Mitigating In-Transit Vision Noise for Enhanced Vehicle Safety

Yichen Luo, Junzhou Chen, Xinyu Chen, Sidi Lu
Department of Computer Science, William & Mary Williamsburg, VA 23185, USA
{yluo11,jchen57,xchen28,sidi}@wm.edu

ABSTRACT

Software-defined vehicles (SDVs) are sensor-enabled systems, with cameras playing a crucial role in intelligent and safety-critical applications. However, they face significant challenges from environmental noise, such as inclement weather and unexpected occlusions. Unlike static sensors, SDV cameras experience dynamically changing noise patterns influenced by driving speed — an aspect often overlooked in previous research on camera-based vision tasks. To address this gap, we conduct a quantitative analysis of the in-transit noise impact using data from public datasets, the CARLA simulator, a robotic vehicle, and a real vehicle. Our findings suggest that maintaining a speed below 40km/h may serve as a threshold for ensuring reliable camera-based applications under noisy urban conditions. In addition, we propose TransitNet, a novel model designed to mitigate in-transit camera noise and enhance driving safety, particularly at higher speeds. Compared to multiple baselines, experimental results show that TransitNet improves the F-measure by 5.1%, mAP@50 by 3.6%, and increases FPS by 56.7% across all datasets. We also provide detailed observations and insights from extensive testing.

CCS CONCEPTS

• **Computer systems organization** → **Embedded and cyber-physical systems**; *Sensors and actuators.*

KEYWORDS

Software-defined vehicles, data noise, vehicle speed

1 INTRODUCTION

Software-defined vehicles (SDVs) are vehicles whose core functionalities are managed by precise, safety-critical software for extensive functionality, supporting continuous software optimization and updates throughout the vehicle's lifecycle [40, 28, 30, 21]. Recently, forward-looking industry leaders (e.g., Tesla [41], General Motors [61], Ford [23], BMW [22], Mercedes-Benz [17], Toyota [42], Audi [7], Arm [5], Amazon [46], and Aptiv [4]) have begun to push a strategic transformation from traditional vehicles to SDVs. Particularly, autonomous vehicles can be considered a type of SDV.

As sensor-enabled smart systems, cameras are among the most critical and widely used sensors in SDVs, supporting various intelligent and safety-critical applications. For instance, Tesla's Full Self-Driving (FSD) software relies entirely on a comprehensive suite of cameras for its autonomous driving capabilities. However, in 2024, even the latest version of the FSD software reportedly failed to recognize red lights via cameras due to external visual factors [19], leading to hundreds of crashes and dozens of fatalities [20].

Specifically, inclement weather (e.g., rain and snow) and unexpected occlusions (e.g., leaves and mud) are typically external factors that introduce noise into sensor data. The former are usually small in size but high in density, while the latter are larger and

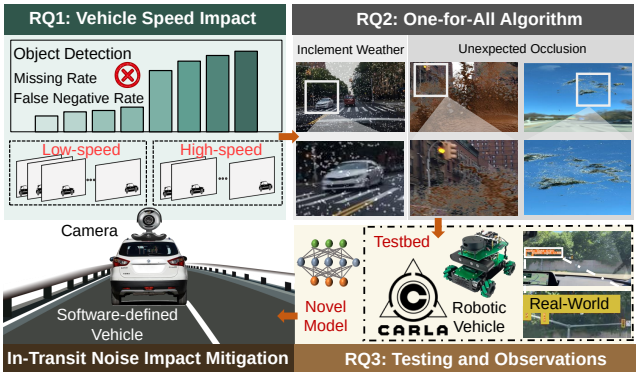


Figure 1: An overview of our structured methodology for understanding the impact of vehicle speed on SDV vision algorithms and mitigating diverse in-transit vision noise types. We address three main research questions (RQs): RQ1 quantifies how vehicle speed affects algorithm performance using public datasets, CARLA simulator data, and data collected from both a robotic vehicle and a real vehicle (top-left). RQ2 introduces TransitNet as a versatile solution to reduce diverse in-transit noise (top-right). RQ3 involves extensive testing and analysis of TransitNet's effectiveness (bottom-right).

cover a portion of the camera's view. Studies show that fatal vehicle crashes increase by 34% in such adverse weather conditions [55].

1.1 Unique Challenges on SDV

Imperatives of a New Strategy. While external data noise is a common issue in consumer electronics like surveillance cameras and windshield screens, handling data noise in SDV presents unique challenges and high requirements due to several key differences.

First, varying vehicle speeds result in rapidly changing noise patterns, even for the same types of noise, such as raindrops. Our analysis of multiple datasets (e.g., BDD100K) shows that raindrops spread outward across the camera lens due to the vehicle's forward motion and air resistance, with dispersion varying with driving speed. This variation directly influences the model's attention on key features, potentially diverting attention away from target objects (detailed in Section 1.2). The impact is often more pronounced for side cameras due to their greater exposure to air turbulence.

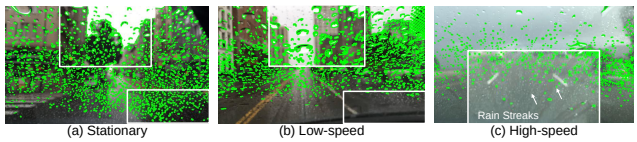
Second, SDV cameras are often exposed to various types of noise depending on their location, especially as they are mounted on the vehicle's exterior, where internal wipers or heaters cannot mitigate external interference. For example, flying insects may obstruct cameras near fields, while salt deposits can accumulate in regions like Salt Lake City. Coastal areas, such as Houston, expose cameras to severe weather, including storms and heavy rainfall. Mud splashes can also obstruct cameras, changing shape and coverage as the vehicle moves, causing prolonged visibility issues.

Third, addressing the aforementioned diverse noise types requires a robust vehicle model, such as an object detection model,

capable of handling noise-degraded data. However, limited computing resources in SDVs make it difficult to maintain high processing speeds, potentially compromising vehicle safety [43].

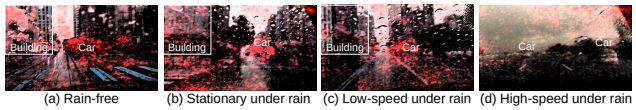
1.2 Motivation: In-Transit Noise Analysis

For the first time, this paper demonstrates that vehicle perception can be improved by accounting for noise patterns at varying speeds. Traditional studies focused on static noise affecting fixed cameras, overlooking the dynamic conditions of driving. As shown in Fig. 2 and Fig. 3, in-transit noise varies with speed, altering the model's focus on key features. By addressing these dynamic noise patterns, our work aims to improve perception accuracy in real-world driving, where noise is inherently variable and influenced by speed.



(a) Stationary (b) Low-speed (c) High-speed

Figure 2: Noise patterns at different driving speeds. Green outlines indicate noise (raindrop) edges, highlighting variations in noise shape, density, and location across the lens.



(a) Rain-free (b) Stationary under rain (c) Low-speed under rain (d) High-speed under rain

Figure 3: Saliency maps highlighting key regions in an image that contribute to the model's decision. Red-highlighted areas represent the most salient regions, indicating where they grab attention for the task at hand, such as object detection.

(1) Noise patterns. Using raindrops as an example of noise, we curate 3,000 BDD100K videos to identify driving scenes involving static vehicles (e.g., stopped at a red light), low-speed urban driving, and high-speed highway driving on rainy days. We use Canny edge detection [52] combined with K-means clustering [2] to visualize raindrop edges. Fig. 2 presents typical noise patterns across these scenarios, with green outlines representing the noise edges.

Specifically, when the vehicle is stationary (Fig. 2(a)), raindrops are often evenly distributed and uniform in size. At low speeds (Fig. 2(b)), they shift upward or to the edges, merging into fewer but larger droplets concentrated at the top of the lens. At high speeds (Fig. 2(c)), raindrops slide outward, forming streaks with sparse noise areas along the streaks and dense concentrations on either side, creating an uneven noise distribution. The frequency of these rain streaks tends to increase with higher driving speeds, further exacerbating the intermittent and uneven noise distribution.

(2) Saliency maps for model attention. To further understand how different in-transit noise patterns affect vehicle model performance on noise-degraded datasets, we generate saliency maps of a pre-trained ResNet-50 model [57] for object detection. As shown in Fig. 3, the red-highlighted areas indicate the regions that the model considers most influential in its decision-making process.

To be concrete, as a baseline, in rain-free conditions (Fig. 3(a)), the red-highlighted regions are concentrated around the target objects, indicating that the model can effectively focus on key elements for successful object detection. Under stationary, rainy conditions (Fig. 3(b)), noise obscures scene details, leading to more scattered model attention for object detection. At low speeds (Fig. 3(c)), the model's attention is further misdirected away from the primary objects. In high-speed, rainy conditions (Fig. 3(d)), rain streaks and uneven noise distribution divert the model's attention, which tends to result in less accurate detection of the key objects.

One might argue that the example evidence, such as noise patterns (Fig. 2) and saliency maps (Fig. 3), is specific to the dataset under study. To test this, we analyze data from other sources, including KITTI, nuScenes, and data from a robotic vehicle and a real vehicle at different speeds. Similar trends are observed across all datasets. While these examples do not imply that all vision noise will have identical effects, they demonstrate that considering driving speeds can enhance scene understanding in vision tasks compared to traditional methods that overlook the dynamic nature of driving, particularly under challenging weather conditions.

1.3 Research Questions and Contributions

Motivated by the in-transit noise analysis (Section 1.2), we focus on the widely adopted camera sensor as a foundational study and select object detection as a case study for SDVs. This work aims to answer three main **research questions (RQs)**: *i*) How does the vehicle's speed influence camera-based object detection results under inclement weather and unexpected occlusion conditions? *ii*) Can we design and implement a "one-for-all" solution to mitigate the impact of various types of in-transit camera noise, considering different vehicle speeds? *iii*) Can this comprehensive solution be both accurate and fast enough for different driving situations?

Technical-wise, we develop research tasks spanning theoretical, simulation, practice, and their interplay to answer the three RQs. Below we describe our major contributions.

