Intelligent Traffic Surveillance: Machine Learning-Based Detection of Vehicle Speed Violations

Niloy Kanti Paul, Dipanwita Saha, Kaushik Biswas, Sultanul Arifeen Hamim, Tanvir Ahmed, Rifath Mahmud
Department of Computer Science, American International University-Bangladesh, Dhaka-1229, Bangladesh

***Abstract*—With the rapid increase in the number of vehicles on roads, traffic management, and safety enforcement have become significant challenges worldwide. Traditional speed violation detection systems either employ high-end hardware, expensive computational resources, or post-processed video data, which are inefficient to implement in real time. This study presents a real-time intelligent vehicle speed violation detection system using YOLOv8 for object detection and SORT for vehicle tracking, and a new Speed Detection Algorithm (SDA). The system can effectively detect vehicles and calculate their speed from video recorded by low-cost fixed cameras. Unlike other models that process simulated or post-processed video data, the new model processes real-life scenarios such as changing lighting and weather conditions. Experimental results indicate that the system achieves 92% to 95% vehicle detection accuracy while maintaining a Mean Absolute Error (MAE) of 1.8 km/h and Root Mean Square Error (RMSE) of 2.5 km/h for speed estimation, and 98% effective at speed detection compared to various other systems that came before it in terms of real-time processing effectiveness. This cost-effective and scalable solution can be incorporated into traffic observation systems for the improvement of road safety and regulation of speed limit compliance.**

***Keywords*—Machine learning; speed detection; object detection; vehicle tracking; violation**

I. Introduction

The rapid technological advancements and urbanization in recent decades have led to a tremendous surge in the number of vehicles traveling along roads worldwide. As transport infrastructure continues to advance, more and more vehicles have come with them, accompanied by terrible problems in traffic flow, traffic jams, and road safety. Speed violation has been one of the worst among such problems, and a high rate of road accidents and casualties has followed from it [1]. Traditional speed detection has been mainly reliant on sensor and radar based technologies, which, with high performance, have high operation and maintenance fees and do not scale. Manual traffic enforcement, on the contrary, is inefficient and labor intensive, especially in high-density urban and rural areas with insufficient surveillance infrastructure. To bridge these gaps, computer vision and AI-based technologies have come with increasing interest in traffic monitoring and police enforcement. These technologies offer real-time and automatic detection and classification of vehicles, thereby increasing efficiency, reducing human interventions, and increasing road safety enforcement [2].

In recent years, various AI-based methods have been explored to estimate vehicle speed, e.g., Optical Flow Analysis, Support Vector Machines (SVMs), and Convolutional Neural Networks (CNNs) [3]. These methods have worked very effectively in moving object detection and tracking; however, there are various limitations. Most such methods require high-performance hardware and thus are not feasible to be implemented in low-end environments. Some methods rely upon synthetic or preprocessed data, which do not depict real-world traffic situations effectively and lead to inconsistencies when implemented in real world settings. Another limitation results from inefficiency in some methods to work in various environmental settings, e.g., in nighttime, rainy, or foggy environments when visibility reduces [4], [5]. To overcome these constraints, in this study, a smart traffic monitoring system using deep learning-based vehicle detection and tracking algorithms to detect vehicles and estimate velocity with high accuracy has been proposed. The system employs You Only Look Once version 8 (YOLOv8) for detection and Simple Online and Real-time Tracking (SORT) for tracking and a Speed Detection Algorithm (SDA) developed and implemented in-house to estimate velocity. Unlike conventional techniques with high-end hardware demands, our system can be implemented in low-cost hardware and, therefore, can be deployed in various traffic environments. Our proposed model has also been tested in various lighting and environmental conditions to provide robust performance in real-world environments. The primary concern in our research is to develop a computationally efficient, real-time, and scalable speeding violation detection system that can be implemented in already deployed traffic monitoring infrastructures with minimal computational overhead. The system employs machine learning-based detection and tracking algorithms to achieve high detection performance with real-time processing efficiency. Experimental results indicate that the proposed system achieves vehicle detection accuracy ranging from 92%–95%, while maintaining low speed estimation errors with an MAE of 1.8 km/h and RMSE of 2.5 km/h.

Key contributions of this work include:

- Development of a real-time, low-cost vehicle speed detection system using YOLOv8 for object detection and SORT for vehicle tracking.
- Design and implementation of a custom Speed Detection Algorithm (SDA) that accurately measures vehicle speed using video footage from low-cost cameras.
- Validation of the proposed model under diverse environmental conditions, including daylight, night-time, rainy, and foggy weather, ensuring robust real-world performance.

- Comparison with state-of-the-art approaches, demonstrating higher accuracy and lower processing time.
- Provision of a scalable solution that can be integrated into existing traffic management systems, contributing to enhanced road safety and effective law enforcement.

The remainder of the study is structured as follows: Section II presents a comprehensive literature review of existing speed detection systems. Section III details the proposed methodology, including dataset creation, model architecture, and speed calculation techniques. Section IV presents experimental results and comparative analysis, followed by discussions on limitations and future work in Section V. Finally, Section VI concludes the study.

II. Literature Review

Efficient traffic regulation has always been a problem in the majority of countries. With a rising number of vehicles traveling roads, traffic detection and speeding detection have always remained priority fields to research. Radar-based detection and man-based detection have always been adopted, which have been ineffective and expensive. This has prompted studies in machine learning and deep learning-based algorithms to automatically detect vehicles and estimate speeding. From conventional image processing to recent deep learning-based techniques have been proposed. This section provides a review of already existing studies in vehicle detection, tracking, and speeding estimation and classifies them into different methodological categories.

Early research in vehicle detection and estimation relied heavily upon frame-by-frame analysis-based techniques in image processing. One early attempt was by P.G. Michalopoulos, who suggested Autoscope, a real-time vehicle detection system that compared frames to determine motion. The system was equipped with a camera with a frame rate of 30 frames per second and a frame rate with a value of 97% accuracy. However, one key shortcoming with such a technique was that it could not be designed to be adaptable to varying weather and therefore was not robust in real-world environments [6]. Another conventional approach was to utilize motion-blurred images to estimate vehicle velocity. The technique, suggested by Lin et al., utilized Sum Modified Laplacian (SML) focus measure to determine the best frame and get estimates for speed. Though such a system had a frame rate with a value of 95% accuracy, it was impacted in terms of reliability in low-visibility environments and when cars had very high speeds [7]. On a related note, a Combination of Saturation and Value (CVS)-based technique was suggested by Karim and Dehghani that compared frames to determine motion. However, such a technique was computationally expensive and could analyze just 12 frames in a second, which had a very negative impact on real-time performance [8]. Though such conventional image processing-based techniques were useful in terms of learning about vehicle detection, they had high computational complexity, low dynamic environment adaptability, and low generalization to different lighting and weather environments.

The evolution of machine learning (ML) has significantly improved vehicle detection and speed estimation, allowing for more robust and adaptable systems. Pavlidis et al. introduced a Fuzzy Neural Network (FNN) classifier, which used infrared imaging signals to detect vehicles and count their numbers. However, despite using a dual-band camera system, the model's accuracy was lower than 50%, making it unsuitable for real-world deployment [9]. Another notable ML-based study was conducted by Li et al., who developed an Intelligent Visual Internet of Things (IVIoT)-based system for vehicle monitoring. Their approach integrated haze removal techniques to enhance image clarity under poor weather conditions. Despite its capability to extract vehicle labels from urban traffic footage, the model only achieved an accuracy rate of 85.80%, limiting its performance for large-scale applications [10]. For speed estimation, Shim et al. proposed an optical character recognition (OCR) based vehicle speed estimation method (VSEM). This method assessed speed by analyzing alterations in video playback speed and achieved an impressive 94.99% accuracy rate. However, its major drawback was its dependency on preprocessed or re-encoded video data, making real-time deployment infeasible [11]. These ML-based approaches introduced significant improvements over traditional image processing methods, particularly in object classification and adaptability. However, challenges such as low real-time efficiency, computational overhead, and dependence on high-quality datasets still need to be addressed.

