International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2025: 7.089

Vehicle Detection Counting and Speed Estimation

Sohail[1], Dr. Satish Kumar Satti[2], Gayathri[3]

[1]Department of CSE, Vignan's Foundation for Science, Technology and Research, Guntur, India

[2]Department of CSE, Vignan's Foundation for Science, Technology and Research, Guntur, India

[3]Department of CSE, Vignan's Foundation for Science, Technology and Research, Guntur, India

Abstract: ***This project introduces a sophisticated vehicular monitoring system that leverages deep learning for real-time vehicle detection, classification, and speed estimation. The implementation is based on the YOLOv8 Nano model for high-performance object recognition, focusing on dynamic vehicle classes such as automobiles, trucks, and buses. A custom tracking algorithm is employed to maintain continuous vehicle trajectory tracking across video frames. Speed estimation is computed through pixel-based distance metrics and temporal analysis, applying distance-time calculations for precise measurements. DenseNet layers, coupled with dropout mechanisms, are utilized to enhance detection accuracy and generalization. The system effectively identifies vehicles violating speed regulations, providing a robust tool for intelligent traffic management, real-time surveillance, and road safety enforcement. Experimental results validate the system's reliability in automating vehicle tracking, velocity estimation, and overspeed detection within traffic monitoring infrastructures.***

Keywords: Vehicle detection, Speed estimation, Traffic monitoring, Deep learning surveillance, Overspeed detection, Vehicle classification, YOLOv8 Nano model, Object recognition, Vehicle tracking.

1. Introduction

In the evolving domain of intelligent transportation systems, our project is dedicated to achieving accurate Vehicle Detection, Counting, and Speed Estimation through state-of-the-art deep learning methodologies. We utilize the YOLOv8n (Nano) model, characterized by its efficient architecture of approximately 30 to 40 layers, which enables swift inference while maintaining a lightweight profile. Importantly, we have strategically opted not to train the initial 20 layers of the model; instead, we concentrate on fine-tuning the subsequent 20 layers through the integration of DenseNet architecture alongside dropout techniques. This methodological approach significantly enhances model performance and mitigates the risk of overfitting. Our project leverages advanced computer vision libraries such as OpenCV and pandas for robust analysis of video streams, enabling us to extract critical insights from the acquired data. Furthermore, our implementation features a sophisticated tracking system that efficiently counts vehicles and accurately estimates their speed, thereby optimizing traffic management and oversight. Through this initiative, we aspire to make significant contributions to the development of intelligent traffic systems that promote road safety and improve vehicular flow.

