Testing the Fault-Tolerance of Multi-sensor Fusion Perception in Autonomous Driving Systems

HAOXIANG TIAN*‡, Institute of Software Chinese Academy of Sciences, University of CAS, Beijing , China and Nanyang Technological University, Singapore
WENQIANG DING, Institute of Software Chinese Academy of Sciences, China and University of CAS, Nanjing, Nanjing Institute of Software Technology, China
XINGSHUO HAN, Continental-NTU Corporate Lab, Nanyang Technological University, Singapore
GUOQUAN WU‡†, Institute of Software Chinese Academy of Sciences, University of CAS, Beijing, China and Nanjing Institute of Software Technology, University of CAS, Nanjing, China
AN GUO, Nanyang Technological University, Singapore and Nanjing University, China
JUNQI ZHANG, University of Science and Technology of China, China
WEI CHEN‡, Institute of Software Chinese Academy of Sciences, University of CAS, Beijing, China
JUN WEI‡†, Institute of Software Chinese Academy of Sciences, University of CAS, Beijing, China
TIANWEI ZHANG, Nanyang Technological University, Singapore

Production-level Autonomous Driving Systems (ADSs), such as Google Waymo [5] and Baidu Apollo [7], typically rely on the multi-sensor fusion (MSF) strategy to perceive their surroundings. This strategy increases the perception robustness by combining the respective strengths of the cameras and LiDAR, directly affecting the safety-critical driving decisions of autonomous vehicles (AVs). However, in real-world autonomous driving scenarios, both cameras and LiDAR are prone to various faults that can significantly impact the decision-making and subsequent behaviors of ADSs. It is important to thoroughly test the robustness of MSF during development. Existing testing methods only focus on the identification of corner cases that MSF fails to detect. However, there is still a lack of investigation on how sensor faults affect the system-level behaviors of ADSs.

To address this gap, we present FADE, the ***first*** testing methodology to comprehensively assess the fault tolerance of MSF perception-based ADSs. We systematically build fault models for both cameras and LiDAR

*This work was done at Nanyang Technological University as a visiting student.
†Guoquan Wu and Jun Wei are the corresponding authors
‡Affiliated with Key Lab of System Software at CAS, State Key Lab of Computer Science at Institute of Software at CAS, Beijing. CAS is the abbreviation of Chinese Academy of Sciences

Authors' Contact Information: Haoxiang Tian, Institute of Software Chinese Academy of Sciences, University of CAS, Beijing, China and Nanyang Technological University, Singapore, tianhaoxiang20@otcaix.iscas.ac.cn; Wenqiang Ding, Institute of Software Chinese Academy of Sciences, China and University of CAS, Nanjing, Nanjing Institute of Software Technology, China, dingwenqiang23@otcaix.iscas.ac.cn; Xingshuo Han, Continental-NTU Corporate Lab, Nanyang Technological University, Singapore, xingshuo001@e.ntu.edu.sg; Guoquan Wu, Institute of Software Chinese Academy of Sciences, University of CAS, Beijing, China and Nanjing Institute of Software Technology, University of CAS, Nanjing, China, gqwu@otcaix.iscas.ac.cn; An Guo, Nanyang Technological University, Singapore and Nanjing University, China, guoan218@smail.nju.edu.cn; Junqi Zhang, University of Science and Technology of China, China, jqzh@ustc.edu.cn; Wei Chen, Institute of Software Chinese Academy of Sciences, University of CAS, Beijing, China, chenwei@otcaix.iscas.ac.cn; Jun Wei, Institute of Software Chinese Academy of Sciences, University of CAS, Beijing, China, wj@otcaix.iscas.ac.cn; Tianwei Zhang, Nanyang Technological University, Singapore, tianwei.zhang@ntu.edu.sg.

ACM 2994-970X/2025/7-ARTISSTA035
https://doi.org/10.1145/3728910

in AVs and inject these faults into MSF-based ADSs to test their behaviors in various testing scenarios. To effectively and efficiently explore the parameter spaces of sensor fault models, we design a feedback-guided differential fuzzer to uncover safety violations of ADSs caused by the injected faults. We evaluate `FADE` on Baidu Apollo, a representative and practical industrial ADS. The evaluation results demonstrate the practical values of `FADE`, and disclose some useful findings. We further conduct physical experiments using a Baidu Apollo 6.0 EDU AV to validate these findings in real-world settings.

CCS Concepts: • **Software and its engineering → Software verification and validation**.

Additional Key Words and Phrases: Autonomous Driving System, Fault Tolerance, Simulation Testing

ACM Reference Format:
Haoxiang Tian, Wenqiang Ding, Xingshuo Han, Guoquan Wu, An Guo, Junqi Zhang, Wei Chen, Jun Wei, and Tianwei Zhang. 2025. Testing the Fault-Tolerance of Multi-sensor Fusion Perception in Autonomous Driving Systems. *Proc. ACM Softw. Eng.* 2, ISSTA, Article ISSTA035 (July 2025), 24 pages. https://doi.org/10.1145/3728910

1 Introduction

In autonomous driving systems (ADSs), perception serves as a foundational module, as its output directly affects the safety-critical driving decisions for autonomous vehicles (AVs). Production-level ADSs, such as Google Waymo [5] and Baidu Apollo [7], typically adopt a Multi-Sensor Fusion (MSF)-based perception strategy. It mainly leverages both cameras and LiDAR as the primary sensors to collect images and 3D point cloud data, respectively. These two modal data are processed separately using different deep learning models and subsequently fused to generate the final perception results. Compared to single-sensor perception, MSF improves the overall perception accuracy and increases the tolerance to sensor-specific errors, making ADSs more robust and reliable. However, in the highly complex and dynamic real-world driving environment, sensors are susceptible to various faults during vehicular operations. For example, the camera lens may become obstructed or damaged; the LiDAR may be misaligned due to the vehicle bump. These faults could compromise the quality of sensor data [11, 47]. It remains unknown whether MSF-based ADS can still make safe decisions and actions in such situations. Therefore, it is crucial to test the fault tolerance of MSF-based ADSs under various types of sensor faults.

Existing works [27, 28, 66, 73] primarily focus on generating corner cases of traffic environments to detect the errors of perception models, while overlooking the impacts of perception errors on system-level safety (e.g., decision-making, action control). A few works [14, 15] test the effects of perception errors on AV crashes. However, they generate adversarial sensor inputs from scenarios rather than considering the inherent sensor faults. Secci [57] and Ceccarelli [16] inject camera failures into the ADS to find safety violations. This approach only focuses on the effects of camera faults on the single-sensor (camera-only) ADS, without considering LiDAR faults and MSF in ADSs. One key advantage of MSF is its ability to compensate for the errors in a single sensor. Thus, this approach cannot accurately test and assess the fault tolerance of industrial-grade MSF-based ADS. Additionally, it does not guarantee that the detected safety violations are indeed caused by the injected sensor faults, as some of them may occur even without fault injection.

To bridge this gap and test the system-level fault tolerance of MSF-based ADS against multi-sensor faults, we model real-world camera and LiDAR faults, and inject them into the ADS to identify their resulting safety violations. However, there are two challenges to be addressed:

- **Challenge 1: how to systematically and comprehensively model the sensor faults in real-world traffic for AVs.** The diverse and complex nature of traffic makes it difficult to comprehensively capture the unpredictable conditions affecting cameras and LiDAR on AVs.

- **Challenge 2: how to accurately identify the system-level safety violations caused by the injected sensor faults.** As the ADS is a highly coupled multi-component deep learning system, it is non-trivial to guarantee the discovered safety violations of the ADS indeed arise from the injected sensor faults.

In this paper, we propose FADE, the ***first*** **FA**ult-tolerance testing metho**D** to **E**valuate the multi-sensor perception of ADSs. **To address Challenge 1,** FADE systematically categorizes sensor faults as *active faults* and *passive faults*. It further subdivides these faults based on sensor components. Subsequently, FADE builds the comprehensive fault models for the camera and LiDAR in real-world traffic. **To address Challenge 2,** FADE designs and implements a differential fuzzer for sensor fault injection and system-level fault tolerance testing. This fuzzer evaluates the functional safety of the ADS under sensor faults and identifies their resulting safety violations. Our technical contributions are elaborated below.

1. Sensor Fault Modeling. Specifically, FADE categorizes the sensor faults that possibly occur in real-traffic driving as active and passive ones, and utilizes FMEA [31, 59] to model them from the perspectives of sensor components and environmental factors. *Active faults* arise from the damage of sensors. For example, it could occur when the lens of the camera is damaged by external objects (e.g., stones or debris hitting the lens), resulting in cracks or partial visual obstruction. Similarly, an active fault in LiDAR enclosure may occur when the AV is driving on a bumpy road, causing a shift in the sensor's mounting position. *Passive faults* originate from the objects in the driving environment (e.g., weather, signals) that directly interfere with the normal operation of sensors. For example, the camera may get a passive fault caused by raindrops, snow grains, or mist on its lens. The LiDAR may have a passive fault in the processing unit when the AV encounters a strong light source (such as a high beam), which makes its detector units occupied by strong light.

2. Differential Fuzzer-based Sensor Fault Tolerance Testing. The goal of FADE is to test whether the MSF-based ADS is capable of functioning safely in the presence of sensor faults during AV driving. To achieve this, FADE designs and employs a genetic algorithm (GA)-guided differential fuzzer, to test the ADS with and without sensor faults, and identify its safety violations caused by injected sensor faults.

We demonstrate the effectiveness of FADE on a widely-used industrial L-4 ADS, Baidu Apollo [7], which adopts the MSF perception strategy [6]. The results of simulation experiments show that FADE can effectively and efficiently discover safety violations of Apollo caused by sensor faults. Furthermore, we conduct the ***first*** physical experiments on multi-sensor faults in MSF-based AVs, to validate the authenticity and practical significance of our sensor fault models and the findings from simulation experiments. More than 60% of our found safety violations of Apollo caused by injected sensor faults can be reproduced in physical experiments, which demonstrates that our approach and findings hold substantial relevance for real-world AVs.

In summary, the paper makes the following contributions:

- **Originality.** To the best of our knowledge, we conduct the first exploration on the fault tolerance of MSF-based ADSs. Our findings can help understand how the system-level safety of MSF-based ADSs is affected by sensor faults in real-world traffic.
- **Approach.** We propose FADE, an automated sensor fault injection and sensor fault-tolerance testing approach for MSF-based ADSs. It systematically models sensor faults that AVs may encounter in real-world traffic, and employs a GA-guided differential fuzzer to identify the safety violations of ADSs caused by sensor faults.
- **Evaluation.** We evaluate FADE on the representative industrial MSF-based ADS, Apollo. The results of our simulation experiments demonstrate that FADE can effectively and efficiently

discover safety violations of Apollo caused by injected sensor faults. The results of our physical experiments demonstrate the practical significance of our findings.