Deep learning has revolutionized vehicle detection and estimation of vehicle speed with successful real-time traffic monitoring applications. Deep learning using Convolutional Neural Networks and advanced detection algorithms has been extensive in high-precision and scalable detection processes. One such pioneering contribution in this direction has been by Grent et al., wherein Region-Based CNN-based models have been applied in detection. It had a detection rate with a rate of accuracy as high as 78% with their approach, which utilized SORT tracking algorithm. It was, however, constrained by poor processing rates and poor real-time responsiveness [12]. A performance-optimized solution with a better use of DL was brought by Luvizon et al., wherein a vehicle speed estimation system in a video has been proposed using a Kanade–Lucas–Tomasi algorithm and T-HOG descriptors to detect license plates. Though it had a precision rate of 0.93 and recall rate of 0.87, it had a drastic limitation in detecting vehicle speeds at a distant location, as it required cameras to be installed very close to vehicles [13]. To enhance detection efficiency even more, Şentaş et al. combined a Histogram of Oriented Gradients (HOG), Support Vector Machines (SVM), and You Only Look Once (YOLO) for vehicle classification. Though it ensured better classification accuracy, it led to a problem with overfitting, reducing generalization across different sets [14]. Though huge breakthroughs were brought by deep learning, a lot of researches continued to rely upon high-end computational power, manipulated sets, or constrained environments, reducing real-world and low-budget deployment effectiveness.

A comparative analysis of prior work reveals intrinsic limitations in existing vehicle detection and speed estimation algorithms. One such limitation is the utilization of synthetic or pre-processed data and thus real-time applicability in question. V2I network-based and OCR-based speed estimation researches were very accurate but tested with simulations or re-encoded video and thus could not be performed in real-time traffic [15], [11], [16]. Most researches lack the flexibility to environmental and illumination factors and per-

form best in day-time and breaks in the night-time, fog, or rainy environments and thus reduces usability in real-time traffic [6], [10], [13]. Another limitation is utilizing high-end hardware and it becomes infeasible to be scaled in low-end environments. Most researchers utilize specialized cameras, IR sensors, or computationally heavy deep networks and thus prove infeasible to be mass-deployed, particularly in developing nations [7], [8], [12]. Furthermore, some researches focus on vehicle detection without real-time speed estimation and thus proves infeasible to deploy in automated policing [9], [17]. This research proposes a real-time, low-cost, and high-accuracy vehicle speed violation detection system trained on real-world data across multiple environmental conditions. The system efficiently runs in low-configuration hardware by utilizing a combination of YOLOv8 and SORT tracking to perform real-time, high-precision detection, tracking, and velocity estimation and thus provides a practical and scalable solution to traffic monitoring.

Table I presents a comparative overview of existing vehicle detection and speed estimation approaches. Earlier image-processing-based techniques suffered from limited adaptability to dynamic traffic conditions and environmental variations. Machine learning approaches improved classification performance but often lacked real-time capability or required computationally expensive hardware. Recent deep learning-based methods achieved higher detection accuracy; however, many systems remained dependent on constrained datasets or post-processed video inputs. In contrast, the proposed system combines YOLOv8, SORT, and a custom Speed Detection Algorithm to provide real-time vehicle monitoring with robust performance under diverse environmental conditions using low-cost surveillance infrastructure.

III. Methodology

To develop a robust and efficient vehicle speed violation detection system, a custom dataset was created from real-world traffic footage. The dataset was carefully designed to include vehicles under various environmental conditions, ensuring the model generalizes well in real-world scenarios. The dataset consists of 12,000 manually labeled images, extracted from video footage recorded on urban roads, highways, and intersections. The dataset includes six distinct vehicle categories: bikes, auto-rickshaws, cars, trucks, buses, and other vehicles.

A. Dataset Collection Process

Since publicly released dataset will be non-real-time applicable to speeding detection, a real-world video dataset has been set. The dataset generation was achieved in three steps in a principal manner: video collection, frame capture, and annotation to encompass traffic scenarios in a holistic fashion.

The first set consisted of high-resolution video frames from fixed roadside cameras installed along urban roads, highways, and intersections. The frames were recorded in various environmental conditions like day, night, fog, and rainy environments to provide real-world variability to the dataset. The video recording was recorded at a frame rate of 30 frames per second to provide a smooth tracking of vehicle motion and vehicle speed estimation. Following the video recording set, frame pulling was performed to pull frames from recorded video. Frames were pulled at regular time steps to provide temporal continuity to support correct tracking. A total of 12,000 frames were selected from different traffic scenarios to provide a representative set in terms of vehicle speeds, densities, and orientations. Frames were manually inspected to discard duplicate or low-quality frames to increase reliability in the dataset.

The final operation in dataset generation was annotation and labelling using LabelImg, which is a free and open-sourced labelling tool. Individual frames captured were manually labelled to generate bounding boxes around detected vehicles and respective class labels for object detection. Six categories of vehicles were present in the dataset: bikes, auto-rickshaws, cars, trucks, buses, and others. Timestamps for each frame were also captured to support vehicle tracking and estimation of vehicle speed precisely. Annotations were in YOLO format with each labelled object described by a Class ID, bounding box coordinates (x-center, y-center, width, height) and detection correctness in terms of a confidence score. Manual checking ensured high-quality labelling and removal of errors that would have otherwise affected model performance. This structured method to create a dataset ensured that a robust dataset representative, diversified, and suitable for real-world traffic environments was established, providing a solid basis to design a good vehicle speed detection system. Table II presents the classification of objects based on vehicle type.

Table III and Table IV summarize the distribution of vehicle categories and environmental conditions in the proposed dataset. The dataset was intentionally designed to include a diverse range of vehicle types and traffic scenarios to improve the model's generalization capability in real-world environments. Cars constituted the largest portion of the dataset due to their high frequency in urban traffic, while motorcycles, trucks, buses, and auto-rickshaws were also sufficiently represented. Additionally, the dataset incorporated challenging environmental conditions, including night-time, rainy, and foggy scenes, enabling the proposed system to learn robust vehicle representations under varying visibility and illumination conditions.