- For RQ1, we conduct a quantitative analysis using the noise-degraded BDD100K data, along with data from the CARLA simulator, a robotic vehicle, and a real vehicle, to assess the impact of speed on vision tasks under two types of in-transit noise: unexpected occlusions and inclement weather (Sec. 3 and Sec. 4). Our findings suggest that 40km/h may serve as a threshold for safe urban driving to ensure accurate camera-based applications under various noisy conditions (Sec. 6).
- For RQ2, we propose TransitNet to mitigate various types of in-transit camera noise while balancing detection accuracy and inference speed. For RQ3, we evaluate TransitNet's effectiveness by comparing it to multiple baselines. TransitNet outperforms state-of-the-art (SOTA) models, such as D-FINE and RT-DETR, achieving a mean Average Precision (mAP@50) improvement of 7.4% and 11.5%, respectively, while nearly doubling the FPS for real-time performance across all datasets.
- We provide an extensive discussion of our experimental observations for RQ1, RQ2, and RQ3, offering in-depth explanations and identifying key trends (Sec. 6). These insights enhance understanding of the challenges faced by in-transit vehicle camera-based applications under inclement weather and unexpected occlusions, contributing to the broader knowledge base for enhanced vehicle safety.

The rest of the paper is organized as follows: Sec. 2 describes the experiment setup. Experimental results of RQ1, RQ2, and RQ3 are shown in Sec. 3, Sec. 4, and Sec. 5, respectively. Sec. 6 discusses experiment results and observations. Related works are reviewed in Sec. 7. Finally, Sec. 8 concludes the entire paper.

2 EXPERIMENT SETUP

Before presenting our experiment design, we summarize the key components of our experiments in this section, including test platforms, experiment tools, and detection methods used.

2.1 Testbed and Tool

In this work, we use multiple experiment testbeds: *i*) CARLA, a widely-used public simulation platform in automotive research (Fig. 4(a)), *ii*) a physical robotic vehicle testbed assembled with Mecanum wheels and developed using Robot Operating System (ROS) 2 (Fig. 4(b)), and *iii*) a real vehicle testbed equipped with a front camera and two side cameras. We provide the robotic vehicle with Level 2 autonomous driving capabilities, including mapping, path planning, obstacle avoidance, etc.

(a) CALRA simulator (b) Robotic vehicle on the track map

Figure 4: CARLA simulator and indoor testbeds.

(a) Robotic vehicle with transparent rain shield (b) Chevrolet Equinox vehicle with front and side cameras

Figure 5: Outdoor testbeds.

2.1.1 CARLA simulator. CARLA, a renowned open-source simulator for vehicle-related research, allows for the structuring of a digital world to meet various requirements [12]. It also provides an urban driving scenario (called "Town 01") along with adjustable settings, such as weather, vehicle speed, and sensor angle, facilitating different SDV testing tasks.

2.1.2 Robotic Vehicle and Track Map. As shown in Fig. 4(b), the robotic vehicle employs the NVIDIA Jetson Orin NX as the computing unit and is equipped with 16GB of memory and high-performance sensors such as LiDAR and a depth camera (Astra Pro Plus). The Jetson Orin NX is among the latest advancements in NVIDIA's Jetson series. Featuring the NVIDIA Ampere architecture GPU with up to 1024 CUDA cores and a 12-core Arm Cortex-A78AE CPU, it delivers up to 100 Tera Operations Per Second (TOPS) of AI performance, compared to the 0.5 TOPS offered by the Jetson Nano [13].

As shown in Fig. 4(b), we have a 2.8m × 3.2m autopilot track map with parking lots, sidewalks, interactions, traffic signs (18cm tall), and traffic lights (24cm tall). It can simulate the traffic scene for vehicles to automatically drive. In addition, to introduce in-transit noise, such as raindrops, the robotic vehicle is equipped with a transparent rain shield for outdoor experiments (Fig. 5(a)), and an electric sprayer is used to continuously apply water during driving.

2.1.3 Real Vehicle with Exterior Cameras. Our testbed also includes a Chevrolet Equinox SUV equipped with three Orbit T100 1080P cameras mounted on the roof to capture forward and side views. The cameras are installed horizontally, each with a 142-degree viewing angle. To align with mainstream datasets such as BDD100K, we crop the images to a resolution of 1280 × 720 pixels. The capability to conduct experiments using both a simulator and multiple physical testbeds enables a comprehensive evaluation.

2.2 Mitigation Solutions and Baselines

To address the impact of in-transit noise on SDV vision tasks, we introduce TransitNet (detailed in Section 5), inspired by the YOLO framework and its limitations (discussed in Section 5.1.1). TransitNet is designed to enhance real-time object detection, optimize hardware resource usage, and improve resilience to dynamic noise.

To further validate the superiority of our approach, we conduct a detailed comparison and analysis with current mainstream and SOTA methods. These methods include SSD [38], which achieves a balanced trade-off between speed and accuracy by detecting objects at multiple scales, making it suitable for real-time applications; RT-DETR [71], which integrates the global feature extraction capabilities of Transformers with an efficient detection mechanism, excelling in both accuracy and speed; and D-FINE [49], a model employing dynamic fine-grained inference strategies, effective for detecting in complex backgrounds and fine-scale objects, representing the advanced standard in real-time object detection.

3 IN-TRANSIT NOISE-DEGRADED DATA

This work focuses on the in-transit SDV vision noises caused by *i*) inclement weather (e.g., raindrops or snowflakes) and *ii*) unexpected occlusions (e.g., leaves and mud). Inclement weather produces small but densely distributed noise that covers the entire image, while unexpected occlusions result in larger, more sporadic coverage of the camera's view.

3.1 CARLA In-Transit Noise-Degraded Data

We first collect the CARLA in-transit noise-degraded dataset using a front-facing camera mounted on the exterior of the ego vehicle, i.e., the primary vehicle controlled by the user or the automated driving system (Fig. 4(a)). The camera captures raw images at 20 frames per second (FPS) with a resolution of 1280 × 720 pixels.

Ego vehicle trajectory. As shown in Fig. 4(a), in our CARLA setup, the ego vehicle starts at two points marked by yellow stars, from "waypoint 62" to "waypoint 15", and proceeds along the designated path marked by the red arrow in "Town 01". Here,"waypoint 62" is located at the junction of a T-shaped road with a traffic light at the next intersection, showing diverse urban road conditions. Here, "waypoint 15" provides a scenario with five surrounding vehicles, representing complex driving conditions.

Diverse driving speed setup in CARLA. Although the CARLA simulator can create various driving scenarios, such as rainy conditions, maintaining a constant speed is challenging in both manual and automated modes. To address this, we use a proportional-integral-derivative (PID) controller to keep the ego vehicle (the white one in Fig. 4(a)), cruising at eight speeds, ranging from 10km/h to 80km/h. Specifically, we initially set the maximum throttle value to accelerate to the desired speed. Once the target speed is reached, the throttle value is adjusted to 0.2 to ensure uniform speed, maintaining the absolute speed deviation within 1km/h.

Simulating noise in CARLA driving scenarios. We use rain as a case study for inclement weather in CARLA, with raindrops representing noise. By adjusting the precipitation levels, we varied the rain intensity, corresponding to different numbers of rain streaks (similar to the real-world scenario in Fig. 2(c)). The intensity levels were set to 0%, 25%, 50%, and 75% of the maximum possible, where 0% represented a clear day, and the other levels represented "light," "medium," and "heavy" rain conditions. The highest level, at 75%, provided a realistic and challenging scenario to test noise effects.

However, we observe that CARLA's precipitation effects were somewhat limited and lacked realism, particularly for heavy rain, where the raindrops were not sufficiently pronounced. To address this, we develop a versatile noise generation mechanism (Algorithm 1 in Section 3.4.3) to create dynamic light, medium, and heavy rain effects, which we apply to the clear CARLA dataset to better investigate the impact of speed on noise-degraded vision tasks.

In addition, to better quantify the impact of driving speed on noise-degraded vision tasks, we also minimize external environmental variables. For instance, we standardize the number and positions of non-player vehicles on the CARLA map, like the pink one in Fig. 4(a), during each driving round.

3.2 Robotic Vehicle In-Trainst Data

Robotic vehicle trajectory. As shown in Fig. 4(b), in the indoor experiment, the robotic vehicle begins its journey on an autopilot track map, proceeding along a straight road to encounter a stop sign, a yellow vehicle, and a traffic light. This design mirrors "waypoint 62" in CARLA, facilitating comparisons between the simulation and real-world testbed.

Diverse driving speed of robotic vehicle. Using the PID method, we capture on-map driving videos at four constant speeds: 0.2m/s, 0.4m/s, 0.6m/s, and 0.8m/s, with a maximum of 1m/s, at a resolution of 1280 × 720. These speeds correspond to real vehicle speeds of 10km/h, 20km/h, 30km/h, and 40km/h, ensuring comparability with the CARLA dataset and real-world vehicle driving scenarios.

Simulating in-transit noise for robotic vehicle. As shown in Fig. 5(a), to introduce raindrops as external noise for the robotic vehicle at varying speeds, we use an electric sprayer to continuously apply water while the robotic vehicle is protected by a transparent rain shield, ensuring no impact on vision tasks such as object detection. The sprayer features three different mist nozzles to simulate light, medium, and heavy rain conditions during driving.

3.3 Real-World Driving Data

We collect real-world driving data using a Chevrolet Equinox with stabilized front and two side cameras (Fig. 5(b)), recording at a resolution of 1280 × 720 pixels. The dataset includes low-speed urban and high-speed highway driving, covering various conditions such as speed limits, traffic lights, stop signs, and sidewalks. Speeds range from 0 to 96km/h (0 to 60mph). Data is collected in sunny and rainy conditions along the same routes for the fair comparisons.

3.4 Noise-Degraded BDD100K Data

The Berkeley DeepDrive BDD100K dataset [69] is one of the largest real driving video collections, featuring 100,000 high-definition videos totaling over 1,100 hours. Each 40-second video, recorded at a resolution of 1280 × 720 pixels, captures driving scenarios on highways as well as in urban and rural areas across major U.S. cities.

3.4.1 Limitations of Existing Datasets. Although the BDD100K dataset contains extensive video footage, only 1.7% includes noise-degraded data under inclement weather conditions [69]. Moreover, it exhibits high redundancy due to the similarity of consecutive frames and lacks specific vehicle speed information. This limitation is also present in other public datasets.

