Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.084054

ARTICLE

Aerial Deep Learning Framework for Vehicle Trajectory Extraction and Data-Driven Urban Transportation Networks Analysis

Rimsha Tayyab[1], Muhammad Hanzla[1], Maha Abdelhaq[2], Raed Alsaqour[3], Mohammed Alonazi[4], Ahmad Jalal[1,5,*] and Hui Liu[6,*]

[1]Department of Computer Science, Air University, Islamabad, Pakistan
[2]Department of Information Technology, College of Computer and Information Sciences, Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia
[3]Department of Information Technology, College of Computing and Informatics, Saudi Electronic University, Riyadh, Saudi Arabia
[4]Department of Information Systems, College of Computer Engineering and Sciences, Prince Sattam bin Abdulaziz University, Al-Kharj, Saudi Arabia
[5]Department of Computer Science and Engineering, College of Informatics, Korea University, Seoul, Republic of Korea
[6]Guodian Nanjing Automation Co., Ltd., Nanjing, China

*Corresponding Authors: Ahmad Jalal. Email: ahmadjalal@mail.au.edu.pk; Hui Liu. Email: hui.liu198510@gmail.com

Received: 15 April 2026; Accepted: 23 June 2026; Published: 22 October 2026

ABSTRACT: Urban transportation networks, typically represented as complex networks of interconnected roads and intersections, have become increasingly vulnerable to congestion propagation, cascading failures, and disturbances due to rapid urbanization. However, conventional static sensor systems provide limited coverage and are difficult to scale, restricting the ability to analyze traffic dynamics and their impact on network integrity. Unmanned Aerial Vehicles (UAVs) offer a flexible solution for large-scale aerial vehicle monitoring, but challenges such as small vehicle sizes, motion blur, occlusions, and diverse urban environments reduce the effectiveness of existing approaches. This paper presents a comprehensive deep learning-based UAV monitoring framework. Using the VisDrone2019 dataset, the proposed framework incorporates preprocessing through Gaussian filtering and Mixture of Gaussians background subtraction, instance segmentation using Mask R-CNN for precise object boundary delineation, and the YOLOv11 algorithm for accurate object detection, achieving AP50 = 38.6% and mAP = 22.8%. The framework further enables robust multi-object tracking through ORB feature matching and ByteTrack algorithms, attaining MOTA = 52.3% and IDF1 = 58.7%. In addition, it supports vehicle counting and trajectory prediction using Long Short-Term Memory (LSTM) networks, providing spatiotemporal insights into vehicle movements. The extracted trajectories are transformed into a spatiotemporal traffic graph, enabling bottleneck identification through betweenness centrality, graph degree distribution analysis, and congestion propagation assessment over time. Consequently, this work offers valuable opportunities for complex network analysis, including network topology characterization, critical node identification, congestion formation analysis, and network resilience evaluation.

KEYWORDS: Aerial vision; LSTM trajectories; aerial surveillance; VisDrone; ByteTrack; complex networks; intelligent transportation systems; traffic dynamics

1 Introduction

Vehicle detection and tracking in aerial imagery has become increasingly critical for intelligent transportation systems (ITS), urban traffic monitoring, and smart city applications [1]. Image processing techniques such as background subtraction and blob detection have been explored for vehicle detection

Copyright © 2026 The Authors. Published by Tech Science Press. This work is licensed under a Creative Commons Attribution 4.0 International License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

in early traffic monitoring systems [2], though these methods were primarily designed for fixed cameras and lack robustness in dynamic aerial scenarios. UAVs offer a flexible, wide-area aerial perspective for urban traffic analysis, providing high-altitude coverage of road networks without the physical constraints of ground-based sensors, particularly valuable for identifying traffic bottlenecks and supporting data-driven mobility management [3,4]. However, small vehicle sizes, frequent occlusions, scale variations, scene clutter, and dynamic backgrounds render conventional computer vision techniques inadequate, necessitating sophisticated deep learning solutions capable of managing dense urban traffic environments [5,6]. Critical challenges, including background removal, shadow interference, high vehicle density, and information loss under occlusion, further limit system reliability [7,8]. Accurate vehicle trajectory extraction from aerial scenes additionally supports urban traffic flow analysis, congestion behavior modeling, and network-level transportation resilience, strongly aligned with modern smart city infrastructures [9,10]. The specific contributions of the proposed research are as follows:

- **UAV-Specific YOLOv11 Integration:** The specific technical contributions of this work extend beyond module combination. ORB is repurposed as a pre-MOG UAV ego-motion stabilizer via RANSAC homography estimation (C1), operating in two procedurally separated roles: Stage A (global frame stabilization before background modeling) and Stage B (post-segmentation geometric correspondence within Mask R-CNN ROI boundaries) (C2). YOLOv11's anchor-free detections replace Mask R-CNN's native RPN entirely, augmented with three UAV-specific adaptations: RPN-bypass cascade coupling, sub-32 × 32 scale-aware Mosaic + MixUp augmentation, and dual-threshold ByteTrack calibration (T_{high} = 0.6, T_{low} = 0.1), d specific to aerial traffic density none present in prior YOLOv11 deployments (C3). The framework introduces the first perception-to-traffic-graph pipeline converting MOT outputs into a betweenness-centrality-analyzable spatiotemporal network (C4), with sensitivity analysis confirming stable centrality rankings (Kendall's T = 0.92) above mAP = 18.4%, and degradation below mAP = 14.2% establishing a minimum perception quality threshold for reliable ITS deployment (C5).
- **Enhanced Spatiotemporal Traffic Analysis:** The framework effectively integrates spatial and temporal information to improve object localization, stable identity preservation, and vehicle motion understanding in dynamic urban traffic scenes.
- **Robust Performance in Complex Urban Environments**: Experimental evaluation on challenging datasets such as VisDrone2019 demonstrates strong robustness against occlusion, scale variation, motion blur, and dense traffic conditions, supporting reliable ITS deployment.
- **Scalable Smart City Applicability:** The developed system is scalable for deployment across multiple UAV platforms and can support traffic monitoring, congestion analysis, emergency response, and smart city transportation management.

2 Literature Review

UAV-based aerial traffic monitoring has gained significant attention for intelligent transportation and smart city applications due to its flexible deployment and wide-area coverage. However, challenges such as small object size, camera motion, occlusion, and background clutter still affect reliability. Despite extensive research, existing methods often lack scalability and robustness. Therefore, this section reviews the literature and identifies key research gaps.

2.1 Vehicle Detection and Tracking Systems

Aerial traffic monitoring demands robust vehicle detection and tracking systems. Early approaches relied on background subtraction and blob detection for stationary cameras Bhaskar and Yong [2], proving inadequate for dynamic UAV motion and high-traffic occlusion scenarios. Chen and Meng [3] advanced this

using handcrafted features (FAST, HoG) with forward-backward tracking, improving stability but limiting detection performance. Mujtaba et al. [4] introduced pixel labeling, particle filtering, and georeferencing for group vehicle tracking, yet remained vulnerable to long-term occlusions, motion blur, and identity switches in crowded scenes.

In addition, Wang et al. [5] proposed image registration and feature extraction for speed and trajectory estimation but required precise camera calibration and were susceptible to background distortion from UAV movement. As deep learning advanced, CNN-based detectors significantly improved detection accuracy. Hamzenejadi and Mohseni [7] optimized YOLOv5 network depth and anchor parameters for small aerial vehicle detection, though at high computational cost with reduced accuracy in dense scenes. Mustafa and Alizadeh [8] integrated attention mechanisms into YOLOv4, enhancing feature discrimination under moderate occlusion, yet boundary estimation errors persisted in dense traffic due to bounding-box constraints. YOLOv8 demonstrated improved speed-accuracy trade-offs for aerial vehicle tracking by Bakirci [9], though most available methods prioritize detection with weak tracking refinement. Subsequent works, such as Cong et al. [11] and Liu et al. [12], extend ByteTrack specifically to address aerial occlusion challenges. However, both the original formulation and its derivatives remain sensitive to upstream detection quality, since faulty bounding boxes directly propagate into tracking errors through the IoU cost matrix. The original ByteTrack method by Zhang significantly advances multi-object tracking by associating every detection box, including low-confidence ones, through a two-stage IoU pipeline, substantially enhancing association stability and minimizing identity switches under occlusion.