B. Data Annotation and Labelling

To ensure proper detection and tracking of vehicles, a manual annotation was carried out to mark vehicles in each frame extracted. The annotation was carried out using LabelImg, a widely used open-source tool for annotating images in deep learning. A bounding box and a class label were assigned to each vehicle in the dataset. The six types of vehicles considered in this study were bikes, auto-rickshaws, cars, trucks, buses, and other vehicles. Each frame was annotated carefully to give the precise location of vehicles to ensure bounding boxes correctly capture them. The annotation was in the YOLO format, in which each item annotated was represented in terms of a Class ID and normalized bounding box coordinates (x-center, y-center, width, height). This format best supports real-time detection of objects and effective model training. To maintain quality and consistency in annotation, different annotators worked simultaneously in annotation and cross-checked each annotation to minimize errors. Any erroneous or ambiguous bounding boxes were rectified to enhance detection. Furthermore, frame timestamps were recorded for each frame that had been annotated to facilitate vehicle tracking and

TABLE I. COMPARATIVE ANALYSIS OF EXISTING VEHICLE DETECTION AND SPEED ESTIMATION METHODS

Study	Methodology	Key Strength	Limitation	Real-Time Support
Michalopoulos [6]	Frame-based image processing	High detection accuracy	Poor adaptability to weather conditions	Partial
Lin et al. [7]	Motion blur analysis	Accurate speed estimation	Sensitive to visibility and motion blur	No
Karim and Dehghani [8]	CVS-based motion detection	Effective motion comparison	Computationally expensive	Limited
Pavlidis et al. [9]	Fuzzy Neural Network	Infrared-based detection	Low detection accuracy	No
Li et al. [10]	IVIoT-based monitoring	Weather enhancement capability	Moderate accuracy	Partial
Shim et al. [11]	OCR-based VSEM	High estimation accuracy	Requires post-processed video	No
Grents et al. [12]	CNN + SORT	Deep learning-based tracking	Low real-time efficiency	Limited
Luvizon et al. [13]	T-HOG + KLT tracking	Good precision and recall	Camera placement dependency	Partial
Şentaş et al. [14]	HOG + SVM + YOLO	Improved classification	Overfitting issues	Partial
Proposed System	YOLOv8 + SORT + SDA	Real-time, multi-weather, low-cost deployment	Minor errors at extreme speeds	Yes

TABLE II. CLASSIFICATION OF THE OBJECTS ACCORDING TO THE VEHICLE TYPE.

Classification Number	Class
0	Bike
1	Auto
2	Car
3	Truck
4	Bus
5	Other Vehicles

TABLE III. DATASET DISTRIBUTION ACROSS VEHICLE TYPES AND ENVIRONMENTAL CONDITIONS.

Category	Number of Images	Percentage (%)
Cars	4,200	35.0
Motorcycles / Bikes	2,300	19.2
Auto-Rickshaws	1,500	12.5
Trucks	1,700	14.2
Buses	1,200	10.0
Other Vehicles	1,100	9.1
Total	**12,000**	**100**

TABLE IV. ENVIRONMENTAL DISTRIBUTION OF THE DATASET

Condition	Number of Images	Percentage (%)
Daylight	5,400	45.0
Night-time	2,600	21.7
Rainy	2,100	17.5
Foggy	1,900	15.8
Total	**12,000**	**100**

velocity estimation. Such timestamps played a very important role in velocity estimation from frame positional differences.

The annotated dataset was then put to use as ground truth to test and train the model to make sure that vehicle detection algorithm would be robust and effective in detecting and classifying vehicles in different real-world settings. The labelling exercise proved useful in making the proposed vehicle speeding violation detection system robust and effective.

C. Data Preparation

In order to ensure optimal performance in the vehicle detection model with YOLOv8, a variety of preprocessing methods was performed across the annotated dataset. The preprocessing methods were important in normalizing input images, enhancing the ability of the model to generalize, and boosting detection performance in different environmental settings. Preprocessing involved resizing, normalization, and augmentation and was important in effectively preparing the dataset for training.

1) Resizing of images and format standardization: Owing to the particular input dimensions required by YOLOv8, each annotated image was resized to a resolution of 640×640 pixels. This helped to provide consistent input dimensions to the model, thus improving detection effectiveness and lowering computational requirements. In addition, each image was converted to the RGB color space in order to provide consistent color representation and to remove inconsistencies with non-standard or grayscale images.

2) Normalization: To facilitate convergence during the training process, pixel intensities were normalized to a value between 0 and 1 by dividing each pixel by 255. This normalization helped to stabilize the gradient descent algorithm and prevented problems associated with vanishing and exploding gradients in terms of deep learning. In addition, normalization enabled adaptive learning in different lighting environments because pixel intensities were normalized uniformly across the dataset.

3) Data augmentation: To enhance the model's performance in recognizing vehicles in different real-world environments, a variety of data augmentation techniques was utilized. Systematic augmentation expanded the set's variety and thus prevented overfitting and improved robustness in different environmental settings. The following augmentations were applied:

- To include perspective changes, random rotations not more than 15 degrees and horizontal reflection were applied.
- Contrast and brightness levels have been altered to mimic different lighting environments, such as dark settings and very brightly lit settings.
- Gaussian blur in conjunction with the use of random noise was utilized to improve robustness to motion blur and poor camera input quality.
- To improve the model's performance in recognizing vehicles in environments with poor visibility, partial occlusions were introduced into some images.

All augmentation techniques were applied in a randomized fashion with a controlled probability, ensuring that no two

batches during training had the same images. This helped to improve the model's capacity to generalize, thus making it more responsive to real-world situations.

D. Dataset Splitting

To facilitate a successful training and testing procedure, the dataset was divided into three unique sets: test set, validation set, and training set. This specific stratification technique was carefully designed to maintain class balance and reduce bias in the learning phase of the model. The dataset setup ensured that each set contained a representative proportion of different vehicle types and environmental settings, thus ensuring strong generalization by the model in terms of real-world traffic scenarios. Additionally, the dataset included a broad mix of vehicle types, road types, and environmental differences, enabling the model to learn efficiently from data representative of real-world scenarios (see Table V).

TABLE V. DATASET SPLIT FOR TRAINING YOLOV8

Dataset Split	Percentage	Number of Images
Training Set	70%	8,400
Validation Set	20%	2,400
Test Set	10%	1,200

E. YOLOv8-Based Object Detection

YOLOv8 (You Only Look Once version 8) was selected for vehicle detection due to its balance between detection accuracy, computational efficiency, and real-time inference capability. Compared to two-stage detectors such as Faster R-CNN, the YOLOv8 network architecture used for vehicle detection is presented in Fig. 1. YOLOv8 offers significantly lower inference latency while maintaining competitive detection performance, making it more suitable for intelligent traffic surveillance applications.YOLOv8 represents a one-stage model for detection of objects that processes an image in a single pass, thus exhibiting high efficiency in real-time applications [18]. Unlike two-stage detectors like Faster R-CNN, YOLOv8 allows for direct predictions in terms of bounding boxes and class scores, thus reducing computational requirements to a great extent. The model features a CSPDarkNet backbone, which improves feature acquisition through cross-stage partial connections (CSP) to promote gradient flow and reduce redundancy in feature maps. The features thus extracted are then passed through a sophisticated Feature Pyramid Network (FPN) and a Path Aggregation Network (PAN) to provide detection across various scales by combining low-level and high-level feature representations [19], [20]. This design ensures effective detection of small and huge vehicles even in complex traffic scenarios. Another major innovation in YOLOv8 includes adoption of anchor-free detection paradigm, which does away with anchor boxes. This modification eases training and eliminates localization errors, thus producing better bounding box predictions. The bounding box prediction mechanism adopted in YOLOv8 involves a direct regression approach, wherein the model predicts bounding box center (x,y), width w, height h, and confidence score C for each detected object. The bounding box B predicted in output can be expressed as given below.