To address these limitations, we first identify key noise patterns for both low-speed urban and high-speed highway scenarios (detailed in Section 1.2 and Section 3.4.2) and develop a versatile noise generation mechanism to simulate inclement weather and unexpected occlusions with adjustable vehicle speeds. These synthesized noises are then applied to the original BDD100K dataset to create noise-degraded datasets for comprehensive quantitative analysis.

3.4.2 In-Transit Noise Patterns: Noise Size and Density. Before introducing the noise generation mechanism, we summarize key insights on in-transit vision noise patterns.

(1) In-transit inclement weather noise patterns. As shown in Fig. 2, at low speeds, due to vehicle motion and lens curvature, raindrops shift upward or towards the edges of the lens, merging into larger droplets, resulting in fewer but larger raindrops concentrated at the top of the lens. At high speeds, raindrops slide outward from the center bottom, forming rain streaks. This creates a sparse distribution along the streaks, while areas on either side have a dense concentration, making the noise distribution noticeably uneven. As speed increases, the frequency of these rain streaks also tends to increase, exacerbating the uneven noise distribution.

(2) In-transit unexpected occlusion patterns. At low speeds, unexpected occlusions such as leaves or mud are relatively large, occupying substantial pixel areas with relatively low noise density, without significant spreading or elongation. At high speeds, occlusions are less affected by vehicle driving speed compared to inclement weather noise, though their intensity may increase slightly due to movement. Despite this, noise shapes and positions remain largely unchanged, resulting in stable noise patterns.

3.4.3 Versatile Noise Generation. Inspired by real-world noise patterns encountered by moving vehicles (Section 1.2 and Section 3.4.2), we propose a versatile noise generation algorithm to create noise masks for unexpected occlusions and inclement weather. Each noise mask is adjusted from the previous frame to ensure temporal continuity, reflecting the influence of vehicle speed on noise patterns. As shown in 1 (lines 1 and 6) and Fig. 6, the camera image, with width W and height H, is divided into an $m \times m$ grid, yielding m^2 cells. Each cell has dimensions cell width = $\frac{W}{m}$ and cell height = $\frac{H}{m}$, with an area of cell area = $\frac{W \times H}{m^2}$.

(1) Unexpected occlusion noise generation. Next, the occlusion noise generation steps (lines 7-8 of Algorithm 1) compute the output for each cell as: $cell_{out} = (1 - M_{cell}) \cdot B_{cell} \odot cell_{in}$. In this process, $cell_{in}$ represents a noise-free BDD100K image used as the background for each grid cell, while B_{cell} serves as a binary template for the selected occluding element, such as mud or a leaf. The mask M_{cell} governs occlusion within each cell: $(1 - M_{cell}) = 1$ indicates the presence of occlusion in the cell, while $(1 - M_{cell}) = 0$

Algorithm 1 Versatile Noise Generation

Input: Grid dimensions m, radius r_o, scaling factor α, drops number num_points
Output: In-transit Noise: position, radius, color
Initialize an empty list for drops
Calculate the width and height of each grid cell
Calculate the density factor $\ell = \beta \times v$
for each cell in the $m \times m$ grid (total m^2 cells) **do**
  if Generating occlusion noise **then**
    $cell_{out} = (1 - M_{cell}) \cdot B_{cell} \odot cell_{in}$
  else
    Determine the number of drops to generate based on ℓ
    for each drop to be generated **do**
      if cell is the bottom central grid cell **then**
        Generate noise streaks with random length
      else
        Randomly determine the drop's position
        Calculate noise size $r_{i,j} = r_o \times \alpha$
        Randomly determine the drop's color
      end if
      Add the drop to the list
    end for
  end if
end for
return In-transit Noise

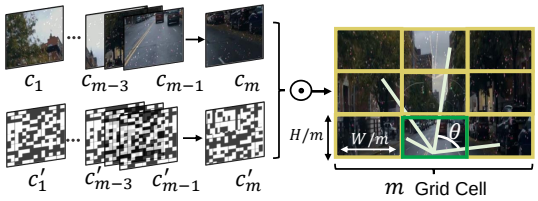


Figure 6: An example of in-transit noise generation in a high-speed scenario, where $\odot$ denotes element-wise multiplication and θ ranges from 0 to π in the first and second quadrants.

denotes its absence. The element-wise multiplication operator $\odot$ blends the occlusion template with the background image, creating a realistic occlusion effect in the output $cell_{out}$ for each grid cell.

(2) Inclement weather noise generation. The generation of in-transit inclement weather noise consists of three key steps below (line 10-19 of Algorithm 1).

i) **In-transit noise size**. Algorithm 1 directly controls the noise size by adjusting the radius $r_{i,j}$ of the noise point within the grid cell of the i-th row and j-th column. The radius $r_{i,j}$ is determined by the default radius r_o and a scaling parameter α: $r_{i,j} = r_o \times \alpha$. Here, r_o ranges from 1 and 2, and α adjusts the radius value according to the noise level, taking values from the set $[1, 2, 4, 8]$ in our experiment.

ii) **In-transit noise density**. Algorithm 1 further determines a suitable noise density factor ℓ for the current vehicle speed v, which controls the number of noise points generated within each grid cell. This factor is defined as: $\ell = \beta \times v$. Here, β represents the default density when the vehicle is static, and $\beta \in (0, 0.5]$. In this work, we define $\beta = 0.1$ and we first theoretically set $v \in [10, 20, 30, 40, 50, 60, 70, 80]$ km/h, resulting in corresponding noise density factors $\ell \in [1, 2, 3, 4, 5, 6, 7, 8]$.

iii) **In-transit noise streaks**. Algorithm 1 generates noise streaks to simulate realistic weather impacts (Fig. 2(c)). As shown in Fig. 6, noise streak generation begins from the bottom center of the image, represented by coordinates $(center_x, center_y)$. Each new streak point is calculated based on the previous point, using the distance parameter $r_{i,j}$ and the angle θ (restricted to $[0, \pi]$ to keep streaks within the first and second quadrants, mimicking realistic motion). The x-coordinate of each subsequent point is determined by adding $r_{i,j} \cdot \cos(\theta)$ to the x-coordinate of the center, while the y-coordinate is calculated by adding $r_{i,j} \cdot \sin(\theta)$ to the y-coordinate. By adjusting the angle θ, multiple rain streaks can be generated in different directions within the specified quadrants, enhancing the authenticity of noise simulation.

3.4.4 In-Transit Noise Generation Results. Using Algorithm 1, we generate noise-degraded datasets from BDD100K for unexpected occlusion and inclement weather, as shown in Fig. 7.

Figure 7: Examples of noise-degraded data based on BDD100K: unexpected occlusion and inclement weather.

4 IN-TRANSIT NOISE EVALUATION

After creating in-transit noise-degraded datasets using data from BDD100K, the CARLA simulator, a robotic vehicle, and a real vehicle, we next quantify the impact of these noises on vision tasks.

4.1 Noised-Degraded BDD100K Evaluation

To address RQ1: How does the vehicle's speed influence camera-based object detection results under inclement weather and unexpected occlusion conditions, we first explore the impact of different noise size ranges on object detection. Then, within the same size range, we examine the impact of varying noise densities.

4.1.1 Evaluation Metrics. To explore the impact of in-transit noise on SDVs, we focus on the fundamental application, object detection, as a case study. We use the missing rate (MR) and false negative rate (FNR) as the key evaluation metrics, defined as follows:

$$\text{Missing Rate} = \frac{\text{Num}_{\text{typex}} - \text{Num}_{\text{base}}}{\text{Num}_{\text{base}}} \quad (1)$$

$$\text{False Negative Rate} = \frac{\text{FN}}{\text{TP} + \text{FN}} \quad (2)$$

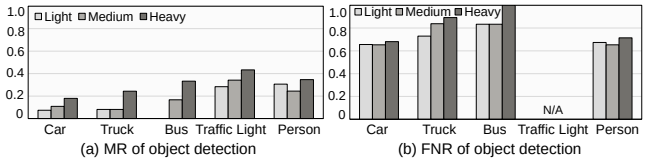
Here, MR measures the proportion of actual objects that are not detected, calculated as the reduction in detection counts due to in-transit noise effects by comparing the difference between detections under adverse conditions ($\text{Num}_{\text{typex}}$) and baseline detections in clear conditions (Num_{base}), normalized by the baseline.

TP and FN represent true positives and false negatives, respectively, where T and F indicate whether the detection result is correct, and P and N denote whether an object is detected (positive) or absent (negative). The FNR metric is essential for assessing detection reliability, particularly in safety-critical applications. Although both FNR and MR measure undetected objects, they differ in focus: MR evaluates the impact of environmental factors by comparing detection counts under clear and adverse conditions, whereas FNR assesses detection performance within a single condition.

In safety-critical applications such as object detection for vehicles, FNR is more critical than the False Positive Rate (FPR), as missing an object (e.g., a pedestrian or another vehicle) directly

compromises safety [68, 29]. In contrast, FPR may cause unnecessary braking or avoidance but typically poses less immediate danger. Hence, minimizing FNR is essential to ensure the safety and reliability, making it a key evaluation metric in this study.

4.1.2 Impact of In-Transit Noise Sizes. We first assess the impact of noise sizes on object detection using YOLOv9. We define and create three experimental groups based on the noise-degraded BDD100K data with "light," "medium," and "heavy" noise levels to represent the light-level, medium-level, and heavy-level noise-degraded datasets. The corresponding noise sizes $r_{i,j}$ (as discussed in Algorithm 1) are categorized into ranges of [1,2], [2,4], and [4,8], respectively.



(a) MR of object detection (b) FNR of object detection

Figure 8: The evaluation results of object detection for five main classes (car, truck, bus, traffic light, and person) across three levels of noise-degraded datasets based on BDD100K.

We use the detection results of YOLOv9 on the original noise-free BDD100K dataset as the ground truth to measure the performance variation of object detection for the three noise-level datasets. The evaluation metrics include MR (Fig. 8(a)) and FNR (Fig. 8(b)).

Based on Fig. 8, we observe that the detection performance (MR and FNR) for cars, trucks, buses, traffic lights, and persons deteriorates as noise levels increase from "light" to "medium" to "heavy," although the degree of impact varies across different categories. Specifically, the detection of cars and persons experience moderate changes in performance. In contrast, trucks and buses face significant challenges, particularly under heavy noise conditions, with sharp increases in both MR and FNR.