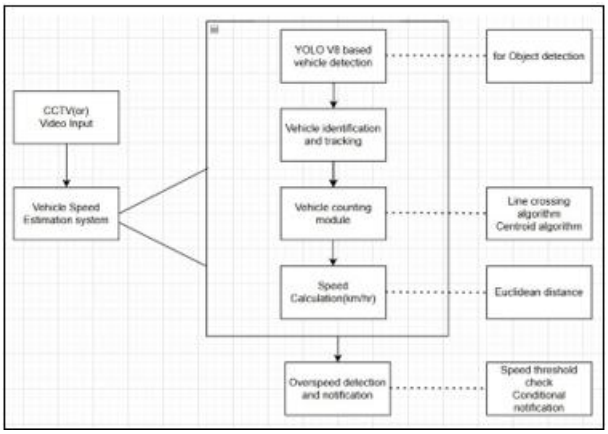
2. Literature Survey

YOLO algorithms, especially YOLO-v7, are highly accurate and fast for vehicle detection, making them useful in traffic systems and autonomous vehicles. They outperform older methods like HOG and SVM in both speed and accuracy. Future research may explore YOLO-v8 for further improvements in detection. As YOLO continues to evolve, it plays an increasingly important role in modern traffic management and safety technologies, YOLO-v7 has the highest accuracy, 95.74 percentage,.[1] Recent research on traffic monitoring systems highlights the use of YOLO for vehicle detection and DeepSORT for tracking, achieving over 90 percentage accuracy in varied conditions. Background subtraction techniques, like Gaussian Mixture Models (GMM), are also utilized but may struggle in low-light scenarios. Methods such as triple-axis reference lines and Regions of Interest (ROI) further enhance accuracy in speed estimation and complex traffic situations.[2] The study presents a real-time traffic monitoring system using a Gaussian Mixture Model (GMM) for vehicle counting and the YOLO algorithm (YOLOv3 and YOLOv4) for vehicle classification. The system achieved high classification accuracy, reaching 98.91 percentage on the MAVD dataset and 99.5 percentage on the GRAM-RTM dataset. Key methods include vehicle speed estimation based on distance and time traveled, utilizing virtual detection zones to enhance accuracy and efficiency under various conditions, such as daytime, nighttime, and rainy days. [3] Recent advancements in object detection and speed estimation show YOLO v8 achieving the highest accuracy of 53.3 percentage, surpassing previous versions and methods like HOG, CNN, R-CNN, and SSD. YOLO's progress from v2 to v8 reflects substantial improvements in mean average precision (mAP) and detection capabilities. YOLO v8, combined with deepSORT, effectively handles challenges like occlusion and perspective issues. This integration enhances real-time vehicle tracking and speed estimation based solely on video data.[4] Recent developments highlight YOLOv2's strong performance in vehicle detection and speed estimation under varied conditions, with impressive accuracy in bounding box detection. YOLOv2 and Faster R-CNN outperform SSD in vehicle counting and classification, with YOLOv2 achieving accuracy rates of 90-98 percentage. While

