Deep Learning Model for Handling Environmental Variability and Tracking Occluded Vehicles for Video Surveillance

EKTA UKEY[1,*], SATISHKUMAR L. VARMA[2]
[1]Pillai College of Engineering, Department of Information Technology,
University of Mumbai,
Navi Mumbai, Maharashtra,
INDIA

[2]Dwarkadas J. Sanghvi College of Engineering, Department of Information Technology,
University of Mumbai,
Mumbai, Maharashtra,
INDIA

**Corresponding Author*

Abstract: - Video Surveillance for Vehicle Detection (VD) and Tracking has many challenges like low brightness, low contrast, noise, occlusions, and identity consistency. This paper presents an adaptive occlusion-aware multi vehicle tracking model that improves robustness under varying illumination, congestion, and occlusion conditions while maintaining efficient performance. The video frame is enhanced using LAB–CLAHE and Contrast Enhancement. The proposed work incorporates an Occlusion Detection and Adaptive Handling module that analyzes motion characteristics and spatial overlap before VD. Occlusion status is recorded for each affected vehicle to support subsequent tracking and performance evaluation. This adaptive mechanism enhances the robustness of the VD by maintaining reliable vehicle localization and improving detection continuity in crowded traffic scenes with frequent object overlap. The proposed model provide the highest metric values of mAP@50 as 98.1%, mAP@50-95 as 71.5%, Precision as 92.1%, Recall as 94%, F1 score as 90% for VD, and MOTA as 85.2%, IDF1 score as 92.6% for vehicle tracking.

Key-Words: - Occlusion, Vehicle Detection, Vehicle Tracking, Surveillance, Environmental, Deep Learning, Multi-Object Tracking.

Received: March 18, 2026. Revised: May 12, 2026. Accepted: June 11, 2026. Available online: September 16, 2026.

1 Introduction

Advanced Intelligent Transportation (ATS) consists of VD as a very important part. These systems need accurate, real-time information to improve traffic flow, safety, and support self-driving car technologies [1], [2], [3]. The number of cars on the road is growing day by day, so it's important to have strong VD systems that can work in different situations, like when the weather changes, the lighting changes, or the types of vehicles change [4]. VD and tracking play a very important role in computer vision tasks that detect, classify, and monitor vehicles. There is a need for the creation of excellent, reliable, and efficient services using deep learning for traffic surveillance, counting, and congestion management. Image processing techniques are a powerful way to do these things because they can look at pictures and videos taken by cameras. Accurate VD is needed for self-driving cars, ITS systems, and traffic monitoring to work [5], [6]. VD, managing traffic jams, and making roads safer all depend on being able to accurately detect, track, and re-identify vehicles. Real-world traffic situations present many problems, such as changing object sizes, obstructions, bad lighting, and computational limitations on edge devices that cannot be fully handled by existing models. The goal of VD and tracking is to automatically find and follow the movement of vehicles in real-time video sequences for security purposes. The paper talks about a video surveillance system that can find and track vehicles even when they are blocked by other objects.

2 Related Works

The traditional vision-based VD methods are not accurate for small and occluded targets. An XGBoost classifier is used to break up the image and find foreground objects in all verified vehicles were then subjected to VGG feature extraction [7], which assigns a unique identifier to each vehicle to enable multi-object tracking across the image frames. Vehicles are counted and categorized into stationary and moving cars by detecting motion in each vehicle using the optical flow algorithm. After that, tracking is done using the simple online and real-time tracker. In CODAN [8], vehicle counting is done by multi-scale-density maps. It is capable of capturing the spatial features of vehicles. In [9], an optimal MSR algorithm is applied to improve the original nighttime images to address weak light and uneven brightness at night, leading to a higher accident rate than during the day. After that, the improved images were given to the pretrained YOLO v3 network, which detects cars in the nighttime pictures. In [10], the COCO dataset was used, having an average precision of 56.8%. [11] proposed a vehicle flow detection and tracking method that integrates an improved YOLOv8n model with the ByteTrack algorithm, achieving a mean Average Precision (mAP) of 62.8%. [12] applied tracking models in traffic monitoring systems and achieved an mAP50 of 80.6% and mAP50-95 of 50.1%. In [13], tracking is performed on images taken from more than one traffic surveillance system on the same road or route the mean Average Precision (mAP) value was observed as 89.20% for the Faster R- CNN algorithm. The mask-based tracking is done in [14] to improve recognition consistency and maintain resilience against occlusions, especially in congested conditions with a mAP of 79.5%.

3 Proposed Approach

The proposed model for handling environmental variability and tracking occluded vehicles for video surveillance is shown in Figure 1. The goal of the proposed model is to record when the detection occurred, what the vehicle class is, where it was detected (BB), how confident the model is, and under what environmental conditions (occluded or non-occluded). The proposed model performs VD on video sequences by integrating environmental preprocessing, motion analysis, and adaptive deep learning-based detection. The environmental preprocessing is done using the color space transformation, where each frame is converted from RGB to LAB color space. The LAB color space is selected due to its perceptual uniformity and explicit separation of luminance from chromatic components, enabling illumination-invariant contrast enhancement and improving the robustness of motion analysis and VD under challenging environmental conditions. Contrast Limited Adaptive Histogram Equalization (CLAHE) is an image contrast enhancement technique which improves visibility without amplifying noise, especially under uneven illumination.

