Intelligence Vehicle Surveillance and Speed Enforcement System

Nandade Pankaj Kumar

Department of IT

MLR Institute of Technology Hyderabad, India.

[nandadeakash5704@gmail.com](mailto:nandadeakash5704@gmail.com)

Thogarla Madhushalini

Department of IT

MLR Institute of Technology Hyderabad, India.

[madhushalini463@gmail.com](mailto:madhushalini463@gmail.com)

Mashetty Venkatesh

Department of IT

MLR Institute of Technology

Hyderabad, India.

[venkymashetty@gmail.com](mailto:venkymashetty@gmail.com)

# ABSTRACT

The main reason for many road accidents in modern times is speeding and negligent driving. The massive increase in traffic on the roads has created a huge demand for control. To address this issue, this paper proposes a vehicle detection and counting system, Classification of vehicles, speed detection using You Only Look Once (YOLO-V8), OpenCV based DeepSORT model for real time vehicle detection and tracking from video sequences and License [plate recognition using YOLOV8 and EasyOCR. Deep learning based Simple Real time Tracker (Deep SORT) algorithm is added, which will track actual presence of vehicles from video frame predicted by YOLO-V8 so the false prediction perform by YOLOV8can be avoid by using DeepSort algorithm. The video will be converted into multiple frames and give as input to YOLO-V8 for vehicle detection, counting and classification of vehicles. It eliminates the need for manual checks by the police to identify speeding vehicles. Intelligence Vehicle Surveillance and Speed Enforcement System helps manage traffic, allowing us to identify peak traffic times and take necessary precautions to avoid long traffic jams.

# CCS CONCEPTS

* Traffic control and Management Systems

# KEYWORDS

YOLOV8, DeepSort, OpenCV, EasyOCR, LPR.

# 1 INTRODUCTION

Nowadays, with the continuous increase of vehicles on the road, traffic management authority requires better traffic surveillance system. With more vehicles, the number of accidents on the road rises up each year. Speed is now the single cause of road accidents. Enforcing speed limit is one of the ways to eliminate speed related accidents. Traffic surveillance systems for speed measurement play an important role in enforcing speed limits.

Vehicle detection and tracking is a common problem with multiple use cases. Government authorities and private establishment might want to understand the traffic flowing through a place to better develop its infrastructure for the ease and convenience of everyone. A road widening project, timing the traffic signals and construction of parking spaces are a few examples where analyzing the traffic is integral to the project. Traditionally, identification and tracking has been carried out manually. A person will stand at a point and note the count of the vehicles and their types.

Recently, sensors have been put into use, but they only solve the counting problem. Those systems are divided in intrusive and non-intrusive sensors. Intrusive sensors are usually based on inductive loop detectors. Although these sensors are used widely, they have complex installation and high maintenance, promotes asphalt deterioration and also can be damaged by wear and tear.

Non-intrusive sensors, which include laser meters and doppler radars, avoid these problems, but are usually more expensive and require frequent maintenance. A part from this, due to high cost of equipment and less accuracy, it is losing its popularity. Sensors will not be able to detect the type of vehicle, their License plate and speed of the vehicle.

A fundamental source of the economic growth of any nation depends on well-planned and resilient transportation systems based on spatial information. Regardless, most cities around the world are still facing a rampant increase in traffic volume and complications in traffic management, resulting in poor quality of life in modern cities. However, recent advancements in internet bandwidth, artificial intelligence, and sensing technologies have minimized these difficulties by collaboratively bringing forward location intelligence for public safety. Automation in location intelligence in road environments using sensing technologies allow authorities to achieve resilience in road safety, controlled commutes, and assessments of road conditions.

# BRIEF INTRODUCTION OF INTELLIGENCE VEHICLE SURVEILLANCE AND SPEED ENFORCEMENT SYSTEM.

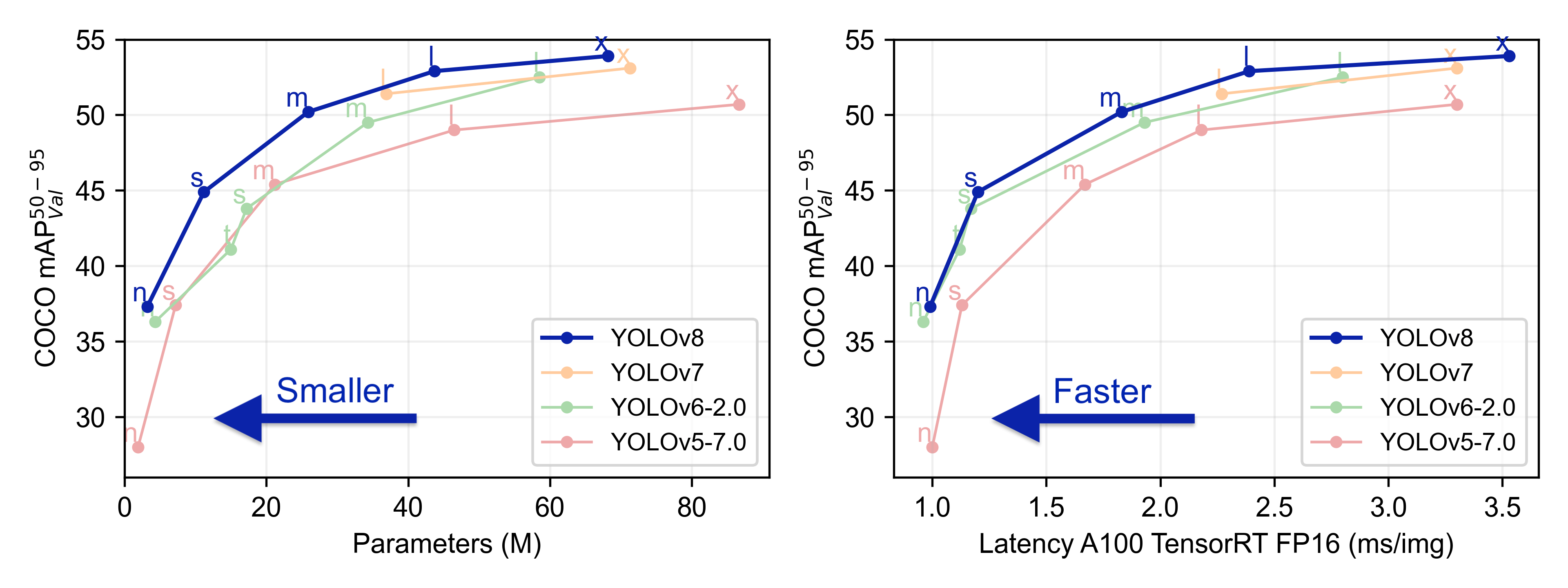
[1] The Intelligent Vehicle Surveillance and Speed Enforcement

CIIS’19, November, 2019, Bangkok, Thailand.

