Speed Violation Detection System

[1]Yash Jaiswal
Electronics and Communication
VNIT
Nagpur, India
yashjaiswal@students.vnit.ac.in

[2] Varun Ramteke
Electronics and Communication
VNIT
Nagpur, India
varunramteke.vnit@students.vnit.ac.in

[3] Romal Gaikwad
Electronics and Communication
VNIT
Nagpur, India
romalgaikwad@students.vnit.ac.in

[4] Raman Kedar
Electronics and Communication
VNIT
Nagpur, India
rk.tech2401@students.vnit.ac.in

[5] Yash Chandekar
Electronics and Communication
VNIT
Nagpur, India
yashchandekar975@gmail.com

***Abstract*—One of the leading causes of accidents that take place on roadways is excessive speeding. In response to the rising number of accidents each year, our study proposes a solution for speed detection and speed violation detection. We used YOLOv5 and computer vision for this project. It can access the live video feed from the surveillance cameras mounted at traffic signal poles and on highway segments. YOLOv5 is used for vehicle detection and tracking vehicles throughout the view. Euclidean concepts have been used to build a tracker to keep track of vehicles at the backend of the system. The speed is then calculated within the region of interest and comparisons are made with respect to the set speed threshold. Any user can access this system with the help of the GUI that has been provided which integrates the YOLOv5 with the tracker system. This aids in the real-time tracking of speed violations and notifies the officer of the law. We can store screenshots and short clips of the violating vehicle for later use.**

***Index Terms*—YOLOV5;Vehicle Detection; Vehicle Tracking;Speed Detection; GUI; Database System ; .**

I. Introduction

According to data published by the World Health Organization (WHO), India accounts for 6% of global road accidents in 2018, despite the fact that the country's entire car population is only 1% of the global total. In 2018, India accounted for roughly 73 percent of all road-related deaths in South Asia. In addition, according to a report released by the World Health Organization (WHO) in 2017, one of the most likely causes of mortality is a car collision. It's also the ninth most common cause of death and the tenth most common cause of long-term impairment. The number of vehicles on the road grows in lockstep with the world's population. According to latest figures released by the Ministry of Road Transport and Highways (India), there were 4,67,044 total road accidents in India in 2018, with 1,51417 individuals killed and 4,69,418 suffering serious injuries. Surprisingly, 45 percent of total car accidents occurred on roads other than national and state highways. Speeding and red-light violations are two of the most common causes of such collisions. To support and improve traffic monitoring, a single system to track and detect numerous breaches is required..

Almost every major city in the globe now has traffic-monitoring cameras installed. Making advantage of the existing observation camera systems will be a viable option. Traffic cops will save a lot of time and effort by incorporating surveillance cameras into an automated Artificial Intelligence Traffic Violation Detection System. Given proof, a law enforcement officer can notice the offence..

We can link our system with the government database and impose fines online instantly because we can combine such a system with vehicle number plate detection.

Since its birth, computer vision has been used in the Intelligent Transportation System. It is helpful when it comes to traffic violation detecting systems. The live video feed may be utilised to identify and analyse traffic in real time. Due to a lack of data and expensive computations, using this technology in the past was extremely challenging. As a consequence of recent breakthroughs in machine learning and deep learning algorithms, as well as the introduction of GPUs and TPUs (Tensor Processing Units) and a substantial growth in information databases, computer vision has become more efficient and practical.

There are five steps to the system we're showing here. In the first stage, YOLOV5 trained on vehicle datasets is used to detect all new automobiles. The centroid of the bounding box obtained after tracking these vehicles is employed in the object tracking technique, and the identified vehicle is then used for Euclidian tracking. The algorithm monitors the vehicle's location in a specified region. In the third stage, the vehicle's speed and direction are both sensed at the same time. Violations are discovered by following a vehicle's movement and position in a certain region.

II. YOLOV5

YOLO, which stands for "You Only Look Once," is an object recognition approach that divides photos into a grid layout. Each grid cell is responsible for detecting things inside its own boundaries. YOLO is one of the most well-known

object detection algorithm due to its speed and precision. Glenn Jocher introduced YOLOv5 utilising the Pytorch framework shortly after the release of YOLOv4.

YOLOv5 differs from its previous versions in that it is built on PyTorch instead of the original Darknet. The YOLOv5 features a CSP backbone and a PA-NET neck, same as the YOLOv4. Mosaic data augmentation and auto learning bounding box anchors are two of the most noticeable enhancements. As a result, YOLO v5 is said to be significantly quicker and lighter than YOLOv4, with similar accuracy to the YOLOv4 test.