2 Background and Related Work

2.1 Perception in Autonomous Vehicles

In ADSs, the perception module relies on various sensors to detect and interpret the surrounding environment [13]. Cameras and LiDAR are the two critical sensors used for this purpose. These sensors form the backbone of the perception module, enabling the AV to perceive the surrounding environment with high fidelity [23, 52].

Camera-based perception. A typical camera mainly comprises five components: lens, camera body, Bayer filter, image sensor, and image signal processor (ISP) [51]. The lens plays a critical role in determining the image quality, focusing light onto the sensor, and enabling image reproduction [9]. The camera body houses and protects internal electronics, while also shielding sensitive parts from environmental exposure. The Bayer filter enables color capture by placing red, green, and blue filters over the sensor's pixels [12, 55]. The image sensor converts captured light into electrical signals, forming the digital image. Finally, the ISP processes this data, enhancing image quality by applying various corrections and producing the final output image [48].

LiDAR-based Perception. A typical LiDAR has five main components, including laser emitter, scanner, receiver, processing unit and LiDAR enclosure [18]. The laser emitter generates laser pulses that are projected into the environment [45]. These pulses reflect off surrounding objects and return to the receiver [40]. The scanner orchestrates the laser's movement to cover a 360-degree field of view or specific sectors, enabling comprehensive spatial mapping. Finally, the processing unit computes the distance and shape of surrounding objects by measuring the time it takes for each pulse to return [10]. LiDAR enclosure is an accessory used to protect the LiDAR lines. Together, these components produce high-resolution, 3D point clouds that enhance the depth perception and allow the system to detect and interpret the vehicle's environment with precision [69].

Multi-Sensor Fusion-based Perception. Cameras capture high-resolution visual data, providing contextual information such as road signs, object appearances and motion changes. This visual input allows for object classification and recognition, which are essential for safe navigation [54]. However, the 2D camera imaging lacks depth information of the 3D driving spaces. LiDAR, on the other hand, uses laser beams to measure distances and generate high-precision 3D point clouds of the traffic environment [67]. This sensor is highly effective in accurately identifying the position and shape of objects, including other vehicles, pedestrians, and obstacles. LiDAR performs well in diverse lighting and weather conditions, which complements the weaknesses of cameras. However, LiDAR struggles to capture detailed texture information (e.g., color), which can be provided by cameras [26]. By integrating LiDAR's depth data with the texture details from cameras, MSF algorithms can enhance object detection beyond the capabilities of either sensor used alone [18, 43].

2.2 Perception Testing of ADSs

The reliability of the perception module is critical for the safety and functionality of ADSs [38], making its testing essential. Existing works focus on two main aspects: (1) testing the perception models and (2) testing the impact of perception errors or sensor failures on ADSs.

2.2.1 Testing Perception Models. Many studies generate adversarial examples or corner cases for the perception models to identify their errors in scenario understanding (e.g., object detection and tracking). These approaches can be categorized into three types based on the perception strategies.

(1) Camera-based perception model testing. (i) Some works [17, 21, 34, 36, 39, 42, 50, 58] generate adversarial perturbations or patches to mislead deep learning-based camera models (e.g., Faster R-CNN, YOLO), particularly targeting real-world objects like traffic signs. (ii) Several works [44, 65, 70, 75] leverage GAN-based perceptual adversarial networks to deceive camera-based perception models. (iii) Additionally, adversarial camouflage patterns [71] are proposed to conceal 3D objects from detection. A metamorphic testing approach [72] is designed to identify inconsistencies in obstacle detection.

(2) LiDAR-based perception model testing. (i) Some studies [19, 64, 68, 74] generate 3D adversarial point clouds to attack LiDAR-based perception models. For example, Zhou et al. [74] employ a metamorphic testing approach combined with fuzzing to detect errors in LiDAR obstacle perception. (ii) Some works [76, 77] identify the impact of critical adversarial locations, and use simple objectives or arbitrary reflective objects to fool LiDAR perception models. (iii) A few works leverage occlusion patterns [60], or affine and weather transformations [32] to generate adversarial inputs to augment LiDAR point clouds. (iv) Moreover, Wang et al. [63] and Li et al [41] use polynomial perturbations on trajectories of NPC vehicles and pedestrians, to test the LiDAR's ability to recognize adversarial dynamic objects and behaviors.

(3) MSF-based perception model testing. Zhong et al. [73] identify fusion errors cased by incorrect multi-sensor data integration. Xiong et al. [66] generate adversarial samples by separately perturbing camera and LiDAR inputs while maintaining data correlation. Gao et al. [28] synthesize real-world data and seek to insert objects into scenarios to uncover perception errors in MSF-based modules, evaluating the perception accuracy under challenging scenarios generated by the metamorphic testing approach. Meanwhile, Gao et al. [27] summarize and implement a range of real-world corruption patterns on the MSF perception module, and test their impacts on the perception results.

However, these approaches primarily focus on generating adversarial examples or corner cases that exploit vulnerabilities in single-modal (camera-based or LiDAR-based) or MSF-based perception models. They do not assess how perception errors propagate to subsequent modules or impact the overall behavior of the ADS.

2.2.2 *Testing the Effects of Perception Errors or Sensor Failures on ADSs.*

A few efforts have examined the effects of perception errors on the system-level consequences of ADSs. Cao et al. [15] categorize different LiDAR spoofing attack patterns from previous studies to assess their impact beyond the perception stage and analyze their influence on the decision-making of ADSs. Additionally, Cao et al. [14] manipulate the shape of 3D meshes by altering vertex positions, synthesizing point clouds and camera images to mislead the ADS into failing to detect objects, ultimately causing crashes. However, these works primarily generate adversarial perception inputs from scenarios to induce the ADS's errors, rather than considering sensor faults that arise in real-world traffic.

Secci [57] and Ceccarelli [16] inject camera failures into the ADS and test its behaviors to find safety violations. However, this approach has two key limitations: (1) it only targets camera-based ADS, disregarding the role of LiDAR and MSF in mitigating camera faults for ADSs; (2) It does not verify whether the identified safety violations are genuinely caused by injected faults. Therefore, it fails to comprehensively and accurately evaluate the fault tolerance of MSF-based ADSs.

Different from existing works, our study is the **first** to systematically test the system-level fault tolerance of MSF-based ADSs, identifying behavioral safety violations caused by various sensor faults. Our goal is to evaluate whether MSF-based ADSs are robust enough to maintain AV safety when encountering real-world sensor faults during driving.

3 Approach

Our objective is to test the fault-tolerance of MSF-based ADSs under camera and LiDAR faults that may occur during AV driving. To this end, we introduce a novel approach: FADE. Its overview is presented in Figure 1, which consists of two parts. ❶ **Sensor fault modeling**: FADE systematically models faults that cameras and LiDARs may encounter in real-world traffic environments, including active faults and passive faults caused by various environment factors. ❷ **Differential Fuzzer-based Sensor Fault Tolerance Testing**: FADE designs and implements a GA-guided differential fuzzer, which uses differential testing to test the performances of the ADS with and without sensor faults, and explores the space of fault models by a GA-based search to discover safety violations of the ADS caused by the injected sensor faults. Below we give details of each component.

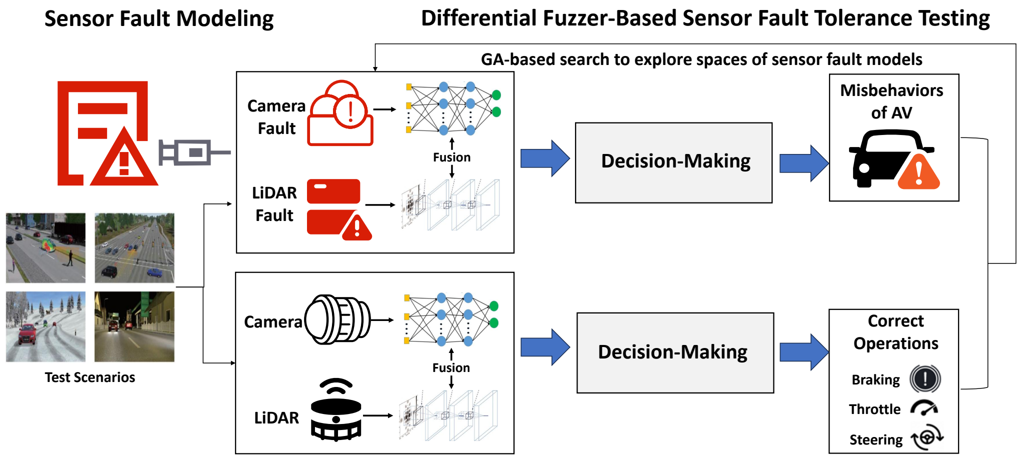


Fig. 1. The overview of FADE.

3.1 Sensor Fault Modeling

Sensors in real-world driving environments may encounter various issues that affect their accuracy and reliability. These issues can be broadly categorized into two main types: active faults and passive faults. Active faults arise from internal sensor malfunctions or damages, directly disabling or degrading their functionality. These faults typically result from component failures or wear. In contrast, passive faults stem from external environmental factors (e.g., an obstacle), rather than sensor damage or malfunction. These environmental factors can degrade the sensor's ability to capture or interpret perception data accurately.

To systematically model these sensor faults in traffic scenarios, we classify them from the perspectives of sensor components and environmental factors. Further, we develop fault models to simulate possible failures in cameras and LiDARs under real-world AV driving conditions. These models enable the construction of a comprehensive fault model library for AV perception, facilitating realistic fault injection experiments. A summary of camera and LiDAR fault-injection models is provided in Table 1 and Table 2.

3.1.1 Camera Fault Model. The camera fault models are categorized into 16 types, including 7 types of active faults and 9 types of passive faults. Due to page limits, we mainly describe one fault model and others are available at *sensor_fault_models.pdf* in https://github.com/ADStesting-test/FADE.