System represents a significant step forward in modern traffic management and road safety. At its core, the system employs state-of-the-art techniques for vehicle detection, counting, classification, speed detection, and license plate recognition. By integrating You Only Look Once (YOLO-V8) for vehicle detection and classification, and OpenCV-based DeepSORT model for real-time vehicle tracking, the system achieves unparalleled accuracy and efficiency in monitoring traffic flow. Additionally, the incorporation of YOLOV8 and EasyOCR for license plate recognition ensures robust enforcement capabilities.[2] The system comprises a camera that captures the traffic scene, and a computer processes the video stream using the YOLO v8 object detection model. The YOLO v8 model is fine-tuned on a large dataset of vehicle images to accurately detect, classify, count, and estimate the speed of vehicles in the scene. The Proposed system uses a single-camera setup to capture the traffic scene, which makes it cost-effective. It is designed to be efficient and capable of processing the video stream in real life, which makes it suitable for real-time applications.

# EVALUATION OF Intelligence Vehicle Surveillance and Speed Enforcement System.

# The methodology for this research involves the use of the YOLOv8 algorithm and DeepSort algorithm for Intelligence vehicle Surveillance System for Traffic Control System. The process can be divided into several steps:

 Comparing different YOLO version, Image from Ultralytics YOLOv8 repo

# Data Collection and Pre-processing: The first step involves collecting a dataset of images and videos that contain various types of vehicles under different driving conditions (DATASET NAME – Cars Detection FROM KAGGLE.COM pre-processed to ensure they are suitable for input into the YOLOv8 model. This includes resizing the images and videos to the required dimensions, normalizing the pixel values, and augmenting the data to increase its size and diversity.

# Model Training: The YOLOv8 model is trained on the pre-processed dataset. The model is trained to detect and classify different types of vehicles. The training process involves feeding the images and videos into the model, which then predicts bounding boxes and class probabilities for each object in the images and videos. The model's predictions are compared to the ground truth labels, and the model's parameters are updated to minimize the difference between the predictions and the ground truth labels. This process is repeated for several epochs until the model's performance on the validation set stops improving. Object Tracking: After the model is trained, it is used to detect and classify vehicles in real-time. The model predicts bounding boxes and class probabilities for each object in the images and videos, and these predictions are used to track the vehicles. The DeepSORT (Simple Online and Realtime Tracking) algorithm is used to track the vehicles across multiple frames. DeepSORT uses the features extracted by the YOLOv8 model to associate detections in consecutive frames, allowing it to track vehicles across time. A real time image was provided to the model to detect the object.

# Evaluation: The performance of the YOLOv8 model and the DeepSORT algorithm is evaluated using various metrics, such as precision, recall, and the mean average precision (mAP). These metrics provide a quantitative measure of the model's ability to detect and classify vehicles accurately and efficiently. The model's performance is also tested under different driving conditions to ensure its robustness.

# Optimization: Based on the evaluation results, the model's parameters are fine-tuned to improve its performance. This involves adjusting the model's hyperparameters, such as the learning rate and the number of epochs, and using techniques such as early stopping and dropout to prevent overfitting. The model is also tested on different hardware platforms to ensure its compatibility and performance. Throughout this process, a balance between the speed of detection and the accuracy of the model is maintained, ensuring that the YOLOv8 model can detect and classify vehicles in real-time while maintaining a high level of accuracy.

# EXISTING PROBLEMS

The current traffic surveillance systems primarily rely on two types of sensors for speed determination: intrusive and non-intrusive sensors.

Intrusive Sensors: These sensors are typically based on inductive loop detectors embedded in the road surface. They work by detecting changes in the magnetic field when a vehicle passes over the loop. While they have been widely used, intrusive sensors suffer from several limitations. They require complex installation procedures, leading to higher setup costs. Additionally, they demand regular maintenance, which can be time-consuming and expensive. Furthermore, they promote asphalt deterioration over time, affecting the road's overall lifespan. Moreover, intrusive sensors need a direct line of sight between the vehicle and the equipment, making them less flexible for various road layouts.

Non-intrusive Sensors: This category includes technologies like laser meters and doppler radars. Non-intrusive sensors offer some advantages over their intrusive counterparts. They do not require road surface installations, mitigating the risk of asphalt damage. However, non-intrusive sensors tend to be more expensive, and they often demand frequent maintenance to ensure accuracy and reliability. This higher cost and maintenance burden have limited their widespread adoption, making them less favorable for budget-conscious traffic management authorities.

* High Cost: Both intrusive and non-intrusive sensors can be expensive to install and maintain. The initial setup costs for intrusive sensors involve complex installations in the road surface, while non-intrusive sensors, such as laser meters and doppler radars, often come with higher price tags. Provide him with the required salary. Also, one cannot completely rely on that person for monitoring purpose.
* Line of Sight Requirement: Intrusive sensors often require a direct line of sight between the sensor and the vehicle, limiting their flexibility for installation in various road layouts.
* Lack of Scalability: Expanding the existing system to cover a larger area or more lanes can be challenging and costly, especially for non-intrusive sensors.
* Expensive: Sensors requires more maintenance.

# IMPLEMENTATION

**Installing Python:**

1.To download and install Python visit the official website of Python [**https://www.python.org/downloads/**](https://www.python.org/downloads/) and choose your version.



FIG-5.1 PYTHON INSTALLATION

2.Once the download is complete, run the exe for install Python. Now click on Install Now.

3.You can see Python installing at this point.

4.When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

**Installing PyCharm:**

1.To download PyCharm visit the website [**https://www.jetbrains.com/pycharm/download/**](https://www.jetbrains.com/pycharm/download/) and Click the "DOWNLOAD" link under the Community Section.



FIG-5.1.1 PYCHARM DOWNLOAD

2.Once the download is complete, run the exe for install PyCharm. The setup wizard should have started. Click “Next”.

3.On the next screen, Change the installation path if required. Click “Next”.

4.On the next screen, you can create a desktop shortcut if you want and click on “Next”.

5.Choose the start menu folder. Keep selected JetBrains and click on “Install”.