speed estimation results are encouraging, better calibration is necessary. Future efforts will aim to refine accuracy by incorporating additional cameras and advancing calibration techniques.[5] Recent studies utilize advanced methods like YOLOv3 and YOLOv4 AF for accurate vehicle detection (up to 95.78 percentage) and CycleGAN for nighttime detection, though with limitations in lighting and visibility. Bluetooth sensors combined with Isolation Forest detect anomalies but lack density and occupancy data. GAIN effectively imputes missing traffic data but needs further validation for diverse scenarios. These approaches underscore the advancements and challenges in traffic monitoring systems. [6] Modern vehicle detection systems utilize background subtraction and YOLO models for accurate real-time object detection and classification. BackgroundSubtractorMOG2 addresses shadow issues effectively, and vehicle counting and speed measurement are based on fixed thresholds. The system achieves over 80 percentage accuracy across diverse conditions, with future work focusing on applying machine learning for deeper traffic pattern analysis. [7] The system uses YOLO algorithms for real-time vehicle detection, achieving high accuracy in vehicle counting and speed detection. YOLO v3's multi-scale detection and advanced architecture enhance object recognition, while Non-Maximum Suppression refines bounding box predictions. Future improvements include transfer learning and number plate recognition to address accuracy limitations and further optimize traffic management. [8] Recent methods in vehicle speed recognition include depth-based systems like Lei Yang's stereoscopic vision and non-depth techniques such as Viktor Kocur's perspective transformations. These approaches minimize hardware costs, with accuracy errors ranging from 7.72 percentage to 14.11 percentage. A proposed system using YOLOv2 and road calibration offers a cost-effective solution with competitive accuracy. [9] Recent methods for vehicle counting and speed estimation use YOLOv5 for accurate vehicle detection, achieving up to 99.57 percentage accuracy in counting. Speed estimation is enhanced by linear regression and threshold filtering, comparing favorably with ground truth from vehicle loop detectors. The system demonstrates high accuracy and reliability in real-world scenarios. [10] Machine learning models, including SVM, RF, GBDT, XGBoost, and ANN, have been used for freeway traffic speed estimation. The RF model showed the highest accuracy and ease of calibration among them. All models effectively capture traffic speed patterns, but RF is preferred for its performance. Future work includes improving accuracy with more data and expanding to traffic flow prediction.[11] The system evaluated vehicle detection using confusion matrices for precision, recall, and accuracy. SSD, Faster R-CNN, and YOLO were compared: SSD excelled in real-time performance with a mAP of 0.78, while Faster R-CNN achieved higher accuracy (mAP of 0.83). YOLO had lower accuracy (mAP of 0.73) and struggled with small objects. Future work aims to integrate machine learning and additional sensors for enhanced accuracy. [12] The ReVISE system leverages RF signal strength and a multi-class SVM classifier, achieving perfect detection accuracy. Speed estimation is performed using a statistical approach and quadratic curve fitting, with the latter yielding 90 percentage accuracy. Future improvements include integrating both methods, refining data quality, and expanding to larger test environments. [13] The study introduces a vehicle detection, tracking, and counting system utilizing YOLOv3-tiny, implemented on a GeForce GTX 950M GPU. This system demonstrates effective real-time performance, processing at 33.5 FPS, and accurately manages vehicle tracking in both single and bidirectional traffic. The approach ensures reliable vehicle counting despite occasional detection misses or duplicate frames.[14] Vehicle tracking using roadside lidar is discussed, highlighting methods like clustering, centroid-based tracking, and accuracy refinement, reaching 0.22 m/s. The study shows advantages of higher lidar beam counts and optimal sensor positioning for urban traffic. Future work includes real-time monitoring improvements and incorporating deep learning for better classification. [15] A model combining YOLO-v3 and SORT was tested on the UA-DETRAC dataset, achieving an 85.45 percentage vehicle counting accuracy. The system struggled with stationary vehicles and intersections, leading to multiple counts and missed detections. Future improvements include using DeepSORT for better handling of occlusions and appearance changes. [16] This vehicle detection and tracking system employs TensorFlow with YOLOv4 and DeepSORT, surpassing YOLOv3 with 82.08 mAP@0.5. YOLOv4-tiny balances accuracy (76.14 percentage mAP@0.5) and speed (40 fps on GTX 1660ti). For improved adaptability, consider using a Raspberry Pi for compact installations and cloud computing with high-performance GPUs for enhanced efficiency.[17] This traffic surveillance system employs machine learning for vehicle number plate recognition and speed detection. It incorporates methods for speed monitoring, accident detection, and license plate identification, with automated notifications sent to control rooms. This solution provides a cost-effective and flexible approach to traffic management and surveillance. [18] This study evaluates a vehicle tracking and speed estimation model applied to Track 1 videos from the 2018 NVIDIA AI City Challenge. The model achieved a detection rate (DR) score of 1.0 but had an RMSE of 12.1094, resulting in an S1 score of 0.6547. While the constant speed (CS) model per-formed well, the predictive speed (PS) model showed higher variability and lower accuracy. Future work includes exploring smoothing techniques and comparing with other detect-then-track algorithms.[19] Vehicle speed estimation using YOLO for detection and RNN for prediction has been explored, demonstrating the effectiveness of bounding box area alone as the input feature. Evaluated on the VS13 dataset, the RNN model achieves an average error of 4.08 km/h, significantly outperforming audio-based methods with an RMSE of 7.39 km/h. The study finds that bounding box position does not enhance estimation accuracy, emphasizing the importance of using only bounding box area. [20]

3. Methodology

The project focuses on developing an automated system for detecting, counting, and estimating vehicle speed in real time, leveraging the YOLO V8 model, a vehicle tracking system, and custom algorithms for speed estimation and overspeed notification.



3.1 Input Data

1) ***Video Input Source:*** The input data for this project is video footage of highway traffic, capturing various vehicle types and movements. Frames from the video are processed individually to analyze vehicle position, direction, and speed across time.
2) ***Preprocessing:*** Before detection, frames are resized and adjusted for optimal performance, and bounding boxes are created around detected objects. Additionally, the center points of each detected object are calculated to support tracking, which enhances detection stability and consistency across frames.

3.2 YOLO V8-Based Vehicle Detection

1) ***Model Selection:*** The YOLO V8 model is chosen for its high-speed performance in object detection, with a pre-trained neural network designed to identify common objects, including vehicles.
2) ***Object Classes:*** The system focuses on specific vehicle classes such as cars, buses, motorcycles, bicycles, and trucks. The YOLO model processes each frame and provides bounding boxes for identified objects.
3) ***Model Configuration and Optimization:*** We use YOLOv8 for vehicle detection due to its quick, accurate identification of objects. This model is trained with 30–40 layers, utilizing dense Net and dropout algorithms to enhance performance on selected layers. Specifically, 20 layers are fine-tuned using these methods, while the remaining layers are left unaltered to maintain foundational feature extraction, improving speed and robustness.