Figure 8(b) shows an FNR of 0 for traffic light detection, as no traffic lights are detected in any noise-degraded dataset (i.e., FN = 0). This highlights the difficulty of detecting small objects like traffic lights under light noise, and their undetectability under medium or heavy noise conditions.

4.1.3 Impact of In-Transit Noise Density. Next, within the same noise size range, we assess the impact of varying noise densities considering the mobility of vehicles. As discussed in Section 3.4.3, we define the noise density factor $\ell \in [1, 2, 3, 4, 5, 6, 7, 8]$ according to different vehicle speeds $v \in [10, 20, 30, 40, 50, 60, 70, 80]$ km/h. Based on the value of ℓ, we generate eight experiment groups for each noise level (i.e., light, medium, and heavy noise). Similarly, we employ YOLOv9 for vehicle detection and person detection and evaluate MR and FNR for each experimental group.

Evaluation results of MR. Figure 9 illustrates vehicle and person detection performance across eight noise levels. Higher speeds increase MR for both detections, leading to more misses. Here, "vehicle" includes cars, trucks, and buses. For vehicle detection (Fig. 9(a)), MR notably rises at higher speeds across all noise levels. A similar pattern is observed for person detection (Fig. 9(b)), where higher speeds increase MR.

Evaluation results of FNR. Compared to MR, the FNR of vehicle detection remains relatively stable with minor fluctuations as vehicle speed increases under light and heavy noise levels. However, it encounters a notable increase under medium noise conditions. Comparing Fig. 9(a) and Fig. 9(b), the FNR for person detection exhibits a more pronounced increase at higher speeds than vehicle detection. This indicates that person detection is more adversely affected by increased vehicle speeds under varying noise conditions.

4.2 Evaluation through CARLA Simulation

Next, for a fair comparison, we use the CARLA simulator to test the performance of object detection under varying rainy scenarios at different driving speeds. The intensity of the simulated rain is set to 0%, 25%, 50%, and 75% of the maximum possible level, where 0% indicates a sunny, daytime scenario and the other three levels represent "light," "medium," and "heavy" noise conditions.

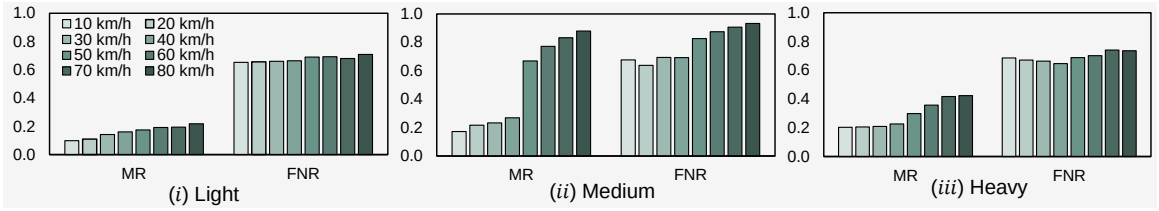
4.2.1 Experiment Groups. As discussed in Section 3.1, CARLA's built-in precipitation effects are not very realistic: CARLA does not accurately depict raindrops but rather represents rain as multiple streaks in the sky and wet road. At the highest level of 75%, the rain streaks transform into rain columns, which are even less realistic.

To address the aforementioned unnatural visual noise in CARLA and enhance noise diversity, we apply Algorithm 1 on the collected CARLA sunny (clear) dataset (discussed in Section 3.1) to generate noise-degraded data with three noise levels (i.e., light, medium, and heavy). In addition, since we define the vehicle driving speed ranging from 10km/h to 80km/h in CARLA, we do not need to apply the noise density factor in Algorithm 1 here.

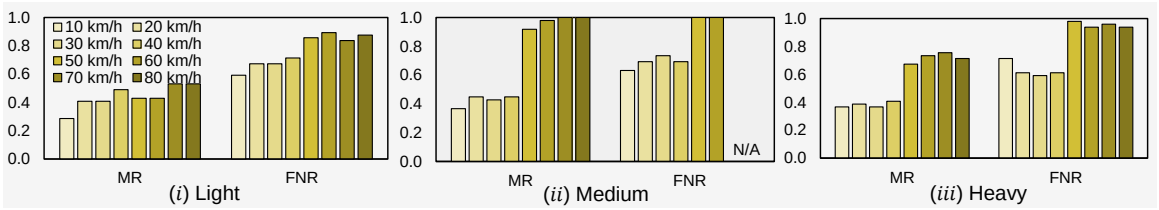
In this way, seven video clips are collected using CARLA. One clip represents the sunny situation and serves as the ground truth (GT). The remaining six clips are divided into two groups: (1) CARLA's built-in precipitation data with light, medium, and heavy precipitation levels, labeled as "Light_CA," "Medium_CA," and "Heavy_CA," respectively, and (2) clips with noise created using our proposed algorithms, with light, medium, and heavy noise levels, labeled "Light_Our," "Medium_Our," and "Heavy_Our," respectively.

4.2.2 Evaluation Metrics. To compare object detection counts between original sunny (clear) images and their noise-degraded counterparts across various noise-level categories in CARLA, we utilize the previously introduced MR as the evaluation metric. Here, Numcar denotes the number of vehicles detected by YOLOv9, while NumGT represents the ground truth count or the actual number of vehicles present in the image.

4.2.3 Evaluation Results on CARLA Dataset. Fig. 10 illustrates the variation in MR as vehicle speed increases, highlighting the challenges in maintaining detection performance at higher speeds and under heavier noise conditions. Specifically, for light noise levels (Light_CA and Light_Our), the MR remains low but increases as speed rises. Medium noise levels (Medium_CA and Medium_Our) have more pronounced impacts, with the MR increasing significantly at higher speeds. Under heavy noise situations (Heavy_CA and Heavy_Our), MR sharply increases, reaching nearly 100% above 40km/h, with detection becoming nearly impossible at higher speeds.

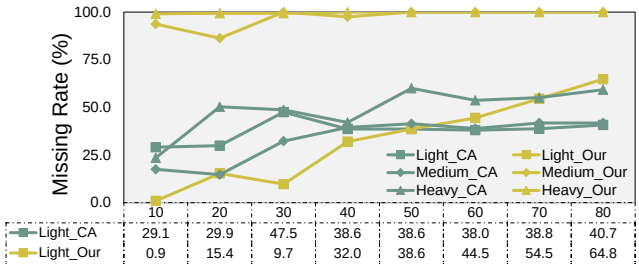


(a) Performance of vehicle detection across eight driving speeds.



(b) Performance of person detection across eight driving speeds.

Figure 9: The performance (MR and FNR) of vehicle detection and person detection at different noise levels and driving speeds. Here, "vehicle" includes cars, trucks, and buses, and "N/A" indicates no object (person) detected.



	10	20	30	40	50	60	70	80
Light_CA	29.1	29.9	47.5	38.6	38.6	38.0	38.8	40.7
Light_Our	0.9	15.4	9.7	32.0	38.6	44.5	54.5	64.8
Medium_CA	17.5	14.7	32.3	39.5	41.4	38.9	41.8	41.8
Medium_Our	93.7	86.3	100.0	97.5	100.0	100.0	100.0	100.0
Heavy_CA	23.5	50.3	48.7	42.1	60.0	53.7	55.1	59.3
Heavy_Our	99.1	99.4	99.4	100.0	100.0	100.0	100.0	100.0

Figure 10: The variation of the missing rate (MR) as vehicle speed increases. The yellow curve indicates changes in CARLA's built-in precipitation data, while the green curve indicates changes in the CARLA data with generated noise.

4.3 Evaluation on the Robotic Vehicle

After testing on the noise-degraded BDD100K dataset (Section 4.1) and through CARLA simulation (Section 4.2), we further address RQ1 using physical testbeds, specifically our assembled robotic vehicle (shown in Fig. 4(b)).

4.3.1 Experiment Preparation on the Robotic Vehicle. SDVs have over 150 million lines of software code for various applications, far surpassing the codebase of traditional IoT devices [24, 59, 62].

To better manage SDV applications and their extensive codebase, we introduce the service-oriented architecture (SOA) [48] to the robotic vehicle. This allows applications to be decomposed into Docker containers for deployment and orchestration on vehicle computing units. We also use ROS 2 on the robotic vehicle, enabling control of the vehicle's wheels by publishing new topics, thus maintaining the target speed [44]. We also deploy the proportional-integral-derivative (PID) method [44] to ensure the robotic vehicle follows a predefined path and speed.

In addition, considering the trade-off between accuracy and inference time for vehicle applications [1, 32], we utilize SSD model [38] with MobileNetv3 on the robotic vehicle, with weights trained on the COCO dataset. We also integrate the TensorFlow Object Detection API [47] to process video footage and record the results.

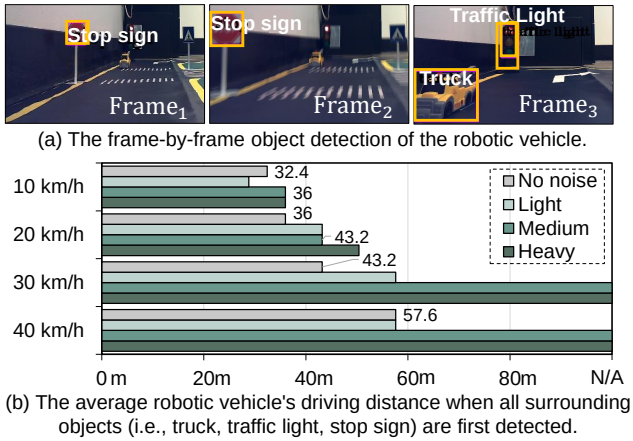
4.3.2 Experiment Groups. As explained in Section 3.3, for testing on the robotic vehicle, we collect autopilot track map data recorded by the driving robotic vehicles. Similar to the previous tests, we create three additional noise-degraded datasets by masking the collected data with different noise levels: light, medium, and heavy.

4.3.3 Evaluation Metrics. When testing on the robotic vehicle in the constrained testing area, the detection rate and MR evaluation metrics are ineffective due to the limited size of the 2.8m × 3.2m autopilot track (as shown in Fig. 4(b) of Section 2.1 and Fig. 11(a)). The confined space restricts the robotic vehicle's ability to encounter and detect a diverse range of objects, leading to a skewed detection rate that does not accurately represent real-world conditions.