6.Wait for the installation to finish.

7.Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the “Run PyCharm Community Edition” box first and click “Finish”.

8.After you click on "Finish," the Following screen will appear.

9.You need to install some packages to execute your project in a proper way.

10. Open the command prompt/ anaconda prompt or terminal as administrator.

11.The prompt will get open, with specified path, type “pip install package name” which you want to install (like numpy, pandas, seaborn, scikit-learn, matplotlib. pyplot).

**1.System**

1. **Capture Images:**

Capture the images using YOLO V8 and open CV libraries.

1. **Train Images:**

The data can’t be trained by the unknown users. The authentication process is implemented here where the training accessibility will be held with only the owner of the application.

We will be using YOLO v8 and Deep Sort Model to train the model.

1. **Download:**

After video processed the video download option would available.

**2.User:**

1. **Enter User Name:**

The user needs to enter the User Name.

1. **Enter Password:**

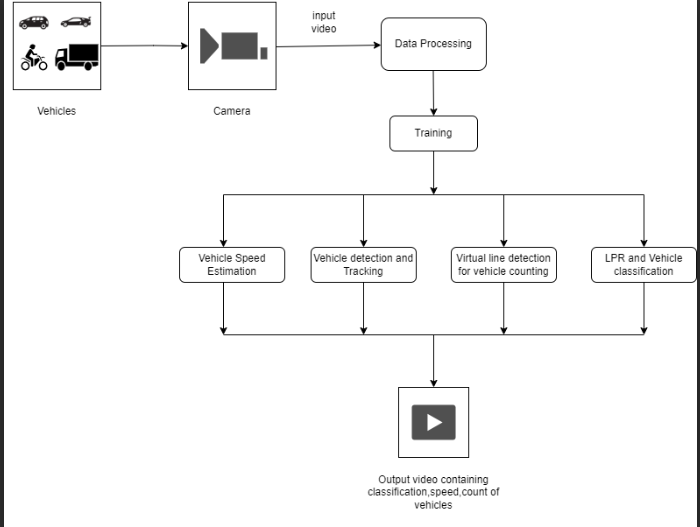
The user needs to enter the Password.

1. **Upload Video:**

User needs to Upload the video.

1. **View Results:**

The known user can download the video once the video processing is done and the result will be saved at desktop. Ex: pip install numpy.



**Figure.3: The Architecture diagram of Intelligence vehicle surveillance and speed Enforcement System.**

The above figure show the overall architecture of the of Intelligence vehicle surveillance and Speed Enforcement System.

At its core, high-quality video footage captured by strategically placed cameras along roadways serves as the primary input. This raw data undergoes a preprocessing stage, where it is cleaned, standardized, and prepared for further analysis. Advanced algorithms are then employed to meticulously detect, count, and track vehicles within the video stream, leveraging sophisticated techniques for license plate recognition. Crucially, utilizing image timestamps and vehicle positions enables the system to accurately calculate vehicle speeds, essential for effective speed enforcement measures. The processed data, including the information such as user credentials for authentication, is stored for future reference and analysis. The alignment of these processes leads to the creation of an output video.

**SOLUTION TO THE PROBLEM**

## Vehicle Detection using YOLOV8

YOLO (You Only Look Once) V8 is a popular real-time object detection algorithm known for its efficiency and accuracy. YOLOv8 is a cutting-edge object detection model that revolutionizes computer vision technology by enabling **real-time object detection**. The **YOLOv8 architecture** follows a single-stage approach and incorporates anchor boxes and non-maximum suppression techniques for accurate object detection. The job of the object detector here is to take the in the input images (frames of the video, in our case), and give us the object classifications and their bounding boxes It takes in the input image, divides it into a grid of cells. Each cell in the grid is responsible for predicting a potential object box and class as shown in the rightwards and downwards flows respectively. These predictions are then combined into proposed class bounding boxes, which are then filtered using non-maximum suppression (NMS) and threshold detection to come to the final.

[1] Data Preparation: We collected a diverse dataset of vehicle images representing a wide range of scenarios typically encountered by autonomous vehicles, including urban streets, highways, and varying weather conditions. The dataset was meticulously annotated with bounding boxes to specify the location and class of each vehicle, providing ground truth for model training.

[2] Training and Fine-Tuning: We employed a two-phase training approach. First, we initialized the YOLOv8 model with pre-trained weights from the training dataset, followed by fine-tuning on our vehicle detection dataset. Data augmentation techniques, including random rotations, flips, and color perturbations, were applied to enhance model robustness and generalization.

[3] Model Evaluation Performance Metrics: To assess the model's performance, we employed standard object detection metrics, including precision. Precision measures the accuracy of detected vehicles, recall quantifies how many actual vehicles are successfully identified, and F1-score balances both precision and recall. map provides a comprehensive assessment of the model's ability to rank and localize vehicle.

[4] Benchmarking and Comparative Analysis: We compared the performance of YOLOv8 with other state-of-the-art object detection models, including YOLOv5, Faster R-CNN under similar evaluation conditions.

YOLOv8 consistently outperformed the competing models in terms of both accuracy and speed, making it a strong candidate for vehicle detection in autonomous vehicles.