3.3 Vehicle Identification and Tracking

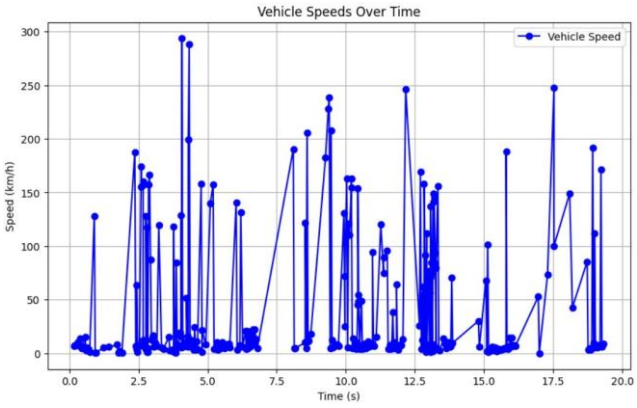
1) ***Tracker Initialization:*** A custom tracker class is implemented to assign unique IDs to vehicles, allowing the system to track them as they move across frames.
2) ***Centroid Calculation:*** For each detected vehicle, the system calculates the centroid of its bounding box, which serves as a key point for tracking.
3) ***ID Assignment:*** The tracker uses a distance threshold to determine if an object is already being tracked. If not, a new ID is assigned. This helps in maintaining accurate counts of vehicles entering and exiting the frame.

3.4 Vehicle Counting Module

1) ***Entry and Exit Lines:*** Virtual lines are defined within the frame to mark entry and exit zones. Vehicles crossing these lines trigger counting events, allowing for an accurate count of vehicles in each direction.
2) ***In-Out Counting:*** By monitoring the position of centroids relative to these lines, the system classifies vehicles as either "in" or "out," updating respective counts for each vehicle class.

3.5 Speed Calculation

1) ***Distance Calculation:*** Using the position of centroids over time, the system calculates the distance traveled by each vehicle. A constant pixels-per-meter (ppm) ratio is used to convert pixel distance into real-world distance.
2) ***Time Interval:*** The time between frames is recorded, allowing for speed calculation as speed = $\frac{\text{distance}}{}$
3) ***Speed Monitoring:*** For each vehicle, the speed is continuously updated based on its latest distance and time values.



3.6 Overspeed Detection and Notification

1) ***Speed Limit Check:*** A predefined speed limit is set for the monitored area. When a vehicle's calculated speed exceeds this limit, it is flagged as over speeding.
2) ***Notification Mechanism:*** Vehicles detected as over-speeding are visually marked with a different bounding box color, and an alert message is displayed on the video

feed.

3.7 Output Video Generation

Annotated Video*:* The processed frames, including vehicle bounding boxes, speed indicators, and counting in-formation, are compiled into an output video. This provides a visual record of vehicle movements and overspeeding events.

3.8 Code Execution and Optimization

The system is implemented using Python, with core libraries including OpenCV for video processing, Pandas for data handling, and Math for geometric calculations. The ultralytics library is used for YOLO V8 model implementation, enabling efficient vehicle detection.

1) ***Efficiency Considerations:*** To achieve real-time processing, the system optimizes the number of frames processed per second by resizing frames and reducing unnecessary computations float

4. Results

Model	Accuracy	Precision	Recall	F1- Score
YOLOVS (Trained by both dense net and drop out)	90.52%	87.34%	82.43%	84.81

4.1 Evaluation Metrics

1) ***Confusion Matrix:*** The confusion matrix is a key evaluation metric that provides insights into the accuracy of the model in terms of:

True Positives (TP): Vehicles correctly detected and tracked.

False Positives (FP): Non-vehicle objects falsely identified as vehicles.

True Negatives (TN): Correctly identified non-vehicle objects.

False Negatives (FN): Vehicles that were present but not detected.