When compared to other object identification algorithms, YOLO's main characteristic is that it only looks at the image once before calculating each item's class probability and bounding box. This is why it is so quick. As illustrated in Fig 1, it first divides the photographs into SxS grid cells. Each cell can have up to B border boxes. The entire bounding box will be SxSxB. If C classes are present, with each box will be having (5+C) characteristics. The confidence score that this box contains is one characteristic, while the other four are box coordinates. A certain level of assurance is required to ascertain the object detection, for which the confidence score is set. If we set the threshold for our system to 0. and the confidence score of the detection is more than 0.5, it will just show the box with the class name. In a single network pass, all of these activities may be done. There is an issue when several detections of the same item. To get around this, a method known as non-max suppression is applied. During this method, the box with the greatest confidence score is picked first. Each cell's boxes will be deleted if they overlap with the previously chosen box by more than a threshold value B bounding box. This function guarantees that each object has just one box..

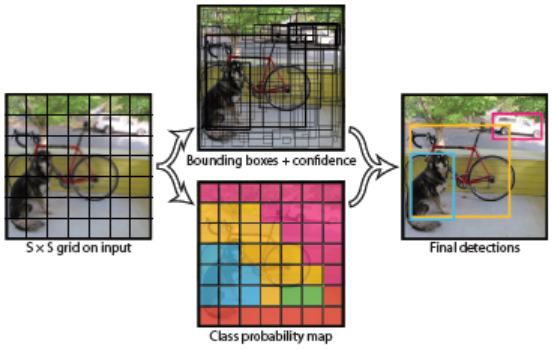

Fig. 1. YOLO separates the image into a SxS grid

YOLOv5 was trained using the COCO dataset, which has 80 unique classes. For our system, we solely used different sorts of vehicles, such as buses, trucks, and cars. In earlier versions of YOLO, little items generated good results, but YOLOv5 gives exceptional results in this circumstance.

III. PROPOSED METHOD

A. Vehicle Detection

The first crucial system, as shown in the flow chart in Figure 6, is vehicle detection. The COCO dataset, which includes automobile and bus classes, was used to train YoloV5. The detector receives the video feed from the security camera and generates a tensor list of coordinates with the height and width of the recognised vehicle..

These are then turned to a numerical list and handed on to be processed further. The working of YOLOv5 is seen in Fig. 4 on an example image.

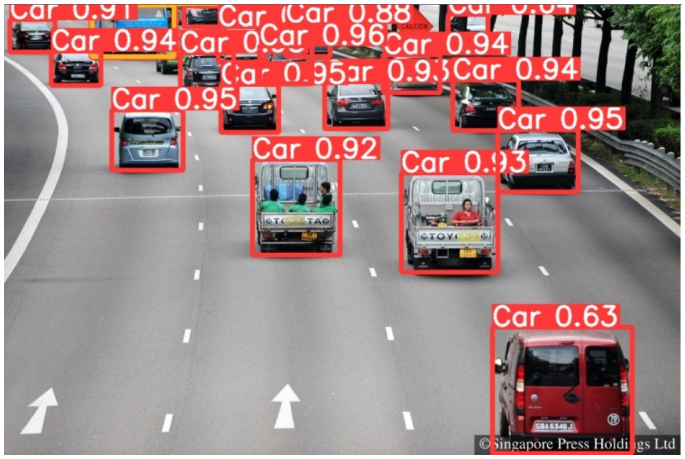

Fig. 2. Tested YOLOv5 on sample image.

B. Vehicle tracking

Work of the Euclidian tracker is depicted in Fig.3. This method takes the centroid of each vehicle into account. The technique calculates the distance between previous centroids first in each detection. If the lowest distance is less than the threshold, the new detection is just the new position of the corresponding centroid with the shortest distance. If the distance is higher than the threshold, the algorithm determines that a new item has been spotted and adds it to the list of detected objects. If any previous item isn't spotted in the current frame, we may assume it is out of the frame and delete the centroid. The supporting statistics for the algorithm's operation are shown in Figures 4 and 5. The distance between any two centroid, say C_1 with coordinates (x_i, y_i) and C_2 with coordinates (x_j y_j), may be calculated using the following equation:

$$d = \sqrt{(x_i - x_j)^2 + (y_i - y - j)^2}$$

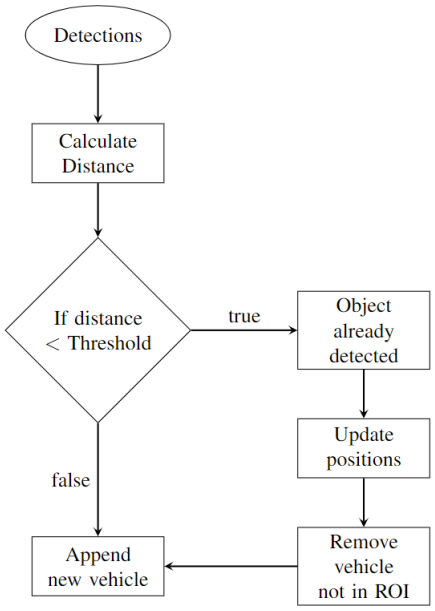

Fig. 3. Flowchart of Euclidian Tracking Algorithm

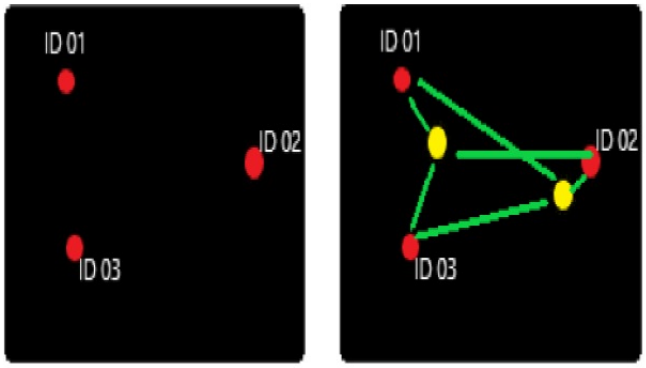

Fig. 4. (a) frame1 (b) frame2

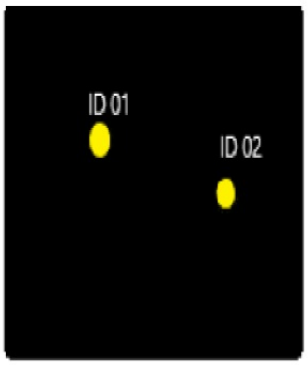

Fig. 5. Updated frame

C. Speed Violation detection

Following the collecting and tracking of vehicle data, the project's next phase was to use this information to identify vehicle speed. Some fundamental physics formulas can be used to do this. Obtaining the quantities required to answer these equations, on the other hand, was a tough task in and of itself. Knowing how far these cars travelled in the frame and how long it took them to do so was part of the job. This operation required some calibration as well.

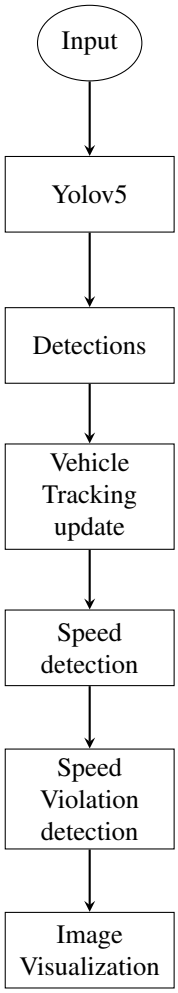

Fig. 6. Flowchart of the Speed Violation Detection system

The recording equipment that would monitor the situation played a role in determining the physical quantities. Before installing this system for a certain highway or traffic square, the installer must calibrate the location of two reference lines seen in the camera video in relation to the real world. The region of interest would be contained inside these reference lines, and the majority of the computations for the vehicles' various properties would be acquired from there. The distance between these two reference lines had to be computed once the positions of these two reference lines were determined. As a result, significant planning was necessary. When the

reference lines on the scene were visually matched with the reference lines drawn on the camera clip, the distance calibration was accomplished with a high degree of precision.

The time it took for the automobiles to traverse that distance was needed now that the distance between the point of interest and the vehicle had been calculated. This was accomplished by using the video footage's frame rate. Because there may be a significant latency due to background calculations, we recorded the number of frames that passed between the automobiles passing these two reference lines. The system begins calculating the number of frames elapsed as soon as the automobile passes the first reference line. The frame count is captured once the vehicle has over the second reference line, and the time passed is computed using the video frame rate. Now that distance travelled by these automobiles was at hand along with the time it took them to travel the same, we could easily calculate the speed of these vehicles using the formula mentioned below.

$$speed = \frac{distance \times FPS}{framecount}$$

Once the speed for individual vehicles is calculated, it was coded into the tracker system to compare these speeds against the set threshold by the user. The provision to set this threshold is provided with the GUI. Once a violation is detected, the vehicle's ID was recorded into the system using a database management system.