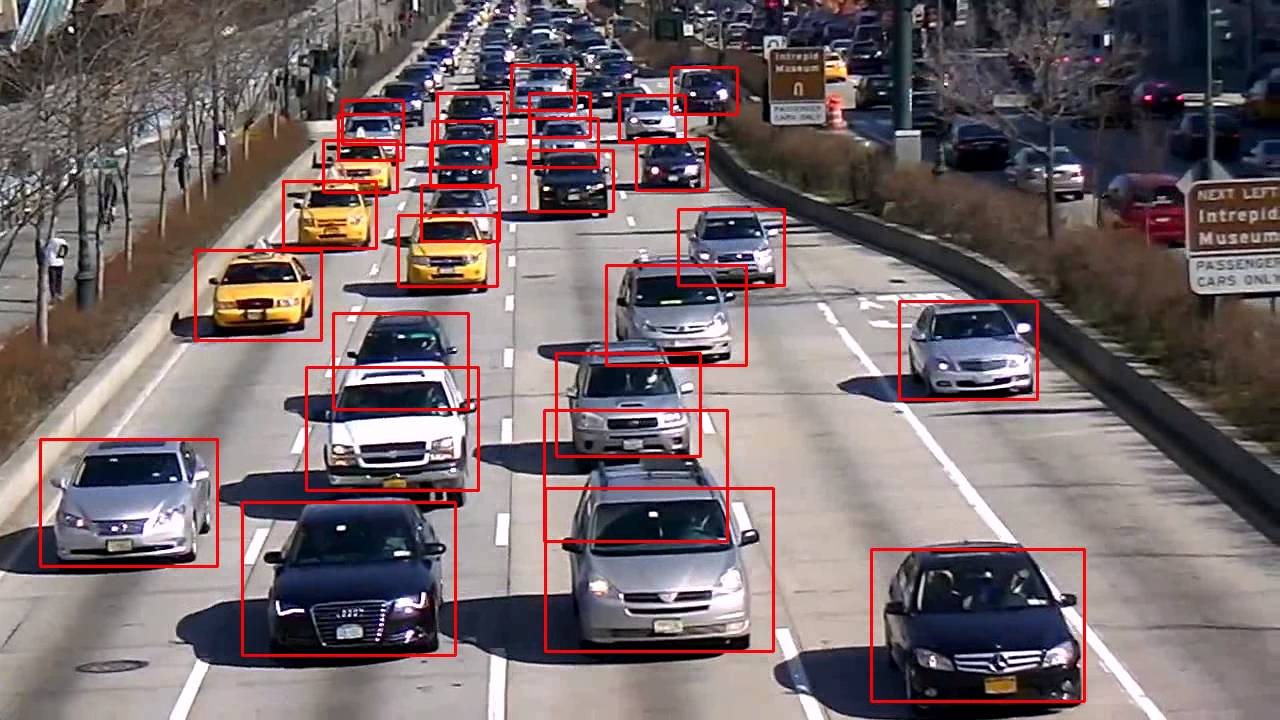


Fig 1: Vehicle detection

## B. Vehicle Tracking using Deepsort

DeepSORT which is an improved version of SORT is one of the most popular state-of-the-art objects tracking frameworks today. While YOLOv8 provides precise object detection results, the Deep SORT algorithm is a separate component that needs integration with YOLOv8 to perform tracking. The algorithm takes the detection outputs from the previous stage and run tracking for each detected object. In tracking by detection scheme, the accuracy of tracking is based on the quality of detection results. The Kalman filter an important role in deep sort.

DeepSORT builds off of a Kalman filter - we plug in the bounding boxes from the object detector at each frame into the filter - the Mahalanobis distance between Kalman predictions and new frame data is used as input into their matching/assignment algorithm for new incoming frames - one approach can be to run this using the Hungarian algorithm. However, DeepSORT builds on just using the Mahalanobis distance by also adding an appearance descriptor for each bounding box, and keeps track of recent appearance descriptors for each track. The aforementioned Mahalanobis using the Kalman predictions as well as a cosine distance metric using the feature descriptors are both fed into their matching cascade algorithm that finally assigns new detections to corresponding tracks - this flow is visualized in Figure 2. It identifies noise in detecting and uses previous states to predict the closed frame surrounding the object best suited. Each time it detects an object it creates a track containing all the necessary information of that object it also tracks and deletes track with detection time exceeds a given threshold due to objects are out of frame. In addition to eliminate duplicates they set a minimum threshold value for detection in the first frame the next problem lies in association between new objects and new predictions from the Kalman filter.

In the tracking procedure, each vehicle is assigned a unique ID to accurately detect its position and derive its speed based on time. Unique IDs are assigned to tracked objects, maintaining their states over time.

Track verification filters out false-positive detections, enhancing tracking system accuracy and reliability. Track termination rules define when objects leave the scene or are no longer detectable. Finally, tracking results are visualized by drawing bounding boxes and IDs on frames or outputting object information like position, velocity, and class labels.

A deque is utilized in implementing trackers like Deep SORT to store the history of object detections or track states. It enables maintaining a sliding window of recent detections or states, facilitating tracking based on that information. A deque can also support temporal analysis of object detections by storing detections from consecutive frames. This enables the analysis of temporal patterns, velocities, or movements of objects over time. Deques provide an efficient way to store and manipulate data based on specific task requirements. However, it's essential to note that the use of a deque depends on the particular implementation and use case. Deques are not inherently part of the YOLOv8 architecture but are employed alongside Deep SORT to enhance tracking and analysis processes.