Table 1. Categorization of camera fault-injection models

Fault Category	Fault Type	An Example in Real Traffic	Faulty Componet
Active Fault	Deflection	Due to bumpy roads, the camera deflects during AV driving	Camera Body
	Displacement	Due to bumpy roads, the camera is misaligned during AV driving	
	Internal Dirt	The dirt accumulation inside caused by driving outside or changing temperature for a long time	
	Broken Lens	The splashing foreign object(e.g.,sand, gravel) hits the lens	Len
	Lens Brightness Change	Long-term usage in high-temperature or seaside cause brightness issues with basic components(e.g.,shutter,diaphragm,iris) of len	
	Blur	The blur introduced by malfunction of complex inside circuit	Image Sensor& ISP
	Internal Scatter	The color noise caused by malfunction of image signal processor	
Passive Fault	Lens Occlusion	The lens is covered with plastic bag or paper during AV driving	Len
	External Scatter	The surface of the camera's lens is contaminated with mud spots	
	Dust	The surface of the camera's lens is contaminated with dust	
	Raindrops	Raindrops appear on the lens along with rainlines during rainfall	
	Snow Grains	The deposit of snow grains on the lens during snowfall	
	Mist	Fogging of lens caused by high humidity and temperature differences	
	Ice	Temperature drops below zero degrees, resulting in ice on the lens	
	Overexposure	Under strong light sources such as high beams or reflective surfaces, camera receives too much light	Image Sensor
	White Balance Shift	At sunset, the camera image appears in red and orange tones due to white balance shift	Bayer Filter

Raindrops on the lens. Raindrops and rainlines act as random streaks of water on the lens, causing scattering, refraction, and partial occlusion of the camera's field of view. Under natural conditions the raindrops exhibit the tilted shape instead of the linear shape [53]. Therefore, when simulating the effect of raindrops and rainlines, the angle and shape changes of raindrops are introduced. Meanwhile, to make the raindrop effect more realistic, the transparency (or brightness) of raindrops is also adjusted [29]. The image I_r captured by the camera with raindrops on the lens is:

$$I_r(x,y) = (1 - L_r(x,y)) \cdot I_e(x,y) + L_r(x,y) \cdot (t_r \cdot I_e(x,y) + (1 - t_r) \cdot N_r(x,y)) \quad (1)$$

$$t_r(x,y) \sim U(t_{min}, t_{max}), \quad N_r(x,y) \sim \mathcal{N}(0, \sigma_r) \quad (2)$$

where $I_e(x,y)$ is the original pixel intensity of the image captured by the camera without raindrops on the lens. $N_r(x,y)$ is a random noise factor simulating the refraction-induced distortion. t_r is a transparency factor, simulating the partial occlusion of the camera's view due to the rain streaks. $L_r(x,y)$ is the rain line mask, calculated as:

$$L_r(x,y) = \Sigma_{i=1}^{n_r} \mathcal{H}\left(\frac{(y - y_i) - tan(\theta_r^i)(x - x_i)}{l_r^2}\right) \quad (3)$$

$$l_r \sim U(l_{min}, l_{max}), \quad \theta_r \sim U(\theta_{min}, \theta_{max}) \quad (4)$$

$L_r(x,y)$ represents the spatial distribution of the rain streaks on the lens. Each streak can be modeled as a linear segment on the image with the length l_r and angle θ_r. (x_i, y_i) is the starting position of a rain line. $\mathcal{H}(f)$ is a Heaviside function (or step function) that returns 1 if f is within the rainline length, and 0 otherwise, controlling the spatial extent of each rainline.

3.1.2 LiDAR Fault Model. The LiDAR fault models are categorized into 8 types, including 4 types of active faults and 4 types of passive faults. We introduce two LiDAR fault models and others are available at *sensor_fault_models.pdf* in https://github.com/ADStesting-test/FADE.

Table 2. Categorization of LiDAR fault-injection models

Fault Category	Fault Type	An Example in Real Traffic	Faulty Component
Active Fault	Deflection	Due to bumpy roads, the orientation of LiDAR changes	LiDAR enclosure
	Displacement	Due to bumpy roads, the position of LiDAR is displaced	
	Beam Loss	Due to long-term wear&tear and aging, laser beam of LiDAR reduces	Emitter
	Line Fault	The noise caused by malfunction of internal circuits	Process-ing Unit
Passive Fault	Electromagnetic Interference	When AV passes airports or power plants,these areas will generate electromagnetic wave interference	Receiver
	Crosstalk	NPC vehicles using LiDAR drive around AV,and their emitted signals cause confusion to receiving channel of AV's LiDAR	
	Rain and Snow Pollution	There are foreign objects (such as rain, snow, mist, mud) covering LiDAR's surface,limiting the LiDAR's field of view	Scanner
	Strong Light Interference	The measurement distance and point-cloud density are reduced due to strong light occupying detector units	Process-ing Unit

Deflection of LiDAR enclosure. This fault is introduced by LiDAR's deflection of the vertical direction to the direct direction and scan angle direction. When the deflection of the vertical direction is perpendicular to the scan angle direction, the resulting vertical error will be negligible. However, when the deflection is parallel to the scan angel direction, the induced vertical error reaches the maximum value [35]. The magnitudes of the rotations are defined as ξ and η components of the deflection of the vertical direction, and the resulting R_G is formalized as:

$$R_G = \begin{bmatrix} cos(\eta) & sin(\xi)sin(\eta) & sin(\xi)cos(\eta) \\ 0 & cos(\xi) & -sin(\xi) \\ -sin(\eta) & cos(\xi)sin(\eta) & cos(\xi)cos(\eta) \end{bmatrix} \quad (5)$$

Displacement of LiDAR enclosure. This fault model is defined by the grid mean approximation and triangular grid approximation, which have been applied experimentally to generate reference data for 3D data in the spatial domain [30, 37, 49]. The grid mean approximation method includes grid point errors and forms grids on the x and y planes based on irregularly distributed spatial data. Thereafter, the z coordinates of the data in the grid are averaged to determine the representative point of each grid. The grid mean approximation method uses a multiple regression analysis technique and calculates the displacement of a structure using structural information such as strain, stress, displacement, and z-coordinates:

$$P_j(x, y, z) = (\varepsilon, Z_j), \quad Z_j = \frac{1}{n_j}\Sigma_{i=1}^{n_j}\Sigma_{j=1}^{m}Z_{ji}, \quad \varepsilon = N_1\varepsilon_1 + N_2\varepsilon_2 + N_3\varepsilon_3 \quad (6)$$

where Z_{ji} represents the Z coordinate value of the i-th coordinate data included in the j-th space. The point P_j is set as reference data in the center of the grid. The shape functions N_1, N_2, N_3 are calculated through natural coordinates.

3.2 Differential Fuzzer-Based Sensor Fault Tolerance Testing

Based on the fault models of camera and LiDAR, FADE injects sensor faults into the MSF-based ADS, and employs a differential fuzzer to assess its fault tolerance and identify the safety violations in the ego vehicle. Our sensor fault-tolerance testing procedure is detailed in Algorithm 1. The explanations of notations used in this algorithm are given as Table 3.

Specifically, FADE first generates test scenarios $\mathbb{TS}$, and instances ($\mathbb{FP}$) of the injected sensor fault sf. It then injects the sensor fault instance fp into the ADS, and performs differential testing by

Table 3. Explanations of notations used in Algorithm 1

Notation	Explanation	Notation	Explanation
ts	a test scenario	AD_o	ego vehicle without fault
sf	a sensor fault	rs_o^{ts}	execution result of AD_o in ts
fp	an instance of sf injected into AD_o	RS_{fp}^{ts}	differential testing result of AD_f and AD_o in ts
AD_f	ego vehicle with the injected fp		
$SVF(RS_{fp}^{ts})$	assertion for safety violation caused by fp in ts	$\Phi_o(rs_o^{ts})$	assertion for no safety violation of AD_o in ts

comparing the behaviors of the ego vehicle with fp (AD_f), and without fp (AD_o) in the same test scenario ts. Safety violations caused by fp are identified by analyzing the system-level performance differences between AD_f and AD_o. Additionally, for each sf in ts, FADE leverages a multi-objective genetic algorithm to optimize its injected instance fp, further exploring its potential to trigger safety violations in the ADS.

Algorithm 1 Sensor fault-tolerance testing

```
Require: Camera fault models $\mathbb{CFM}$, LiDAR fault models $\mathbb{LFM}$
Ensure: $\mathbb{SV}$: Safety violations of ego vehicle caused by injected sensor faults
 1: $\mathbb{SV} \leftarrow \emptyset$, $terminate \leftarrow False$
 2: $\mathbb{TS} \leftarrow generate\_scenarios(num)$    ▷ generate a set of test scenarios
 3: $\mathbb{SF} \leftarrow \mathbb{CFM} \bigcup \mathbb{LFM}$, $\mathbb{ISF} \leftarrow combination(\mathbb{CFM}, \mathbb{LFM})$
 4: for $isf \in \mathbb{ISF}$ do
 5:     if $\forall i, j \in isf, i.pre = j.pre$ then
 6:         $\mathbb{SF} \leftarrow \mathbb{SF} \bigcup isf$    ▷ determine the sensor faults to be injected into the ADS
 7: for $\forall sf \in \mathbb{SF}$ do
 8:     for $\forall ts \in \mathbb{TS}$ do
 9:         $rs_o^{ts} \leftarrow execute(ts, AD_o)$
10:         if $\Phi_o(rs_o^{ts})$ then
11:             $\mathbb{FP} \leftarrow initialization(sf, k)$    ▷ create initial instances of the injected sensor fault $sf$
12:             for $\forall fp \in \mathbb{FP}$ do
13:                 $AD_f \leftarrow inject\_faults(fp, AD_o)$    ▷ inject the instance of $sf$ into the ADS
14:                 $RS_{fp}^{ts} = differential\_testing(AD_f, AD_o, ts)$    ▷ perform differential testing
15:             $\mathbb{RS} \leftarrow differential\_fuzzer(sf, ts, \mathbb{FP})$ ▷ optimize instances of $sf$ to find safety violation of $AD_f$
16:             for $RS_{fp}^{ts} \in \mathbb{RS}$ do    ▷ compare the results to identify ADS's safety violations caused by $sf$
17:                 $monitor \leftarrow SVF(RS_{fp}^{ts})$
18:                 if $monitor$ then
19:                     $\mathbb{SV} \leftarrow \mathbb{SV} \bigcup RS_{fp}^{ts}$
20: return $\mathbb{SV}$
```

Considering that the simultaneous faults of multiple sensors are rare in real-traffic driving, FADE primarily injects each single-sensor fault into the ADS. However, to balance practicality and testing thoroughness, FADE also considers sensor co-faults, which refer to the multiple faults arising from the same environmental conditions. For example, both the *overexposure* of camera and *strong light interference* of LiDAR, arise from the strong light in the driving environment. Thus the two faults constitute a sensor co-fault, and FADE injects the two faults simultaneously into the ADS.

3.2.1 Generating Test Scenarios. To evaluate the ADS's performance under sensor faults, FADE generates test scenarios that reflect diverse real-world traffic conditions, leveraging naturalistic driving data [24, 25]. These scenarios are parameterized by variables including road topology, traffic density, and the behaviors of dynamic participants (e.g., actions, speeds, accelerations, and relative positions of NPC vehicles and pedestrians). The scenarios are then generated through a constraint-based sampling of parameters within the Operational Design Domain (ODD) [20]. To ensure the validity and relevance of the generated scenarios, FADE enforces constraints to prevent unrealistic cases that violate the ODD and invalid cases that lack meaningful interactions with the ego vehicle. Specifically, FADE imposes the following constraints:

(1) $\forall i \in k, p_i(0) - p_{ego}(0) \leq d_{max}$. $p_i(0)$ represents the position of the i-th participant at the initial time, and k is the total number of participants in the scenario. d_{max} represents the maximal distance of the sensors on the ego vehicle. This constraint requires the initial positions of participants not to be too distant from the ego vehicle's initial position, ensuring the participants in the test scenario are more likely to enter the sensing range of the ego vehicle's sensors.