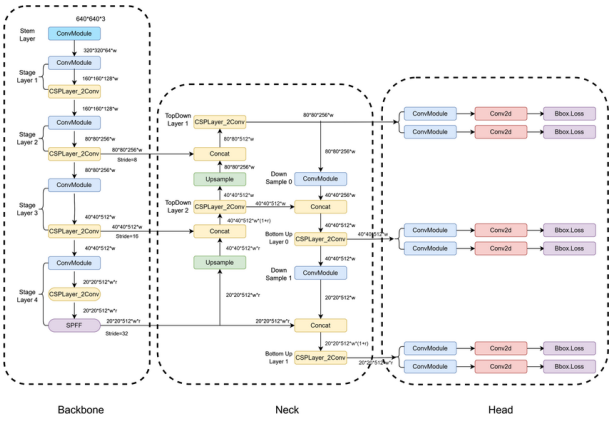

Fig. 1. YOLOv8-network-architecture-diagram

The bounding box B is defined as:

$$B = (x, y, w, h, C) \tag{1}$$

where, x, y represent the **center coordinates** of the bounding box. The values w, h define the **width and height** of the bounding box, and C represents the **confidence score** associated with the detection.

To improve detection accuracy, YOLOv8 utilizes the **Complete Intersection over Union (CIoU) Loss**, which refines the bounding box predictions by considering not only the overlap between predicted and ground truth boxes but also their distance and aspect ratio. The CIoU loss function is given by:

$$L_{\text{CIoU}} = 1 - \text{IoU} + \frac{\rho^2(\mathbf{b}, \mathbf{b}^{gt})}{c^2} + \alpha v \tag{2}$$

where, IoU is the Intersection over Union between the predicted and ground truth bounding boxes. The term $\rho(\mathbf{b}, \mathbf{b}^{gt})$ represents the Euclidean distance between the center points of the predicted and ground truth bounding boxes, while c is the diagonal length of the smallest enclosing box covering both predicted and ground truth boxes. The term v accounts for aspect ratio consistency, and α is a trade-off parameter.By optimizing the detection pipeline using the CIoU loss function, YOLOv8 achieves better object localization, reducing the chance of misalignment between predicted bounding boxes and actual vehicle positions. The combination of CSPDarkNet backbone, anchor-free detection, and CIoU-based loss optimization makes YOLOv8 well-suited for vehicle detection in dynamic traffic environments.

F. Model Training and Optimization

The training protocol defined for the YOLOv8 model was designed with a strategic emphasis on detection accuracy with real-time operational effectiveness. To realize this aim, the model was trained using a custom vehicle dataset that was previously created. This method of training was performed in the framework of supervised learning, with the model being carefully optimized to reduce differences between bounding box predictions and associated ground truth labels. Implementation and training of the YOLOv8 model was performed using

the PyTorch framework for deep learning, taking advantage of GPU acceleration to improve computational effectiveness. Stochastic Gradient Descent (SGD) was used as the optimizer for model optimization, systematically adjusting model parameters according to evaluated loss function. Training was performed with a resolution input image of 640×640 pixels for a period of 100 epochs with a batch size of 16. The AdamW optimizer was also adopted for adaptive learning rate optimization with the initial learning rate set to 0.001 and following a cosine schedule for learning rate decay. Regarding bounding box regression, Complete Intersection over Union (CIoU) Loss function was adopted by the model. A strategy using a mini-batch gradient descent was adopted, which involved weight updates following processing by small batches of images and not by the entire set in a single pass. This method helped in stabilizing convergence and improving the model's generalization.

The model was trained using a mini-batch gradient descent approach, where weights were updated after processing small batches of images rather than the entire dataset at once. This method helped stabilize convergence and improve generalization performance.The total loss function used during training was a combination of three key components:

$$L_{\text{total}} = L_{\text{cls}} + L_{\text{obj}} + L_{\text{CIoU}} \tag{3}$$

where, L_{cls} is the classification loss, which measures how accurately vehicles are classified into different categories, such as cars, bikes, and trucks. The term L_{obj} represents the objectness loss, ensuring that the model correctly identifies whether a region contains a vehicle or not. Finally, L_{CIoU} is the Complete IoU loss, refining the bounding box prediction as described in the previous section.

To further improve performance, learning rate scheduling was applied, where the learning rate was gradually reduced after each epoch using the cosine decay method:

$$\eta_t = \eta_{\text{init}} \times \frac{1 + \cos\left(\frac{t\pi}{T}\right)}{2} \tag{4}$$

The model was trained using the training set (70% of the dataset) and validated on the validation set (20%). After each epoch, the model was evaluated based on Mean Average Precision (mAP@50), precision, and recall metrics. The best-performing model checkpoint was saved, and early stopping was applied if the validation loss stopped improving for a predefined number of epochs.By leveraging optimized hyperparameters, loss function adjustments, and validation monitoring, the YOLOv8 model was effectively trained for accurate and efficient vehicle detection in real-world traffic environment

G. Object Detection and Bounding Box Generation

Following the training, the YOLOv8 model was applied to real-time vehicle detection in traffic videos. The model processed each video frame, recognized cars, and marked them using bounding boxes and confidence values for accurate vehicle localisation before tracking and speed measurement [21]. Unlike typical anchor-based strategies, YOLOv8 employed a direct regression strategy whereby each vehicle identified was assigned a bounding box that included a centre coordinate, height, width, and confidence level. Detections with a confidence greater than a threshold of 0.5 were considered valid for analysis. The idea of YOLO in object classification, object detection and segmentations is presented in Fig. 2.

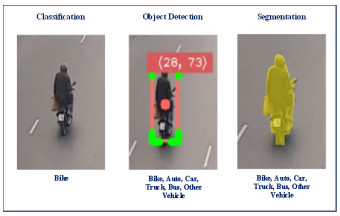

Fig. 2. YOLO in object classification, object detection and segmentations.

For improved detection accuracy, Non-Maximum Suppression (NMS) discarded duplicate detections by preserving high-confidence bounding boxes and suppressing overlapping ones by using an IoU threshold of 0.45. Resizing each video frame to 640×640 pixels was performed, and features were extracted using the CSPDarkNet backbone. Boxes were constructed and low-confidence detections were rejected. When a car was detected, bounding box information about it was propagated to the tracking module, which assigned a special tracking ID so that consistency over frames was preserved. This tracking ID was invaluable for estimating velocity over time. With good bounding box generation and filtering, YOLOv8 achieved proper real-time localization of vehicles accurately, which in turn formed the basis for the tracking of the vehicles using SORT.

H. SORT-Based Vehicle Tracking

SORT was selected as the vehicle tracking algorithm due to its simplicity, low computational overhead, and strong real-time performance. Although advanced tracking algorithms such as DeepSORT and ByteTrack provide improved identity preservation through appearance-based feature matching, they require additional computational resources and more complex feature extraction pipelines. In contrast, SORT combines Kalman filtering and the Hungarian assignment algorithm to achieve efficient multi-object tracking with minimal latency, making it suitable for real-time traffic surveillance systems operating on low-cost hardware platforms.After detection of vehicles in all frames using YOLOv8, their tracking from one frame to the other is required for speed measurement and motion estimation. For this purpose, Simple Online and Real-time Tracker (SORT) was chosen since it is light in weight, capable of real-time, and tracks several moving objects. SORT assigns a unique track ID to all detected vehicles and keeps updating the bounding box coordinates of each track constantly, rendering it consistent frame by frame [22].