2.2 Vehicle Detection, Segmentation, Counting, and Trajectory Prediction Systems

While vehicle detection remains paramount, segmentation, counting, and tracking is equally critical for comprehensive traffic intelligence. Transformer-based methods like Mask2Former Cheng et al. [13] achieve precise instance segmentation in complex scenarios but demand prohibitive computational resources for real-time UAV deployment. Mask R-CNN offers a practical balance between boundary precision and efficiency, particularly for overlapping vehicles, though its added latency limits are adopted in motion tracking pipelines. For vehicle counting, DBSCAN-based centroid clustering Sun et al. [14], performs reliably in sparse scenes but misclassified grouped vehicles traveling in proximity. Optical flow provides pixel-level motion estimation but remains susceptible to UAV vibrations, causing false motion detections. LSTM-based trajectory prediction Cheng effectively models nonlinear vehicle dynamics, yet most implementations target ground-based video, failing to account for scale variation and aerial noise characteristic of UAV footage. Critically, existing works predominantly address individual sub-problems in isolation; few propose integrated pipelines encompassing preprocessing, detection, segmentation, tracking, counting, and trajectory prediction within a unified aerial framework.

No prior work addresses all five dimensions simultaneously. YOLOv11, Mask R-CNN, ByteTrack, and LSTM are established methods; their specific architectural coupling motion-compensated MOG, dual-role ORB, RPN-bypass cascade, and perception-to-network graph construction constitute methodological contributions collectively absent from all cited works on VisDrone2019.

3 Proposed Methodology

The proposed framework is a modular, deployment-aware multi-stage aerial traffic intelligence pipeline integrating classical computer vision with advanced deep learning to address key UAV challenges: small object size, high vehicle mobility, occlusion, and complex urban environments. The pipeline spans preprocessing, background modeling, detection, segmentation, feature matching, tracking, and trajectory prediction, with outputs supporting spatiotemporal traffic flow analysis and ITS applications (Fig. 1).

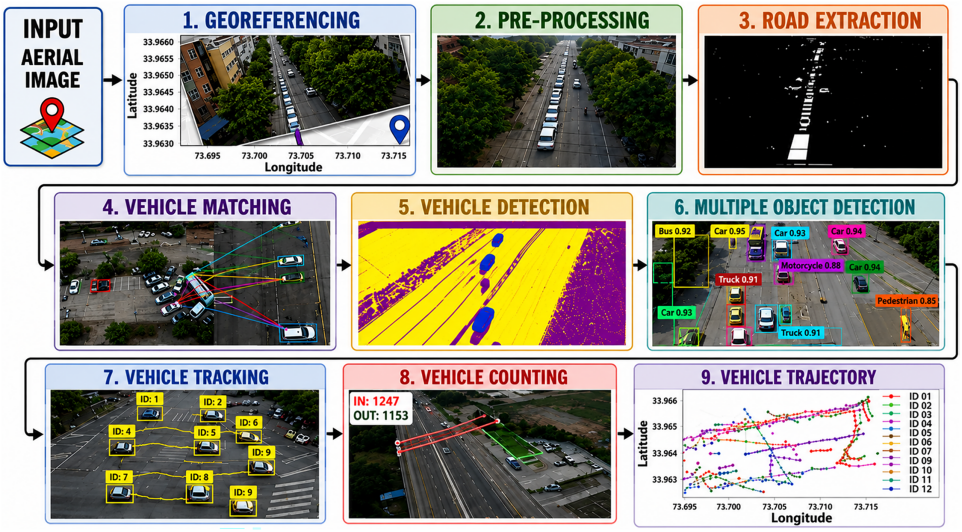


Figure 1: Integrated aerial perception pipeline that transforms raw UAV imagery into high-fidelity spatiotemporal vehicle trajectories and traffic flow data for urban transportation networks.

Two operational modes are supported: full geospatial mode for GPS-enabled UAV platforms providing KML waypoint files, GCPs, and georeferenced imagery, and vision-only mode for benchmark evaluation where geospatial metadata is unavailable. All VisDrone2019 experiments operate in vision-only mode commencing from Gaussian preprocessing, since VisDrone2019 provides no KML, GCP, or GPS-tagged imagery, ensuring the framework is simultaneously benchmarkable on standard academic datasets and deployable in real-world ITS scenarios.

3.1 Geo-Referencing

The Georeferencing connects each UAV frame to its precise geographic location using Ground Control Points (GCPs) extracted from KML files, transforming image coordinates (x, y) to map coordinates (X, Y), which is established through the following homography transformation in Eq. (1).

$$\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = \frac{1}{h_{31}x + h_{32}y + h_{33}} \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{1}$$

The homography matrix $H = [h_{ij}]$ defines the transformation between the image plane and ground reference plane, with values computed from GCP pairs extracted from KML files. Coordinate information is embedded in UAV images and stored in GeoTIFF format, preserving both visual and geospatial data for seamless integration with GIS tools (ArcGIS, QGIS), supporting traffic network topology analysis. Fig. 2 shows the resulting georeferenced aerial image mapped to real-world coordinates for precise vehicle localization. This module activates exclusively on survey-grade GPS-enabled platforms (e.g., DJI Phantom 4 RTK, DJI Matrice 300 RTK), providing KML waypoint files and GCP survey data. For datasets lacking geospatial metadata, including VisDrone2019, this module is bypassed entirely, with the pipeline proceeding directly to Gaussian preprocessing (Section 3.2). All detection, segmentation, tracking, and trajectory prediction modules (Sections 3.2–3.9) operate independently of this stage and require no geospatial input.

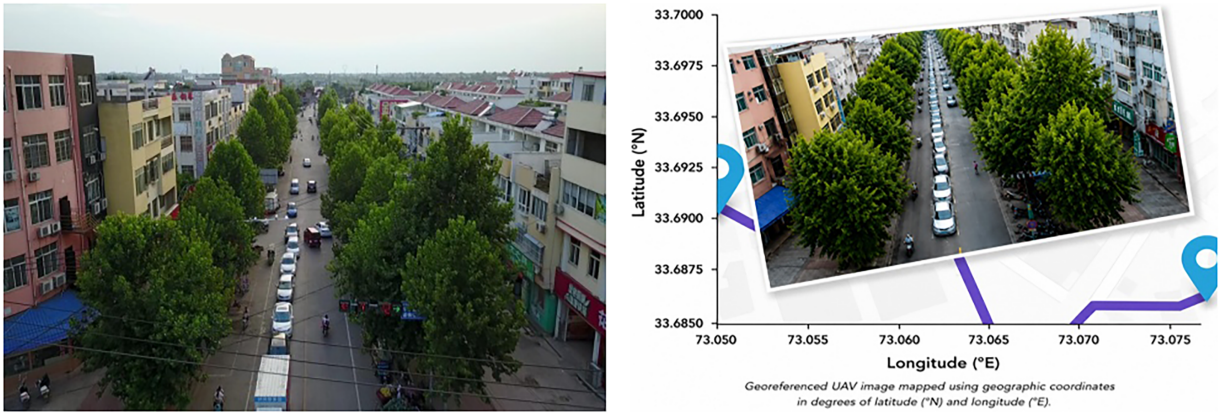


Figure 2: Geo-referenced image aligned using KML-based ground control points.