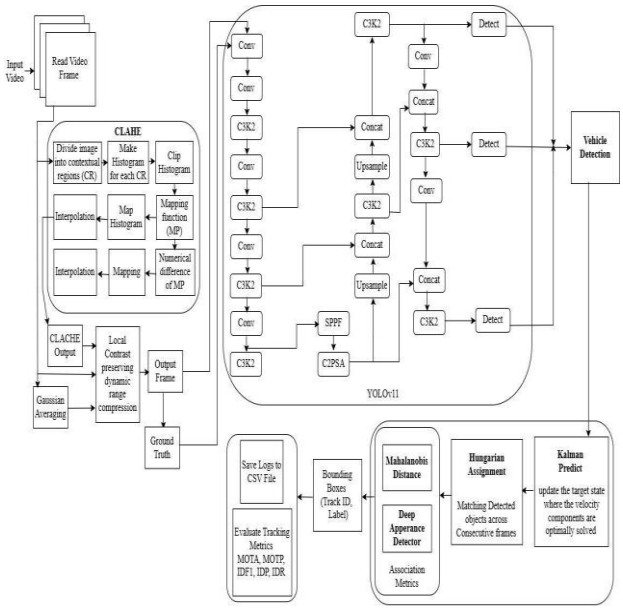

Fig. 1: Proposed Model for Handling Environmental Variability and Tracking Occluded Vehicles for Video Surveillance
Source: Created by the authors

Each input frame is first enhanced using CLAHE-based luminance correction and Gaussian smoothing to reduce illumination variations and noise. CLAHE increases contrast locally (small regions instead of the whole image) and prevents over-amplification using a contrast limit. It works very well for dark, shadowed, foggy, or nighttime scenes. Then, a background subtraction model is used to make a foreground motion mask, which is then used to get motion blobs. Occlusion is estimated using two criteria: the motion density ratio and the Intersection over Union (IoU) between motion blobs. Based on the occlusion state, the detection is passed to an object detector and only vehicle classes are retained. The detected vehicles are annotated on the frame, and their spatial and confidence information, along with occlusion status, is recorded in a structured log. This adaptive pipeline improves detection robustness under varying environmental conditions and traffic

densities while maintaining real-time performance.

The model's tracking framework performs multi-object tracking. Given an input video sequence V = $\{F_1, F_2, \ldots, F_t\}$ and a corresponding set of frame-wise detections D = $\{D_1, D_2, \ldots, D_t\}$ obtained from a detector, the system initializes a tracker with predefined motion and appearance thresholds. For each frame F_t, the detections Z_t are retrieved from the CSV source and provided to the tracker. The tracker first predicts the states of existing tracks using a Kalman filter and then associates the current detections with predicted tracks using a joint cost function based on IoU, motion similarity and appearance-based cosine distance. The optimal assignment is solved using Hungarian optimization, after which confirmed tracks are updated and assigned persistent identities. For each confirmed track, the BB, class label, and occlusion state are recorded, and the corresponding annotation is rendered on the output frame. The above process is continued for all the frames, which produces an annotated video and a structured track record set containing frame index, track identity, class, BB coordinates, and occlusion status. The resulting record set is stored in a CSV file for further analysis.

Input Video is defined as:

$$V = \{F_t \mid t = 1,2,..,T\}$$

where $F_t \in R^{H\times W\times 3}$ is the RGB frame at time, t

$$F_t^{LAB} = \Phi_{RGB\to LAB}(F_t) \quad (1)$$

After transforming the frame F_t into LAB color space, the resulting representation is decomposed into luminance and chromatic components, i.e.,

$$F_t^{LAB} = (L_t, a_t, b_t) \quad (2)$$

where L_t denotes the luminance channel.

L_t is the luminance channel obtained by decomposing the LAB-transformed frame F_t^{LAB}.

This removes all ambiguity.

LAB separates brightness from color, where:
$L^* \to Lightness$ which ranges from 0 (black) to 100 (white)
$a^* \to Green \leftrightarrow Red\ axis$ Negative values are green and positive values are red.
$b^* \to Blue \leftrightarrow Yellow\ axis$ Negative values are blue and positive values are yellow.

So the color is represented as:

$$Color = (L^*, a^*, b^*) \quad (3)$$

LAB is used in the proposed system as illumination changes (shadows, sunlight, night) mainly affect L^* and the object color (vehicle body) is mostly preserved in a^* and b^*. So instead of using an RGB image, which distorts color, enhance only L^* using CLAHE, keep a* and b* unchanged, convert back to RGB. This gives better contrast, less color distortion, and more stable detection under lighting variation. For contrast enhancement in CLAHE, Adaptive Histogram Equalization is applied as:

$$L_t' = H(L_t) \quad (4)$$

where $H(.)$ is the CLAHE operator.