(2) $\forall i \in k, \exists t \in F \bigwedge rs \in R, p_i(t) \in rs \bigwedge p_{ego}(t) \in rs$. F represents the total time steps of the test scenario execution, and R represents the road map of the test scenario consisting of several road segments (represented by rs). This requires that each participant's trajectory has a same road segment as the ego vehicle's trajectory. Constraint (1) and constraint (2) ensure the ego vehicle has interactions with participants in the test scenario.

(3) $\forall i \in k, t \in F, (v_{t+1} - v_t \leq a_{max} \bigwedge v_i(t) \leq v_{max}) \bigwedge (d_i(t) = dir_{rs}, p_i(t) \in rs)$. $v_i(t)$ represents the speed of participant i at time t and v_{max} represents the speed limit of the road. a_{max} represents the maximal acceleration of the participant (vehicle or pedestrian). $d_i(t)$ represents the direction of participant i at time t and dir_{rs} represents the direction of the road segment rs where $p_i(t)$ is located (for each pedestrian, dir_s includes all directions). This constraint requires the speeds and directions of participants not to violate the realistic traffic dynamics during their motions.

3.2.2 GA-Based Differential Fuzzer. For each sensor fault and co-fault sf in a generated test scenario, FADE initializes k instances of sf randomly and injects each fault instance fp into the ADS to create AD_f. It then performs differential testing between AD_f and AD_o in the same scenario ts. Using the recorded differential testing results, FADE employs a multi-objective GA [61] to optimize the instances of sf to expose ADS's safety violations caused by fp in ts.

Individual Encoding and Representation. In ts, each instance of the injected sensor fault is encoded as an individual, consisting of one or two chromosomes. FADE encodes each injected single sensor fault as a chromosome and each co-fault as two chromosomes. Each chromosome consists of a series of genes, and one gene corresponds to a parameter of the sensor fault model.

The scores of each individual are calculated by a **multi-objective fitness function**. The optimal individuals are selected as parents, by building improved Pareto-optimal solutions from the individuals of the current and previous generations. The individuals of the next generation are generated by **variation operators** on parents. When the selected top k excellent individuals remain the same in three consecutive generations, the optimization of instances for sf in ts terminates.

Multi-objective Fitness Function. The fitness function optimizes two objectives: the additional risk introduced by AD_f compared to AD_o, and the motion deviation between them. Based on these objectives, FADE constructs successfully-improved Pareto-optimal solutions. Specifically, fp represents an instance of the injected sensor fault sf.

(1) Objective 1: the additional risk introduced by AD_f compared to AD_o. The additional risk introduced by fp during the driving of AD_f compared to AD_o in the same test scenario ts, is defined as $I(fp, ts)$:

$$I(fp, ts) = METTC(AD_o, ts) - METTC(AD_f, ts) \tag{7}$$

METTC is the minimal estimated time for collision [62], which is widely used to measure the risk of the AV during driving [46, 56]. We define an improved calculation of METTC as follows:

$$METTC(AD_f, ts) = \min_{\substack{i\in[1,k]\\ t\in(0,F)}} \left(\frac{d.x(b_t^{AD_f}, b_t^{p_i})}{v.x_t^{p_i} - v.x_t^{AD_f}} + \frac{v.x_t^{p_i} - v.x_t^{AD_f}}{ac.x_t^{p_i} - ac.x_t^{AD_f}}, \frac{d.y(b_t^{AD_f}, b_t^{p_i})}{v.y_t^{p_i} - v.y_t^{AD_f}} + \frac{v.y_t^{p_i} - v.y_t^{AD_f}}{ac.y_t^{p_i} - ac.y_t^{AD_f}} \right) \quad (8)$$

where p_i is the i-th participant in the scenario ts. $d.x(b_t^{AD_f}, b_t^{p_i})$ and $d.y(b_t^{AD_f}, b_t^{p_i})$ respectively denote the lateral and longitudinal Euclidean distances between the bounding boxes of the ego vehicle and the i-th participant at time t. $v.x$ and $v.y$ represent the lateral and longitudinal speeds respectively. $ac.x$ is the lateral acceleration and $ac.y$ is the longitudinal acceleration. The larger $I(fp, ts)$, the higher the fitness score.

(2) Objective 2: the motion deviation between AD_f and AD_o. The motion deviation between AD_f and AD_o in the same test scenario ts, is defined as:

$$L(fp, ts) = \sum_{t=1}^{n} \sqrt{(x_{AD_f}^t - x_{AD_o}^t)^2 + (y_{AD_f}^t - y_{AD_o}^t)^2} \quad (9)$$

where $(x_{AD_f}^t, y_{AD_f}^t)$ represents the position of AD_f's waypoint at time t in the scenario ts. The larger $L(fp, ts)$, the higher the fitness score of the test scenario.

Variation Operators. The variation consists of two operators: crossover and mutation.

(1) Crossover. It is applied between two individuals. For each parent individual, FADE randomly generates a crossover rate $r_c \in (0, 1)$. If $r_c > threshold_c$, the crossover is performed. Specifically, for a single sensor fault, FADE applies uniform crossover on chromosomes across the two individuals. For sensor co-fault, it exchanges chromosomes corresponding to the same sensor fault.

(2) Mutation. This is applied inside an individual. For each parent individual, FADE randomly generates a mutation rate $r_m \in (0, 1)$. If $r_m > threshold_m$, gene mutation is performed by modifying one parameter of a sensor. Specifically, FADE randomly selects one gene from one chromosome and applies Gaussian mutation, introducing adaptive perturbations to efficiently explore the sensor fault model's parameter space.

3.2.3 Comparing Results. During the execution of test scenarios, FADE continuously monitors and records the states of the ego vehicle and participants in real-time, including the waypoint sequences of the ego vehicle and participants. Each waypoint is recorded as a 4-tuple [x-coordinate, y-coordinate, speed, orientation]. FADE builds the test oracle by a formal safety specification to compare the execution results based on the recorded data. In test scenario ts, the safety violation of the ADS caused by an instance fp of the injected sensor fault sf, is identified by $SVF(RS_{fp}^{ts})$:

$$SVF(RS_{fp}^{ts}) = \Phi_f(rs_{AD_f}^{ts}) \land \Phi_o(rs_{AD_o}^{ts}) \quad (10)$$

$$\Phi_f(rs_{AD_f}^{ts}) = \mathbf{F}_{t\in[0,L]}(Co(rs_{AD_f}^{ts}.t) \lor TS(rs_{AD_f}^{ts}.t) \lor TV(rs_{AD_f}^{ts}.t) \lor TD(rs_{AD_f}^{ts}.L)) \quad (11)$$

$$\Phi_o(rs_{AD_o}^{ts}) = \mathbf{G}_{t\in[0,L]}(\neg Co(rs_{AD_o}^{ts}.t) \land \neg TS(rs_{AD_o}^{ts}.t) \land \neg TV(rs_{AD_o}^{ts}.t) \land \neg TD(rs_{AD_o}^{ts}.t)) \quad (12)$$

where RS_{fp}^{ts} records the execution results of AD_f and AD_o in ts, represented as $rs_{AD_f}^{ts}$ and $rs_{AD_o}^{ts}$ respectively. $rs_{AD_f}^{ts}.t$ denotes the state of AD_f at time step t in ts, including its position, velocity, orientation, and relative distances to participants. $rs_{AD_o}^{ts}.t$ denotes the state of AD_o at t in ts. $\Phi_f(rs_{AD_f}^{ts})$ is defined as a temporal logic formula, which asserts whether AD_f violates any safety specification during the execution time of ts. Similarly, $\Phi_o(rs_{AD_o}^{ts})$ is to assert whether AD_o violates no safety specification during the execution time of ts. Co, TS, TV, TD are the safety specifications. Their descriptions for AD_f are given below, which are similar as those for AD_o.

- $Co(rs_{AD_f}^{ts}.t)$ represents AD_f colliding with any object in ts at time t.
- $TS(rs_{AD_f}^{ts}.t)$ represents AD_f blocking or interrupting any participant in ts at time t (comparing to the normal driving of the participant in the scenario where AD_o operates).
- $TV(rs_{AD_f}^{ts}.t)$ represents the speed of AD_f exceeding the speed limit of the road or AD_f running the red light at an intersection in ts at time t.
- $TD(rs_{AD_f}^{ts}.t)$ represents AD_f failing to arrive at the destination where AD_o arrives at the ending time of ts.

Note that for ts, among the instances fp generated by the GA-based optimization of sf, if the number of fp whose $SVF(RS_{fp}^{ts})$ is true, is more than M, sf is considered to be capable of inducing safety violations of the ADS in ts.

4 Evaluation

To comprehensively evaluate FADE, we explore the following research questions:

- **RQ1:** Can FADE discover sensor fault-tolerance issues in ADSs against various sensor faults?
- **RQ2:** How effective is FADE in sensor fault modeling and injection compared to baselines?
- **RQ3:** How effective is the differential fuzzer-based sensor fault-tolerance testing of FADE?
- **RQ4:** What are the practical impacts of sensor faults injected into ADSs in the physical world?

4.1 Experiment Settings

ADS Under Test. We select an industry-grade MSF-based Level-4 ADS, Baidu Apollo [7] as the test target. Apollo leverages MSF perception to recognize and understand objects in the surrounding environment, primarily integrating data from camera and LiDAR sensors. It has been widely recognized and adopted in the autonomous driving industry with the following evidences. (1) The Apollo community ranks among the top-four leading industrial ADS developers [4], while the other three ADSs are not publicly released. (2) Apollo can be readily installed on vehicles for driving on public roads [3]. It has been commercialized for many real-world self-driving services [1, 2].

Simulation Environment. We conduct the simulation experiments on Ubuntu 20.04 with 500 GB memory, an Intel Core i9 CPU, and an NVIDIA GTX3090 TI. We adopt SORA-SVL [8], an end-to-end AV simulation platform that supports connection with Apollo.

Parameter Settings. We consider and set the following parameters in FADE: (1) num: this is the number of generated test scenarios for each injected sensor fault. We set it as 100 for the balance of scenario coverage and time cost. (2) $threshold_m$ and $threshold_c$: these are the thresholds for mutation and crossover, respectively. We test different values of these two parameters recommended by existing genetic algorithms [33, 61], and choose 0.3 for $threshold_m$ and 0.4 for $threshold_c$. (3) k is the number of selected excellent individuals in each generation, and M is the threshold for determining a sensor fault capable of causing ADS's safety violations. To balance the search effects and evolving costs, we set k as 4 and M as 5.