SORT applies Kalman filtering and the Hungarian algorithm together to predict and match vehicle locations. The Kalman filter predicts a tracked vehicle's next location from its previous movement. It describes each vehicle's movement with a state vector having position and velocity components. Upon receiving a new detection, the Hungarian algorithm is applied to associate it with an ongoing tracked object by Intersection

over Union (IoU). If there is association, the tracking ID is retained; otherwise, a new ID is generated. The tracking mechanism is able to effectively handle occlusions and vehicle exits. When a vehicle goes out of detection for several frames, its motion is predicted through the Kalman filter. If it goes undetected more than a specified limit (e.g., 10 frames), the tracking ID is deleted. This real-time tracking process allows precise speed estimation, and thus, the system is capable of operating in high-traffic areas with numerous moving vehicles.

1) Camera calibration and homography transformation: Accurate vehicle speed estimation requires converting pixel displacement in video frames into real-world distance measurements. Since surveillance cameras capture a two-dimensional projection of a three-dimensional road scene, a homography transformation is applied to establish the relationship between image coordinates and actual ground-plane coordinates. The homography matrix is computed using four reference points selected from the road surface with known real-world distances. These reference points are manually identified from lane markings and fixed roadside structures visible in the surveillance footage. The corresponding real-world distances are measured directly from the road environment to ensure accurate scaling. Let (x, y) denote the pixel coordinates in the image plane and (X, Y) represent the corresponding real-world coordinates on the road plane. The homography transformation is defined as:

$$s \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = H \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{5}$$

where, H is the 3×3 homography matrix and s is a non-zero scaling factor.

The homography matrix is estimated using perspective transformation techniques based on corresponding point pairs between the image plane and the ground plane. After calibration, the pixel displacement of vehicles between consecutive frames is transformed into real-world displacement measured in meters. To validate the calibration accuracy, the estimated vehicle travel distances are compared with manually measured ground distances across multiple test sequences. The calibration process produces stable and consistent distance estimation with minimal deviation, enabling reliable vehicle speed computation under varying traffic conditions. Although homography-based calibration significantly improves real-world speed estimation accuracy, minor estimation errors may still arise due to perspective distortion, camera vibration, motion blur, and variations in road elevation.

2) Speed Detection Algorithm (SDA) and calculation: After vehicle tracking and camera calibration, the system estimates vehicle speed by measuring the real-world displacement of vehicles between consecutive frames. The calibrated homography transformation enables conversion of pixel movement into actual ground-plane distance, allowing accurate speed computation in real-time traffic scenarios, a crucial part of the proposed traffic monitoring system. The Speed Detection Algorithm (SDA) calculates the speed of a vehicle by measuring the time between predefined tracking points and dividing this by video frame rate (FPS) to get velocity. This ensures real-time identification of speed offenses at high accuracy.

The speed of a vehicle is determined by the distance traveled between two instants and the time taken to cover that distance. The equation used is

$$v = \frac{d}{t} \tag{6}$$

where, v represents the speed of the vehicle in meters per second, d is the distance traveled between two tracking points (in meters), and t denotes the time taken to cover the distance d (in seconds).

Since the video frame rate (FPS) is known, the time between two frames is given by:

$$t = \frac{1}{\text{FPS}} \times \Delta f \tag{7}$$

where, Δf represents the number of frames between two tracking points.

By substituting this into the speed equation, the vehicle speed can be expressed as:

$$v = \frac{d \times \text{FPS}}{\Delta f} \tag{8}$$

To convert the speed from meters per second (m/s) to kilometers per hour (km/h), the conversion equation is:

$$v_{\text{km/h}} = v \times 3.6 \tag{9}$$

I. Model Architecture

The speed violation detection vehicle designed system has a deep learning-based, modular structure consisting of YOLOv8 as the object identifier, SORT for tracking vehicles, and SDA for speed computation. The system operates in real-time to provide effective and accurate traffic violation monitoring.

The first process is video preprocessing, through which real-time video stream of traffic observation cameras is stripped and preprocessed for analysis. All the frames in the video stream are resized and normalized before feeding into the detection model. YOLOv8 object detection model identifies the vehicle, includes bounding boxes and class names, and rejects low-confidence detections. YOLOv8 deviates from mainstream object detection architecture by using an anchor-free scheme at the expense of no decreased precision with a rising level of computational simplicity. Following vehicle detection, SORT tracking algorithm assigns a distinct tracking ID to every vehicle and updates every vehicle's position over frames. Kalman filter predicts the motion, and Hungarian algorithm optimizes data association to enable robust tracking even in cluttered environments. The tracking ID remains valid for as long as the vehicle is tracked in the frame.To estimate speed, Speed Detection Algorithm (SDA) calculates the real motion of a vehicle in the world between two consecutive frames. Pixel distance is mapped to meters using a homography transformation, and speed is calculated based on frame rate (FPS) and time. Vehicles that are speeding above the set

speed limit are identified, and their number plates are read for enforcement.

The final output is stored in a database where traffic reports, violation records, and speed information are represented and processed. The structure has a low-cost, high-performance traffic surveillance system that is able to work in real road environments with variable environmental conditions.The overall architecture of the proposed intelligent traffic surveillance framework is illustrated in Fig. 3.

IV. Results

The performance of the suggested intelligent traffic monitoring system was evaluated in terms of vehicle detection accuracy (YOLOv8), tracking performance (SORT), estimation accuracy of speed (SDA), and real-time processing capability. The system was also evaluated in day, night, foggy, and rainy conditions to ensure its reliability towards real-world traffic scenarios. They measured the results against ground truth data obtained from radar-based speed sensors to estimate accuracy. mAP@50 and mAP@50:95, Precision, and Recall were utilized to analyze the accuracy of vehicle detection. The YOLOv8 model achieved an mAP@50 of 94.3% and an mAP@50:95 of 89.7%, indicating that it can detect multiple vehicle types sturdily. The model performed reasonably for different classes, with cars, trucks, and buses registering a higher mAP rate, while motorcycles and auto-rickshaws were marginally lower in mAP score due to their small dimensions and variable motion characteristics. The performance evaluation of YOLOv8 across different vehicle types is summarized in Table VI.

TABLE VI. Performance Evaluation of YOLOv8 for Different Vehicle Types.

Vehicle Type	mAP@50 (%)	Precision (%)	Recall (%)
Cars	95.2	96.1	94.5
Trucks	94.8	95.4	93.9
Buses	94.5	94.9	93.5
Motorcycles	92.1	93.0	91.2
Auto-Rickshaws	91.7	92.5	90.6
Overall Avg.	**94.3**	**94.7**	**93.9**

Environmental conditions also played a role in detection performance. As seen in Table VII, accuracy was highest during daylight conditions (95.6%) and decreased slightly under night-time (92.8%), foggy (89.4%), and rainy (87.9%) conditions due to reduced visibility and motion blur. To evaluate the consistency and reliability of the proposed framework, experiments were conducted across multiple test sequences captured under different environmental conditions. Detection accuracy values reported in Table VII represent the average performance across repeated evaluation runs. The observed standard deviation remained low across all environmental scenarios, indicating stable and reliable vehicle detection performance under varying traffic and visibility conditions.

The system effectively detected vehicles across multiple categories, achieving the highest accuracy for larger vehicles such as trucks and buses, while motorcycles and auto-rickshaws exhibited slightly lower precision due to their smaller size and rapid movement, as shown in Fig. 4.

The vehicle detection accuracy rate varied across different environmental conditions, with the highest accuracy observed in daylight conditions, while performance slightly decreased in night-time, foggy, and rainy environments, as shown in Fig. 5.

TABLE VII. Detection Accuracy and Variance Across Environmental Conditions.