To address this, we record and calculate the average driving distance at which the robotic vehicle first detected all surrounding objects (e.g., truck, stop sign, traffic light as shown in Fig. 11(a)) for each experiment at different speeds, instead of using the previous MR metric. This adjustment is inspired by prior research showing that braking distance is related to driving speed [8, 66], and by the key principle of advanced driver assistance systems (ADAS), where early detection of obstacles is critical for decision-making [39].

As shown in Fig. 11(b), the driving distance to detect the first object reflects the impact of in-transit noise on the SSD algorithm's effectiveness: a longer detection distance indicates greater difficulty for the algorithm in identifying objects under noise conditions for that experimental group. In this context, the distance is calculated as $Index_{Fr} \times v$, where $Index_{Fr}$ denotes the video frame index at which the first object is detected, and v represents the robotic vehicle's original speed (Fig. 11(b)). The robotic vehicle speed is mapped to real-world driving speed, enabling consistent SSD algorithm evaluations under varying in-transit noise. A frame rate of 1 FPS ensures consistent detection distance measurements.

4.3.4 Evaluation Results. Figure 11 (a) shows frame-by-frame SSD detection on the robotic vehicle, while Fig. 11(b) illustrates how



(a) The frame-by-frame object detection of the robotic vehicle.

(b) The average robotic vehicle's driving distance when all surrounding objects (i.e., truck, traffic light, stop sign) are first detected.

Figure 11: The impact of driving speed and noise levels on object detection in robotic vehicles. Here, N/A indicates that the entire route was driven without detecting any objects.

noise levels and vehicle speed impact the detection distance for surrounding objects. Longer average distances indicate poorer detection ability. From Fig. 11 (b), we observe that at 10 km/h, the vehicle is resilient to all noise levels, showing the shortest detection distances. At 20 km/h, the detection distance increases slightly, reflecting a minor decline in effectiveness. Generally, higher speeds and heavier noise lead to longer detection distances. At 30 km/h and 40 km/h under medium and heavy noise, the robotic vehicle fails to detect any objects along the route, marked as N/A. This shows that such noise levels severely impair detection effectiveness at these speeds. This aligns with prior results from the noise-degraded BDD100K dataset (Fig. 9) and CARLA simulation (Fig. 10)

5 TRANSITNET: "ONE-FOR-ALL"

In Sec 1.2 we analyze noise patterns at low and high speeds, showing how they misdirect model attention and degrade fine-grained details, leading to incorrect and missed detections. Small objects are disproportionately affected, with experiments revealing that YOLOv9 struggles more with detecting traffic lights and pedestrians than vehicles under moderate noise (Fig.8, Fig.9) due to their smaller pixel occupancy [11]. To address this, we propose TransitNet to enhance vision task performance in noisy environments.

5.1 Proposed Methodology

5.1.1 YOLOv9 Structure and Challenges. Similar to its predecessors, YOLOv9 [63, 65] comprises three main components: backbone network, feature pyramid network (FPN) [34], and head network. Specifically, there are two key innovations: programmable gradient information (PGI) and a generalized efficient layer aggregation network (GELAN) to mitigate feature degradation in its design. GELAN enhances feature aggregation within backbone, making extracted features critical for subsequent processing. These features are utilized by FPN for multi-scale integration and by PGI in auxiliary reversible branches for hierarchical feature retention. Together, these enhancements improve the backbone's ability to extract high-quality features, boosting model performance.

However, YOLOv9 struggles with object detection in noisy environments, particularly small ones [45, 65]. This limitation is partly due to conventional downsampling operations (ADown) in backbone, which introduce information loss and increase the likelihood of missed detections. Likewise, traditional convolution in the GELAN module further degrades shallow spatial details, making objects undetectable [70]. Moreover, YOLOv9 may overlook the real-time SDV constraints, which require high processing speeds and low memory overhead [37]. One potential factor contributing to this limitation is GELAN's design, with its multiple parallel convolutional paths and concatenation operations, significantly increasing computational complexity and memory usage [67, 3], posing challenges for real-time deployment.

5.1.2 TransitNet. To address these challenges, we propose **TransitNet** (In-Transit Noise Adaptive Network), inspired by YOLOv9, integrating SPD-Conv and Self-Calibrated Convolution (SC-Conv) (Fig. 12). TransitNet enhances the YOLOv9 backbone through a dual approach: *i*) replacing standard convolutions with SPD-Conv, and *ii*) substituting GELAN in the ninth layer with SC-Conv. SPD-Conv preserves spatial information during downsampling, while SC-Conv improves computational efficiency and feature representation.

(1) SPD-Conv module. The loss of spatial distribution variance and inability to preserve feature intensity are major limitations of ADown, causing ineffective object detection. To address these issues, the SPD-Conv method [56] incorporates strided convolutions and pooling layers, utilizing a space-to-depth layer followed by a non-strided convolution to enhance feature representation.

Space-to-depth layer: This technique enables the transformation method [53] to downsample feature maps within a CNN. Given an $S \times S \times C_1$ feature map donated by X, the process involves slicing out a sequence of sub-feature maps based on a specified scale value (Fig. 12, left-bottom).

Non-strided convolutional layer: After the space-to-depth operation, the intermediate feature map X' with dimensions $(S/2,\ S/2,\ 4C_1)$ proceeds to a non-strided convolutional layer with a stride of 1. This layer uses C_2 filters, where $C_2 < 4C_1$. The stride of 1 preserves discriminative feature information, ensuring symmetrical sampling from each row or column of the feature map and maintaining the size of each sampled pixel by using an $n \times n$ filter (e.g., 3×3).

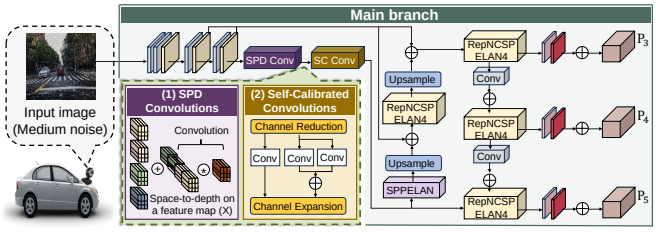


Figure 12: The TransitNet architecture comprises two key modules: SPD-Conv and SC-Conv. The SPD-Conv module provides detailed operational functionality, while the SC-Conv module enhances adaptability and robustness by dynamically adjusting feature weights in response to the input.

(2) SC-Conv module. It comprises four stages: channel reduction, convolution, latent space calibration, and channel expansion [36]. Channel reduction lowers complexity by reducing feature

map channels while retaining key information. Convolution enhances feature extraction, enriching representation. Latent space calibration down-samples and refines the feature map to minimize information loss. Channel expansion restores channel count, ensuring input-output consistency, and making it ideal for YOLOv9 integration. Compared to GELAN, it provides a lightweight design with precise feature calibration and layer-by-layer integration. It minimizes overhead by reducing inter-layer connections and complex aggregations, enabling efficient feature adjustment for resource-constrained environments.

5.2 Testing of TransitNet

To address RQ3, we assess the accuracy and inference speed of TransitNet by comparing it with multiple mainstream and state-of-the-art methods, such as SSD [38], RT-DETR [71], D-FINE [49], etc. We evaluate across different datasets, considering scenarios involving inclement weather, unexpected occlusions, and varying driving speeds. Moreover, we test TransitNet using our real-world driving dataset, which includes a wide range of speeds from 0km/h to 96km/h across both urban and highway environments.

5.2.1 Testing with Inclement Weather Noise. Based on our previous finding that medium-level inclement weather noise significantly affects object detection in SDV (Fig. 9), we aim to further quantify the effectiveness of our proposed TransitNet model in handling datasets most affected by this type of noise.

Testing metrics. We train each model, including TransitNet, Faster-RCNN [50], SSD [38], D-FINE [49], RT-DETR [71], and YOLOv9 [63] models specifically on medium noise-degraded images from the BDD100K dataset. These models are tasked with detecting vehicles, traffic lights, and pedestrians. We use the original (untrained) YOLOv9's detections as GT. Predictions with an intersection-over-union (IoU) [51] greater than 0.5 are considered true positives (TP). Evaluation metrics include precision, F-measure, mAP@50, Giga Floating Point Operations (GFLOPs), and FPS, defined as follows:

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \qquad \text{Recall} = \frac{\text{TP}}{\text{TP+FN}}$$
$$\text{F-measure} = \frac{2*\text{Precision}*\text{Recall}}{\text{Precision+Recall}} \qquad \text{mAP@50} = \frac{1}{\text{N}} \sum_{\text{i=1}}^{\text{N}} \text{AP@50}_\text{i}$$

Here, N denotes the total number of categories, and mAP@50 represents the mean average precision at an IoU threshold of 0.5. In machine learning, GFLOPs measure the total floating-point operations (in billions) a model performs, indicating its computational complexity and resource demands. Higher GFLOPs generally imply more intensive computation. Additionally, a higher FPS signifies faster processing, essential for SDV applications. To standardize computational complexity comparisons, we use a 640×640 input size for all models. Each convolution operation involves a multiplication and addition, so we double the GFLOPs count to reflect both, following the convention in YOLOX [6, 9, 16].

(1) Quantitative results. Table 1 provides a comprehensive evaluation, divided into two sections: *i*) a comparison of TransitNet with mainstream and SOTA methods, presented in the upper section above the dashed line; and *ii*) ablation study results in the lower section, analyzing the impact of specific modifications within YOLO model variants, including YOLO-SPD, YOLO-SC, and TransitNet.

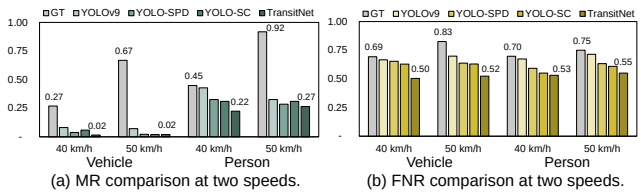
As shown in Table 1, the comparison experiments reveal that TransitNet surpasses all other models in accuracy. Specifically, TransitNet achieves the highest mAP@50 (0.87), outperforming D-FINE-X (0.81) and RT-DETR (0.78). As mentioned before, there is a trade-off between accuracy and speed in SDV. Although SSD achieves the highest FPS (129.33), it falls short in detection accuracy relative to the YOLO-series models. Therefore, we focus on the YOLO series models in the following evaluations and further discuss the reasons behind their performance in Sec. 6.