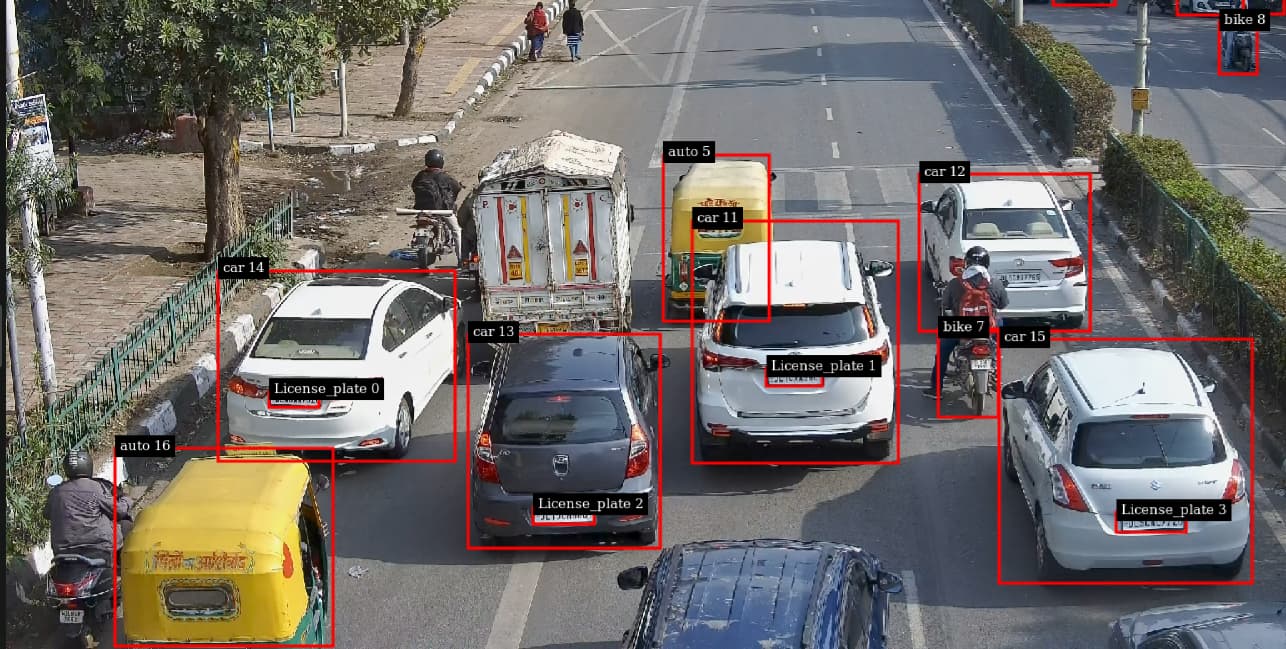


Fig2: Object assigned with a unique ID

*C. Vehicle Counting Using (YOLOV8 +DeepSort)*

Counting vehicles using YOLOv8 and DeepSORT involves a sophisticated process that integrates object detection and object tracking techniques to accurately tally the number of vehicles in a given scene. First, YOLOv8, a state-of-the-art object detection model, is employed to detect and localize vehicles within video frames. YOLOv8 is capable of identifying vehicles and drawing bounding boxes around them, providing precise spatial information about their locations in the scene. Once the vehicles are detected, DeepSORT (Deep Simple Online and Realtime Tracking) is utilized for object tracking. DeepSORT assigns unique IDs to each detected vehicle and tracks their movements across frames, effectively maintaining the identity of each vehicle as it navigates through the scene. This tracking process enables the system to understand the flow of individual vehicles over time. To count the vehicles, the system analyzes the tracking data produced by DeepSORT. By monitoring the entry and exit points of vehicles within the scene and associating these movements with the unique IDs assigned by DeepSORT, the system can accurately count the number of vehicles passing through specific areas or regions of interest. The combination of YOLOv8 for accurate vehicle detection and DeepSORT for robust object tracking allows for precise and reliable vehicle counting. This approach is valuable for applications such as traffic flow analysis, congestion monitoring, and urban planning, providing actionable insights for traffic management and infrastructure development.

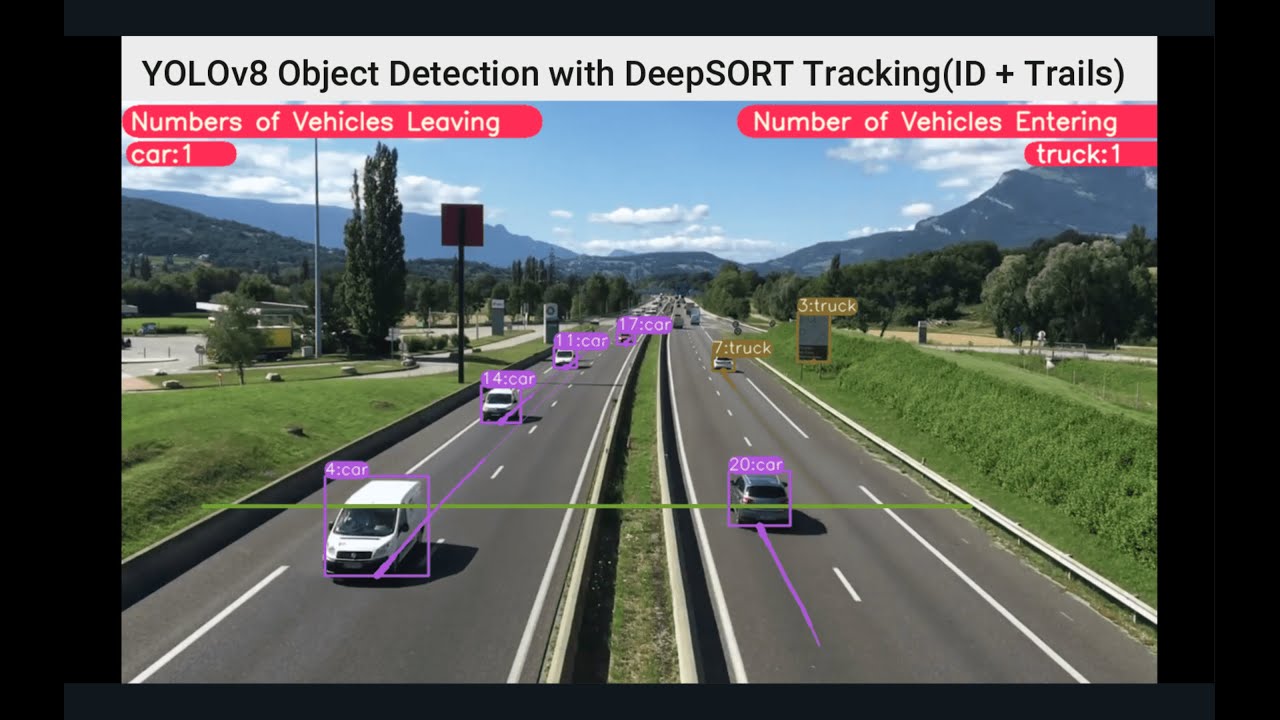


Fig3: Counting vehicles

*D. Vehicle Speed Detection using YOLOV8 and OpenCV*

Vehicle speed estimation using YOLOv8 and OpenCV integrates the YOLOv8 object detection model with OpenCV's video processing capabilities to estimate the speeds of vehicles in a traffic surveillance or monitoring system. YOLOv8 is utilized to detect and localize vehicles within video frames, providing bounding boxes around the vehicles of interest. OpenCV is then employed to track the detected vehicles across consecutive frames, allowing for the calculation of their displacements.

By applying the Euclidean distance formula to the tracked positions of vehicles, the distance traveled by each vehicle between frames is computed. Leveraging a predefined pixels-per-meter conversion factor, this distance is converted into real-world units. Additionally, the frame rate of the video stream is taken into account to scale the speed estimation from pixels per frame to a real-world speed metric, typically kilometers per hour.

The integration of YOLOv8 and OpenCV enables the real-time estimation of vehicle speeds, making it suitable for applications such as traffic flow analysis, road safety monitoring, and intelligent transportation systems. This approach provides a robust and efficient solution for extracting valuable insights from video data, contributing to enhanced traffic management and safety measures.