Condition	Detection Accuracy (%)	Standard Deviation (%)
Daylight	95.6	0.8
Night-time	92.8	1.1
Foggy	89.4	1.4
Rainy	87.9	1.5

The confusion matrix provides further insights into the classification performance of YOLOv8, demonstrating strong differentiation between vehicle types and minimal misclassification errors, as shown in Fig. 6.

The SORT tracking algorithm performance was evaluated for ID consistency, handling occlusions, and overall tracking accuracy. It was found that the system maintained unique IDs for 97.1% of vehicles across consecutive frames while tracking errors were occasionally induced by occlusions under conditions of high vehicle density. Kalman filter estimates were confirmed by comparing positions of the calculated bounding boxes with respective actual vehicle positions with a mean positioning error of 2.1 pixels. Tracking stability was slightly reduced during night conditions (94.3%), the major reason being reduced feature contrast. But the system dealt with occlusions effectively, 92.7% of cars being successfully re-identified on temporary loss of visibility from the frame. The initial frames cut from the traffic surveillance video serve as input to the tracking module, where YOLOv8 detects vehicles and assigns distinctive tracking IDs before passing them along to the SORT algorithm for continued tracking, as shown is Fig. 7. After going through the tracking pipeline, the system can achieve successful distinctive tracking IDs for all vehicles, including when there are occlusions or changes in motion, demonstrating the effectiveness of the SORT algorithm.

Estimation of vehicle speed is a main component of the planned intelligent traffic surveillance system. Speed Detection Algorithm (SDA) estimates the speed of the vehicle based on its displacement between two consecutive frames and correlating pixel movement to real world speed measurement. The accuracy of SDA was investigated using ground truth speed measurements of radar-based speed sensors and vehicle's estimated speeds. The system had a total Mean Absolute Error (MAE) of 1.8 km/h and Root Mean Square Error (RMSE) of 2.5 km/h, which is indicative of high accuracy in speed estimation. The best was at low- and mid-speeds (0–60 km/h), where estimated and actual values were almost identical. Minor inaccuracies were also observed at high-speed cases (above 90 km/h) due to the influence of motion blur and different camera angles. Despite the challenges, the SDA recorded an accuracy level appropriate for use in real traffic monitoring scenarios. In order to further illustrate the performance of the system, Fig. 8 is a plot of estimated speed against actual speed for a test set. The ideal (y = x) case, represented by the dashed diagonal line, is where the estimated and actual speeds are perfectly matched. The blue line represents the model's estimate, and it has high correlation with ground truth and zero bias for high-speed cases. The consistency of the results confirms the effectiveness of the SDA algorithm in efficiently detecting

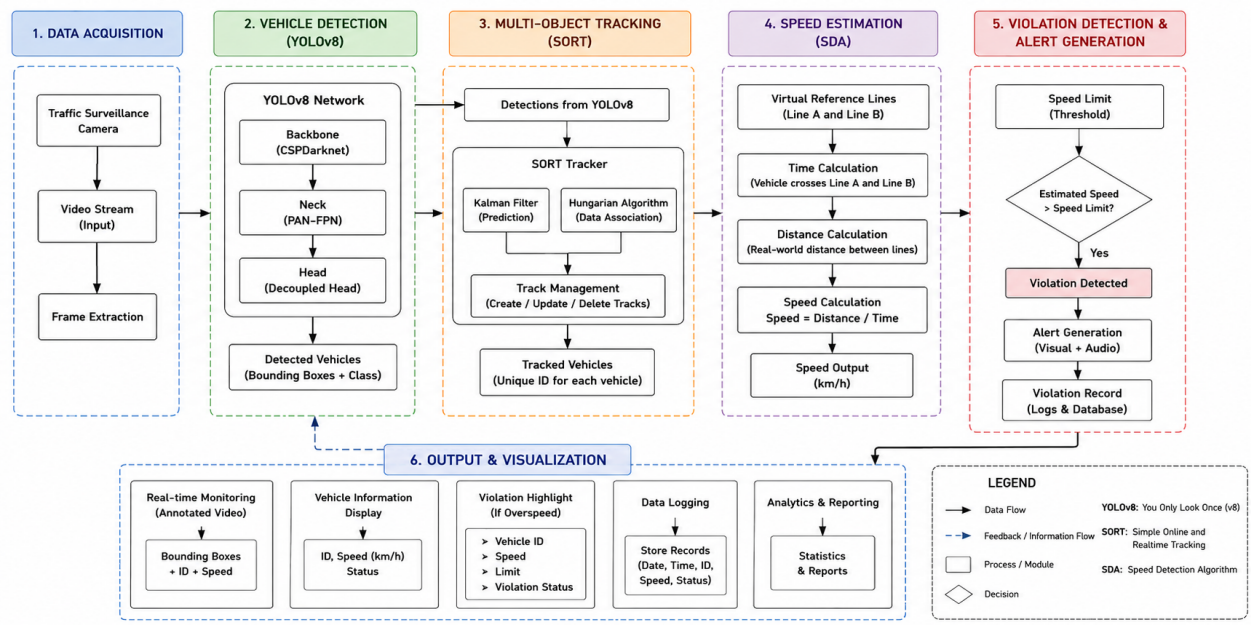

Fig. 3. Overall architecture of the proposed intelligent traffic surveillance and speed violation detection system.

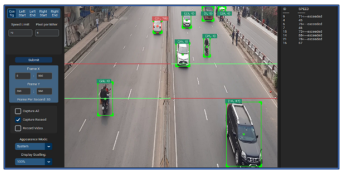

Fig. 4. Implemented speed violation detection application.

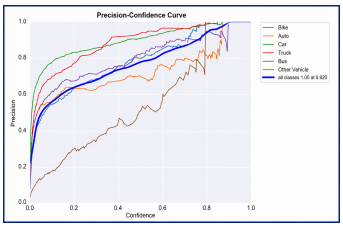

Fig. 5. Vehicle detection accuracy rate

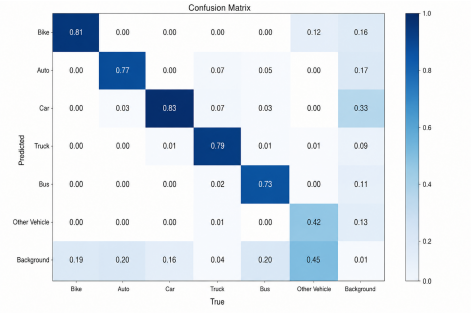

Fig. 6. Confusion matrix over mAP

speed offenses and facilitating enforcement practicability. As shown in Table VIII, the MAE increases as the speed range increases, indicating higher errors at higher speeds.

TABLE VIII. MEAN ABSOLUTE ERROR (MAE) AND ROOT MEAN SQUARE ERROR (RMSE) ACROSS DIFFERENT SPEED RANGES.

Speed Range (km/h)	MAE (km/h)	RMSE (km/h)
0 - 30	1.2	1.9
30 - 60	1.5	2.2
60 - 90	2.3	2.8
Above 90	2.9	3.4

The system performed exceptionally well for low- and medium-speed vehicles, with minimal errors. However, slight deviations were observed for high-speed vehicles (above 90 km/h) due to motion blur effects and variations in camera angle.