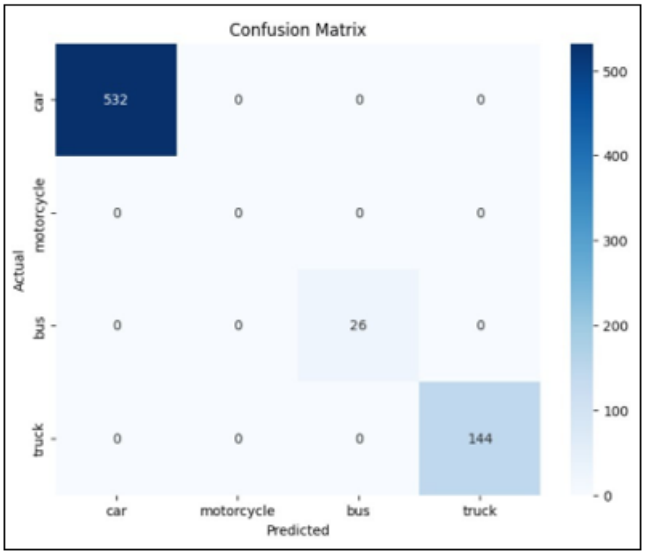
The confusion matrix helps in identifying the accuracy of the model in detecting vehicles correctly and avoiding false detections.

Accuracy: The accuracy of the model is calculated by dividing the sum of true positives and true negatives by the total number of instances. Accuracy helps in quantifying how well the system identifies vehicles correctly and excludes non-vehicles.

Precision: Measures the proportion of true positive vehicle detections out of all detected instances.

Recall: Indicates the proportion of actual vehicles detected out of the total vehicles present.

F1 Score: The harmonic mean of precision and recall, balancing the two metrics for an overall view of model performance.



References

[1] A. Dodia and S. Kumar, "A Comparison of YOLO Based Vehicle Detection Algorithms," 2023 Int. Conf. Ar-tif. Intell. Appl. ICAIA 2023 Alliance Technol. Conf. ATCON-1 2023 - Proceeding, pp. 1–6, 2023, doi: 10.1109/I-CAIA57370.2023.10169773.

[2] K. Kumar, M. T. Talluri, B. Krishna, and V. Karthikeyan, "A Novel Approach for Speed Estimation along with Vehicle Detection Counting," 2022 IEEE Stu-dents Conf. Eng. Syst. SCES 2022, pp. 1–5, 2022, doi: 10.1109/SCES55490.2022.9887707.

[3] C. J. Lin, S. Y. Jeng, and H. W. Lioa, "A Real-Time Vehicle Counting, Speed Estimation, and Classification System Based on Virtual Detection Zone and YOLO," Math. Probl. Eng., vol. 2021, 2021, doi: 10.1155/2021/1577614.

[4] K. Soma, L. Shibu, and N. Meenakshi, "A Real-Time Ve-hicle Detection and Speed Estimation Using YOLO V8," 2024 Int. Conf. Adv. Data Eng. Intell. Comput. Syst. ADICS 2024, pp. 1–6, 2024, doi: 10.1109/ADICS58448.2024.10533551.

[5] C. Liu, D. Q. Huynh, Y. Sun, M. Reynolds, and S.Atkinson, "A Vision-Based Pipeline for Vehicle Counting, Speed Estimation, and Classification," IEEE Trans. Intell. Transp. Syst., vol. 22, no. 12, pp. 7547–7560, 2021, doi: 10.1109/TITS.2020.3004066.

[6] T. Nadu, T. Nadu, and T. Nadu, "Advanced Traffic monitoring: Precise vehicle counting, speed estimation, color recognition and Type classification utilizing ROC curve analysis," pp. 349–354, 2024.

[7] A. Ghosh, M. S. Sabuj, H. H. Sonet, S. Shatabda, and D. M. Farid, "An Adaptive Video-based Vehicle Detection, Classification, Counting, and Speed-measurement System for Real-time Traffic Data Collection," Proc. 2019 IEEE Reg. 10 Symp. TENSYMP 2019, vol. 7, pp. 541–546, 2019, doi: 10.1109/TENSYMP46218.2019.8971196.

[8] K. Darwhekar, A. Patil, S. Ghodke, R. Bawkar, and S.

Rudrawar, “Computer Vision based Intelligent Traffic Man-agement System,” 6th Int. Conf. Electron. Commun. Aerosp. Technol. ICECA 2022 - Proc., no. Iceca, pp. 1051–1056, 2022, doi: 10.1109/ICECA55336.2022.10009105.