3.2 Preprocessing Using Gaussian Filtering

Gaussian filtering suppresses sensor noise and environmental artifacts in raw UAV imagery while preserving structural features. The filter assigns spatially weighted averages using a probability distribution greater weight to central pixels and lesser to distant ones, as defined in Eq. (2).

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{2}$$

where x, y indicate the pixel location, and σ control smoothing level. The variable exp represents the exponential function, and $2\pi\sigma^2$ represents the normalization factor. This function is applied using the convolution process, in which the value of each pixel is calculated using the weighted sum of the neighboring pixels according to the Gaussian function, as given in Eq. (3).

$$R(x, y) = \sum_{i,j} G(i, j) \cdot S(x - i, y - j) \tag{3}$$

where $R(x, y)$ is the refined signal, and S represents the original pixel values in the kernel window. The symmetric Gaussian kernel ensures uniform smoothing in all directions, minimizing edge distortions compared to alternative filters while maintaining computational stability as demonstrated in Fig. 3.

3.3 Background Elimination Using MOG

Background elimination via the Mixture of Gaussians (MOG) model requires a stationary background assumption that is fundamentally violated in UAV settings, where continuous camera ego-motion causes MOG to incorrectly classify the entire background as foreground. The proposed pipeline addresses this through an explicit pre-MOG global motion compensation (GMC) stage entirely separate from georeferencing (Section 3.1), in which ORB keypoints from consecutive frames I_t and I_{t-1} are matched via Hamming distance minimization. The resulting keypoint pairs estimate a global homography H_t via RANSAC-based robust fitting (Eq. (4)), warping I_{t-1} into alignment with I_t It before MOG is applied, ensuring the background model receives a stabilized input where only independently moving vehicles produce foreground responses.

$$\hat{H}_t = \arg\min_H \sum_i \rho(\left|\left|p_i^t - H \cdot p_i^{t-1}\right|\right|^2) \tag{4}$$

where p_i^t, p_i^{t-1} are matched keypoint coordinates and $\rho(\cdot)$ is the Huber robust loss function that down-weights outliers caused by independently moving vehicles. H_t warps I_{t-1} into the coordinate space of I_t:

$$\hat{I}_{t-1} = W(I_{t-1}, H_t) \tag{5}$$

where W denotes the bilinear warp operator. MOG is subsequently applied to the stabilized input I_{t-1}, ensuring residual pixel differences correspond exclusively to independently moving vehicles rather than camera motion. This motion-compensated MOG formulation is defined as Eq. (6).

$$B(x, y, t) = \sum_{i=1}^{N} w_i(t).N(x, y; \mu(t), \sum_i(t)) \tag{6}$$

where $B(x, y, t)$ denotes the background model at time t, N the number of Gaussian components, $w_i(t)$ the weight of the i component, and $N(x, y; \mu(t), \sum(t))$ a Gaussian distribution with mean $\mu(t)$ and covariance $\sum_i(t)$. Fig. 4 demonstrates accurate vehicle background separation.



Figure 3: Gaussian smoothing effect on aerial imagery for a complex network.

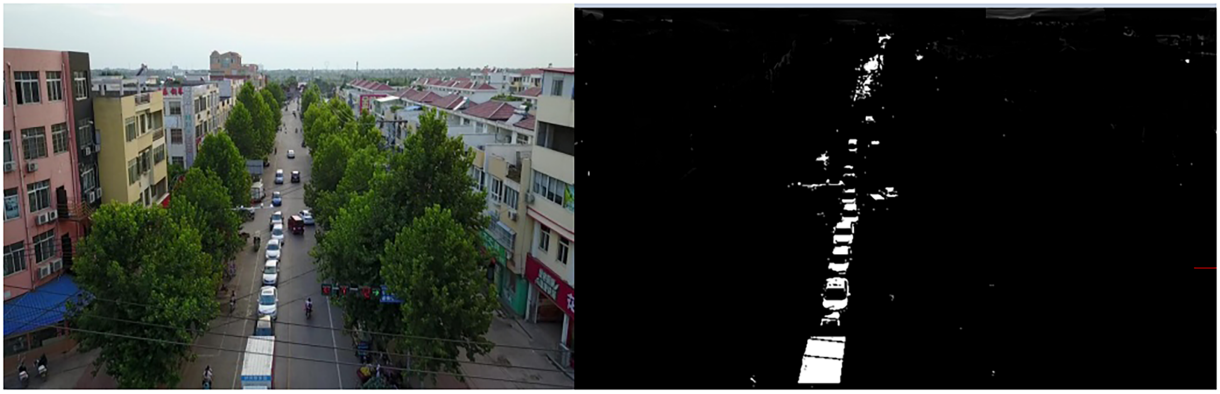


Figure 4: Showing moving vehicles from the static background.

3.4 Vehicle Detection Using YOLOv11 Framework

YOLOv11 is an advanced real-time object detection architecture designed to offer high accuracy and efficient inference in a range of scenarios, particularly in UAV-based aerial imagery, where motion blur, heavy traffic, and small-object detection present significant challenges. The model enhances the traditional

YOLO pipeline by adding a stronger backbone, an optimized multi-scale feature fusion neck, and a decoupled detection head that separates localization and classification for more stable convergence. Using CSP-enhanced convolutional blocks, the backbone extracts rich multi-resolution features, producing feature maps at scales P3, P4, and P5, shown in Fig. 5.

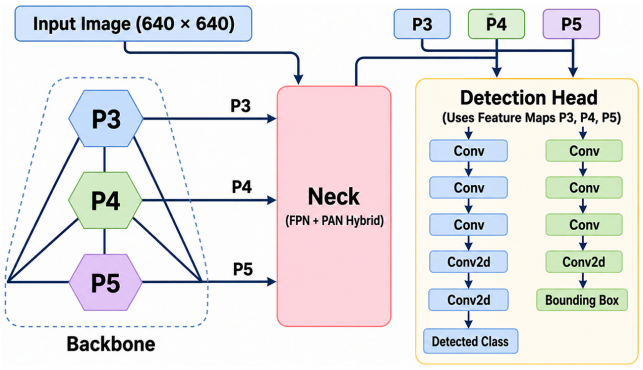


Figure 5: Illustration of the multi-scale YOLOv11 detection architecture.

A hybrid FPN-PAN neck network is used to fuse spatial and semantic features across different levels, which helps to improve the detection of small and overlapping objects like vehicles. The objectness prediction is computed as defined in Eq. (7).

$$P_{obj} = \sigma(f_{obj}(F_i)) \tag{7}$$

where P_{obj} is the probability of object existence in the predicted region, σ stands for the sigmoid activation function, f_{obj} is the object prediction function, and F_i for the feature vector from the feature map. YOLOv11 employs an anchor-free detection head that directly regresses bounding box coordinates from grid cell offsets without predefined anchor priors, improving small-object detection in aerial scenes. The predicted center coordinates are given by Eq. (8):

$$b_x = x_{cell} + \sigma(\Delta x), b_y = y_{cell} + \sigma(\Delta y) \tag{8}$$

$$w_{pred} = e^{\Delta w}, h_{pred} = e^{\Delta h} \tag{9}$$

where x_{cell} and y_{cell} are the grid cell center, σ is the sigmoid activation, and (Δx, Δy, Δw, Δh) are raw network output offsets. No anchor prior are used. The class probability is calculated as Eq. (10):

$$P_{(c|obj)} = \frac{e^{zc}}{\sum_{k=1}^{K} e^{zc}} \tag{10}$$

where $P_{(c|obj)}$ is the conditional probability of class c given the presence of an object, and e^{zc} is the classification prediction function for the feature vector K is the total number of classes. Training is guided by a composite loss function defined in Eq. (11):

$$L = \lambda_1 L_{CIoU} + \lambda_2 L_{cls} + \lambda_3 L_{dfl} \tag{11}$$

where L is the total training loss. l_{GIoU} stands for the Generalized Intersection over Union loss used for precise bounding box regression. The λ_1, λ_2, and λ_3 coefficients are used for weighing each component of the total loss. L_{CIoU} is the Complete IoU regression loss for precise bounding box alignment, L_{cls} is the focal classification loss, and L_{dfl} is the Distribution Focal Loss for accurate boundary prediction. Unlike

older YOLO versions, YOLOv11's decoupled head removes the objectness loss. Training parameters are detailed in Table 1. Fig. 6 demonstrates the successful detection of all vehicle classes under dense aerial traffic conditions.