Table 1: Comparative analysis of multiple models and ablation study of YOLO variants across key evaluation metrics.

Model	F-measure	mAP@50	GFLOPs	FPS
Faster-RCNN [14]	0.11	0.28	201.08	68.28
SSD [38]	0.44	0.49	281.97	129.33
D-FINE-L [49]	0.47	0.51	91.674	45.12
RT-DETR [71]	0.72	0.78	259.31	67.01
D-FINE-X [49]	0.77	0.81	205.42	36.06
YOLOv9 [63]	0.79	0.84	237.70	47.62
YOLO-SPD	0.81	0.85	243.70	60.98
YOLO-SC	0.80	0.84	239.14	65.79
TransitNet **(Ours)**	**0.83**	**0.87**	236.40	74.63

Among the ablation study, each variant demonstrates improvements over YOLOv9, with TransitNet showing particular advantages by combining the strengths of YOLO-SPD and YOLO-SC (Table 1). Specifically, YOLO-SPD achieves a high F-measure (0.81) and mAP@50 (0.85), while YOLO-SC offers an optimal balance between accuracy and efficiency, reaching the highest mAP@50 (0.84), a solid F-measure (0.80), competitive FPS (65.79), and lower GFLOPs (239.14) compared to YOLO-SPD (243.70). TransitNet surpasses both, achieving the highest F-measure (0.83) and mAP@50 (0.87) across all models, while maintaining an efficient FPS (74.63) and a relatively low GFLOPs (236.4). These results underscore the effectiveness of YOLO-SPD, YOLO-SC, and especially TransitNet in delivering both high accuracy and computational efficiency.

(2) Testing with critical driving speeds. Previous research highlights that the performance gap in MR and FNR between driving speeds of 40km/h and 50km/h is particularly significant compared to other speed ranges, as illustrated in Fig. 9. Building upon this finding, we conduct an in-depth evaluation of the MR and FNR of each model for vehicle and person detection at these two speeds under medium-level inclement weather conditions, as shown in Fig. 13. The result reveals that our proposed methods outperform YOLOv9 in both MR and FNR across the two speeds, with particularly low MR values in person detection at 50 km/h, highlighting the robustness of the TransitNet model with noises.



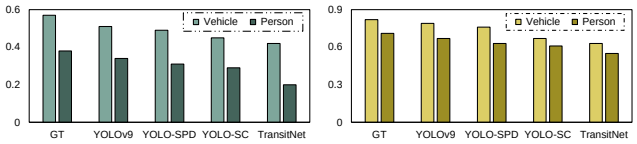
(a) MR comparison at two speeds. (b) FNR comparison at two speeds.

Figure 13: The MR and FNR of the vehicle and person detection at two important speeds (40km/h and 50km/h) with medium-level inclement weather noise.

As shown in Fig. 13, YOLOv9 experiences a significant increase in both MR and FNR at higher speeds, while our proposed models

(YOLO-SPD, YOLO-SC, and TransitNet) consistently maintain or improve performance. For vehicle detection, YOLO-SPD and YOLO-SC keep the MR as low as 0.02 at both 40km/h and 50km/h, whereas YOLOv9's MR rises sharply to 0.67 at 50km/h. Among our models, TransitNet achieves the best overall performance, maintaining low MR and FNR values across both vehicle and person detection, even at higher speeds. For person detection, TransitNet holds steady MR values and achieves the lowest FNR, significantly outperforming YOLOv9, which reaches 0.75 at 50km/h.

5.2.2 Testing with Unexpected Occlusion Noise. Next, we focus on testing the proposed TransitNet algorithm under unexpected occlusion conditions. Fig. 14 illustrates MR and FNR for vehicle and person detection under unexpected occlusion noise, comparing the YOLOv9-Ori, YOLOv9, YOLO-SPD, YOLO-SC, and TransitNet.



(a) Vehicle and person MR in occlusion. (b) Vehicle and person FNR in occlusion.

Figure 14: The MR (a) and FNR (b) of vehicle and person detection with unexpected occlusion noise.

In this test, the dataset is generated with unexpected occlusion noise caused by mud (Fig. 7). As shown in Fig. 14, TransitNet demonstrates a notable improvement in both MR and FNR for vehicle and person detection compared to YOLOv9-Ori and YOLOv9. These results highlight that TransitNet significantly enhances detection accuracy under unexpected mud occlusion noise, outperforming YOLOv9 in both vehicle and person detection.

5.2.3 Testing with Real-World Driving Datasets. As detailed in Section 3.3, we collect diverse real-world driving videos using an internal camera. These videos capture driving scenarios under both sunny and rainy conditions along identical routes in urban and highway settings, varying in traffic volumes and densities. Hence, we further validate our proposed solution TransitNet using this comprehensive real-world driving dataset, covering a wide range of driving speeds from 0km/h to 96km/h. The detection results of YOLOv9 in sunny datasets are treated as the baseline.

(1) Quantitative analysis. Table 2 compares the detection rates of YOLOv9 (baseline) and TransitNet in real-world rainy driving datasets across urban and highway environments. TransitNet shows significantly superior detection performance under rainy conditions in Table 2. In this context, a lower MR reflects enhanced model performance. Specifically, TransitNet achieves nearly twice the reduction in MR compared to YOLOv9 in urban environments for both vehicles and traffic lights. On highways, TransitNet further boosts vehicle detection accuracy by approximately 40% compared to YOLOv9, achieving a substantial MR reduction from 95.24% to 4.77%. We also perform similar experiments in sunny conditions covering identical routes in both urban and highway. Our model also consistently outperforms YOLOv9 on sunny days.

(2) Detection visualization. Figure 15 presents detection examples in real-world rainy-day scenarios under both urban and highway conditions. In the urban scenario, our model successfully recognizes

Table 2: MR of YOLOv9 (baseline) and TransitNet (our solution) in real-world rainy driving datasets.

Model	Urban	Highway	MR (Vehicle)	MR (Traffic Light)
YOLOv9	✓		81.50%	71.20%
TransitNet (Ours)	✓		**58.93%**	**49.64%**
YOLOv9		✓	51.92%	95.24%
TransitNet (Ours)		✓	**32.69%**	**4.77%**

traffic lights that YOLOv9 misses. Similarly, in the highway scenario, our model outperforms YOLOv9 in detecting vehicles under heavy rain, where YOLOv9 fails to detect them effectively. Thus, both Table 2 and Fig. 15 underscore TransitNet robustness and superior performance in challenging real-world conditions.

(a) Detection examples under urban scenario. (b) Detection examples under highway scenario.

Figure 15: The real-world detection examples of YOLOv9 and our model under urban and highway conditions.

6 OBSERVATIONS AND DISCUSSIONS

In this section, we illustrate and summarize our answers to **RQ1**, **RQ2**, and **RQ3**, discuss the main observations, and present our explanation for our experiment results and observed trends.

★ **Observation 1:** *As vehicle speed increases, the camera-based application's ability to accurately detect objects diminishes, with a more severe impact on person detection compared to vehicle detection.*

This observation answers **RQ1** and is supported by Fig. 9 on BDD100k, Fig. 10 on CARLA, and Fig. 11(b) on the robotic vehicle.

With rising speed, both vehicle and person detection experience increases in MR and FNR (Fig. 9) as well as the decreases in detection rate (Fig. 10), and the impact is more pronounced for pedestrian detection (Fig. 9). Similarly, we observe that at low speeds, the vehicle demonstrates resilience to all noise levels, maintaining the shortest detection distances (Fig. 11(b)). However, at high speeds, the robotic vehicle increasingly struggles to detect objects along the entire route, often failing to identify any (Fig. 11(b)).

Discussion: At high speeds, capturing clear images and retaining key information becomes challenging, even without noise. SDVs must address these issues, particularly under inclement weather and unexpected occlusion conditions. High-speed motion exacerbates these challenges, as rapid movement can introduce motion blur and reduce the effectiveness of object detection algorithms.

★ **Observation 2:** *40km/h may be considered a threshold for safe urban driving speed, to ensure accurate camera-based applications in SDVs under different noisy conditions.*

This observation also answers **RQ1** and is backed by both Fig. 9, Fig. 10, and Fig. 11(b). It shows that when speeds exceed 40km/h, especially under medium and high noise levels, both MR and FNR for vehicle and person detection increase sharply (Fig. 9), while the detection rate drops significantly (Fig. 10). Besides, at 40km/h under medium and heavy noise scenarios, the robotic vehicle fails to detect any objects along the entire route (Fig. 11(b)).

Discussion: Vehicle motion amplifies inclement weather noise (raindrops and snowflakes). Though initially small, these particles accumulate within a grid, intensifying their impact and disrupting

the model's learning from shallow layers, where initial feature extraction occurs. This observation highlights the need for models to handle medium-sized noise effectively, which is often overlooked. While individual particles seem insignificant, their collective impact challenges detection, especially at speeds over 40 km/h. Higher speeds increase noise frequency and density, degrading detection and emphasizing robust noise handling in model design.

⋆ **Observation 3:** *The impact of inclement weather noise is more significant than unexpected occlusion noise for SDVs, particularly when the vehicle is in motion.*

This observation answers **RQ1** and is supported by comparisons among Fig. 8, Fig. 9(a), Fig. 9(b), and Fig. 14, suggesting that sparse and high-intensity noise (inclement weather noise) can pose significant challenges for SDVs.

Explanation and discussion: When the vehicle is static, both person and vehicle detection are impeded, though the severity decreases in that order (Fig. 8 and Fig. 14). This is because YOLO struggles to detect objects occupying small pixel areas due to the lack of contextual information these targets present.

In addition, higher speeds increase the likelihood of inclement weather (raindrops or snowflakes) covering the entire lens, making weather conditions more problematic than unexpected occlusions (Fig. 9(a), Fig. 9(b), and Fig. 14). While unexpected occlusion noise typically only affects a specific area, inclement weather noise is dynamic and impacts the entire screen. This type of noise disrupts the relationships between grid cells, particularly when detecting objects such as pedestrians, making detection more challenging.