The enhanced frame is reconstructed as:

$$\hat{F}_t = \Phi_{RGB\to LAB}(L_t, A_t, B_t) \quad (5)$$

Gaussian smoothing is applied to reduce noise as follows:

$$\hat{F}_t = \hat{F}_t * G_\sigma \quad (6)$$

where G_σ is a Gaussian kernel with standard deviation σ

$$G_\sigma(x,y) = \frac{1}{2\Pi\sigma^2} exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \quad (7)$$

where (x, y) are the spatial coordinates.

Motion detection is achieved by modeling each pixel's temporal appearance using a Gaussian Mixture Model and classifying deviations from the dominant background distribution as foreground. Median filtering is applied to suppress noise and enforce spatial coherence in the resulting motion mask. Motion Detection via Background Subtraction, where the foreground mask generation can be done as follows:

$$M_t = \left|\hat{F}_t - B_t\right| \quad (8)$$

where B_t is the adaptive background model of Mixture of Gaussians, which is a probabilistic background modeling technique in which each pixel's intensity is represented by a weighted sum of Gaussian distributions to distinguish foreground motion from background variations. After background subtraction, the raw foreground mask M_t may contain noise. A median filter is applied to suppress impulse noise while preserving object boundaries.

$$\widetilde{M_t} = Median\ filter(M_t)$$

For a pixel location (x, y):

$$\widetilde{M_t}(x,y) = median\{M_t(i,j) \mid (i,j) \in N(x,y)\} \quad (9)$$

where $N(x, y)$ is a local neighborhood window.

The output preserves motion regions while removing salt-and-pepper noise.

For Motion BLOB extraction, the connected component analysis is applied to the filtered binary motion mask $\widetilde{M_t}$ to identify contiguous moving regions. Each connected component corresponds to a distinct motion blob. Components with an area below a predefined threshold A_{min} are discarded to suppress noise. For each remaining component, a tight axis-aligned BB, $b_i = (x_1^i, y_1^i, x_2^i, y_2^i)$ is extracted and used for subsequent occlusion and overlap analysis. Let the connected components in $\widetilde{M_t}$ be $C_t = \{c_1, c_2, \ldots, c_n\}$ where BB are extracted as $B_t = \{bi = (x_1^i, y_1^i, x_2^i, y_2^i)\}$

subject to $Area(c_i) > A_{min}$

The occlusion detection role is to automatically decide whether the current video frame contains occluded vehicles (e.g., vehicles partially blocking each other, dense traffic, or heavy motion clutter). This decision is later used to adapt the model's confidence threshold.

Occlusion is detected using two complementary cues:

1. Motion density in the frame
2. Overlap between motion BB

If either cue indicates congestion, the frame is considered occluded.

Motion density ratio at the time instant t is defined as the proportion of foreground (moving) pixels to the total number of pixels in the frame, and is calculated as:

$$\rho_t = \frac{\Sigma_{x,y}\ 1\,[\widetilde{M_t}(x,y) \neq 0]}{H \times W} \quad (10)$$

where ρ_t is Motion density ratio at frame t, $\widetilde{M_t}(x,y)$ is the Binary foreground mask obtained after background subtraction and filtering at pixel location (x,y), H and W are height and width of the video frame, respectively.

For two BBs, the Intersection over Union (IoU) is defined as the ratio between the area of their intersection and the area of their union.

$$IoU(b_i, b_j) = \frac{Area(b_i \cap b_j)}{Area(b_i \cup b_j)} \quad (11)$$

where $\boldsymbol{b_i} \cap \boldsymbol{b_j}$ denotes the overlapping region between the two BBs and $\boldsymbol{b_i} \cup \boldsymbol{b_j}$ denotes the total region covered by both BBs.

If $\rho_t > T_m$ where T_m is the maximum tolerable density before assuming occlusion.

There is a high probability of occlusion due to traffic congestion and multiple moving vehicles, which increases the probability of partial or full occlusion, so the frame is marked as occluded.

$$\exists\,(i,j) : IoU(b_i, b_j) > T_{iou} \quad (12)$$

where T_{iou} is the IoU threshold used to decide occlusion.

There exists at least one pair of BBs whose overlap exceeds the predefined IoU threshold. IF it is true then the occlusion is detected.

$$\rho_t \leq T_m \wedge \forall(i,j), IoU(b_i, b_j) \leq T_{iou} \quad (13)$$

The frame is not occluded because motion is low and no detected objects overlap significantly, as vehicles are well separated, No vehicle is blocking another, Detection conditions are favorable so the frame is classified as non-occluded. The occlusion state is set to 1 if the motion density exceeds a predefined threshold or if any pair of motion BB has an IoU greater than T_{iou} .

The proposed occlusion detection module combines global motion density and local BB overlap to robustly identify occluded frames. Motion density captures scene-level congestion, while IoU-based overlap captures object-level occlusion. This hybrid strategy enables adaptive confidence thresholding thus improving recall and mAP under dense traffic conditions without introducing additional learned parameters. Adaptive Confidence Thresholding T_t dynamically adjusts the minimum detection confidence required by the object detector based on scene conditions, specifically occlusion. Instead of using a fixed confidence threshold for all frames, the system adapts the threshold according to whether occlusion is present or not. The adaptive confidence threshold T_t at frame t is 1 if occlusion is detected and 0 if no occlusion is detected.