Table 1: Parameter configuration table.

Parameter	Value Used	Purpose/Explanation
Input Resolution	640 × 640	Balanced trade-off between accuracy and inference speed
Batch Size	16	Suitable for stable gradient updates
Optimizer	AdamW	Better generalization and weight regularization
Learning Rate	0.01	Default optimal LR for convergence stability
Epochs	300	Sufficient training time for UAV datasets
Confidence Threshold	0.25	Reduces low-quality detections
NMS IoU Threshold	0.45	Maintains separation between nearby vehicles
Loss Function	CIoU + (DFL)	Precise box regression without objectness branch
Dropout	0.1	Prevents overfitting on small object categories
Model Variant	YOLOv11-S (Small)	Maximizes FPS while retaining good accuracy
Activation Function	SiLU	Smooth, stable gradients
Pretrained Weights	COCO Pretrained	Accelerates UAV-specific fine-tuning

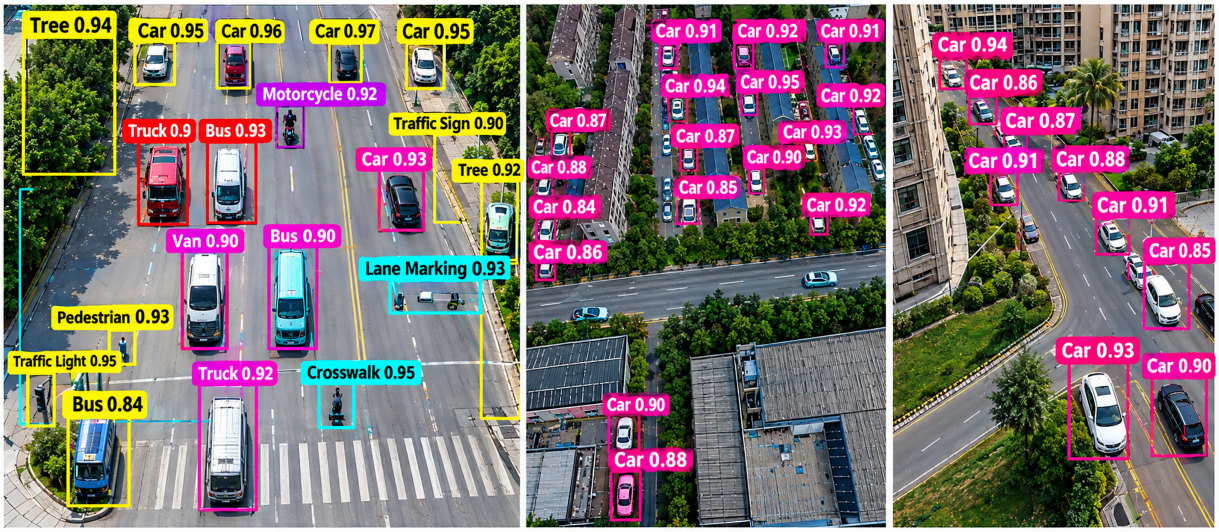


Figure 6: Multiple object detection using a YOLOv11 framework.

3.5 Instance Segmentation Using Mask R-CNN

The proposed system employs Mask R-CNN for pixel-level vehicle boundary delineation in a YOLOv11-cascaded RPN-bypass design. Rather than processing full aerial frames, Mask R-CNN receives only YOLOv11-confirmed bounding boxes bi as ROI proposals, each cropped with 8 px padding from frame F_k, and processed via RoI Align to extract fixed-size feature representations. Predicted masks are reprojected to original frame coordinates, keeping computational cost $O(M \times s_{roi}^2)$, rather than $O(M \times W)$. The mask branch optimizes pixel-wise binary cross-entropy in Eq. (12).

$$L_{mask} = -\frac{1}{m^2}\sum_{i=1}^{m}\sum_{j=1}^{m}[y_{ij}\log(y_{ij}) + (1-y_{ij})\log(1-\hat{y}_{ij})] \tag{12}$$

where L_{mask} mask is the mask loss, y_{ij} and $\hat{y}_{ij}$ denote ground-truth and predicted pixel values at position (i, j) within the $m \times m$ mask. The unified multi-task loss in below Eq. (13):

$$L_{total} = L_{RPN} + L_{bbox} + L_{mask} \tag{13}$$

where L_{total}, L_{RPN}, L_{bbox} represent region proposal, bounding box regression, and segmentation losses, respectively. Fig. 7 demonstrates accurate vehicle instance separation in dense aerial scenes.

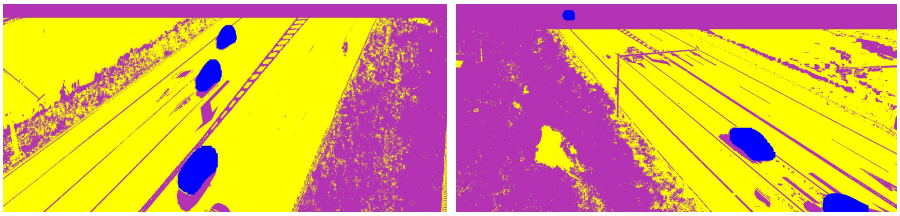


Figure 7: Showing precise vehicle contours extracted from aerial images.

3.6 Feature Matching Using Oriented FAST and Rotated BRIEF (ORB)

The proposed system employs ORB for keypoint correspondence between consecutive UAV frames, combining FAST detection with BRIEF description for computational efficiency. Feature matching is performed via Hamming distance in Eq. (14).

$$D(p,q) = \sum_{i=1}^{N}(p_i \oplus q_i) \tag{14}$$

where p, q are binary descriptors, p_i, q_i are their i^{th} bits, and $\oplus$ denotes XOR. Keypoint orientation is computed as Eq. (15):

$$\theta = \tan^{-1}\left(\frac{\sum_{x\varepsilon p} yI(x,y)}{\sum_{x\varepsilon p} xI(x,y)}\right) \tag{15}$$

In the above equation, θ represents the keypoint orientation, and $I(x, y)$ represent the pixel intensity at the location (x, y) within the surrounding area of the keypoint. Ensuring rotation invariance is essential for dynamic UAV viewpoints. ORB serves two distinct roles in the pipeline: Stage A (Section 3.3) performs global homography-based ego-motion compensation before MOG; Stage B (this section) performs post-segmentation inter-frame geometric verification of tracked vehicle regions within Mask R-CNN ROI boundaries. Feature matching results are shown on Fig. 8.