4.2 Experiment Design

To answer RQ1, we apply FADE to test Apollo's tolerance of different sorts of sensor faults. For each sensor fault and co-fault, we generate 100 test scenarios, encompassing various driving situations with different types of roads, participants, and weather conditions. Road types include highways, urban streets, and intersections. Participants consist of behaviors of NPC vehicles (including following lanes, changing lanes, crossing, turning around, overtaking, and parking) and pedestrians (including walking along, walking across, and standing).

To answer RQ2, considering that there are no available approaches that could test the system-level performance of MSF-based ADSs with injected sensor faults, we select an MSF robustness and reliability testing benchmark [27], as the baseline for comparison. This benchmark summarizes and implements a range of real-world corruption patterns on MSF perception modules, and tests their impacts on the MSF results in representative perception tasks (e.g., object detection, object tracking, and depth completion). They conclude 14 corruption patterns, which simulate corrupted data of scenarios and input into the perception module to obtain the output result. 12 of these 14 corruption patterns exhibit similar characteristics to a part of the sensor faults implemented in FADE, e.g., *rain in environment* in the benchmark corresponds to *rain on lens and LiDAR* in FADE. We refer to those overlapping patterns as *common patterns*. We implement the baseline by integrating the common patterns in the benchmark into Apollo, and test its safety with the input of corrupted sensor data. For fairness, we test Apollo with each common pattern in the same test scenarios as FADE, and explore it using the differential fuzzer of FADE.

To answer RQ3, we conduct the ablation experiment that compares FADE with FADE_r, which replaces the differential fuzzer with a random sampling of sensor fault parameters. For the sensor fault sf in the test scenario ts, we use the random-based baseline to generate the same number of instances of sf as those generated by FADE in ts, and test the fault tolerance of Apollo.

Note that for RQ1, RQ2, and RQ3, to account for the randomness of the differential fuzzer in FADE, each experiment is repeated ten times. Meanwhile, across the ten runs of experiments, for RQ1, we vary the parameters' values of the 100 test scenarios considering the randomness of the parameter sampling of test scenarios. For RQ2 and RQ3, we use the same scenarios in RQ1 to test the baselines for fair comparisons.

To answer RQ4, we conduct the experiments on an actual AV in real-world roads. As illustrated in Figure 2, our AV is equipped with a 32-line LiDAR, 1920*1080p HD camera, Huace GI-410 INS, and a Nuvo-8111 industrial PC with an Intel Core i9-9900K CPU, NVIDIA RTX 3060 GPU, 32GB RAM, and 1TB SSD, integrated with Pix Hooke Chassis and Apollo 6.0 Edu Platform.

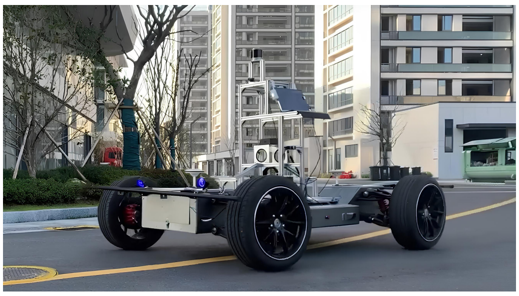
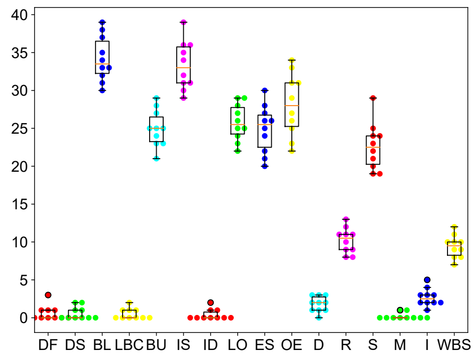


Fig. 2. The autonomous vehicle equipped with Apollo 6.0 Edu for our physical experiment.

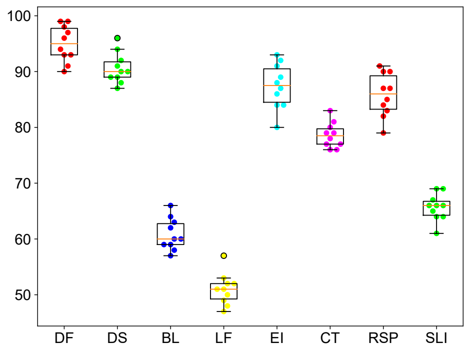
4.3 RQ1: Effectiveness Experiment

Our effectiveness evaluation results are shown in Tables 4 and 5, where SV is short for "safety violation". We also show the average numbers of SVs and variance for each fault in Figure 3 (where the sequences of camera and LiDAR faults in the x-axis correspond to the ones in Tables 1 and 2). We have two high-level observations. First, MSF exhibits different degrees of fault tolerance against different types of faults. Second, for each run of the experiments, the results remain consistent, implying that the camera and LiDAR faults have predictable and deterministic impacts on the

ADS's behaviors. Some experiment videos are available at https://zenodo.org/uploads/14015455. Below we present more in-depth analysis and findings about MSF's fault tolerance.



(a) Number of SVs caused by camera faults



(b) Number of SVs caused by LiDAR faults

Fig. 3. The number of safety violations and variance under camera and LiDAR fault injection

Table 4. The average numbers of safety violations caused by camera faults

Fault Type	Deflec-tion	Displace-ment	Broken Lens	Lens Bright-ness Change	Blur	Internal Scatter	Internal Dirt	Lens Occl-usion
Number of SVs	0.6	0.6	34.2	0.5	25	33.3	0.4	25.8
Fault Type	External Scatter	Overexpo-sure	Dust	Rain	Snow	Mist	Ice	White Bal-ance Shift
Number of SVs	24.9	28.1	1.8	10.2	22.6	0.2	2.7	9.4

4.3.1 Camera Fault. The ADS exhibits significant different behaviors under various types of camera faults. Among the sixteen camera faults, we observe that only half of them can cause a large number of safety violations in Apollo, and their impacts on Apollo's behaviors vary greatly.

First, from Table 4 we observe that some camera faults, including deflection, displacement, lens brightness change, internal dirt, dust, mist, and ice, do not frequently lead to safety violations of Apollo, which suggests that the MSF perception-based ADS demonstrates a notable degree of resilience to such slight distortions. Specifically, the perception pipeline can tolerate slight distortions in image quality or slight obstructions in the camera lens well, such as slight shifts in camera alignment (deflection or displacement) or gradual brightness changes, because the complementary LiDAR input can mitigate visual defects. Additionally, small particles or slight occlusions (such as dust or fog) on the camera lens may not block a large part of the field of view, allowing the ADS to maintain a consistent scenario interpretation.

Finding 1: The MSF perception-based ADS has strong fault tolerance on unstructured visual disturbances to the camera.

Finding 1 indicates that the MSF perception-based ADS is inherently robust to low-level visual noise or occlusions, ensuring that minor disturbances do not trigger erroneous system behaviors. This resilience is critical in real-world applications, where *minor visual obstructions are inevitable* *due to varying environmental conditions.* Consequently, the ADS's capacity to manage these subtle distortions suggests that multi-sensor fusion frameworks enhance fault tolerance and contribute to safer operational performance by reducing the impact of minor camera faults.

Second, some camera faults, including broken lens, blur, internal and external scatter, lens occlusion, and overexposure, can distort the structural integrity of image. They introduce significant distortions to the image data, affecting critical visual features (e.g., object edges, textures, spatial clarity). From Table 4, we observe that these faults can lead to more safety violations in Apollo. Such safety violations occur when the motion of objects near Apollo changes. These faults can lead to MSF's misdetection and misclassification of objects in scenarios. Furthermore, they disrupt the ADS's ability to perform precise spatial localization and safe navigation. For example, a broken lens or severe blurring can obscure the boundaries of objects, making it difficult for the ADS to discern the presence or position of pedestrians, vehicles, or road obstacles. Overexposure and lens occlusion faults exacerbate this issue by introducing areas of high brightness or visual blockage, leading to a limited field of view that prevents the ADS from obtaining a complete and reliable representation of its surroundings. As a result, these structural distortions not only degrade image quality but also lead to frequent safety violations as the ADS makes incorrect or delayed decisions.

Finding 2: Camera faults that significantly compromise the structural integrity of the visual data can easily cause misbehaviors of ADSs in response to motion changes of nearby objects.

Third, the impacts of passive camera faults (e.g., snow, ice, raindrops, mist, or dust on the lens) on ADS's behaviors vary greatly. Specifically, heavy snow and ice on the lens tend to accumulate in larger volumes, creating substantial occlusion on the lens and severely obstructing light transmission, which can change the structural aspects of the visual data. Raindrops, although typically smaller in volume than snow and ice, possess unique optical properties that cause the light to refract and scatter as it passes through or around the droplets. This scattering effect can distort object shapes and positions, leading to erroneous behaviors of the ADS. In contrast, mist and dust on the lens form a thin, diffuse layer that moderately reduces the light transmittance but with a relatively lower scattering rate, often resulting in a softened image that preserves object outlines and only reduces the contrast and detail. So MSF-based ADSs can resolve them well.

Finding 3: The passive faults that cause contamination to the lens have significant different impacts on ADS's behaviors, which are related to the inherent properties of the pollutants (e.g., volume, transmittance, and scattering rate).

Fourth, the white balance shift can only cause safety violations of Apollo during driving in two traffic situations. When this fault causes the camera background color to be light red, the MSF perception identifies the green light as a red light, causing the ADS to stop at the intersection and disrupt the traffic flow. When the white balance shift causes the camera background color to be light yellow, the MSF-based perception recognizes the yellow light as a green light, causing the ADS's decision of running through even if the ego vehicle does not exceed the stopping line, which violates the traffic regulation and increases the risk of collisions at intersections.

Finding 4: White balance shift can affect ADS's accuracy in identifying traffic lights under specific circumstances.

4.3.2 LiDAR Faults. The LiDAR faults have much stronger impacts than camera faults on MSF-based ADSs. As shown in Table 5, nearly all types of LiDAR faults are capable of inducing erroneous

Table 5. The average numbers of safety violations caused by LiDAR faults

Fault Type	Deflection	Displacement	Beam Loss	Line Fault
Number of SVs	95	90.6	60.8	51
Fault Type	Electromagnetic Interference	Crosstalk	Rain and Snow Pollution	Strong Light Interference
Number of SVs	87.4	78.6	85.8	65.7

behaviors of ADSs. The point clouds captured by LiDAR provide detailed 3D spatial context, and are directly used by the ADS for obstacle recognition and avoidance. Camera faults, while potentially affecting object recognition in certain visual conditions, do not have a critical impact on real-time obstacle avoidance.