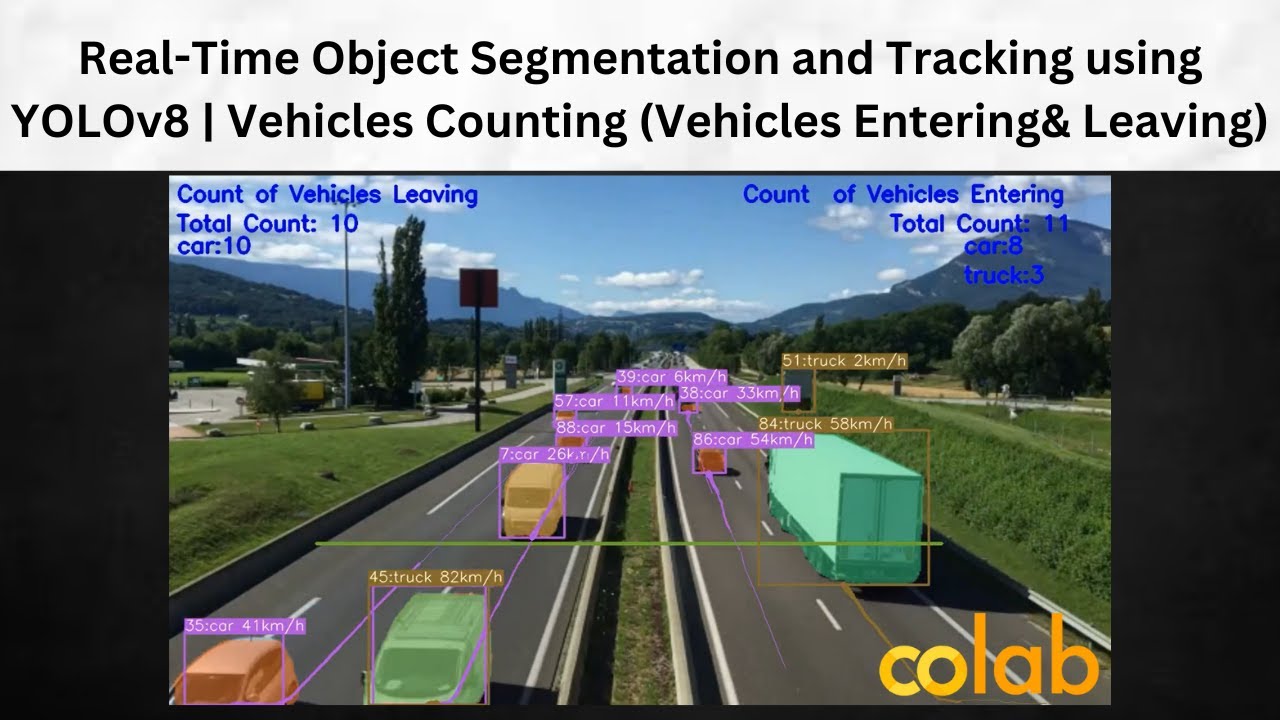


Fig4: Vehicle Speed is Detected

*E. Vehicle classification:*

TheYOLOv8 vehicle classification procedure involves several key steps. First, a dataset of vehicle images must be collected and annotated with class labels to represent a diverse range of vehicle types and variations. The YOLOv8 architecture, specifically designed for object detection and classification tasks, is then selected. The model is initialized with pre-trained weights, which may have been trained on a large-scale dataset like COCO or ImageNet, to help the model learn features useful for vehicle classification. The process of classifying objects using a new weight file by training images and creating labels is explained. The training procedure for YOLO typically involves dataset preparation, selecting the appropriate YOLO version and network architecture, initializing with pre-trained models, defining the loss function, splitting the data into training and validation sets, and performing the training process using optimization algorithms like SGD or Adam. The trained model is then evaluated on the validation set to assess its performance using metrics such as accuracy, precision, recall, and F1 score. Finally, the model is tested on the test set to obtain a final assessment of its performance. Following these steps, the YOLOv8 model can be effectively trained for vehicle classification tasks.



Fig 5: Vehicles are Classified into Different types

*F. License Plate Recognition using YOLOV8 and EasyOCR*

An License Plate Recognition system consists of a digital video recorder and different systems for video analysis. The steps are as follows.

• Photo input and image capture: First, the camera license certificate camera captures a photo or with one or more licenses.

• Scan and collect licenses: Scan licenses on images using machine learning and computer vision. Different methods differ in material requirements, including complexity, speed, and accuracy. One of the best ways is to first use the product search to find the vehicles and then find the license on the box with this border . When the license is detected it is clipped and normalized (sharpened, warped and enhanced).

• Remove plate and read: Then use OCR software for plate area, return plate with number of letters read. OCR software can be optimized for different characters so that the same LPR can be used indifferent countries. The output of an LPR system is usually a license plate number along with a regional or national identifier

Plate Localization: Plate Localization is the step of connecting the plate on the big stage. The location of the license is confirmed, and the output is just a thumbnail containing the license. The database contains 1000 license images distributed during training and testing. For the training, we took 800 images and labeled them with tags, similar to the process done in vehicle detection. After the diagnosis is complete. All detected licenses are stored in separate files to identify the character. We need to use different product search algorithms to determine that the license plate is a product and to recognize it. There are two different points in the picture, scatter and area.

Classification is nothing more than classifying an object such as a car, bicycle, or person, and location is where an object is represented in a picture by drawing a bounding box on it. Object detection is integration of classification and localization of objects in an image. A useful GPU-centric algorithm we recommend for detecting regions of images is the YOLO algorithm (You Only Look Once)

• Character Segmentation: Convert license plate image to grayscale. Then, a two-dimensional filter is applied to the grayscale image. The double-sided filter is a non-linear, noise reduction and smoothing filter that preserves edges. Each pixel changes with the weighted value of neighboring pixels.

The extracted segmented characters are sent as input for character recognition.