⋆ **Observation 4:** *Compared to YOLO variants, TransitNet achieves higher accuracy and FPS, maintaining performance in challenging conditions. It closes the detection gap between 40 km/h and 50 km/h in noisy environments, significantly reducing MR and FNR for vehicle and pedestrian detection, even under occlusions and inclement weather conditions. TransitNet also ensures robust and consistent performance across both urban (low-speed) and highway (high-speed) scenarios.*

This observation confirms **RQ2** and **RQ3**, supported by Table. 1, Fig. 13 (inclement weather situation), Fig. 14 (unexpected occlusion) and Fig. 15 (real-world rainy day data).

Discussion: Replacing the original ADown operation with SPD-Conv enhances our proposed model's ability to detect more objects (e.g., traffic lights and pedestrians) during driving by preventing the loss of spatial distribution variance and preserving the feature intensity, both of which are critical for accurate small object detection. Additionally, the SC-Conv module addresses a common limitation of channel attention mechanisms, which tend to prioritize objects with larger areas or more prominent features while overlooking small object features. Specifically, SC-Conv overcomes the limitations of standard convolutions (e.g., 3×3 convolutions) in the GELAN module, which primarily captures local features, by introducing self-calibrated weights. As a result, TransitNet's enhanced ability to identify relational patterns significantly improves detection performance, in noisy and rainy conditions, addressing diverse camera noise challenges.

⋆ **Observation 5:** *The success of TransitNet lies in striking a balance between high detection accuracy and fast processing speeds, achieved through comparisons with mainstream and advanced models.*

This observation confirms **RQ3** and is affirmed by Table. 1.

Discussion: The precision gap between the two-stage detector (Faster R-CNN) and one-stage models (SSD and YOLO) on the BDD100K dataset highlights architectural differences. Faster R-CNN is more prone to data labeling errors and demands greater computational resources [35], limiting its performance in complex traffic scenarios. While SSD shares YOLO's one-stage design, it falls short in detection accuracy, making it less suitable for challenging conditions. Though RT-DETR demonstrates strong global feature modeling capabilities, its higher computational cost and limited performance on small object detection present challenges in resource-constrained and fine-grained detection scenarios.

D-FINE-L, a lightweight version of D-FINE, has lower computational demands, making it suitable for resource-constrained environments. However, as shown in the Table. 1, it falls short in accuracy, which may limit its accuracy in more demanding detection tasks. Generally, lower GFLOPs result in higher FPS, as reduced computation per frame speeds up processing. However, FPS also depends on the network's structural design and optimization. This explains why D-FINE-L, despite achieving the lowest GFLOPs (91.67), has the lower FPS (45.12). D-FINE-X focuses on higher detection accuracy, showing an improvement over D-FINE-L. Yet, this comes at the cost of computational efficiency, resulting in reduced real-time performance. It is more suitable for scenarios where accuracy is prioritized over strict real-time requirements.

Compared to the above models, YOLO's architecture excels in robust feature extraction and adapts well to diverse, noisy environments, achieving high precision [58, 31]. However, it faces challenges in detecting small objects, often missing traffic lights or pedestrians, and its FPS performance remains suboptimal. TransitNet is specifically designed to overcome its limitations in detecting small objects and managing dynamic noise efficiently to meet real-time tasks. The proposed model introduces enhanced relational pattern modeling, enabling more accurate detection of small-scale targets even under inclement weather or unexpected occlusion noise. By incorporating advanced feature aggregation and adaptive noise handling mechanisms, TransitNet demonstrates superior accuracy in addressing in-transit challenges, managing diverse camera noise and real-world rainy conditions across low-speed and high-speed scenarios. This adaptability makes it particularly effective for complex scenarios where traditional models struggle, ensuring both robustness and precision in object detection.

7 RELATED WORK

Recent advancements in deep learning and sensor technology have advanced autonomous driving to levels 3 and 4 [27, 26, 25, 10]. However, severe weather conditions like fog, rain, snow, sandstorms, and low light still degrade image quality, with effects such as raindrop patterns blurring details and snow obscuring objects. While Tesla claims level 4 autonomy, reports [19, 20] highlight challenges in such conditions. Studies now focus on enhancing images [60] and improving algorithm resilience [29] to increase robustness. Building on these efforts, previous studies have developed specialized

solutions for weather-related challenges. One study has proposed approaches for vehicle detection and tracking in adverse weather conditions [18]. Other studies narrow in on single-weather models to address specific weather conditions [54, 33, 15, 34, 64]. However, these works overlook the critical influence of vehicle speed on detection performance.

To the best of our knowledge, prior research has neither explicitly evaluated nor quantified the impact of vehicle speed on perception. A notable exception is only one prior study [66] has incorporated vehicle speed during experimental testing; however, it treats speed merely as a parameter without providing a detailed quantitative analysis of its effects. To bridge this gap, our work provides an extensive analysis of how noise patterns at varying speeds affect camera-based perception, along with quantitative evaluations to ensure the safe and efficient operation of SDVs at higher speeds.

8 CONCLUSION

This study provides an extensive analysis of the impact of vehicle speed and in-transit camera noise on vision tasks, utilizing diverse testing environments, involving BDD100K dataset, CARLA simulator, a robotic vehicle, and a real vehicle. Our findings highlight that increased speed amplifies the challenges of object detection, with a notable performance drop at speeds above 40km/h, particularly in detecting smaller objects like pedestrians. To address these challenges, we introduce TransitNet, a novel model integrating SPD-Conv and SC-Conv, excelling in object detection under adverse weather and occlusion noise, with extensive testing confirming its robustness and generalizability. Designed as a general framework, TransitNet has the potential to be applied across various domains requiring small object detection.

ACKNOWLEDGEMENT

The authors are thankful to the reviewers and our shepherd for their constructive comments and suggestions. This work is supported in part by the National Science Foundation (NSF) grant CNS-2348151 and Commonwealth Cyber Initiative (CCI) grant HC-3Q24-048.

REFERENCES

[1] Amir Pooyan Afghari, Md Mazharul Haque, and Simon Washington. 2020. Applying a joint model of crash count and crash severity to identify road segments with high risk of fatal and serious injury crashes. *Accident Analysis & Prevention*, 144, 105615.

[2] Mohiuddin Ahmed, Raihan Seraj, and Syed Mohammed Shamsul Islam. 2020. The k-means algorithm: a comprehensive survey and performance evaluation. *Electronics*, 9, 8, 1295.

[3] Mujadded Al Rabbani Alif and Muhammad Hussain. 2024. Yolov1 to yolov10: a comprehensive review of yolo variants and their application in the agricultural domain. *arXiv preprint arXiv:2406.10139*.

[4] APTIV. 2020. White paper: smart vehicle architecture overview (online). https://www.aptiv.com/en/insights/article/white-paper-smart-vehicle-architecture-overview. (Mar. 2020).

[5] Konstantinos Arakadakis, Pavlos Charalampidis, Antonis Makrogiannakis, and Alexandros Fragkiadakis. 2021. Firmware over-the-air programming techniques for IoT networks-a survey. *ACM Computing Surveys (CSUR)*, 54, 9, 1–36.

[6] Andrea Asperti, Davide Evangelista, and Moreno Marzolla. 2021. Dissecting flops along input dimensions for greenai cost estimations. In *International Conference on Machine Learning, Optimization, and Data Science*. Springer, 86–100.

[7] Flavio Bonomi and Adam T Drobot. 2023. Infrastructure for digital twins: data, communications, computing, and storage. In *The Digital Twin*. Springer, 395–431.

[8] 2023. Brake distance (online). http://www.csgnetwork.com/. (2023).

[9] Jierun Chen, Shiu-hong Kao, Hao He, Weipeng Zhuo, Song Wen, Chul-Ho Lee, and S-H Gary Chan. 2023. Run, don't walk: chasing higher flops for faster neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 12021–12031.

[10] Long Chen et al. 2022. Milestones in autonomous driving and intelligent vehicles: survey of surveys. *IEEE Transactions on Intelligent Vehicles*, 8, 2, 1046–1056.

[11] Tausif Diwan, G Anirudh, and Jitendra V Tembhurne. 2023. Object detection using yolo: challenges, architectural successors, datasets and applications. *multimedia Tools and Applications*, 82, 6, 9243–9275.

[12] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. 2017. Carla: an open urban driving simulator. In *Conference on robot learning*. PMLR, 1–16.

[13] Anurag Dutt. 2023. Evaluating the energy impact of device and workload parameters for dnn inference on edge rpe report.

[14] RCNN Faster. 2015. Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 9199, 10.5555, 2969239–2969250.

[15] Xueyang Fu, Jiabin Huang, Delu Zeng, Yue Huang, Xinghao Ding, and John Paisley. 2017. Removing rain from single images via a deep detail network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 3855–3863.

[16] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. 2021. Yolox: exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*.

[17] Oliver Haslam. 2023. Mercedes-Benz rolls its flagship EV's infotainment system out to other cars via ota update (online). https://www.pocket-lint.com/mercedes-benz-rolls-its-flagship-evs-infotainment-system-out-to-other-cars-via-ota-update/. (Feb. 2023).

[18] Mahmoud Hassaballah, Mourad A Kenk, Khan Muhammad, and Shervin Minaee. 2020. Vehicle detection and tracking in adverse weather using a deep learning framework. *IEEE transactions on intelligent transportation systems*, 22, 7, 4230–4242.

[19] Andrew J. Hawkins. 2023. Elon musk's fsd v12 demo includes a near miss at a red light and doxxing mark zuckerberg (online). https://www.theverge.com/2023/8/28/23848882/elon-musk-tesla-fsd-v12-demo-red-light-zuckerberg-house. (Aug. 2023).

[20] Andrew J. Hawkins. 2024. Tesla's autopilot and full self-driving linked to hundreds of crashes, dozens of deaths (online). https://www.theverge.com/2024/4/26/24141361/tesla-autopilot-fsd-nhtsa-investigation-report-crash-death. (Apr. 2024).

[21] Yuankai He and Weisong Shi. 2024. A vision for transformative intersections. *Computer*, 57, 12, 58–68.

[22] Jacqueline Henle, Mona Gierl, Houssem Guissouma, Felix Müller, Goutham Bharadwaj Ramesh, and Eric Sax. 2023. Concept for an Approval-Focused Over-The-Air Update Development Process. Tech. rep. SAE Technical Paper.

[23] Emma Himes. 2021. Nhtsa up in the clouds: the formal recall process & over-the-air software updates. *Mich. Tech. L. Rev.*, 28, 153.