Finding 5: The MSF perception-based ADS has low fault tolerance of LiDAR faults.

The most significant distinction between camera faults and LiDAR faults lies in how deflection and displacement influence ADS behaviors and create system-level consequences. Specifically, while deflection and displacement in the camera rarely lead to safety violations, similar faults in the LiDAR are among the highest in inducing safety-critical failures. We analyze the main reason for such difference: when a deflection or displacement fault occurs in LiDAR, it skews these distance measurements and point cloud data. As a result, the ADS experiences a fundamental misinterpretation of object positions and distances. Such inaccuracies have a direct and high-impact consequence on the decision-making of the ADS. The camera only contributes to the 2D spatial understanding by supplementing object classification and contextual information. This makes the camera's spatial misalignments less impactful, and MSF-based perception can correct this fault by the MSF algorithm and LiDAR data. Therefore, the deflection and displacement faults in camera have a low likelihood of propagating severe errors into the decision-making process. Compared to the camera, when LiDAR data is corrupted due to deflection or displacement, it introduces inconsistencies in the sensor fusion results, which cannot be resolved well by the MSF algorithm.

Finding 6: The MSF perception-based ADS is more sensitive to LiDAR deflection and displacement than to camera misalignment.

Besides, the beam loss and line fault are the conditions of LiDAR aging and degradation, where single laser beams weaken or malfunction. These faults result in reduced partial density in the point cloud, compromising the system's ability to detect smaller or distant objects reliably. Compared with other faults of LiDAR, Beam loss and line fault causes fewer safety violations. This suggests that the MSF-based ADS is tolerant against aging and degradation to some extent. Such fault tolerance may remain until the long-term aging and degradation reach a critical threshold. As aging and degradation faults of LiDAR develop over extended periods, the MSF perception-based ADS can adapt incrementally as the sensor's performance gradually shifts. This progressive nature makes it possible for the ADS to identify and compensate for these slow changes. Specifically, the design of MSF perception-based ADS typically incorporates sensor redundancy. The ADS can adjust MSF perception to recalibrate its confidence weights for LiDAR data, relying more on input from other sensors when a decrease in LiDAR reliability is detected.

Finding 7: The MSF perception-based ADS exhibits a better fault tolerance for LiDAR aging and degradation than for external interferences.

4.3.3 Co-faults. Next we consider the combination of multiple sensor faults triggered by the same condition. This includes: deflection and displacement (bumpy roads), raindrops and snow on sensors (sleety weather), overexposure and strong light interference (under high beam). The safety violations of Apollo caused by these combinations are shown in Figure 4.

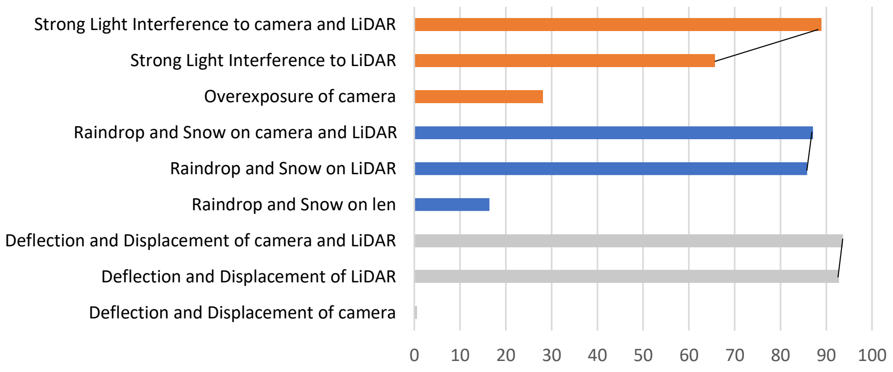


Fig. 4. The average numbers of safety violations of Apollo caused by different co-faults.

The MSF-based ADS leverages complementary data from both the camera and LiDAR, effectively mitigating the impact of missing different partial information from one sensor alone. In situations where partial data loss occurs in either the camera or LiDAR input, the ADS can compensate for it by relying on the other sensor to retrieve the data for the missing part. This complementary data enables the localization, prediction, and decision-making modules of the ADS to generate accurate and reliable operational commands, thus sustaining the vehicle's safe navigation and decision-making capabilities.

However, under strong light conditions, both the camera and LiDAR sensors experience concurrent data degradation simultaneously, which overexposes camera inputs and saturates LiDAR sensors. This lack of alternative input due to identical data loss from the camera and LiDAR may cause the ADS's localization and prediction modules to fail to detect obstacles and misinterpret distances, or generate inaccurate predictions of nearby objects' movements. Consequently, the decision-making module, which relies on accurate perception data, may issue incorrect operational commands, which critically undermine the correctness and safety of ADS's behaviors.

Finding 8: Strong light interference that simultaneously disrupts both camera and LiDAR significantly increases the frequency of safety violations of MSF perception-based ADSs.

4.4 RQ2: Comparison Experiment

Table 6 shows the sensor faults in FADE that correspond to the corruption patterns in the baseline benchmark, as well as their number of violations. Note that there exist some sensor faults that contain more than one corruption pattern (e.g., both *Motion Blur* and *Defocus Blur* in the benchmark are included in *Blur* of FADE). For these sensor faults, we accumulate the impacts of the corresponding corruption patterns as the baseline's input data of test scenarios.

Table 6. Comparison results of FADE and the baseline benchmark

Sensor Type	**Corruption in Benchmark**	**Number of SVs**				**Sensor Faults in FADE**	**Number of SVs**			
		avg	min	md	max		avg	min	md	max
Camera	Brightness Increasing	9.6	6	9.5	12	Overexposure	28.1	22	28	34
	Darkness	0.4	0	1	1	Lens Brightness Change	0.5	0	0	1
	Image Noise	18.6	11	18	29	Internal Scatter	33.3	29	33	39
	Motion&De-focus Blur	18.5	9	18	24	Blur	25	21	25	29
LiDAR	Point Cloud Gaussian Noise	41.1	30	43.5	47	Line Fault	51	47	51	57
	Point Cloud Impulse Noise	60.5	48	63	69	Electromagnetic Interference	87.4	80	87.5	93
	Signal Loss	50.6	40	51	59	Beam Loss	60.8	57	60	66
Camera+ LiDAR	Fog	0.3	0	0	1	Mist	0.2	0	0	1
	Rain	25.8	15	25.5	35	Rain on lens &LiDAR	38.7	36	38.5	41
	Spatial Misalignment	52.5	45	52	59	Deflection and Displacement	95.9	93	95.5	99

We observe FADE demonstrates more effectiveness in safety violation identification than the baseline benchmark. Specifically, for the common patterns that have significant impacts on Apollo, FADE can discover a larger number of safety violations than the benchmark. For the common patterns that rarely lead to safety violations of Apollo, such as darkness (lens brightness change) and fog (mist), both FADE and the benchmark discover almost the same number of safety violations. The results demonstrate that the sensor fault injection of FADE is significantly more effective in discovering safety violations than the benchmark.

4.5 RQ3: Ablation Experiment

Figure 5 shows the comparison results of FADE (blue bars in $F.x$) and FADE_r (orange bars in $R.x$), where the sequences of camera and LiDAR faults in the x-axis correspond to the ones in Tables 1 and 2. Overall, compared to FADE_r, under the same test scenarios, FADE can discover more safety violations of Apollo caused by injected sensor faults.

Specifically, for each camera fault, in each run, FADE consistently identifies more types of critical faults and safety violations of Apollo. For LiDAR faults, FADE_r identifies all types of LiDAR faults that can induce safety violations of Apollo. However, for each fault, the number of Apollo's safety violations detected by FADE_r is significantly lower than that by FADE. We conclude that FADE exhibits superior performance in discovering critical camera faults, as well as safety violations under both camera and LiDAR faults. The results demonstrate that the differential fuzzer of FADE is more effective and efficient in exploring the space of sensor faults, and more comprehensive in testing the fault-tolerance of MSF-based ADSs.

4.6 RQ4: Physical Experiment

To ensure adherence to safety concerning of AV integrity, equipment protection, and safety of the test site and participants, we select the following sensor faults based on the results of RQ1, which can significantly cause safety violations of Apollo in simulation experiments: camera faults of lens

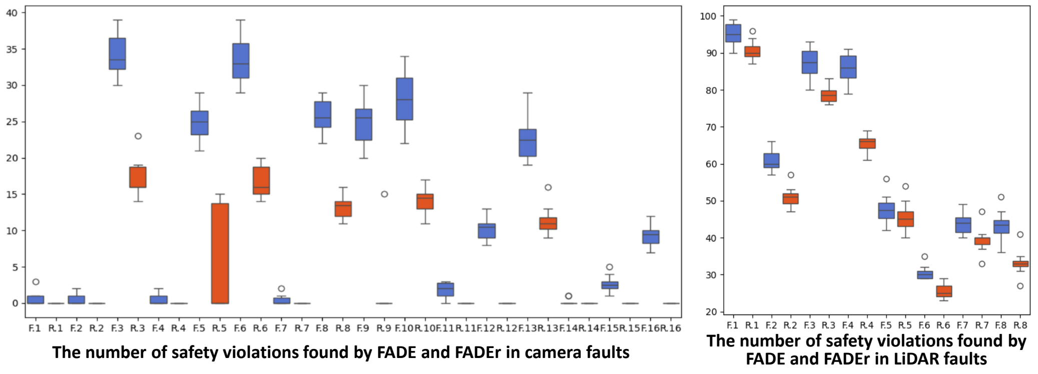


Fig. 5. Comparisons of FADE and FADE$_r$.

occlusion and external scatter, LiDAR faults of deflection and displacement, camera overexposure and LiDAR interference due to strong light. For each selected sensor fault, we randomly choose one of Apollo's safety violations identified by our approach from each of the 10 runs in RQ1. To ensure diversity, the selected 10 cases have distinct parameter values in the sensor fault model and test scenarios, reducing the risk of occasionality in the physical evaluation. This selection process is repeated five times to account for randomness.

For each selected sensor fault, we validate whether the found safety violations of Apollo caused by the injected sensor fault will occur in the physical world. The sensor faults are applied to the actual AV in the following ways, to replicate the simulated counterpart as closely as possible:

- **Lens Occlusion:** we use paper sheets to cover specific portions of the lens, ensuring that the occluded area and position match the safety-violation scenarios caused by lens occlusion.
- **External Scatters:** according to the safety-violation scenarios caused by external scatters, we disperse colored confetti at predefined densities and locations on the lens of the camera.
- **LiDAR Deflection:** we manually adjust LiDAR's orientation angle to introduce specific deflection in safety-violation scenarios caused by LiDAR deflection.
- **LiDAR Displacement:** we modify LiDAR's mounting position according to safety-violation scenarios caused by LiDAR displacement, while ensuring that LiDAR is securely attached to the ego vehicle.
- **Strong Light Interference:** we use a high-intensity flashlight to project specific intensity of illumination from a predefined angle above the vehicle's front, replicating the impacts of strong light interference on camera and LiDAR.