The proposed model formulates object detection as a single-stage regression and classification problem, where object localization and recognition are performed simultaneously in one forward pass of the network.

Model produces detections as:

$D_t = \{(b_k, c_k, s_k)\}$

where b_k is BB, c_k is the predicted class, $s_k \epsilon [0,1]$ confidence scores.

The detection condition is:

$$s_k \geq T_t \wedge c_k \epsilon\, C$$

Each valid detection is stored as:

$$R = \{t, c_k, b_k, s_k, o_t\}$$

where t is frame_id, c_k is class, $b_k$is the BB, s_k represents the model's confidence in the detection and o_t is the occlusion detector.

The computational performance metrics are:

Total Processing time (T_{total}) is calculated as:

$$T_{total} = t_{end} - t_{start} \quad (14)$$

where t_{start} is the time instant at which video processing begins, t_{end} is the time instant at which video processing terminates.

Average time per frame T_{avg} quantifies the expected computational cost of processing a single frame. It is a critical metric for evaluating frame-level efficiency, especially for video-based detection systems.

$$T_{avg} = \frac{T_{total}}{N_{processed}} \quad (15)$$

where T_{total} is total processing time, $N_{processed}$ is the number of video frames actually processed.

Each detection at frame t is defined as:

$$d_i^t = (b_i^t, s_i^t, c_i^t)$$

where $b_i^t = (x_i^t, y_i^t, w_i^t, h_i^t)$ is a BB,
$s_i^t \epsilon [0, 1]$ is the detection confidence,
c_i^t is the vehicle class

The detection set is: $Z_t = \{d_1^t, d_2^t, \ldots., d_{N_t}^t\}$

The COCO Dataset is used to facilitate precise VD. It groups 80 object categories into 12 super categories. Annotation Types in COCO dataset are Bounding Boxes, Segmentation Polygons, Object Class Labels, Crowd Annotations, Image-level Metadata (image ID, dimensions, license info, etc.).

Algorithm 1 for adaptive VD and environmental processing deals with occlusion by adjusting the thresholds for detection confidence.

Algorithm 1: Adaptive VD with Environment Preprocessing

Input: Video sequence V = {F_1 , F_2 , …, F_t },
Pretrained object detection model M,
Base confidence threshold τ_b,
Occluded confidence threshold τ_o
IoU threshold τ_{iou} ,
Motion area threshold τ_m, Vehicle class set C

Output: Annotated video $\hat{V}$
Detection record set R

1: Initialize background subtractor B
2: Initialize CLAHE operator H
3: Initialize empty record set $R \leftarrow \phi$
4: frame_id ← 0
5: for each frame $F_t \epsilon V$ do
6: frame_id ← frame_id + 1
7: $\hat{F}_t$ ← Preprocess(F_t ,H)8: $M_t \leftarrow B(\hat{F}_t)$ 9: $M_t \leftarrow Median\ filter(M_t)$10: $B_t \leftarrow ExtractBoundingBoxes(M_t)$11: $o_t \leftarrow DetectOcclusion(M_t, B_t, \tau_{iou}, \tau_m)$12: if o_t = TRUE then13: $\tau \leftarrow \tau_o$ 14: else15: $\tau \leftarrow \tau_b$ 16:end if17: $D_t \leftarrow M(\hat{F}_t, confidence = \tau)$18:for each detection $d \epsilon D_t$ do19: `if Class(d)` $\in$ `C then`20: Store (frame_id, class, BB, score, o_t) in R21: Draw BB and label on $\hat{F}_t$ 22: end if23:end for24: Append $\hat{F}_t$ to output video $\hat{V}$25:end for26:Save R to CSV file27:Return $\hat{V}$, R

In multi-object tracking, each object trajectory T_k is modeled as a dynamic system whose motion evolves over time. To capture both the current object location and its temporal dynamics, each track is represented using an 8-dimensional state vector.

Kalman Filter Prediction: Each track T_k maintains a motion state vector:

$$X_k^t = [x_k^t \;\; y_k^t \;\; a_k^t \;\; h_k^t \;\; x^{\cdot t}_k \;\; y^{\cdot t}_k \;\; a^{\cdot t}_k \;\; h^{\cdot t}_k]$$

where x_k^t , y_k^t are Center coordinates of the BB

Aspect ratio of the BB $a_k^t = \frac{w_k^t}{h_k^t}$ (16)

where h_k^t the Height of the BB, w_k^t is width of BB, $x^{\cdot t}_k$, $y^{\cdot t}_k$ are Velocities of the BB center, $a^{\cdot t}_k$ is Rate of change of aspect ratio, $h^{\cdot t}_k$ is the rate of change of height.

This allows the tracker to model both position and scale changes over time, which is crucial when vehicles move toward or away from the camera.

The Kalman filter prediction step estimates the state at frame t using the previous state:

$\hat{X}_k^t = F\, x_k^{t-1}$ (17)

where F is a constant-velocity transition matrix.