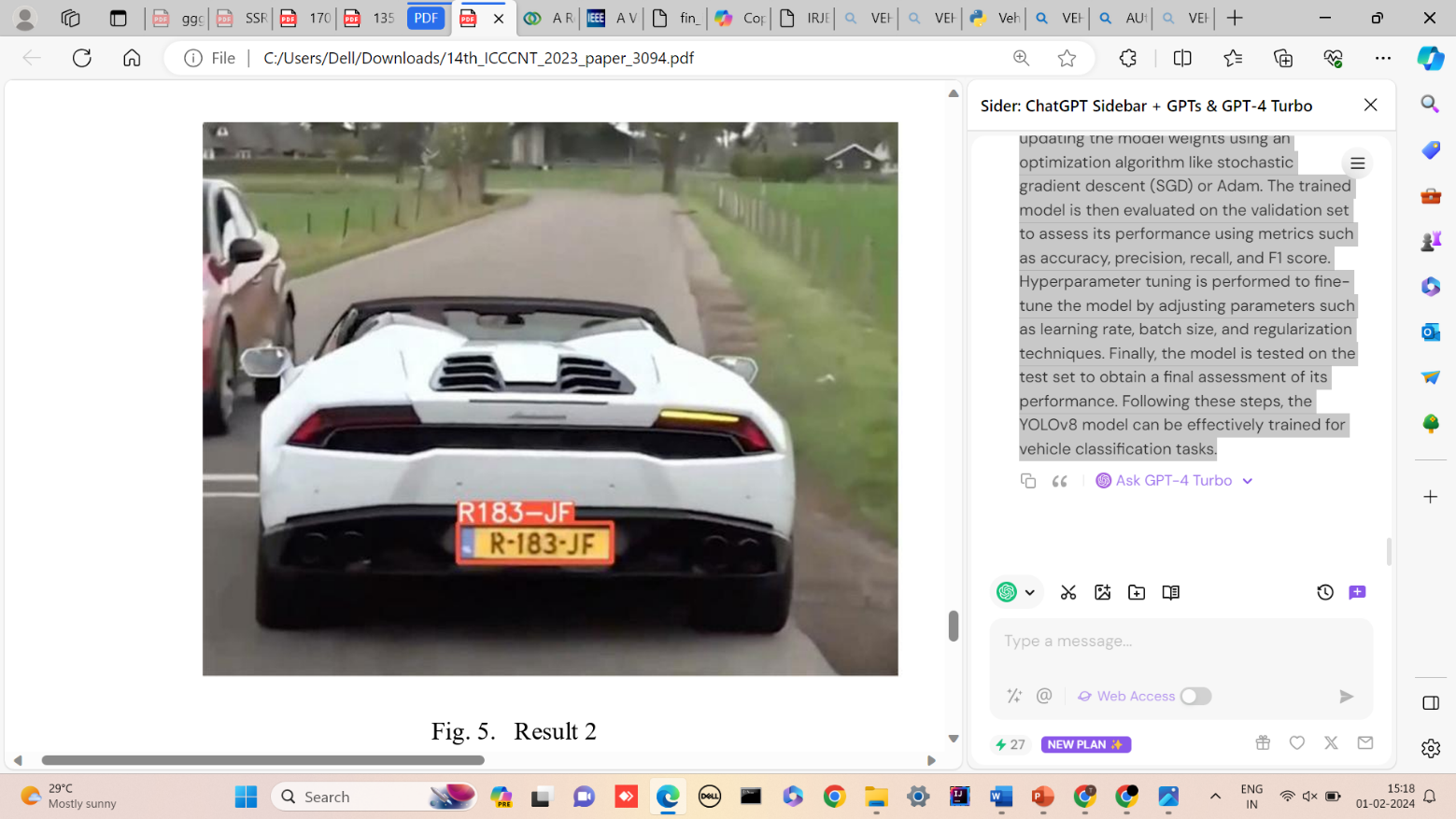


Fig 6: License plate Detection of Vehicle

5**.RESULT ANALYSIS**

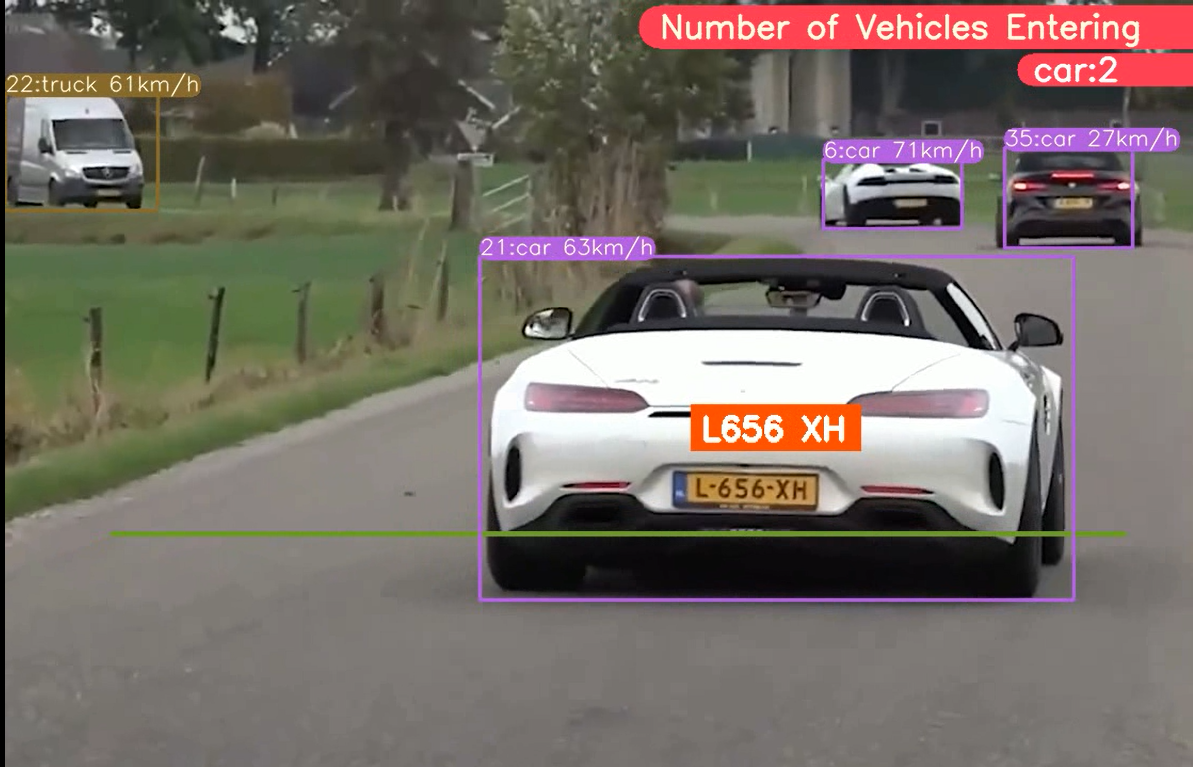


Fig 7: OCR License Plate + speed + obj Tracking + counting

The proposed Intelligent Vehicle Surveillance and Speed Enforcement System, leveraging YOLO-V8, DeepSORT, and EasyOCR technologies, showcases promising outcomes in enhancing traffic management and road safety. Through real-time vehicle detection, counting, classification, and speed measurement, the system provides accurate and efficient traffic flow monitoring. The integration of YOLO-V8 ensures robust vehicle identification, while the DeepSORT algorithm addresses false predictions, enhancing overall precision. The use of EasyOCR for license plate recognition adds a layer of enforcement capability. The system's cost-effectiveness, single-camera setup, and real-time processing capabilities make it a viable solution for practical applications. The result analysis underscores the system's potential to revolutionize traffic surveillance, curb speeding incidents, and contribute to the overall improvement of urban road safety and transportation infrastructure.