The results of physical experiments are shown as Table 7. The AV failure rate (AFR) for sensor fault sf is represented as AFR_{sf}, calculated as $AFR_{sf} = \frac{N_{SV}}{N_{sf}}$, where N_{SV} is the number of safety-violation scenarios induced by sf successfully reproduced in the physical experiment, and N_{sf} is the number of selected safety-violation scenarios induced by sf in simulation testing.

In the five runs of physical experiments, LiDAR faults cause higher AFRs than camera faults: the former leads to an average AFR of more than 80%, and the AFR of the later can also reach over 60% and the minimum is 50%. The AFR of overexposure and strong light interference ranks the highest. These physical experiment results demonstrate that the safety violations of the ADS caused by sensor faults discovered by FADE have strong practical significance. The effects of sensor faults are reliable and deterministic in real-world environments, mirroring their behaviors in simulations.

Table 7. The evaluation results of physical experiments

Sensor Fault	Lens Occlusion		External Scatter		LiDAR Deflection		LiDAR Displacement		Strong Light to Camera and LiDAR	
AV Failure Rate	min	50%	min	50%	min	70%	min	70%	min	80%
	avg	60%	avg	62%	avg	82%	avg	80%	avg	92%
	max	70%	max	70%	max	90%	max	90%	max	100%

5 Threats to Validity

Selective validation of physical experiments. One primary threat is that not all sensor faults are injected into the physical AV to validate their impacts on the ADS in the real world. Due to sensor component costs and the safety of vehicles and pedestrians in physical experiments, we selectively validate those sensor faults where the sensor components will not be damaged. For the safety of vehicles and pedestrians involved in physical experiments, we choose cardboard boxes to replace participants in simulation scenarios. While this threat exists, we conduct an in-depth analysis of the underlying causes of our findings, which can provide convincing recommendations for developers, testers, and safety researchers in the field of ADSs.
Parameter ranges of sensor fault models. Another potential threat is that the difficulty of identifying safety violations caused by the injected sensor faults could depend on the parameter ranges of the fault model. The GA-based differential fuzzer of FADE inherently explores the parameter spaces of each sensor fault model, which can adapt to varying ranges of them. Consequently, regardless of the size of the parameter ranges, FADE guarantees thorough exploration of sensor fault models. We plan to conduct a detailed analysis of parameter range variations for sensor fault models, exploring the robustness of ADS to different parameter ranges of sensor fault models.

6 Conclusion

This paper proposes FADE, the first approach to test the fault tolerance of MSF-based ADSs against different types of sensor faults. FADE designs sensor fault models for injecting camera and LiDAR faults into the MSF-based ADS, and implements a GA-guided differential fuzzer to explore the parameter spaces of sensor fault models. We evaluate FADE on an industrial MSF-based ADS. The evaluation results demonstrate that FADE can effectively and efficiently discover Apollo's safety violations caused by the injected sensor faults. To validate the findings in real-world AVs, we conduct the physical experiments and the results show the practical significance of our findings.
Future Work. Based on these findings, we can prioritize which sensor faults require further analysis. We plan to conduct the follow-up research work to FADE from the following aspects. (1) Currently, our evaluation focuses on the occurrences of the ADS's safety violations induced by different sensor faults. The evaluation of the severity of the discovered safety violations is orthogonal to the objective of this paper. The further analysis includes detailed assessments of their severity. (2) In further analysis, we will also conduct an empirical study on the robustness of the overall ADS to sensor fault model parameters.

Data Availability

The source code of FADE is available at https://github.com/ADStesting-test/FADE or https://zenodo.org/records/15168648. The experiment results are available at https://zenodo.org/uploads/14015455 [22].

Acknowledgments

This work is supported by National Natural Science Foundation of China (62472412, U20A6003), Major Project of ISCAS (ISCAS-ZD-202302) and Project of ISCAS (ISCAS-JCMS-202402). This work is also supported by the National Research Foundation, Singapore and DSO National Laboratories under its AI Singapore Programme (AISG Award No: AISG2-GC-2023-008). The authors are grateful for the financial support provided by the China Scholarship Council Program (Grant No. 202304910498).

References

[1] [n. d.]. *Autoware Self-driving Vehicle on a Highway.* Retrieved Sepetem 1, 2023 from https://www.youtube.com/watch?v=npQMzH3jd8
[2] [n. d.]. *Baidu Launches Public Robotaxi Trial Operation.* Retrieved April 1, 2024 from https://www.globenewswire.com/news-release/2019/09/26/1921380/0/en/Baidu-Launches-Public-Robotaxi-Trial-Operation.html
[3] [n. d.]. *Baidu launches their open platform for autonomous cars–and we got to test it.* Retrieved April 1, 2024 from https://technode.com/2017/07/05/baidu-apollo-1-0-autonomous-cars-we-test-it/
[4] [n. d.]. *Navigant Research Names Waymo, Ford Autonomous Vehicles, Cruise, and Baidu the Leading Developers of Automated Driving Systems.* Retrieved April 1, 2024 from https://www.businesswire.com/news/home/20200407005119/en/Navigant-Research-Names-Waymo-Ford-Autonomous-Vehicles
[5] [n. d.]. *WAYMO's virtual world to test self-driving cars: Simulation City.* Retrieved July 23, 2024 from https://www.d1ev.com/news/jishu/150890
[6] 2013. *Baidu apollo given another 20 licenses by beijing for autonomous car road tests.* Retrieved March 16, 2024 from https://autonews.gasgoo.com/china_news/70015513.html
[7] 2013. *An open autonomous driving platform.* Retrieved May 16, 2024 from https://github.com/ApolloAuto/apollo
[8] 2023. *SORA-SVL Simulator.* Retrieved July 30, 2024 from https://github.com/YuqiHuai/SORA-SVL
[9] Marie Altenburg. 2013. The Lens: A Practical Guide for the Creative Photographer. *PSA Journal* 79, 7 (2013), 9–10.
[10] Alireza Asvadi, Cristiano Premebida, Paulo Peixoto, and Urbano Nunes. 2016. 3D Lidar-based static and moving obstacle detection in driving environments: An approach based on voxels and multi-region ground planes. *Robotics and Autonomous Systems* 83 (2016), 299–311.
[11] Saeed Asadi Bagloee, Madjid Tavana, Mohsen Asadi, and Tracey Oliver. 2016. Autonomous vehicles: challenges, opportunities, and future implications for transportation policies. *Journal of modern transportation* 24 (2016), 284–303.
[12] Daniel Baumgartner, Peter Roessler, Wilfried Kubinger, Christian Zinner, and Kristian Ambrosch. 2009. Benchmarks of low-level vision algorithms for DSP, FPGA, and mobile PC processors. *Embedded Computer Vision* (2009), 101–120.
[13] Sean Campbell, Niall O'Mahony, Lenka Krpalcova, Daniel Riordan, Joseph Walsh, Aidan Murphy, and Conor Ryan. 2018. Sensor technology in autonomous vehicles: A review. In *2018 29th Irish Signals and Systems Conference (ISSC).* IEEE, 1–4.
[14] Yulong Cao, Ningfei Wang, Chaowei Xiao, Dawei Yang, Jin Fang, Ruigang Yang, Qi Alfred Chen, Mingyan Liu, and Bo Li. 2021. Invisible for both camera and lidar: Security of multi-sensor fusion based perception in autonomous driving under physical-world attacks. In *2021 IEEE symposium on security and privacy (SP).* IEEE, 176–194.
[15] Yulong Cao, Chaowei Xiao, Benjamin Cyr, Yimeng Zhou, Won Park, Sara Rampazzi, Qi Alfred Chen, Kevin Fu, and Z Morley Mao. 2019. Adversarial sensor attack on lidar-based perception in autonomous driving. In *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security.* 2267–2281.
[16] Andrea Ceccarelli and Francesco Secci. 2022. RGB cameras failures and their effects in autonomous driving applications. *IEEE Transactions on Dependable and Secure Computing* 20, 4 (2022), 2731–2745.
[17] Shang-Tse Chen, Cory Cornelius, Jason Martin, and Duen Horng Chau. 2019. Shapeshifter: Robust physical adversarial attack on faster r-cnn object detector. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2018, Dublin, Ireland, September 10–14, 2018, Proceedings, Part I 18.* Springer, 52–68.
[18] Xiaozhi Chen, Huimin Ma, Ji Wan, Bo Li, and Tian Xia. 2017. Multi-view 3d object detection network for autonomous driving. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition.* 1907–1915.
[19] Garrett Christian, Trey Woodlief, and Sebastian Elbaum. 2023. Generating Realistic and Diverse Tests for LiDAR-Based Perception Systems. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE).* IEEE, 2604–2616.
[20] Krzysztof Czarnecki. 2018. Operational design domain for automated driving systems. *Taxonomy of Basic Terms ", Waterloo Intelligent Systems Engineering (WISE) Lab, University of Waterloo, Canada* 1 (2018).
[21] Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. 2018. Robust physical-world attacks on deep learning visual classification. In *Proceedings of the IEEE conference on computer vision and pattern recognition.* 1625–1634.

[22] FADE. 2024. *Testing the Fault-Tolerance of Multi-Sensor Fusion Perception in Autonomous Driving Systems.* doi:10.5281/zenodo.14015455

[23] Di Feng, Ali Harakeh, Steven L Waslander, and Klaus Dietmayer. 2021. A review and comparative study on probabilistic object detection in autonomous driving. *IEEE Transactions on Intelligent Transportation Systems* 23, 8 (2021), 9961–9980.

[24] Shuo Feng, Yiheng Feng, Haowei Sun, Shan Bao, Yi Zhang, and Henry X Liu. 2020. Testing scenario library generation for connected and automated vehicles, part II: Case studies. *IEEE Transactions on Intelligent Transportation Systems* (2020). doi: 10.1109/TITS.2020.2988309.

[25] Shuo Feng, Yiheng Feng, Chunhui Yu, Yi Zhang, and Henry X Liu. 2020. Testing scenario library generation for connected and automated vehicles, Part I: Methodology. *IEEE Transactions on Intelligent Transportation Systems* (2020), 1573–1582. doi: 10.1109/TITS.2020.2972211.

[26] Davi Frossard and Raquel Urtasun. 2018. End-to-end learning of multi-sensor 3d tracking by detection. In *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 635–642.

[27] Xinyu Gao, Zhijie Wang, Yang Feng, Lei Ma, Zhenyu Chen, and Baowen Xu. 2023. Benchmarking Robustness of AI-Enabled Multi-Sensor Fusion Systems: Challenges and Opportunities. In *Proceedings of the 31st ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 871–882.