3.7 Vehicle Tracking Using ByteTrack

In order to guarantee stable vehicle tracking in complex aerial scenes, the proposed system employs ByteTrack, which associates every detection box, including low-confidence detections discarded by conventional trackers, through a two-stage hierarchical IoU-based pipeline. ByteTrack maintains tracklets T = {T1, ..., Tn} and a detection set D = {D1, ..., Dm} for each frame. Detections are divided into high-confidence set D^{high} and low-confidence set D^{low} using two thresholds T^{high} and T^{low}:

$$D^{high} = \left\{D_i | s_i \geq T_{high}\right\}, D^{low} = \left\{D_i | T_{low} < s_i < T_{high}\right\} \tag{16}$$

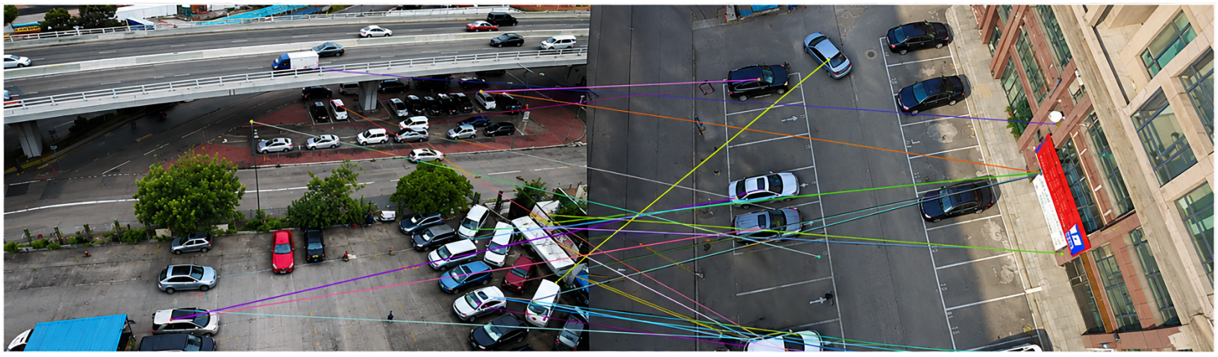


Figure 8: Matched keypoints across frames showing motion paths.

First association high-score detections are matched to active tracklets using IoU between Kalman-predicted track let positions $\hat{T}_j$ and detected boxes D_i^{high}:

$$Cost_{ij} = 1 - IoU(\hat{T}_j, D_i^{high}) \tag{17}$$

The Hungarian algorithm is applied to minimize total assignment cost. Second association unmatched tracklets from the first stage are matched against D^{low} using IoU again, recovering occluded or briefly lost objects. Tracklet state transitions of the object from the 'New' to the 'Tracked,' 'Lost,' and then 'Removed' states after 30 frames of inactivity. No appearance embedding is involved. We also clarify that ORB feature matching (Section 3.6) serves a separate geometric registration purpose and is not part of the ByteTrack association step,

$$A^* = \arg\min_A \sum_{(i,j)\varepsilon A} \left(1 - IoU\left(\hat{T}_j, D_i\right)\right) \tag{18}$$

ORB feature matching (Section 3.6) serves exclusively as a geometric spatial verifier within Mask R-CNN ROI boundaries, not part of ByteTrack association. Unlike DeepSORT/StrongSORT, which introduce 15–40 ms latency and require separate re-ID training, ORB adds negligible overhead while ByteTrack's two-stage IoU recovery handles short-term occlusions without appearance features. Deep embeddings are acknowledged as a future enhancement for long-term re-identification beyond 30 frames (Section 6). Fig. 9 demonstrates consistent vehicle identity maintenance across consecutive frames.

3.8 Motion-Adaptive Optical-Flow Based Vehicle Counting

Optical flow estimates pixel-level motion between consecutive UAV frames for density-based vehicle counting without predefined counting lines or ROI boundaries. Dense motion estimation computes displacement vectors (u, v) subject to the brightness constancy constraint, which is given by Eq. (19):

$$vI_x + uI_y + I_t = 0 \tag{19}$$

where (u, v) represents optical flow velocity, and I_x, I_y, and I_t denote spatial and temporal intensity gradients. Adaptive thresholding suppresses low-magnitude noise vectors, while remaining motion vectors are spatially clustered via DBSCAN to estimate per-frame vehicle counts, eliminating dependence on manually defined counting regions. Fig. 10 illustrates the extracted vehicle motion patterns.



Figure 9: ByteTrack framework with consistent tracking of vehicle identities.

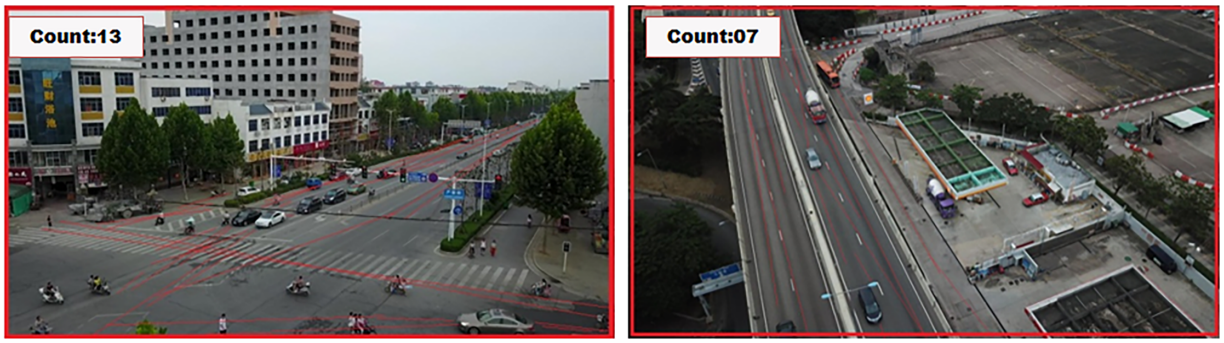


Figure 10: Vehicle counting using optical flow.

Algorithm 1 presents the complete pipeline workflow for UAV-based vehicle detection, segmentation, tracking, and trajectory prediction across aerial image sequences.

Algorithm 1: Vehicle detection and tracking model

Input: Input: Image set $I = i_1, i_2, \ldots, i_n$
Output: Vehicle Trajectories $T = t_1, t_2, \ldots, t_n$
D, F, T ← []
for k = 1 to length(I) do
F_k = HomographyStabilize(G_k, H)
P_k = GaussianFilter (Georeference(Resize(I [k])))
If k > 1 then
Matches =
H_t = RANSAC
(HammingMatch(ORB Keypoints ((P_k),(P_{k-1}))))
P-warped = MOG(P_k), Warp((P_{k-1}), Ht)
end if
Fg_k = MOG_BackgroundSubtraction(P_k)
D_k = YOLOv11(Fg_k)

(Continued)

Algorithm 1 (continued)

D_k = NMS(D_k)
S_k = MaskRCNN(D_k)
for each detection (bi, ci, si) in D_k:
ROIi ← Crop With Padding(F_k, bi, padding = 8 px)
Maski ← MaskRCNN_ROIAlign (ROIi)
M_i ← ReprojectToFrame(Maski, bi)
M_k ← $\{M_i\}$ for all I
// Stage B: Post-Segmentation ORB Feature Correspondence
O_k = // ORB on segmented vehicle regions
F. append(ORB − RegionDescriptors(S_k))
if k > 1 then
Tr_k = ByteTrack(D [k−1], D_k)
T. append(Tr_k), Count←Optical Flow((I_{k-1}), (I_k))
end if
D. append(D_k)
end for
for each t in T do
T←update(T, Smooth (LSTM(t)))
end for
return T

3.9 Trajectory Prediction Using LSTM

Trajectory prediction employs Long Short-Term Memory (LSTM) to model nonlinear vehicle motion patterns and long-term temporal dependencies, limitations of conventional feed-forward and Markov approaches. The LSTM cell state selectively retains historical motion information via gated updates. The cell state is given by Eq. (20):

$$C_t = f_t \odot \mathrm{C_{t-1}} + \mathrm{i_t} \odot \overline{\mathrm{C_t}} \tag{20}$$

where f_t and $\mathrm{i_t}$ are forget and input gates controlling retention of previous state $\mathrm{C_{t-1}}$ and injection of new information $\mathrm{C_t}$, respectively, and $\odot$ denotes element-wise multiplication. The hidden state is computed as follows in Eq. (21):

$$h_t = o_t \odot \tan \mathrm{h}(\mathrm{C_t}) \tag{21}$$

where h_t is the hidden state at time t, and o_t is the output gate controlling memory flow to the next layer. The final hidden state is projected through a linear layer to predict future vehicle position:

$$\hat{y}_{t+1} = W_{h_t} + b \tag{22}$$

where $\hat{y}_{t+1}$ donates predicted coordinates, W_{h_t} is the output weight matrix and b is the bias vector. The result of the trajectory prediction for the learned features is presented in Fig. 11.