Each vehicle track is modeled using a constant-velocity state vector, enabling robust prediction of object position and scale during temporary occlusions through Kalman filter–based motion propagation.

The motion similarity via IoU between the predicted track T_k and detection d_i is:

$IoU_{(T_k,\ d_i)} = \frac{|B_k \cap B_i|}{|B_k \cup B_i|}$ (18)

The IoU based distance is:

$d_{IoU}(k,i) = 1 - IoU_{(T_k,\ d_i)}$ (19)

where $d_{IoU} > T_{iou}$ are discarded during association.

This IoU-based motion similarity ensures temporal continuity of object motion, reduces false associations caused by spatial inconsistency, and acts as the first gating stage before appearance-based matching. Algorithm 2 for occlusion-aware vehicle tracking makes occlusion more robust by explicitly modeling track survival with a maximum age limit, which ensures that identities are preserved across occluded frames.

Algorithm 2: Occlusion-Aware Vehicle Tracking

Input: Video sequence V = {F_1 , F_2 , …, F_t }
CSV-based detection set D = {D_1 , D_2 , . . ., D_T }
Maximum track age A_{max}, IoU threshold $\square_{iou}$,
Appearance distance threshold $\square_{app}$

Output: Annotated video $\hat{V}$, Track record set R

Initialize tracker with parameters (A_{max}, $\square_{iou}$, $\square_{app}$)
Initialize empty record set $R \leftarrow \phi$
frame_id ← 0
for each frame $F_t \epsilon\, V$ do
    frame_id ← frame_id + 1
    if D_t exists then
        $Z_t \leftarrow \{(b_i^t, s_i^t, c_i^t)\}$ for all detections in D_t
    else
        $Z_t \leftarrow \phi$
    end if
    Predict track states using Kalman filter
    Compute association cost using IoU-based motion similarity, Appearance-based cosine distance
    Assign detections to tracks via Hungarian optimization
    Update confirmed tracks and maintain identities
    for each confirmed track T_k^t do
        Extract BB b_k^t
        Determine occlusion state o_k^t from CSV
        Store (frame_id, ID_k, c_k, b_k^t, o_k^t) in R
        Draw BB and label on F_t
    end for
    Append annotated frame $\hat{F}_t$ to output video $\hat{V}$
end for
Save track record set R to CSV file
Return $\hat{V}$, R

To enhance data association robustness under occlusion and appearance ambiguity, each detected object is represented using a deep appearance feature extracted by a convolutional neural network (CNN). Each detection is encoded using a CNN appearance model as:

$$f_i^t \in R^d$$

where f_i^t denotes the d dimensional appearance embedding of the detected object, d is the feature dimensionality of the CNN embedding space.

Each active track T_k maintains an appearance descriptor f_k computed as the running average of previously associated detection embeddings.

The cosine distance is defined as:

$$d_{app}(k,i) = 1 - \frac{f_k \cdot f_i}{||f_k||\ ||f_i||} \quad (20)$$

where $f_k \cdot f_i$ is the dot product between feature vectors, $||\ .\ ||$ denotes the euclidean distance.

A detection-to-track association is considered valid if the appearance distance satisfies the following condition:

$$d_{app}(k,i) < T_{app}$$

where T_{app} is a predefined appearance similarity threshold. So the associations exceeding this threshold are rejected, thereby preventing identity switches caused by visually dissimilar objects.

To associate detections with existing tracks at the frame t, a joint cost function is defined that combines motion consistency and appearance similarity. For track T_k and detection d_i, the combined association cost is defined as:

$$C_{k,i} = \lambda \cdot d_{IoU}(k,i) + (1-\lambda) \cdot d_{app}(k,i) \quad (21)$$

where $d_{IoU}(k,i)$ is the IoU-based motion distance, $d_{app}(k,i)$ is the appearance-based cosine distance, $\lambda \in [0,1]$ is a weighting parameter controlling the relative importance of motion and appearance cues; optimal matching is solved using the Hungarian algorithm. When $\lambda \to 1$ it prioritizes motion consistency, if $\lambda \to 0$ it prioritizes appearance similarity.

The occlusion state of the track T_k at frame t is defined as $O_k^t = 1$ indicates that the object corresponding to the track T_k is occluded at t frame, $O_k^t = 0$ indicates normal visibility. The occlusion flag is obtained directly from the CSV-based detection annotations corresponding to the frame t. At each frame t tracker detects several objects. Each object is assigned a unique ID, a class label, a BB, an occlusion state. This information forms a record r_k^t. All records from a frame form R_t. Combining records from all frames produces the final set R. This final set R is what gets stored in the output CSV file.

For each confirmed track T_k at frame t, a tracking record is stored as:

$$r_k^t = (t, ID_k, c_k, x_1^t, y_1^t, x_2^t, y_2^t, o_k^t)$$

where t is the frame index, ID_k is the unique identity assigned to track k, c_k is the class label of the tracked vehicle, (x_1^t, y_1^t) top left corner of BB, (x_2^t, y_2^t) bottom-right corner of BB, o_k^t occlusion state of track k at frame t.