[9] W. P. Wu, Y. C. Wu, C. C. Hsu, J. S. Leu, and J. T. Wang, “Design and Implementation of Vehicle Speed Estima-tion Using Road Marking-based Perspective Transformation,” IEEE Veh. Technol. Conf., vol. 2021-April, pp. 1–5, 2021, doi: 10.1109/VTC2021-Spring51267.2021.9448813.

[10] C. W. Peng, T. Y. Lin, C. C. Hsu, and S. C. Huang, “Enhanced Vision-Based Speed Estimation By Road-side Surveillance Cameras,” GCCE 2023 - 2023 IEEE 12th Glob. Conf. Consum. Electron., pp. 889–890, 2023, doi: 10.1109/GCCE59613.2023.10315516.

[11] Z. Zhang and X. Yang, “Freeway Traffic Speed Es-timation by Regression Machine-Learning Techniques Using Probe Vehicle and Sensor Detector Data,” J. Transp. Eng. Part A Syst., vol. 146, no. 12, 2020, doi: 10.1061/jtepbs.0000455.

[12] M. Vijayalakshmi, D. Suvitha, and C. Gowtham Krishna, “Moving Vehicle Speed and Distance Estimation in Autonomous Vehicles,” 12th IEEE Int. Conf. Adv. Comput. ICoAC 2023, pp. 1–5, 2023, doi: 10.1109/I-CoAC59537.2023.10249241.

[13] N. Kassem, A. E. Kosba, and M. Youssef, “RF-based vehicle detection and speed estimation,” IEEE Veh. Technol. Conf., pp. 1–5, 2012, doi: 10.1109/VETECS.2012.6240184.

[14] G. Oltean, C. Florea, R. Orghidan, and V. Oltean, “Towards Real Time Vehicle Counting using YOLO-Tiny and Fast Motion Estimation,” SIITME 2019 - 2019 IEEE 25th Int. Symp. Des. Technol. Electron. Packag. Proc., no. October, pp. 240–243, 2019, doi: 10.1109/SIITME47687.2019.8990708.

[15] J. Zhang, W. Xiao, B. Coifman, and J. P. Mills, “Vehicle Tracking and Speed Estimation from Roadside Li-dar,” IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens., vol. 13, no. November, pp. 5597–5608, 2020, doi: 10.1109/JS-TARS.2020.3024921.

[16] J. M. Anil, L. Mathews, R. Renji, R. M. Jose, and S. Thomas, “Vehicle Counting based on Convolution Neu-ral Network,” Proc. 7th Int. Conf. Intell. Comput. Control Syst. ICICCS 2023, pp. 695–699, 2023, doi: 10.1109/ICI-CCS56967.2023.10142302.

[17] M. A. Bin Zuraimi and F. H. Kamaru Zaman, “Vehicle detection and tracking using YOLO and DeepSORT,” ISCAIE 2021 - IEEE 11th Symp. Comput. Appl. Ind. Electron., pp. 23–29, 2021, doi: 10.1109/ISCAIE51753.2021.9431784.

[18] P. S. Kumar, S. Shanmugasundaram, and S. Ma-halakshmi, “Vehicle Number Plate Identification and Speed Detection for Traffic Surveillance Prevention Using Ma-chine Learning,” 2023 Int. Conf. Syst. Comput. Autom. Net-working, ICSCAN 2023, pp. 1–5, 2023, doi: 10.1109/IC-SCAN58655.2023.10395451.

[19] S. Hua, M. Kapoor, and D. C. Anastasiu, “Ve-hicle tracking and speed estimation from traffic videos,” IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recog-nit. Work., vol. 2018-June, pp. 153–160, 2018, doi: 10.1109/CVPRW.2018.00028.

[20] A. Perunicic, S. Djukanovic, and A. Cvijetic, “Vision-based Vehicle Speed Estimation Using the YOLO Detector and RNN,” 2023 27th Int. Conf. Inf. Technol. IT 2023, pp. 15–18, 2023, doi: 10.1109/IT57431.2023.10078639.