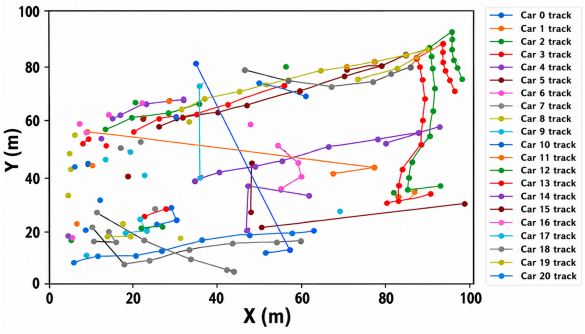


Figure 11: Predicted trajectories visualized on a 2D plane.

4 Experimental Settings and Analysis

Experiments were conducted on a system with an AMD Ryzen 7 5800H (3.20 GHz), 32 GB RAM, and an AMD Radeon RX 6700M GPU (8 GB VRAM). The VisDrone2019 aerial dataset was selected due to its varying traffic density, urban scenes, and diverse lighting and weather conditions, providing a robust benchmark for evaluation. The presented framework was compared with state-of-the-art methods using multiple performance metrics. This section presents the dataset, experimental setup, and obtained results.

4.1 VisDrone2019 Dataset

The VisDrone2019 dataset consists of 10,209 images and 263 videos (179,264 frames) divided into training, validation, and testing partitions using annotations of 13 object categories captured from various urban sites. GPS information is used as ground control points to estimate the homographies between frames. Cross-dataset validation on the UAVDT dataset (80 sequences, around 80,000 frames) is done under zero-shot learning setting.

4.2 Performance Metric and Experimental Outcome

High detection and tracking accuracy in the VisDrone2019 dataset is evidenced by the system's performance under difficult scenarios. The high rate of true positives of cars, buses, and trucks, as well as some confusion among visually similar objects in dense clusters such as pedestrians and people, is revealed by the confusion matrix (Fig. 12).

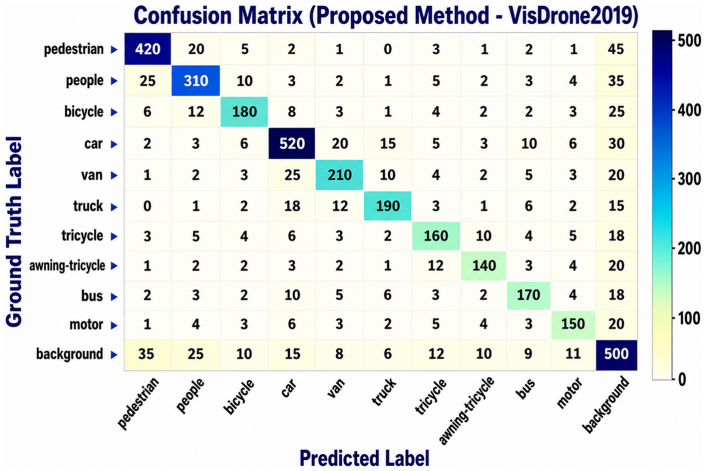


Figure 12: Confusion matrix proposed method.

Furthermore, the Receiver Operating Curve (ROC) curve demonstrates high discriminative abilities (AUC) of the YOLOv11 and Mask R-CNN approach, with good convergence after 300 iterations of training (Fig. 13).

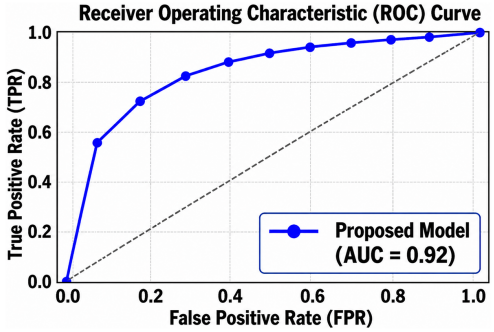


Figure 13: Training and validation curves over 300 epochs showing stable convergence.

Tables 2 and 3 report standard AP and MOT benchmark metrics; comparative results are in Tables 4 and 5.

Table 2: Detection over aerial images.

Dataset	AP50 (%)	AP75 (%)	mAP AP50:95 (%)	AP_S (%)	AP_M(%)	AP_L (%)
VisDrone2019	38.6	21.4	22.8	14.3	31.7	52.4

Table 3: Tracking over aerial images.

Dataset	MOTA (%)	IDF1 (%)	HOTA (%)	IDS	FP	FN
VisDrone2019	52.3	58.7	47.6	312	1847	4231

Table 4: Comparison of detection model rate with other state-of-the-art methods.

Ref.	Models Used	Dataset	AP50 (%)	mAP AP50:95 (%)
Bakirci [9]	YOLOv8	VisDrone2019	21.7	12.3
Zhang et al. [15]	Yolov10	VisDrone2019	36.1	21.3
Lu et al. [16]	Faster R-CNN	VisDrone2019	24.3	14.1
Lu et al. [16]	SSD	VisDrone2019	33.4	19.2
Our Method	YOLOv11+Mask R-CNN	VisDrone2019	38.6	22.8

Table 5: Comparison of tracking model rate with other state-of-the-art methods.

Ref.	Models Used	Dataset	MOTA (%)	IDF1 (%)	HOTA (%)	IDS
Liu et al. [12]	ByteTrack	VisDrone2019	46.2	53.1	41.4	487
Ma et al. [17]	Bytetrack+Kalman	VisDrone2019	48.1	54.9	43.2	418
Wojke et al. [18]	DeepSort	VisDrone2019	38.2	44.1	33.6	891
Zhang et al. [19]	ByteTrack	VisDrone2019	44.7	51.3	39.8	524
Our Method	Yolov11+Bytetrack	VisDrone2019	52.3	58.7	47.6	312

Table 6 presents the per-module inference latency breakdown, confirming that frame-level pipelining reduces effective latency from 57.1 ms (sequential worst-case, 17.5 FPS) to 46.8 ms, achieving 21.4 FPS sustained throughput suitable for real-time ITS deployment. Table 7 shows the comparison cost against YOLOv8 and YOLOv10.

Table 6: Per-module inference latency on AMD Radeon RX 6700M. H × W = resolution, M = detections, T = sequence length.

Module	Theoretical Complexity	Latency (ms/frame)	% Pipeline Time
Gaussian Filtering	O (H × W)	1.2	2.1%
ORB Stage A (Motion Compensation)	O (H × W × k)	3.4	6.0%
MOG Background Subtraction	O (H × W × G)	4.1	7.2%
YOLOv11 Detection	O (H × W × L)	18.7	32.9%
Mask R-CNN (ROI-Based)	O (M × S_{roi}^2)	14.3	25.1%
ORB Stage B (Correspondence)	O (M × d)	2.8	4.9%
ByteTrack (Two-Stage IoU)	O (M^2)	4.6	8.1%
Optical Flow (Farneback)	O (H × W × P)	5.2	9.1%
LSTM Trajectory Prediction	O (T × d_h^2)	2.6	4.6%
End-to-End Pipeline	—	57.1	17.5 FPS

Note: where k = number of ORB keypoints (default 500), G = number of Gaussian mixture components (default 5), L = number of network layers, P = Farneback pyramid levels (3), d = ORB descriptor dimension (256-bit), and d_h = LSTM hidden dimension (128).