At each frame t multiple tracks may exist. The set of all track records at the frame t is:

$$R_t = \cup_{k=1}^{N_t} \quad r_k^t \quad (22)$$

where N_t is number of confirmed tracks at the frame t.

The final tracking record over the entire video is:

$$R_t = \cup_{t=1}^{T} \quad \cup_{k=1}^{N_t} \quad r_k^t \quad (23)$$

where T is the total number of frames, N_t is the number of tracks at the frame t, r_k^t is the record of the track k at frame t.

The proposed model enables robust VD, even under partial occlusion, illumination changes, and dense traffic scenes.

4 Results and Discussion

The various metrics are evaluated as follows:

$$P\ (Precision) = \frac{TP}{TP + FP} \quad (24)$$

where TP = True Positives, FP = False Positives

$$R\ (Recall) = \frac{TP}{TP + FN} \quad (25)$$

where FN = False Negatives

$$F1\ (F1\ score) = \frac{2 * P * R}{P + R} \quad (26)$$

Tracking time is measured using total processing duration and average frame time, while tracking accuracy is evaluated using standard MOT metrics such as Multiple Object Tracking Accuracy (*MOTA) and* Multiple Object Tracking Precision *(MOTP).*

$$MOTA = 1 - \frac{FN + FP + IDSW}{GT} \quad (27)$$

where $IDSW$ is ID switches, GT is the total number of true object instances

$$MOTP = \frac{\Sigma_{i,t}\ d_{i,t}}{\Sigma_t\ c_t} \quad (28)$$

where $d_{i,t}$ is a localization error between the predicted and ground truth box for the object i at frame t, c_t is number of correct matches at the frame t.

$$IDF1\ (ID\ F1\ score) = \frac{2 * IDTP}{2 * IDTP + IDFP + IDFN} \quad (29)$$

where $IDTP$ is a correct identity match, $IDFP$ is False identity matches, $IDFN$ is Missed identity matches, $IDF1$ measures how well object identities are preserved.

IDP measures how many predicted identities are correct. IDR measures how many true identities are correctly tracked.

$$IDP\ (ID\ Precision) = \frac{IDTP}{IDTP + IDFP} \quad (30)$$

$$IDR\ (ID\ Recall) = \frac{IDTP}{IDTP + IDFN} \quad (31)$$

The results of the proposed model are shown in Figure 2. During training, the loss components that were used are BB loss and classification loss. The BB loss measures localization accuracy. The box loss measures the difference between predicted and actual BB positions, helping the model learn accurate object localization. Thus, it measures how accurately the model predicts the location and size of the BB around an object. It compares predicted BB coordinates and Ground truth BB coordinates. The classification loss evaluates object class prediction. The total training loss is the combination of these components, enabling the model to learn both object localization and classification effectively. The classification loss evaluates how correctly the model predicts the object class and the class of the detected object. It compares the predicted class probability and the ground truth class label.

Table 1 shows the results of existing YOLO models, YOLOv5s, YOLOv8s, and YOLOv10s, with the proposed model after 20 epochs. In Figure 3 the graph for the same is shown in terms of P, R, mAP@50, mAP@50-95. The tracking metrics of the proposed model, compared with the existing YOLO modelsYOLOv5s, YOLOv8s, and YOLOv10s, are shown in Figure 4.

The proposed model has the highest metric values of mAP@50 as 98.1% and mAP@50-90 as 71.5%, where the P is 92.1%, R is 94% and F1-score is 90%. The tracker achieved a MOTA of 85.2% and an IDF1 score of 92.6% indicating strong identity preservation and stable multi vehicle tracking performance.