D. Graphical User Interface

The entire system's graphical user interface is written in Python using the Tkinter framework. The graphical user interface is incredibly user-friendly. Take a look at Fig. 7 for a view of the primary window. Two distinct areas make up the main frame. The video frame is the first, and the setting region is the second.

The Video section includes a video frame area as well as pause and play buttons. The pause and play buttons will stop and start the system in a real-time scenario. In the case of video, the buttons will respond to their names. In the setting area, there are three setting buttons that go to their respective dialogue boxes. Camera settings, parameter settings, database settings, these are the only settings the user is allowed to change.

Fig. 7. Result

The first and second Lines of Interest (LOI) are represented by two black lines in the result frame. These are the reference lines used to determine the vehicle's speed. The presence of a vehicle inside the box is shown by a green box, and the colour of the box denotes the detection of a violation. If the box is green, no infraction has been detected; otherwise, it is red, indicating that a speed infringement has been detected. The temporary id of the vehicle is indicated on the top left of the box, while the speed of the vehicle after it has crossed the LOI2 is mentioned on the right side.

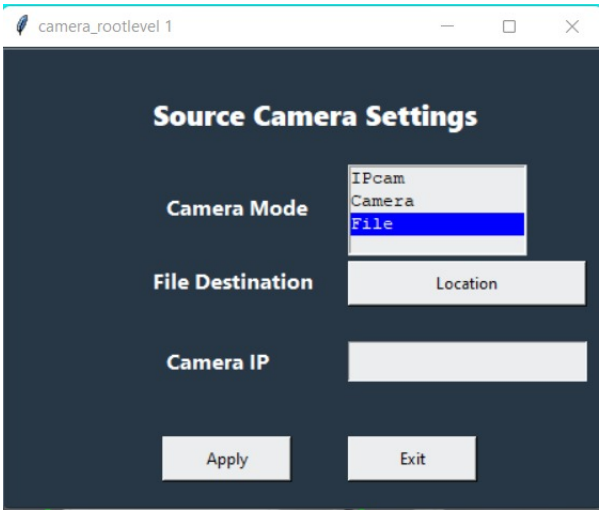

Fig. 8. Camera Settings

We gave three ways for the user to use the system for violation detection in the camera settings. The first is an IP camera that is connected to the local network, the second is a camera that is connected to the system, and the third is a source file that allows the user to specify the video location and test the violation detection system on it.

Fig. 9. Parameters Setting

There are five parameters that the user can control in the parameters setting. The first is the LOI1 position, which may be altered using the trackbar. The same can be said for the LOI2 position. In the Euclidean tracker, the pixel threshold is represented by a slider called "Fixed pixel dist." In the real world, the other entry boxes are used to specify the distance between the two LOIs and the speed violation limit. Finally, the database configuration includes an entry box for entering Google drive or spreadsheet API tokens.

IV. Results

The vehicle dataset was used to train YOLOv5, with promising results seen in Fig. 7. Vehicle detection with YOLOv5 yields results that are nearly 100 percent accurate. The method works according to the flowchart in Figure 6. The vehicle tracking algorithm, on the other hand, is based on the Euclidian algorithm, as shown in Fig. 3,4 and 5. The main task of speed detection is based on simple equations of physics concerning distance and time. The speed is then compared against the threshold set by the user. All of this is packaged into a GUI. Any violations recorded are then sent over to be stored in google sheets. We put the entire system to the test by testing a speed infringement in a video footage. Figures 10 shows the final result.