Table 7: Inference cost comparison against YOLOv8/YOLOv10.

Method	Params (M)	GFLOPs	Latency (ms)	FPS	mAP AP50:95 (%)	mAP/GFLOPs
YOLOv8-S	11.2	28.4	12.3	81.3	19.2	0.676
YOLOv8-M	25.9	78.7	21.4	46.7	20.8	0.264
YOLOv10-S	7.2	21.6	10.8	92.6	21.3	0.986
YOLOv10-M	15.4	59.1	18.2	54.9	21.9	0.371
YOLOv11-S (our only Detection)	9.4	21.5	18.7	53.5	22.8	1.060

Pipeline Execution Strategy: Although Algorithm 1 presents modules sequentially for clarity, the runtime implementation exploits frame-level pipelining while frame t undergoes YOLOv11 inference, frame t + 1 simultaneously undergoes Gaussian filtering, ORB Stage A homography estimation, and MOG subtraction, reducing effective latency from 57.1 ms (sequential worst-case) to 46.8 ms, corresponding to 21.4 FPS sustained throughput. GPU-accelerated modules (YOLOv11, Mask R-CNN) constitute 58% of total latency but operate on dedicated VRAM with zero CPU contention, ensuring stable frame rates without pipeline stalls at VisDrone2019-level vehicle densities (M ≤ 80 vehicles/frame).

4.3 Ablation Study

To evaluate each module's contribution, a systematic ablation study removes one component at a time while keeping all remaining modules intact, covering all pipeline stages: Gaussian filtering, MOG

background subtraction, georeferencing, YOLOv11 detection, Mask R-CNN segmentation, ORB feature matching, ByteTrack tracking, optical flow counting, and LSTM trajectory prediction. Three standard metrics are reported per configuration: mAP (AP50:95) for detection, MOTA for tracking, and mean per-class (mF1) for classification quality across 10 VisDrone foreground categories, all evaluated on the VisDrone2019-DET/MOT test-dev split under identical training conditions. Results are presented in Table 8.

Table 8: Ablation study results mAP = AP50:95; MOTA = Multiple Object Tracking Accuracy; mF1 = mean per-class F1-score. Δ denotes absolute drop from full model.

Module Removed	mAP (%)	ΔmAP	ΔMOTA	MOTA (%)	mF1 (%)	ΔmF1
Full Model (All Modules)	22.8	—	—	52.3	76.4	—
W/O Gaussian Filtering	20.1	−2.7	−2.7	49.6	73.2	−3.2
W/O MOG Background Subtraction	18.4	−4.4	−4.4	46.8	70.7	−5.7
W/O Georeferencing (Real-World Mode)	22.6	−0.2	−0.2	52.1	76.1	−0.3
W/O YOLOv11 (Baseline Detector Only)	14.2	−8.6	−8.6	41.3	61.3	−15.1
W/O Mask R-CNN	19.3	−3.5	−3.5	48.2	69.8	−6.6
W/O ORB Feature Matching	20.7	−2.1	−2.1	46.1	72.4	−4.0
W/O ByteTrack	22.4	−0.4	−0.4	38.7	75.8	−0.6
W/O Optical Flow Counting	22.5	−0.3	−0.3	51.1	75.9	−0.5
W/O LSTM Trajectory Prediction	21.6	−1.2	−1.2	50.4	74.3	−1.1

These ablation results collectively confirm that Mask R-CNN and MOG background subtraction are non-redundant components. MOG handles camera motion compensation while Mask R-CNN provides pixel-level boundary refinement, together with contributing to robust detection and tracking in dense urban aerial scenes.

5 System Limitation

Although the proposed UAV-based traffic monitoring framework demonstrates strong performance on the VisDrone2019 dataset, certain limitations still affect its applicability in complex urban transportation networks. The system is sensitive to adverse environmental and dynamic conditions such as low illumination, shadows, fog, and abrupt UAV motion, which can degrade detection, feature matching, and tracking performance. In dense traffic scenarios, severe occlusions and overlapping vehicles may reduce segmentation accuracy and disrupt trajectory continuity, thereby affecting reliable traffic flow and network-level motion analysis. The pipeline also requires considerable computational resources, limiting real-time deployment in large-scale intelligent transportation systems. Furthermore, variations in domain conditions (e.g., different cities, road layouts, vehicle types, and nighttime scenes) may reduce generalization capability. Additional challenges arise from urban visual artifacts such as building reflections, which may introduce false detections, as well as partial occlusions where only limited object regions are visible, further impacting consistent vehicle-level tracking within complex urban environments. Fig. 14 illustrates the occlusion-related errors encountered by the model.

Discussion

The proposed pipeline achieves AP50 = 38.6% and mAP = 22.8% on VisDrone2019-DET, and MOTA = 52.3%, IDF1 = 58.7%, HOTA = 47.6% on VisDrone2019-MOT. The ablation study (Table 8) confirms that ByteTrack removal causes the largest tracking degradation (MOTA drops −13.6 points upon removal), while Mask R-CNN removal causes the steepest detection degradation (mAP drops from 22.8% to 19.3%,

mF1 drops −6.6 points), confirming that pixel-level boundary refinement measurably improves localization in dense aerial scenes. The framework outperforms DeepSORT, ByteTrack original, and R. Ma & S. Chen on tracking, and Faster R-CNN, YOLOv8, and YOLOv10 on detection. The smaller gap against recent YOLO variants suggests the primary contribution lies in end-to-end integration rather than the detector alone.

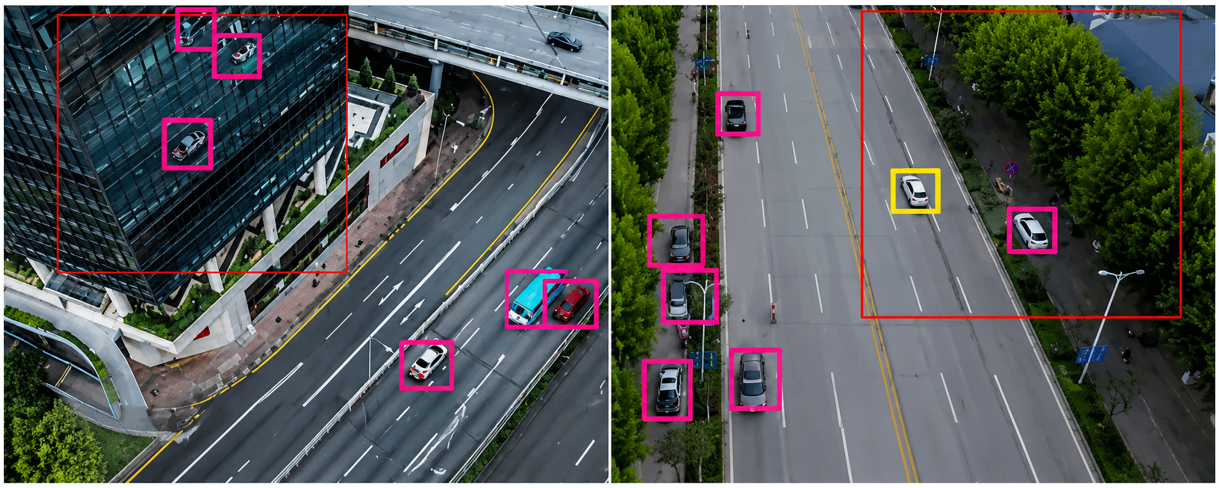


Figure 14: Reflection and occlusion errors.