# CONCLUSION AND FURTHER SCOPE

A real-time traffic monitoring system is proposed, utilizing a virtual detection zone and YOLO v8, to enhance the efficiency of vehicle counting, detection, and classification. Additionally, the distance and time traveled by each vehicle are utilized to estimate their speed. YOLO v8 demonstrates superior accuracy in both classification and detection when compared to other algorithms, and deep sort is employed alongside YOLO v8 to track vehicles effectively. As a result, this method can be effectively applied to real-life scenarios for vehicle counting, speed estimation, and classification and LPR.

**Future Enhancements:**

To further improve the system's capabilities and extend its potential impact, several future enhancements can be considered are:

Multi-Camera Integration: Integrating data from multiple cameras to create a comprehensive traffic monitoring network for larger areas and better traffic flow analysis.

Violation Detection: Identifying vehicles exceeding speed limits based on pre-defined thresholds.

Alert System: Notifying authorities of speed violations for immediate action.

# ACKNOWLEDGMENTS

I like to thank my Team for continuous work and support in construction of paper and also extend my sincere gratitude for continuous encouragement and support.

# REFERENCES

1. Akhil Reddy Kalva, Jyothi Swarup Chelluboina, B. Bharathi.” Smart Traffic Monitoring System using YOLO and Deep Learning Techniques”. 2023 7th International Conference on Trends in Electronics and Informatics (ICOEI).
2. V. K. Madasu and M. Hanmandlu, "Estimation of vehicle speed by motion tracking on image sequences," 2010 IEEE Intelligent Vehicles Symposium, La Jolla, CA, USA, 2010.
3. Mahmudol H. Tusar, Md. T. Bhuiya, Md. S. Hossain, Anika Tabassum, Riasat Khan, "Real Time Bangla License Plate Recognition with Deep Learning Techniques", *2022 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET)*, pp.1-6, 2022.
4. Amit Ghosh, MD. shahinuzzaman, Hamudi Hasan Sonet, Swakkhar Shatabda. “An Adaptive Video-based Vehicle Detection, Classification, Counting, and Speed-measurement System for Real-time Traffic Data Collection”.2019 IEEE Region 10 Symposium (TENSYMP).
5. Raju, P. Daniel Ratna, and G. Neelima. "Image segmentation by using histogram thresholding." International Journal of Computer Science Engineering and Technology 2, no.1 (2012): 776-779.
6. Adnan, Muhammad Akram, Norliana Sulaiman, Nor Izzah Zainuddin, and Tuan Badrul Hisyam Tuan Besar. "Vehicle speed measurement technique using various speed detection instrumentation." In Business Engineering and Industrial Applications Colloquium (BEIAC), 2013 IEEE, pp. 668-672. IEEE, 2013.
7. Xinyue Zhao, Guangli Wang, Zaixing He.” A survey of moving object detection methods: A practical perspective”. Neurocomputing Volume 503, 7 September 2022, Pages 28-48.
8. Mohit Chandorkar , Shivam Pednekar , Dr. Sachin Bojewar, 2021, Vehicle Detection and Speed Tracking, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 10, Issue 05 (May 2021),
9. Patel Roy, Saurab Dutta, Nilanjan Dey, Goutami Dey, Sayan Chakraborty, Ruben Ray.”Adaptive thresholding: A comparative Study**”.**2014 International Conference On Control, Instrumentation, Communication and Computational Technologies (ICCICCT)
10. Esha Kapoor, Eklavya Sharma, Ekansh Gupta, Eesha Srivastava, Eisha Patel, "Automated license plate detection and recognition using machine”.
11. Liu, D. Q. Huynh, Y. Sun, M. Reynolds and S. Atkinson, "A Vision-Based Pipeline for Vehicle Counting, Speed Estimation, and Classification," in IEEE Transactions on Intelligent Transportation Systems, vol. 22, no. 12, pp. 7547-7560, Dec. 2021.
12. Anchal Baliyan, Anjali Saini, Amit Yadav, Akash Rao, Vishal Jayaswal , “Automatic licence plate detection and recognition".Han Chen d Shaun Wang "A Network Model Approach to Systemic Risk in the Financial System", January 2013,
13. S. Ren, K. He, R. Girshick, and J. Sun, “Faster r-cnn: towards real-time object detection with region proposal networks,” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137–1149, 2017.
14. X. Hu, Z. Wei, and W. Zhou, “A video streaming vehicle detection algorithm based on YOLOv4,” in *Proceedings of the 2021 IEEE 5th Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, pp. 2081–2086, Chongqing, China, March 2021.
15. H. Abdel-Gawad, A. Khamis, L. A. Said and A. G. Radwan, "Vulnerable Road Users Detection and Tracking using YOLOv4 and Deep SORT," 2021 9th International Japan-Africa Conference on Electronics, Communications, and Computations (JAC-ECC), 2021, pp. 140 145, doi: 10.1109/JAC-ECC54461.2021.9691441.
16. Du, L.; Chen, W.; Fu, S.; Kong, H.; Li, C.; Pei, Z. Real-time detection of vehicle and traffic light for intelligent and connected vehicles based on YOLOv3 network. In Proceedings of the 5th International Conference on Transportation Information and Safety (ICTIS), Liverpool, UK, 14–17 July 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 388–392.
17. C.-Y. Chen, Y.-M. Liang, and S.-W. Chen, “Vehicle classification and counting system,” in *Proceedings of the 2014 International Conference on Audio, Language and Image Processing (ICALIP)*, pp. 485–490, Shanghai, China, July 2014.
18. Grents, V. Varkentin, and N. Goryaev, “Determining vehicle speed based on video using convolutional neural network,” *Transportation Research Procedia*, vol. 50, pp. 192–200, 2020.
19. Liu, T.; Liu, Y. Deformable model-based vehicle tracking and recognition using 3-D constrained multiple-Kernels and Kalman filter. IEEE Access 2021, 9, 90346–90357
20. Kilic and G. Aydin, "Turkish Vehicle License Plate Recognition Using Deep Learning," 2018 International Conference on Artificial Intelligence and Data Processing (IDAP), Malatya, Turkey