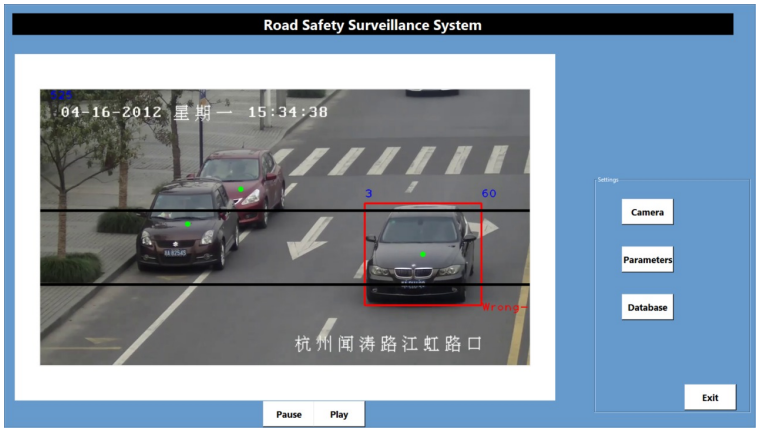

Fig. 10. Result with Speed Violation Detected

The test is run on an Intel Core i7 processor with 8GB of RAM, resulting in a frame rate of approximately 12 frames per second. It demonstrates that using a GPU to run this method produces promising real-time results.

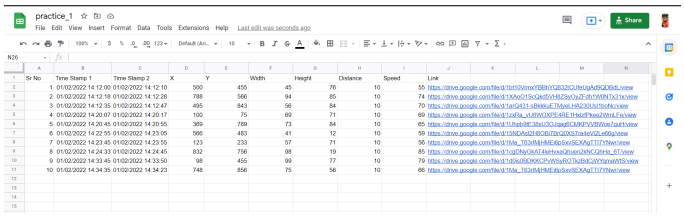

Fig. 11. live data feeding using google sheets api

In figure 11, timestamps of vehicle passing, vehicle's speed and the snapshot of vehicle is shown. Live data is fed using google sheet api and the image was uploaded to drive using google drive api.

V. Conclusion

This paper demonstrates a technique for recognising and monitoring moving cars,speed estimates and data input through Google Sheets and Google Drive API. The technique is unique in that it identifies the Region of Interest and monitors adaptive changes in the vehicle's position for rule breaches. The suggested method detects vehicles with an average accuracy of 99.17 percent and violations with an average accuracy of 97.5 percent. To restrict the extent of any false positive detection and to focus on only one side of the road, the suggested technique supports manual cropping. In order to detect infractions, the recommended technique for recognising and monitoring moving vehicles extracts information from the vehicle's location, height, and breadth. There are fewer calculations and memory requirements as a result. The recommended technique maintains vehicle characteristics, predicted automobile speeds, and speed limit infractions in the database.

The YOLO V5 object identification technology from Ultralytics is one of the quickest on the market. YOLO v5 beats its

predecessors in terms of accuracy and latency. The proposed method is straightforward to include into an existing traffic management system. With this method, the technology will be integrated with a number plate detection system in the future.

REFERENCES

[1] Arash Gholami Rad, Abbas Dehghani and Mohamed Rehan Karim, "Vehicle speed detection in video image sequences using CVS method"

[2] Shilpa Jahagirdar, Sanjay Koli, "Automatic Accident Detection Techniques using CCTV Surveillance Videos: Methods, Data sets and Learning Strategies", International Journal of Engineering and Advanced Technology (IJEAT), ISSN: 2249 – 8958, Volume-9 Issue-3, February, 2020.

[3] Earnest Paul Ijjina, Dhananjai Chand, Savyasachi Gupta, Goutham K, "Computer Vision-based Accident Detection in Traffic Surveillance", arXiv:1911.10037v1 [cs.CV] 22 Nov 2019.

[4] Pranith Kumar Thadagoppula, Vikas Upadhyaya, Speed Detection using Image Processing, 2016 International Conference on Computer, Control, Informatics and its Applications.

[5] Tarun Kumar, Tarun Kumar, "An Efficient Approach for Detection and Speed Estimation of Moving Vehicles", Twelfth International Multi-Conference on Information Processing-2016 (IMCIP-2016).

[6] Zillur Rahman, Amit Mazumder Ami and Muhammad Ahsan Ullah, "A Real-Time Wrong-Way Vehicle Detection Based on YOLO and Centroid Tracking", 2020 IEEE Region 10 Symposium (TENSYMP), 5-7 June 2020, Dhaka, Bangladesh.

[7] Dr. S. Raj Anand, Dr. Naveen Kilari, Dr. D. Udaya Suriya Raj Kumar, "TRAFFIC SIGNAL VIOLATION DETECTION USING ARTIFICIAL INTELLIGENCE AND DEEP LEARNING", International Journal of Advanced Research in Engineering and Technology (IJARET).Volume 12, Issue 2, February 2021, pp. 207-217.