The effectiveness of an intelligent traffic surveillance system depends on its ability to process video streams in real-time. The proposed system was evaluated based on its frames per second (FPS), latency breakdown, and computational efficiency across different hardware configurations. The results indicate that when executed on a GPU (NVIDIA RTX 3060), the system achieved an average processing speed of 31.2 FPS, ensuring smooth real-time performance. In contrast, on a CPU (Intel i7-10th Gen), the FPS dropped to 11.8, highlighting the necessity of GPU acceleration for high-speed applications.The system's latency was analyzed in three key stages: vehicle detection (YOLOv8), tracking (SORT), and speed estimation (SDA). The YOLOv8 detection module required an average of 12.5 milliseconds per frame, while SORT tracking took 5.8 milliseconds per frame, and SDA computed speed within 3.6 milliseconds per frame. The combined processing latency remained well within real-time constraints, allowing the system to operate seamlessly at 30 FPS on GPU-enabled devices. These results confirm that the system is scalable and de-

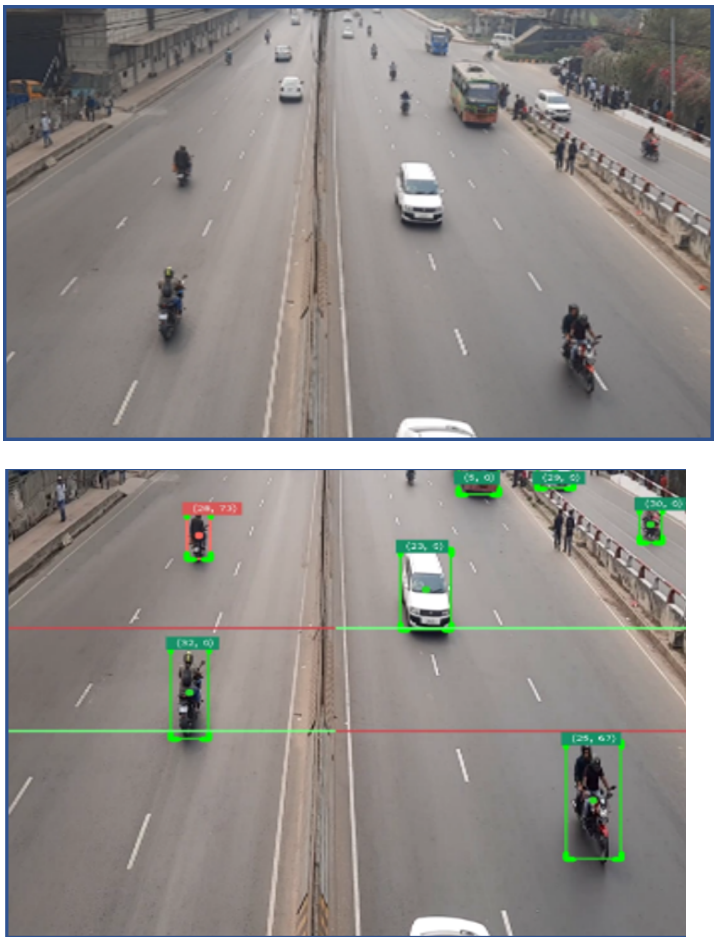

Fig. 7. Comparison of two images captured from traffic surveillance footage.

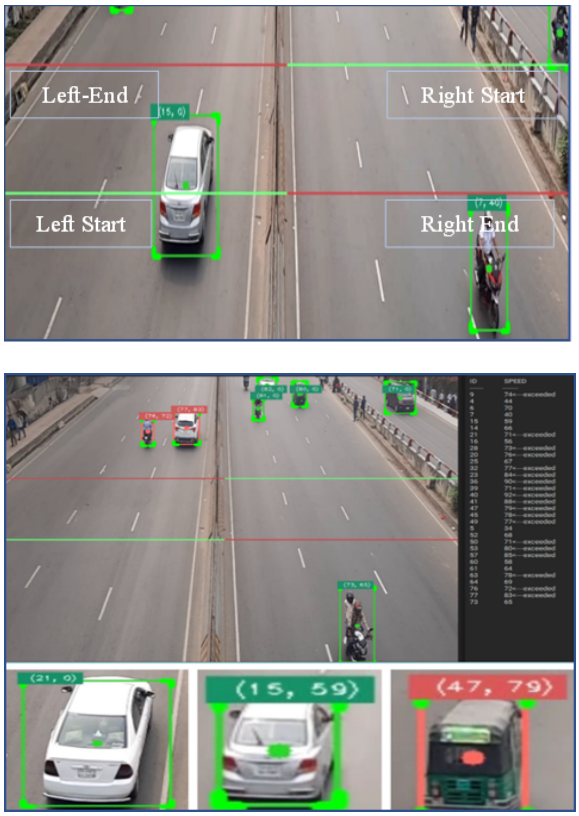

Fig. 8. Graph comparing estimated speed vs. actual speed.

ployable for real-world traffic monitoring, with the potential to handle multiple camera feeds in high-traffic zones while maintaining accurate and efficient processing. The processing speed comparison between GPU and CPU is presented in Table IX.

TABLE IX. PROCESSING SPEED COMPARISON BETWEEN GPU AND CPU

Hardware	FPS (Frames Per Second)
RTX 3060 GPU	31.2
Intel i7 CPU	11.8

Table X compares the proposed system with several existing vehicle detection and speed estimation approaches. Traditional image processing and OCR-based systems demonstrated reasonable detection performance but often lacked real-time applicability or robustness under varying environmental conditions. Deep learning-based methods improved detection capability; however, many required computationally expensive hardware or relied on constrained experimental settings. In contrast, the proposed system combines YOLOv8 and SORT to achieve robust real-time vehicle detection and tracking while maintaining low speed estimation error. The system operates effectively under multiple environmental conditions, including daylight, night-time, foggy, and rainy scenarios. Furthermore, the achieved processing speed of 31.2 FPS on GPU hardware demonstrates the suitability of the proposed framework for real-time intelligent traffic surveillance applications.

V. DISCUSSION

This study discussed the development and evaluation of an intelligent traffic surveillance system integrating deep learning-based vehicle detection, tracking, and speed estimation for real-time traffic monitoring. The study confirms that the system achieves high detection accuracy (mAP@50 of 94.3%), robust tracking (97.1% ID consistency), and precise speed estimation (MAE of 1.8 km/h, RMSE of 2.5 km/h), demonstrating its feasibility for automated traffic law enforcement. One critical finding is the effectiveness of YOLOv8 in vehicle detection, where single-stage detection ensures real-time efficiency. The confusion matrix analysis highlights that larger vehicles such as trucks and buses achieve higher accuracy, while smaller objects like motorcycles and auto-rickshaws show slight performance drops due to motion variations. Enhancing multi-scale feature learning could further improve detection for smaller vehicles. The SORT tracking algorithm successfully maintains vehicle identity across frames, handling occlusions and motion variations. However, minor ID mismatches occur in high-density traffic, where overlapping vehicles interfere with tracking. Future improvements such as Deep SORT with Re-ID networks could enhance long-term tracking stability, especially in dense urban settings.Speed estimation is highly accurate, with minimal deviations from ground truth radar-based measurements. However, higher-speed vehicles (¿90 km/h) exhibit minor estimation errors, likely due to motion