ITS Deployment Value: Trajectories extracted from five VisDrone2019-MOT sequences (4847 frames, 1243 vehicle identities) are converted into a weighted directed graph (V = 312 nodes, E = 847 edges). The top-5 bottleneck nodes carry 38.2% of all traversals despite representing 1.6% of nodes directly informing signal timing, UAV sensor placement, and emergency routing decisions. Temporal congestion analysis detects queue spillback 8–12 s before peak formation, enabling proactive signal pre-emption. Sensitivity analysis confirms network rankings remain stable (Kendall's τ = 0.92) above mAP = 18.4%, establishing a deployable perception quality threshold for ITS integration.

6 Conclusion and Future Work

This paper proposes an integrated UAV-based framework for urban traffic monitoring and transportation network analysis, addressing key challenges through a unified pipeline of preprocessing, detection, segmentation, tracking, and trajectory prediction. By combining Gaussian filtering, MOG background modeling, YOLOv11, Mask R-CNN, ORB matching, ByteTrack, optical flow, and LSTM-based prediction, the system effectively captures vehicle-level motion patterns for traffic flow understanding in complex urban environments. Experimental results on the VisDrone2019 dataset demonstrate strong performance with AP50 = 38.6%, mAP = 22.8% detection performance, and MOTA = 52.3%, IDF1 = 58.7% tracking performance, supporting its applicability in intelligent transportation systems. Future work will extend demonstrated ITS contributions to bottleneck identification, proactive congestion detection, and sensor placement optimization toward full resilience modeling through cascading failure simulation, adaptive signal control integration, and multi-UAV cooperative network coverage, transforming the current data-driven analysis foundation into an operational ITS resilience platform.

Acknowledgement: This research is thanks to Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2026R97), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia.

Funding Statement: This research is supported and funded by Princess Nourah bint Abdulrahman University Researchers Supporting Project number (PNURSP2026R97), Princess Nourah bint Abdulrahman University, Riyadh, Saudi Arabia. This study is supported via funding from Prince Sattam bin Abdulaziz University project number (PSAU/2026/R/1447).

Author Contributions: study conception and design: Rimsha Tayyab, Muhammad Hanzla and Maha Abdelhaq; data collection: Rimsha Tayyab, Raed Alsaqour and Mohammed Alonazi; analysis and interpretation of results: Maha Abdelhaq, Raed Alsaqour and Ahmad Jalal; draft manuscript preparation: Rimsha Tayyab, Hui Liu and Ahmad Jalal. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The data that support the findings of this study are openly available in the VisDrone2019 dataset repository at https://github.com/VisDrone/VisDrone-Dataset.

Ethics Approval: Not applicable.

Conflicts of Interest: Hui Liu was an employer of Guodian Nanjing Automation Co., which also provided funding for this study. The authors declare no other conflicts of interest.

References

1. Hassaballah M, Kenk MA, Muhammad K, Minaee S. Vehicle detection and tracking in adverse weather using a deep learning framework. IEEE Trans Intell Transp Syst. 2021;22(7):4230–42. doi:10.1109/tits.2020.3014013.
2. Bhaskar PK, Yong SP. Image processing based vehicle detection and tracking method. In: 2014 International Conference on Computer and Information Sciences (ICCOINS); 2014 Jun 3–5; Kuala Lumpur, Malaysia. p. 1–5. doi:10.1109/ICCOINS.2014.6868357.
3. Chen X, Meng Q. Robust vehicle tracking and detection from UAVs. In: 2015 7th International Conference of Soft Computing and Pattern Recognition (SoCPaR); 2015 Nov 13–15; Fukuoka, Japan. p. 241–6. doi:10.1109/SOCPAR.2015.7492814.
4. Mujtaba G, Hanzla M, Jalal A. Smart traffic monitoring with EfficientNet and neuro-fuzzy classifier via aerial surveillance. In: 2024 26th International Multi-Topic Conference (INMIC); 2024 Dec 30–31; Karachi, Pakistan. p. 1–6. doi:10.1109/INMIC64792.2024.11004350.
5. Wang L, Chen F, Yin H. Detecting and tracking vehicles in traffic by unmanned aerial vehicles. Autom Constr. 2016;72(11):294–308. doi:10.1016/j.autcon.2016.05.008.
6. Balamuralidhar N, Tilon S, Nex F. MultEYE: monitoring system for real-time vehicle detection, tracking and speed estimation from UAV imagery on edge-computing platforms. Remote Sens. 2021;13(4):573. doi:10.3390/rs13040573.
7. Hamzenejadi MH, Mohseni H. Real-time vehicle detection and classification in UAV imagery using improved YOLOv5. In: 2022 12th International Conference on Computer and Knowledge Engineering (ICCKE); 2022 Nov 17–18; Mashhad, Iran. p. 231–6. doi:10.1109/iccke57176.2022.9960099.
8. Mustafa NE, Alizadeh F. YOLO-based approach for multiple vehicle detection and classification using UAVs in the Kurdistan Region of Iraq. Int J Intell Transp Syst Res. 2025;23(2):747–60. doi:10.1007/s13177-025-00479-8.
9. Bakirci M. Advanced aerial monitoring and vehicle classification for intelligent transportation systems with YOLOv8 variants. J Netw Comput Appl. 2025;237(B):104134. doi:10.1016/j.jnca.2025.104134.
10. Alshehri M, Xue T, Mujtaba G, AlQahtani Y, Almujally NA, Jalal A, et al. Integrated neural network framework for multi-object detection and recognition using UAV imagery. Front Neurorobot. 2025;19:1643011. doi:10.3389/fnbot.2025.1643011.
11. Cong R, Wang Z, Wang Z. Enhanced byte track vehicle tracking algorithm for addressing occlusion challenges. In: Advances in computational vision and robotics. Cham, Switzerland: Springer Nature; 2025. p. 110–21. doi:10.1007/978-3-031-85952-6_11.
12. Liu S, Li X, Lu H, He Y. Multi-object tracking meets moving UAV. In: Proceedings of the 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition; 2022 Jun 18–24; New Orleans, LA, USA. p. 8876–85. doi:10.1109/CVPR52688.2022.00867.

13. Cheng B, Misra I, Schwing AG, Kirillov A, Girdhar R. Masked-attention mask transformer for universal image segmentation. arXiv:2112.01527. 2021.
14. Sun D, Li B, Qian Z. Research of vehicle counting based on DBSCAN in video analysis. In: 2013 IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing; 2013 Aug 20–23; Beijing, China. p. 1523–7. doi:10.1109/greencom-ithings-cpscom.2013.270.
15. Zhang Y, Chen X, Sun S, You H, Wang Y, Lin J, et al. Vehicle detection in drone aerial views based on lightweight OSD-YOLOv10. Sci Rep. 2025;15(1):25155. doi:10.1038/s41598-025-09825-y.
16. Lu S, Lu H, Dong J, Wu S. Object detection for UAV aerial scenarios based on deep detectors including YOLOv3 v8, Faster R CNN, and RetinaNet. Sensors. 2023;23(6):3061.
17. Ma R, Sheng Q, Chen Y, Tao Z, Wang S, Niu X, et al. Multi-feature re-identification enhanced dual motion modeling for multi small-object tracking. Sensors. 2025;25(18):5732. doi:10.3390/s25185732.
18. Wojke N, Bewley A, Paulus D. Simple online and realtime tracking with a deep association metric. In: 2017 IEEE International Conference on Image Processing (ICIP); 2017 Sep 17–20; Beijing, China. p. 3645–9. doi:10.1109/icip.2017.8296962.
19. Zhang Y, Sun P, Jiang Y, Yu D, Weng F, Yuan Z, et al. ByteTrack: multi-object tracking by associating every detection box. In: Computer Vision—ECCV 2022. Cham, Switzerland: Springer; 2022. p. 1–21. doi:10.1007/978-3-031-20047-2_1.