':
None, 'size': [0, 0]}])
----------------------------------------
('tracked_blobs', [{'trail': [(436, 255), (436, 262)], 'bumper_x': 501, 'first_seen': 1576554319.113, 'last_seen': 1576554319.26, 'speed': [0], 'id': '37f86d42', 'dir'
: 'right', 'size': [0, 0]}, {'trail': [(431, 242), (433, 250)], 'bumper_x': 361, 'first_seen': 1576554319.413, 'last_seen': 1576554319.529, 'speed': [0], 'id': 'e4f5fb
7f', 'dir': 'left', 'size': [0, 0]}])
----------------------------------------
('tracked_blobs', [{'trail': [(436, 255), (436, 262)], 'bumper_x': 501, 'first_seen': 1576554319.113, 'last_seen': 1576554319.26, 'speed': [0], 'id': '37f86d42', 'dir'
: 'right', 'size': [0, 0]}, {'trail': [(425, 237), (431, 242), (433, 250)], 'bumper_x': 357, 'first_seen': 1576554319.413, 'last_seen': 1576554319.591, 'speed': [0], '
id': 'e4f5fb7f', 'dir': 'left', 'size': [0, 0]}])
----------------------------------------
('tracked_blobs', [{'trail': [(436, 255), (436, 262)], 'bumper_x': 501, 'first_seen': 1576554319.113, 'last_seen': 1576554319.26, 'speed': [0], 'id': '37f86d42', 'dir'
: 'right', 'size': [0, 0]}, {'trail': [(417, 222), (425, 237), (431, 242), (433, 250)], 'bumper_x': 352, 'first_seen': 1576554319.413, 'last_seen': 1576554319.66, 'spe
ed': [0], 'id': 'e4f5fb7f', 'dir': 'left', 'size': [0, 0]}])
----------------------------------------
('tracked_blobs', [{'trail': [(436, 255), (436, 262)], 'bumper_x': 501, 'first_seen': 1576554319.113, 'last_seen': 1576554319.26, 'speed': [0], 'id': '37f86d42', 'dir'
: 'right', 'size': [0, 0]}, {'trail': [(409, 211), (417, 222), (425, 237), (431, 242), (433, 250)], 'bumper_x': 345, 'first_seen': 1576554319.413, 'last_seen': 1576554
319.729, 'speed': [0], 'id': 'e4f5fb7f', 'dir': 'left', 'size': [0, 0]}, {'trail': [(301, 264)], 'bumper_x': None, 'first_seen': 1576554319.729, 'last_seen': 157655431
9.729, 'speed': [0], 'id': '1da2b309', 'dir': None, 'size': [0, 0]}])
----------------------------------------
('tracked_blobs', [{'trail': [(436, 255), (436, 262)], 'bumper_x': 501, 'first_seen': 1576554319.113, 'last_seen': 1576554319.26, 'speed': [0], 'id': '37f86d42', 'dir'
: 'right', 'size': [0, 0]}, {'trail': [(399, 200), (409, 211), (417, 222), (425, 237), (431, 242), (433, 250)], 'bumper_x': 345, 'first_seen': 1576554319.413, 'last_se
```

Calculated results of speed tracking



VII. CONCLUSION

In this project, we suggest that the computer program be able to determine the precise speed of a moving object. For accurate representation of the moving objects, this technique was combined with a Gaussian mix model. Even with poor visual quality, the combination of the optical stream and the Kalman channel aids in outcome prediction. In our ongoing work, we hope to enhance the DBSCAN division so that it can recognize each component of the collection of cars, as well as employ flexible heaps of pixels to sense vertical advancements' speed.

VIII. REFERENCES

[1] Automatic detection of the direction and speed of moving objects in the video," Sixth International Conference on Current Computing (IC3), 2013, pp. 86–90. O. Smirg, Z. Smekal, M. K. Dutta, and B. Kakani.

[2] J. X. Wang, "Study of vehicle speed detection method in video surveillance," 2016 International Conference on Audio, Language and Image Processing (ICALIP), July 2016, pp. 349–352.

[3] Vehicle speed detection system," 2009 IEEE International Conference on Signal and Image Processing Applications, C. Pornpanomchai and K. Kongkittisan, pp. 135–139, Nov 2009.

[4] A new vehicle detect approach based on gaussian mixture model combined with estimate moment velocity using optical flow," by M. A. Alavianmehr, A. Zahmatkesh, and A. Sodagaran.

[5] Video size comparison for embedded vehicle speed detection traveltime estimation system by utilising raspberry pi," in 2016 International Conference on Robotics, Automation and Sciences (ICORAS), pp. 1-4, Nov 2016. I. Iszaidy, A. Alias, R. Ngadiran, R. B. Ahmad, M. I. Jais, and D. Shuhaizar

[6] Vehicle speed identification utilizing corner detection, K. V. K. Kumar, P. Chandrakant, S. Kumar, and K. J. Kushal, Proceedings of the 2014 Fifth

[7] IEEE Computer Society, 2014, International Conference on Signal and Image Processing, ICSIP '14, Washington, DC, USA, pp. 253–258.

[8] Vehicle speed recognition from a single motion blurred image," Image and Vision Computing, vol. 26, no. 10, pp. 1327–1337, 2008. H.-Y. Lin, K.-J. Li, and C.–H. Chang

[9] Adaptive background mixing models for real-time tracking," by C. Stauffer and W. E. L. Grimson, 1999, in Computer Vision and Pattern Recognition IEEE Computer Society Conference on, volume 2, 1999 IEEE, pp. 246-252.

[10] A. Burton and J. Radford, Critical Essays in the Study of Thought Processes: Thinking in Perspective. 1978's Methuen.

[11] Electronic Spatial Sense for the Blind: Contributions from Perception, Rehabilitation, and Computer Vision, by D. H. Warren and E. R. Strelow, vol. 99. the year 2013; Springer Science & Business Media.

[12] Learning OpenCV: Computer vision with the Open library," by G. Bradski and A. Kaehler, O'Reilly Publishing, Inc., 2008.