TABLE X. COMPARISON WITH EXISTING VEHICLE SPEED DETECTION SYSTEMS

Study	Detection Method	Tracking Method	Speed Estimation Accuracy / Error	Real-Time Capability	Environmental Robustness
Michalopoulos [6]	Image Processing	Frame Differencing	97% detection accuracy	Partial	Limited
Lin et al. [7]	Motion Blur Analysis	No Tracking	95% speed estimation accuracy	No	Poor in low visibility
Shim et al. [11]	OCR-based VSEM	OCR Tracking	94.99% accuracy	No (post-processed video)	Moderate
Grents et al. [12]	CNN-based Detection	SORT	78% detection accuracy	Limited	Moderate
Luvizon et al. [13]	T-HOG + KLT	Feature Tracking	Precision: 0.93	Partial	Camera-distance sensitive
Proposed System	YOLOv8	SORT	MAE: 1.8 km/h, RMSE: 2.5 km/h	Yes (31.2 FPS, GPU)	High

blur and perspective distortion. Motion compensation and adaptive calibration could improve accuracy at extreme speeds. Although the system can operate on CPU-only hardware, the reduced processing speed (11.8 FPS) may limit deployment in high-density traffic environments requiring strict real-time responsiveness, emphasizing the importance of GPU acceleration for large-scale deployments. Despite its success, the system's performance slightly degrades in low-light conditions. Future work should explore infrared-based detection, adaptive calibration, and integration with license plate recognition (LPR) to enhance traffic enforcement capabilities.While achieving strong real-time performance, the system faces several challenges. In high-density traffic, the SORT framework occasionally suffers from tracking instability and ID switching due to its reliance on motion rather than appearance. Environmental factors like heavy rain, glare, and low light also impact detection accuracy, while high-speed vehicles (over 90 km/h) introduce estimation errors from perspective distortion. Furthermore, the system's real-time consistency depends on GPU acceleration, limiting its efficiency on CPU-only hardware. Future enhancements could integrate DeepSORT or ByteTrack for better occlusion handling and infrared-enhanced detection to ensure robustness in diverse surveillance environments.

VI. CONCLUSION

In recent time, traffic rule violation is one of the most alarming news. Even this is so much big that human interference is even harder to manage the problem. To eliminate the problem, this overspeed issue should be solved as soon as possible. With a view to solving the problem, a system has been proposed where the system starts its working flow by detecting a vehicle. In tracking the vehicle, the latest YOLO model, the custom YOLOv8n, has been used. Using a tracking algorithm named SORT, tracking of the detected vehicle can be ensured. The model was trained with 12000 images. After that, the main cause of the road accident, overspeed, can be measured by calculating the frames per second of the video. This technology will contribute to a decrease in overspeed-related fatalities. The system will also include certain further future developments, such as lane violation detection, pedestrian lane detection, traffic light detection, etc. In order to enhance road safety, the proposed system provides a practical and cost-effective traffic surveillance solution that can operate using standard camera infrastructure. While CPU-based execution is feasible for low-density or non-critical monitoring scenarios, GPU acceleration is recommended to achieve consistent real-time performance in high-traffic environments

DECLARATIONS

Competing Interests

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this study.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Author Contribution

N. K. Paul conceptualized the study, designed the methodology, and performed the machine learning modeling. D. Saha and K. Biswas contributed to data collection, preprocessing, and experimental validation. S. A. Hamim and T. Ahmed worked on the implementation, optimization, and technical improvements. R. Mahmud supervised the project, provided critical insights, and revised the manuscript. All authors contributed to the writing, reviewing, and approval of the final manuscript.

Data Availability Statement

The datasets used and analyzed during the current study are available from the corresponding author upon reasonable request.

Research Involving Human and/or Animals

This study did not involve any human participants or animals.

REFERENCES

[1] A. Sayar and Y. Dikilitas, "A real-time traffic violation detection system on highways using surveillance cameras and message-oriented middleware," *IEEE Access*, 2026.

[2] M. K. Jha, P. K. Jha, and R. K. Yadav, "A grid-enabled vision and machine learning framework for safer and smarter intersections: Enhancing real-time roadway intelligence and vehicle coordination," *Infrastructures*, vol. 11, no. 2, p. 41, 2026.

[3] H. Zhang, M. Liang, and Y. Wang, "Yolo-bs: a traffic sign detection algorithm based on yolov8," *Scientific Reports*, vol. 15, no. 1, p. 7558, 2025.

[4] Y. Liu and S. Shen, "Vehicle detection and tracking based on improved yolov8," *IEEE Access*, 2025.

[5] K. Pawar and V. Attar, "Deep learning based detection and localization of road accidents from traffic surveillance videos," *ICT Express*, vol. 8, no. 3, pp. 379–387, 2022.

[6] P. Michalopoulos, "Vehicle detection video through image processing: the autoscope system," *IEEE Transactions on Vehicular Technology*, vol. 40, no. 1, pp. 21–29, 1991.

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

[8] M. Karim and A. Dehghani, "Vehicle speed detection in video image sequences using cvs method," *International Journal of the Physical Sciences*, vol. 5, no. 17, pp. 2555–2563, 2010.

[9] I. Pavlidis, V. Morellas, and N. Papanikolopoulos, "A vehicle occupant counting system based on near-infrared phenomenology and fuzzy neural classification," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 2, pp. 72–85, 2000.

[10] Q. Li, H. Cheng, Y. Zhou, and G. Huo, "Road vehicle monitoring system based on intelligent visual internet of things," *Journal of Sensors*, 2015.

[11] K. Shim, N. Park, J. Kim, O. Jeon, and H. Lee, "Vehicle speed measurement methodology robust to playback speed-manipulated video file," *IEEE Access*, vol. 9, pp. 132 862–132 874, 2021.

[12] A. 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.

[13] D. Luvizon, B. Nassu, and R. Minetto, "A video-based system for vehicle speed measurement in urban roadways," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 6, pp. 1393–1404, 2016.

[14] A. Sentas, I. Tashiev, F. Kucukayvaz, S. Kul, S. Eken, A. Sayar, and Y. Becerikli, "Performance evaluation of support vector machine and convolutional neural network algorithms in real-time vehicle type and color classification," *Evolutionary Intelligence*, vol. 13, pp. 83–91, 2020.

[15] K. Sridharamurthy, A. Govinda, J. Gopal, and G. Varaprasad, "Violation detection method for vehicular ad hoc networking," *Security and Communication Networks*, vol. 9, no. 3, pp. 201–207, 2016.

[16] M. Dahl and S. Javadi, "Analytical modeling for a video-based vehicle speed measurement framework," *Sensors*, vol. 20, no. 1, p. 160, 2019.

[17] S. Srividhya, C. Kavitha, W. Lai, V. Mani, and O. Khalaf, "A machine learning algorithm to automate vehicle classification and license plate detection," *Wireless Communications and Mobile Computing*, pp. 1–12, 2022.

[18] U. Nishitha, V. Lokesh, T. Kaushik, and R. P. Singh, "Vehicle detection in unmanned aerial imagery through advance you only look once architectures," in *2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, June 2024, pp. 1–6.

[19] N. Chatterjee, A. V. Singh, and R. Agarwal, "You only look once (yolov8) based intrusion detection system for physical security and surveillance," in *2024 11th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO)*, 2024, pp. 1–5.

[20] V. Babanne, N. S. Mahajan, R. L. Sharma, and P. P. Gargate, "Machine learning based smart surveillance system," in *2019 Third International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC)*. IEEE, 2019, pp. 84–86.

[21] J. Xie, Y. Zheng, R. Du, W. Xiong, Y. Cao, Z. Ma, D. Cao, and J. Guo, "Deep learning-based computer vision for surveillance in its: Evaluation of state-of-the-art methods," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 4, pp. 3027–3042, 2021.

[22] Z. Min, G. M. Hassan, and G.-S. Jo, "Rethinking motion estimation: An outlier removal strategy in sort for multi-object tracking with camera moving," *IEEE Access*, 2024.