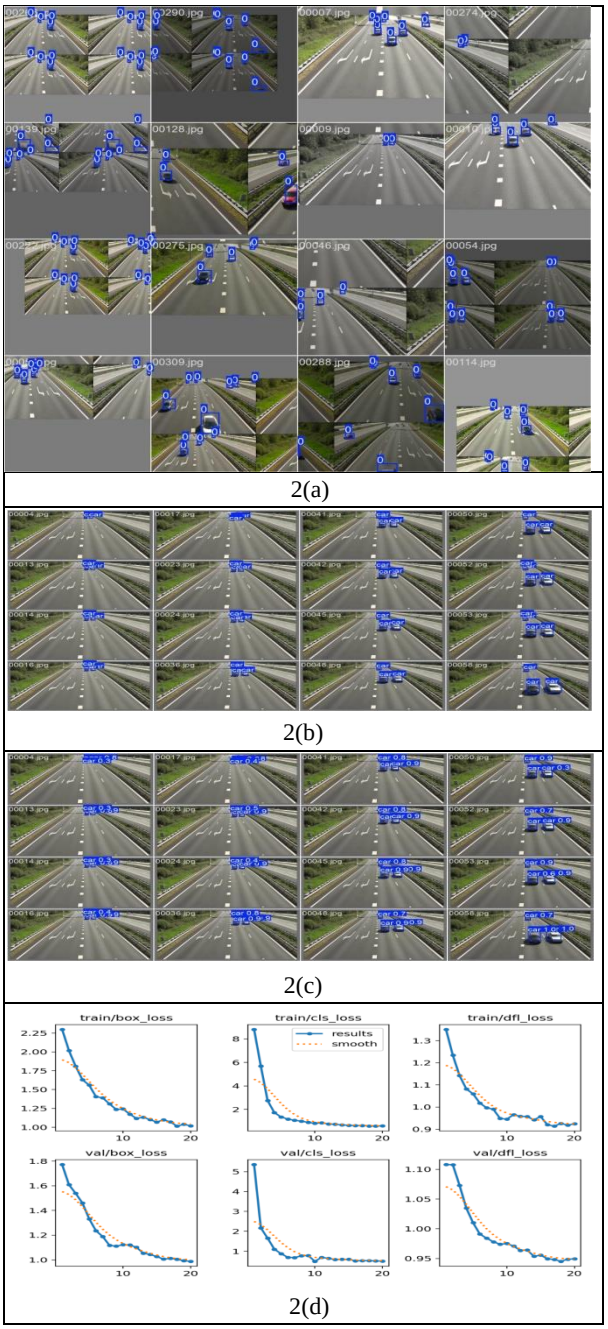

Fig. 2: Proposed model (a) train_batch0, (b) val_batch0_labels, (c) val_batch0_pred, (d) loss
Source: Created by the authors

Table 1. Results of Proposed model with recent approaches for VD

Model	P (%)	R (%)	F1 score (%)	mAP@50 (%)	mAP@50-95 (%)
YOLOv5s	83	85.8	85.9	96.6	64.6
YOLOv8s	83.6	88.3	88.1	97.5	67.6
YOLOv10s	86.6	93.9	88.9	97.6	69.3
Proposed Model	92.1	94	90	98.1	71.5

Source: Created by the authors

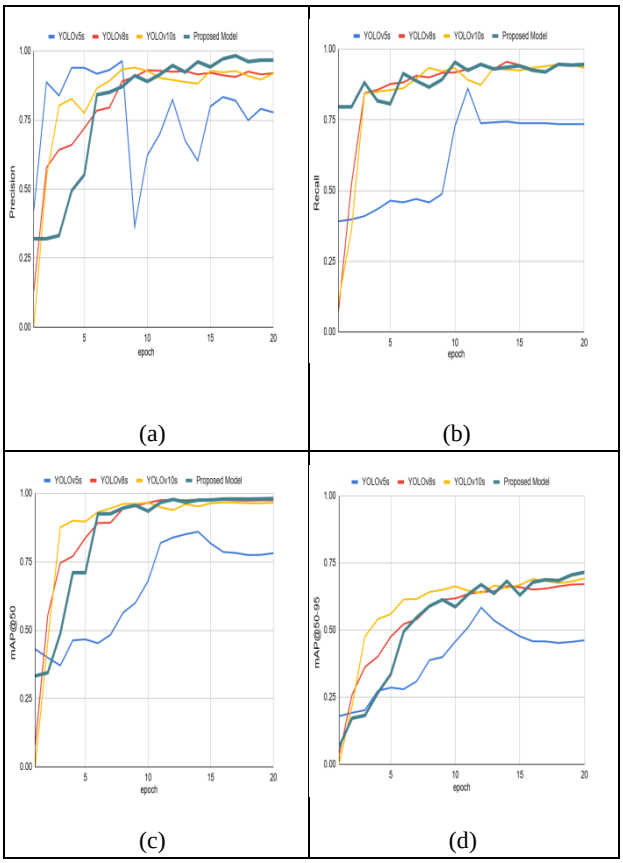

Fig. 3: Proposed model compared with the existing YOLO models YOLOv5s, YOLOv8s, YOLOv10s after 20 epochs in terms of (a)Precision, (b)Recall, (c)mAP@50, (d)mAP@50-95

Source: Created by the authors

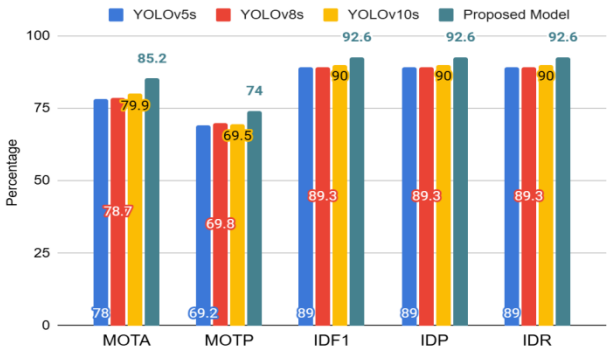

Fig. 4: Comparative analysis of Proposed model with existing YOLO models YOLOv5s, YOLOv8s, YOLOv10s

Source: Created by the authors

5 Conclusion

This paper presents a framework for occlusion-aware VD and tracking that integrates environmental preprocessing, adaptive deep learning-based detection, occlusion handling and multi-object tracking. In the proposed model the video frames are enhanced using contrast normalization and noise reduction, followed by background subtraction to identify motion regions and estimate occlusion. Based on the occlusion state, the detection confidence threshold is dynamically adjusted to improve the detection of partially visible vehicles using our model. The resulting detections are then processed by a tracker, which associates objects across frames using Kalman filter based motion prediction and appearance similarity, with assignment solved via the Hungarian algorithm. Occlusion is handled explicitly by maintaining tracks for a specified duration and recording occlusion states. The model outputs annotated video and structured detection and tracking logs which enables robust performance under varying illumination, congestion and occlusion conditions while maintaining efficient processing.