[24] Ling Hu and Qiang Ni. 2017. Iot-driven automated object detection algorithm for urban surveillance systems in smart cities. *IEEE Internet of Things Journal*, 5, 2, 747–754.

[25] Yihan Hu et al. 2023. Planning-oriented autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 17853–17862.

[26] Yu Huang and Yue Chen. 2020. Autonomous driving with deep learning: a survey of state-of-art technologies. *arXiv preprint arXiv:2006.06091*.

[27] Silviu Ionita. 2017. Autonomous vehicles: from paradigms to technology. In *IOP Conference Series: Materials Science and Engineering* number 1. Vol. 252. IOP Publishing, 012098.

[28] Reza Jafarpourmarzouni, Yichen Luo, Sidi Lu, Zheng Dong, et al. 2024. Towards real-time and efficient perception workflows in software-defined vehicles. *IEEE Internet of Things Journal*.

[29] Hyeonjae Jeon, YaeOhn Kim, Minyoung Choi, Donggeon Park, Sungho Son, Jungki Lee, Gyeungho Choi, and Yongseob Lim. 2022. Carla simulator-based evaluation framework development of lane detection accuracy performance under sensor blockage caused by heavy rain for autonomous vehicle. *IEEE Robotics and Automation Letters*, 7, 4, 9977–9984.

[30] Günter Keßler, Dominik Sieben, Anand Bhange, and Elmar Börner. 2023. The software defined vehicle–technical and organizational challenges and opportunities. In *International Stuttgart Symposium*. Springer, 414–426.

[31] Boshra Khalili and Andrew W Smyth. 2024. Sod-yolov8–enhancing yolov8 for small object detection in traffic scenes. *arXiv preprint arXiv:2408.04786*.

[32] Xiaoqiang Kong, Subasish Das, Kartikeya Jha, and Yunlong Zhang. 2020. Understanding speeding behavior from naturalistic driving data: applying classification based association rule mining. *Accident Analysis & Prevention*, 144, 105620.

[33] Siyuan Li et al. 2019. Single image deraining: a comprehensive benchmark analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3838–3847.

[34] Yu Li, Robby T Tan, Xiaojie Guo, Jiangbo Lu, and Michael S Brown. 2017. Single image rain streak decomposition using layer priors. *IEEE Transactions on Image Processing*, 26, 8, 3874–3885.

[35] Huixin Liu, Guohua Lu, Mingxi Li, Weihua Su, Ziyi Liu, Xu Dang, and Dongyuan Zang. 2024. High-precision real-time autonomous driving target detection based on yolov8. *Journal of Real-Time Image Processing*, 21, 5, 174.

[36] Jiang-Jiang Liu, Qibin Hou, Ming-Ming Cheng, Changhu Wang, and Jiashi Feng. 2020. Improving convolutional networks with self-calibrated convolutions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10096–10105.

[37] Liangkai Liu, Sidi Lu, Ren Zhong, Baofu Wu, Yongtao Yao, Qingyang Zhang, and Weisong Shi. 2020. Computing systems for autonomous driving: state of the art and challenges. *IEEE Internet of Things Journal*, 8, 8, 6469–6486.

[38] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. 2016. Ssd: single shot multibox detector. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14*. Springer, 21–37.

[39] Zhenguang Liu, Shuang Wu, Shuyuan Jin, Qi Liu, Shijian Lu, Roger Zimmermann, and Li Cheng. 2019. Towards natural and accurate future motion prediction of humans and animals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10004–10012.

[40] Zongwei Liu, Wang Zhang, and Fuquan Zhao. 2022. Impact, challenges and prospect of software-defined vehicles. *Automotive Innovation*, 5, 2, 180–194.

[41] Lurah Lowery. 2023. OEMshift to OTArecall fixes predicted to occur by 2028 (online). https://www.repairerdrivennews.com/2023/05/09/oem-shift-to-ota-recall-fixes-predicted-to-occur-by-2028/. (May 2023).

[42] Sidi Lu, Nejib Ammar, Akila Ganlath, Haoxin Wang, and Weisong Shi. 2022. A comparison of end-to-end architectures for connected vehicles. In *2022 Fifth International Conference on Connected and Autonomous Driving (MetroCAD)*. IEEE, 72–80.

[43] Yichen Luo, Yongtao Yao, Junzhou Chen, Sidi Lu, and Weisong Shi. 2024. An efficient data transmission framework for connected vehicles. In *2024 IEEE/ACM Symposium on Edge Computing (SEC)*. IEEE, 306–320.

[44] Rajesh Kannan Megalingam, Deepak Nagalla, Katta Nigam, Vamsi Gontu, and Phanindra Kumar Allada. 2020. Pid based locomotion of multi-terrain robot using ros platform. In *2020 Fourth International Conference on Inventive Systems and Control (ICISC)*. IEEE, 751–755.

[45] Jianjun Ni, Shengjie Zhu, Guangyi Tang, Chunyan Ke, and Tingting Wang. 2024. A small-object detection model based on improved yolov8s for uav image scenarios. *Remote Sensing*, 16, 13, 2465.

[46] Jörg Ohlsen. 2022. The software-defined vehicle is overwhelming the automotive industry. *ATZelectronics worldwide*, 17, 6, 56–56.

[47] Bo Pang, Erik Nijkamp, and Ying Nian Wu. 2020. Deep learning with tensorflow: a review. *Journal of Educational and Behavioral Statistics*, 45, 2, 227–248.

[48] Marco Pau, Markus Mirz, Jan Dinkelbach, Padraic Mckeever, Ferdinanda Ponci, and Antonello Monti. 2022. A service oriented architecture for the digitalization and automation of distribution grids. *IEEE Access*, 10, 37050–37063.

[49] Yansong Peng, Hebei Li, Peixi Wu, Yueyi Zhang, Xiaoyan Sun, and Feng Wu. 2024. D-fine: redefine regression task in detrs as fine-grained distribution refinement. *arXiv preprint arXiv:2410.13842*.

[50] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28.

[51] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. 2019. Generalized intersection over union: a metric and a loss for bounding box regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 658–666.

[52] Weibin Rong, Zhanjing Li, Wei Zhang, and Lining Sun. 2014. An improved canny edge detection algorithm. In *2014 IEEE international conference on mechatronics and automation*. IEEE, 577–582.

[53] Mehdi SM Sajjadi, Raviteja Vemulapalli, and Matthew Brown. 2018. Frame-recurrent video super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 6626–6634.

[54] Christos Sakaridis, Dengxin Dai, and Luc Van Gool. 2018. Semantic foggy scene understanding with synthetic data. *International Journal of Computer Vision*, 126, 973–992.

[55] Scott E Stevens, Carl J Schreck, Shubhayu Saha, Jesse E Bell, and Kenneth E Kunkel. 2019. Precipitation and fatal motor vehicle crashes: continental analysis with high-resolution radar data. *Bulletin of the American Meteorological Society*, 100, 8, 1453–1461.

[56] Raja Sunkara and Tie Luo. 2022. No more strided convolutions or pooling: a new cnn building block for low-resolution images and small objects. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 443–459.

[57] Dhananjay Theckedath and RR Sedamkar. 2020. Detecting affect states using vgg16, resnet50 and se-resnet50 networks. *SN Computer Science*, 1, 2, 79.

[58] Duy Anh Tran, Pascal Fischer, Alen Smajic, and Yujin So. 2021. Real-time object detection for autonomous driving using deep learning. *Institute of Computer Science, Department of Computer Science and Mathematics, Goethe University Frankfurt*, 15.

[59] Shreshth Tuli, Nipam Basumatary, and Rajkumar Buyya. 2019. Edgelens: deep learning based object detection in integrated iot, fog and cloud computing environments. In *2019 4th International Conference on Information Systems and Computer Networks (ISCON)*. IEEE, 496–502.

[60] Jeya Maria Jose Valanarasu, Rajeev Yasarla, and Vishal M Patel. 2022. Transweather: transformer-based restoration of images degraded by adverse weather conditions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2353–2363.

[61] Lindsay VanHulle. 2023. GM joins collaborative for vehicle software development (online). https://www.autonews.com/automakers-suppliers/gm-joins-effort-develop-shared-software-auto-industry. (Apr. 2023).

[62] Chengjia Wang, Shizhou Dong, Xiaofeng Zhao, Giorgos Papanastasiou, Heye Zhang, and Guang Yang. 2019. Saliencygan: deep learning semisupervised salient object detection in the fog of iot. *IEEE Transactions on Industrial Informatics*, 16, 4, 2667–2676.

[63] Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. 2024. Yolov9: learning what you want to learn using programmable gradient information. *arXiv preprint arXiv:2402.13616*.

[64] Guoqing Wang, Changming Sun, and Arcot Sowmya. 2019. Erl-net: entangled representation learning for single image de-raining. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5644–5652.

[65] Juan Wang, Hao Yang, Minghu Wu, Sheng Wang, Ye Cao, Shuyao Hu, Jixiang Shao, and Chunyan Zeng. 2024. Ur-yolo: an urban road small object detection algorithm. *Pattern Analysis and Applications*, 27, 4, 121.

[66] Hao Yan, Feng Lin, Jin Li, Meng Zhang, Zhisheng Yan, Jian Xiao, and Kui Ren. 2023. Ghost-probe: nlos pedestrian rushing detection with monocular camera for automated driving. In *Proceedings of the 21st ACM Conference on Embedded Networked Sensor Systems*, 390–402.

[67] Muhammad Yaseen. 2024. What is yolov9: an in-depth exploration of the internal features of the next-generation object detector. *arXiv preprint arXiv:2409.07813*.

[68] Keisuke Yoneda, Naoki Suganuma, Ryo Yanase, and Mohammad Aldibaja. 2019. Automated driving recognition technologies for adverse weather conditions. *IATSS research*, 43, 4, 253–262.

[69] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, Trevor Darrell, et al. 2018. Bdd100k: a diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2, 5, 6.

[70] Xia Zhao, Limin Wang, Yufei Zhang, Xuming Han, Muhammet Deveci, and Milan Parmar. 2024. A review of convolutional neural networks in computer vision. *Artificial Intelligence Review*, 57, 4, 99.

[71] Yian Zhao, Wenyu Lv, Shangliang Xu, Jinman Wei, Guanzhong Wang, Qingqing Dang, Yi Liu, and Jie Chen. 2024. Detrs beat yolos on real-time object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16965–16974.