[28] Xinyu Gao, Zhijie Wang, Yang Feng, Lei Ma, Zhenyu Chen, and Baowen Xu. 2024. MultiTest: Physical-Aware Object Insertion for Testing Multi-sensor Fusion Perception Systems. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*. 1–13.

[29] Kshitiz Garg and Shree K Nayar. 2006. Photorealistic rendering of rain streaks. *ACM Transactions on Graphics (TOG)* 25, 3 (2006), 996–1002.

[30] Pelagia Gawronek, Maria Makuch, Bartosz Mitka, and Tadeusz Gargula. 2019. Measurements of the vertical displacements of a railway bridge using TLS technology in the context of the upgrade of the polish railway transport. *Sensors* 19, 19 (2019), 4275.

[31] Warren Gilchrist. 1993. Modelling Failure Modes and Effects Analysis. *International Journal of Quality & Reliability Management* 10 (1993). https://api.semanticscholar.org/CorpusID:108702726

[32] An Guo, Yang Feng, and Zhenyu Chen. 2022. LiRTest: augmenting LiDAR point clouds for automated testing of autonomous driving systems. In *Proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis*. 480–492.

[33] Randy L Haupt. 2000. Optimum population size and mutation rate for a simple real genetic algorithm that optimizes array factors. In *IEEE Antennas and Propagation Society International Symposium. Transmitting Waves of Progress to the Next Millennium. 2000 Digest. Held in conjunction with: USNC/URSI National Radio Science Meeting (C)*, Vol. 2. IEEE, 1034–1037. doi: 10.1109/APS.2000.875398.

[34] Jung Im Choi and Qing Tian. 2022. Adversarial attack and defense of yolo detectors in autonomous driving scenarios. In *2022 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 1011–1017.

[35] Christopher Jekeli. 2019. Deflections of the vertical from full-tensor and single-instrument gravity gradiometry. *Journal of Geodesy* 93 (2019), 369–382.

[36] Wenbo Jiang, Hongwei Li, Sen Liu, Xizhao Luo, and Rongxing Lu. 2020. Poisoning and evasion attacks against deep learning algorithms in autonomous vehicles. *IEEE transactions on vehicular technology* 69, 4 (2020), 4439–4449.

[37] DS Kang, HM Lee, Hyo Seon Park, and I Lee. 2007. Computing method for estimating strain and stress of steel beams using terrestrial laser scanning and FEM. *Key Engineering Materials* 347 (2007), 517–522.

[38] Philip Koopman and Michael Wagner. 2017. Autonomous vehicle safety: An interdisciplinary challenge. *IEEE Intelligent Transportation Systems Magazine* 9, 1 (2017), 90–96.

[39] K Naveen Kumar, Chalavadi Vishnu, Reshmi Mitra, and C Krishna Mohan. 2020. Black-box adversarial attacks in autonomous vehicle technology. In *2020 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*. IEEE, 1–7.

[40] Bo Li, Tianlei Zhang, and Tian Xia. 2016. Vehicle detection from 3d lidar using fully convolutional network. *arXiv preprint arXiv:1608.07916* (2016).

[41] Yiming Li, Congcong Wen, Felix Juefei-Xu, and Chen Feng. 2021. Fooling lidar perception via adversarial trajectory perturbation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 7898–7907.

[42] Yujie Li, Xing Xu, Jinhui Xiao, Siyuan Li, and Heng Tao Shen. 2020. Adaptive square attack: Fooling autonomous cars with adversarial traffic signs. *IEEE Internet of Things Journal* 8, 8 (2020), 6337–6347.

[43] Ming Liang, Bin Yang, Shenlong Wang, and Raquel Urtasun. 2018. Deep continuous fusion for multi-sensor 3d object detection. In *Proceedings of the European conference on computer vision (ECCV)*. 641–656.

[44] Aishan Liu, Xianglong Liu, Jiaxin Fan, Yuqing Ma, Anlan Zhang, Huiyuan Xie, and Dacheng Tao. 2019. Perceptual-sensitive gan for generating adversarial patches. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 33. 1028–1035.

[45] 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.

[46] SM Sohel Mahmud, Luis Ferreira, Md Shamsul Hoque, and Ahmad Tavassoli. 2017. Application of Proximal Surrogate Indicators for Safety Evaluation: A Review of Recent Developments and Research Needs. *IATSS Research* (2017), 153–163.

[47] Francisco Matos, Jorge Bernardino, João Durães, and João Cunha. 2024. A Survey on Sensor Failures in Autonomous Vehicles: Challenges and Solutions. *Sensors* 24, 16 (2024), 5108.

[48] Alex May. 2004. *Fotografia digitale*. Apogeo Editore.

[49] J Notbohm, A Rosakis, S Kumagai, S Xia, and G Ravichandran. 2013. Three-dimensional Displacement and Shape Measurement with a Diffraction-assisted Grid Method. *Strain* 49, 5 (2013), 399–408.

[50] Tri Minh Triet Pham, Bo Yang, and Jinqiu Yang. 2024. Perception-Guided Fuzzing for Simulated Scenario-Based Testing of Autonomous Driving Systems. *CoRR'24* (2024).

[51] Jonathan B Phillips and Henrik Eliasson. 2018. *Camera image quality benchmarking*. John Wiley & Sons.

[52] Rui Qian, Xin Lai, and Xirong Li. 2022. 3D object detection for autonomous driving: A survey. *Pattern Recognition* 130 (2022), 108796.

[53] Martin Roser and Andreas Geiger. 2009. Video-based raindrop detection for improved image registration. In *2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops*. IEEE, 570–577.

[54] Francisca Rosique, Pedro J Navarro, Carlos Fernández, and Antonio Padilla. 2019. A systematic review of perception system and simulators for autonomous vehicles research. *Sensors* 19, 3 (2019), 648.

[55] Floyd F Sabins Jr and James M Ellis. 2020. *Remote sensing: Principles, interpretation, and applications*. Waveland Press.

[56] Chris Schwarz. 2014. On Computing Time-To-Collision for Automation Scenarios. *Transportation Research part F: traffic psychology and behaviour* (2014), 283–294. doi: 10.1016/j.trf.2014.06.015.

[57] Francesco Secci and Andrea Ceccarelli. 2020. On failures of RGB cameras and their effects in autonomous driving applications. In *2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 13–24.

[58] Dawn Song, Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Florian Tramer, Atul Prakash, and Tadayoshi Kohno. 2018. Physical adversarial examples for object detectors. In *12th USENIX workshop on offensive technologies (WOOT 18)*.

[59] Diomidis H Stamatis. 2003. *Failure mode and effect analysis*. Quality Press.

[60] Jiachen Sun, Yulong Cao, Qi Alfred Chen, and Z Morley Mao. 2020. Towards robust {LiDAR-based} perception in autonomous driving: General black-box adversarial sensor attack and countermeasures. In *29th USENIX Security Symposium (USENIX Security 20)*. 877–894.

[61] Haoxiang Tian, Yan Jiang, Guoquan Wu, Jiren Yan, Jun Wei, Wei Chen, Shuo Li, and Dan Ye. 2022. MOSAT: finding safety violations of autonomous driving systems using multi-objective genetic algorithm. In *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 94–106.

[62] Richard Van Der Horst and Jeroen Hogema. 1993. Time-To-Collision and Collision Avoidance Systems. (1993).

[63] Jingkang Wang, Ava Pun, James Tu, Sivabalan Manivasagam, Abbas Sadat, Sergio Casas, Mengye Ren, and Raquel Urtasun. 2021. Advsim: Generating safety-critical scenarios for self-driving vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 9909–9918.

[64] Xupeng Wang, Mumuxin Cai, Ferdous Sohel, Nan Sang, and Zhengwei Chang. 2021. Adversarial point cloud perturbations against 3D object detection in autonomous driving systems. *Neurocomputing* 466 (2021), 27–36.

[65] Chaowei Xiao, Bo Li, Jun-Yan Zhu, Warren He, Mingyan Liu, and Dawn Song. 2018. Generating adversarial examples with adversarial networks. *arXiv preprint arXiv:1801.02610* (2018).

[66] Zuobin Xiong, Honghui Xu, Wei Li, and Zhipeng Cai. 2021. Multi-source adversarial sample attack on autonomous vehicles. *IEEE Transactions on Vehicular Technology* 70, 3 (2021), 2822–2835.

[67] Bin Yang, Wenjie Luo, and Raquel Urtasun. 2018. Pixor: Real-time 3d object detection from point clouds. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*. 7652–7660.

[68] Kaichen Yang, Tzungyu Tsai, Honggang Yu, Max Panoff, Tsung-Yi Ho, and Yier Jin. 2021. Robust roadside physical adversarial attack against deep learning in lidar perception modules. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*. 349–362.

[69] Tianwei Yin, Xingyi Zhou, and Philipp Krahenbuhl. 2021. Center-based 3d object detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11784–11793.

[70] Handi Yu and Xin Li. 2018. Intelligent corner synthesis via cycle-consistent generative adversarial networks for efficient validation of autonomous driving systems. In *2018 23rd Asia and South Pacific design automation conference (ASP-DAC)*. IEEE, 9–15.

[71] Yang Zhang, PD Hassan Foroosh, and Boqing Gong. 2019. Camou: Learning a vehicle camouflage for physical adversarial attack on object detections in the wild. In *ICLR*.

[72] Yifan Zhang, Dave Towey, Matthew Pike, Jia Cheng Han, George Zhou, Chenghao Yin, Qian Wang, and Chen Xie. 2023. Metamorphic Testing Harness for the Baidu Apollo Perception-Camera Module. In *2023 IEEE/ACM 8th International Workshop on Metamorphic Testing (MET)*. IEEE, 9–16.

[73] Ziyuan Zhong, Zhisheng Hu, Shengjian Guo, Xinyang Zhang, Zhenyu Zhong, and Baishakhi Ray. 2022. Detecting multi-sensor fusion errors in advanced driver-assistance systems. In *proceedings of the 31st ACM SIGSOFT International Symposium on Software Testing and Analysis*. 493–505.

[74] Zhi Quan Zhou and Liqun Sun. 2019. Metamorphic testing of driverless cars. *Commun. ACM* 62, 3 (2019), 61–67.

[75] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*. 2223–2232.

[76] Yi Zhu, Chenglin Miao, Foad Hajiaghajani, Mengdi Huai, Lu Su, and Chunming Qiao. 2021. Adversarial attacks against lidar semantic segmentation in autonomous driving. In *Proceedings of the 19th ACM conference on embedded networked sensor systems*. 329–342.

[77] Yi Zhu, Chenglin Miao, Tianhang Zheng, Foad Hajiaghajani, Lu Su, and Chunming Qiao. 2021. Can we use arbitrary objects to attack lidar perception in autonomous driving?. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*. 1945–1960.

Received 2024-10-31; accepted 2025-03-31