References:

[1] X. Liu & N. Rodelas, "Vehicle Detection and Tracking Techniques Based on Deep Learning in Road Traffic Surveillance," Academic Journal of Science and Technology, vol. 10, pp. 125–129, 2024. https://doi.org/10.54097/86akec97.

[2] E. Ukey and S. L. Varma, "Review on Evolution of Vehicle Detection Methods in Computer Vision for Surveillance," in 2nd International Conference on Emerging Trends in Engineering and Medical Sciences (ICETEMS), Nagpur, India, 2024, pp. 1–6. https://ieeexplore.ieee.org/document/10965053.

[3] A. Jazayeri, H. Cai, J. Y. Zheng, & M. Tuceryan, "Vehicle Detection and Tracking in Car Video Based on Motion Model," IEEE Trans. on Intell. Transp. Syst., vol. 12, no. 2, pp. 583–595, Jun. 2011. [Online]. https://doi.org/10.1109/TITS.2011.2113340.

[4] I. Ogunrinde & S. Bernadin, "Improved DeepSORT-Based Object Tracking in Foggy Weather for AVs Using Semantic Labels and Fused Appearance Feature Network," Sensors, vol. 24, no. 14, Art. no. 4692, 2024. https://doi.org/10.3390/s24144692.

[5] D. P. Carrasco, H. A. Rashwan, M. Á. García, & D. Puig, "T-YOLO: Tiny Vehicle Detection Based on YOLO and Multi-Scale Convolutional Neural Networks," IEEE Access, vol. 11, pp. 22430–22440, 2023. https://doi.org/10.1109/ACCESS.2021.3137638.

[6] Z. Wu, F. Li, Y. Zhu, K. Lu, & M. Wu,

"Design of a robust system architecture for tracking vehicle on highway based on monocular camera," Sensors, vol. 22, no. 9, Art. no. 3359, 2022. https://doi.org/10.3390/s22093359.

[7] M. Alonazi, A. Qureshi, S. Alotaibi, N. Almujally, N. Almudawi, A. Alazeb, A. Jalal, J. Kim, & M. Min, "A Smart Traffic Control System Based on Pixel-Labeling and SORT Tracker," IEEE Access, vol. 11, pp. 80973-80985, 2023. https://doi.org/10.1109/ACCESS.2023.3299488.

[8] W. Li, Z. Wang, X. Wu, J. Zhang, Q. Peng, & H. Li, "CODAN: Counting-driven Attention Network for Vehicle Detection in Congested Scenes," in Proc. 28th ACM Int. Conf. Multimedia (MM), Seattle, WA, USA, Oct. 2020, pp. 73–82. https://doi.org/10.1145/3394171.3413945.

[9] Y. Miao, F. Liu, T. Hou, L. Liu, and Y. Liu, "A Nighttime Vehicle Detection Method Based on YOLO v3," Chinese Automation Congress (CAC), Shanghai, China, Nov. 2020, pp. 6617–6621. https://doi.org/10.1109/CAC51589.2020.9326819.

[10] C. Y. Wang, A. Bochkovskiy, & H.Y. M. Liao, "YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Vancouver, BC, Canada, Jun. 2023, pp. 7464–7475. https://doi.org/10.1109/CVPR52729.2023.00721.

[11] J. Liu, Y. Xie, Y. Zhang, & H. Li, "Vehicle flow detection and tracking based on an improved YOLOv8n and ByteTrack framework," World Electric Vehicle Journal, vol. 16, no. 1, Art. no. 13, Jan. 2025. https://doi.org/10.3390/wevj16010013.

[12] H. Gao, "Vehicle Detection and Tracking Based on YOLOv11," in Proc. of the 2nd Int. Conf. on Data Science and Engineering (ICDSE 2025), Lisbon, Portugal, Apr. 23–25, 2025, pp. 481–486. SCITEPRESS – Science and Technology Publications, ISBN: 978-989-758-765-8.

[13] S. Ay & M. Karabatak, "A new automatic vehicle tracking and detection algorithm for multi-traffic video cameras," *Traitement du Signal*, vol. 40, no. 2, pp. 457–468, 2023. https://doi.org/10.18280/ts.400205.

[14] R. Li, "A multi-stage deep learning approach for real-time vehicle detection, tracking, and speed measurement in intelligent transportation systems," Sci. Rep., vol. 15, Art. no. 22531, 2025. https://doi.org/10.1038/s41598-025-07343-5.

Contribution of Individual Authors to the Creation of a Scientific Article (Ghostwriting Policy)

The authors equally contributed in the present research, at all stages from the formulation of the problem to the final findings and solution. No ghostwriting occurred.

Sources of Funding for Research Presented in a Scientific Article or Scientific Article Itself

The authors declare that no dedicated external funding was received for this research.

Conflict of Interest

The authors have no conflicts of interest to declare that are relevant to the content of this article.