Testing Autonomous Driving Systems Through Blind-Spot Guided Fuzzing

Sali Moussa
School of Information Engineering
Chang'an University
Xi'an, China
musasali@chd.edu.cn

***Abstract*—Autonomous Driving Systems (ADS) must reliably perceive and react to complex environments, even when sensor blind spots obscure critical objects. While existing testing methods often focus on dynamic interactions, they significantly underestimate safety risks arising from both dynamic occlusions (e.g., vehicles) and persistent static occlusions (e.g., buildings). This research proposal introduces a novel, unified framework for occlusion-aware ADS testing. We present Blind-Spot Guided Fuzzing, a technique that systematically generates critical test scenarios by leveraging Large Language Models (LLMs) to synthesize realistic seeds from accident reports and employs multi-objective optimization to evolve them. This approach is implemented in our Occlusion-Sensitive Fuzzing (OS-Fuzz) framework, which encompasses two specialized modules: BlindSpotFuzz (BSF) for dynamic occlusions and StaticOccluFuzz (SOF) for static environmental occlusions that persistently hide Vulnerable Road Users. At its core, OS-Fuzz integrates a generalized occlusion model and innovative metrics to guide test generation and quantify the exploration of obscured inputs. Preliminary results against Apollo show BSF identifies over 50% more blind-spot-related collisions. Our comprehensive evaluation plan will benchmark OS-Fuzz against state-of-the-art techniques, rigorously analyzing the relative impact of static versus dynamic occlusions. This research aims to significantly enhance ADS safety by incorporating the critical dimension of sensor and environmental limitations into automated testing.**

***Index Terms*—Software testing, Autonomous driving system, Fuzzing.**

I. INTRODUCTION

Autonomous Driving Systems (ADS) represent a complex integration of software, hardware, and mechanical components, promising to revolutionize transportation by enhancing safety, convenience, and efficiency. However, ensuring their reliability and safety remains a paramount challenge, as high-profile failures underscore. An ADS typically comprises four principal modules: sensing, perception, planning, and control, which function in a tightly coupled pipeline [1]. A failure in any one module, whether due to software bugs, sensor noise, or adversarial interference, can compromise the entire system's safety.

Testing is widely recognized as an indispensable technique for uncovering latent faults and verifying ADS behavior under realistic conditions [2]. A significant body of research has emerged to address this, often employing fuzzing techniques to generate diverse scenarios involving dynamic object interactions [4], [8]–[12]. However, a critical and frequently overlooked vulnerability stems not only from the behavior of other objects but also from the inherent physical limitations of the sensors themselves and the structure of the environment—specifically, sensor blind spots. These occluded regions can be categorized into two fundamental types: Dynamic Occlusions, caused by other moving objects (e.g., a bus obscuring a pedestrian), and Static Occlusions, caused by persistent environmental features (e.g., a building corner or a large parked vehicle obscuring a crosswalk).

While some prior work has begun to explore environmental factors [13], the systematic generation and evaluation of tests specifically designed to exploit both types of sensor limitations remain an open challenge. Static occlusions present a uniquely challenging problem, creating persistent blind spots that conceal Vulnerable Road Users (VRUs) and necessitate explicit reasoning from the ADS, forming a combinatorial state space that is poorly addressed by traditional fuzzing methods. Current testing methodologies lack a unified approach to generate and evaluate scenarios that specifically target this full spectrum of blind-spot-induced vulnerabilities.

To address this comprehensive gap, this research proposal introduces Blind-Spot Guided Fuzzing, a novel testing framework designed to systematically uncover failures caused by sensor occlusions. Our approach, implemented in Occlusion-Sensitive Fuzzing (OS-Fuzz), encompasses two integrated modules: BlindSpotFuzz (BSF), which targets failures induced by dynamic occlusions, and StaticOccluFuzz(SOF), which expands the paradigm to target failures induced by static environmental occlusions.

Our approach leverages Large Language Models (LLMs) to generate realistic initial scenarios from accident reports and employs multi-objective optimization to evolve test cases that maximize occlusion severity, diversity, and realism. The primary contributions of this work are: (1) an LLM-driven technique for seeding occlusion-critical scenarios, (2) a novel fuzzing framework with specialized metrics for quantifying blind-spot exploration, and (3) a rigorous evaluation protocol to benchmark OS-Fuzz against state-of-the-art methods and to analyze the relative impact of static versus dynamic occlusions.

This proposal outlines our research methodology, presents encouraging preliminary results for dynamic occlusions, details the plan for expanding into static occlusions, and defines our comprehensive evaluation strategy. The remainder of this

2643-1572/25/$31.00 ©2025 IEEE
DOI 10.1109/ASE63991.2025.00396

document is structured as follows: Section II discusses related work, Section III details our methodology, Section IV outlines the evaluation plan, and Section V concludes with progress and future work.

II. Related Work

A. Testing for Autonomous Driving Systems

The multidisciplinary nature of ADS, spanning hardware, embedded software, machine learning, and real-time control, renders traditional testing approaches insufficient [3]. In recent years, a growing body of literature has proposed novel testing frameworks to address this challenge [4], [5], [11]. These works often concentrate on generating diverse dynamic object interactions and traffic scenarios, effectively exploring the behavioral space of other actors on the road. For instance, tools like ScenoRITA [4] focus on generating diverse and mutable test scenarios, while BehAVExplor [5] uses behavior diversity to guide the testing process. Though valuable, these approaches often operate under the assumption of a perfectly functioning sensor suite, neglecting a fundamental source of real-world failures.

B. Fuzzing in ADS Testing

Fuzzing provides a compelling black-box approach for stimulating complex systems, such as ADS, without requiring internal access to proprietary or third-party components [14]. By systematically generating diverse input stimuli, fuzzing can probe an ADS's operational robustness and detect faults that manifest only under rare or unexpected conditions [15]. Frameworks such as DoppelTest [9] and AV-FUZZER [10] demonstrate the effectiveness of fuzz testing for identifying safety violations in autonomous driving software. However, a key challenge in ADS fuzzing is the test oracle problem—determining whether the system's behavior under test is correct. The absence of fine-grained insight into internal module states means that subtle safety violations induced by perceptual uncertainties (such as occlusions) may go undetected without sophisticated, domain-aware oracles.

C. Addressing Sensor Limitations and Blind Spots

One critical but often overlooked aspect of ADS testing is the effect of sensor blind spots—regions occluded or outside the field of view of one or more sensors. Blind spots can arise from static occlusions (e.g., buildings, parked vehicles) or dynamic occlusions (e.g., pedestrians darting from behind obstacles). While some recent work begins to address environmental factors [13], the systematic generation and evaluation of tests specifically designed to exploit these sensor limitations remain an open challenge. Most existing fuzzing techniques, including those mentioned above, are behavior-centric, focusing on the actions of other objects rather than the perceptual capabilities of the ego vehicle itself. This work directly addresses this gap by proposing a sensor-aware fuzzing paradigm that explicitly targets the vulnerability space created by physical sensor constraints.

III. Research Methodology

Our research will be guided by a set of core questions designed to validate the effectiveness of our proposed approach across different occlusion types.

A. Research Questions

- **RQ1:** To what extent does leveraging LLMs for seed generation from accident reports improve the relevance and efficiency of discovering blind-spot-related failures in ADS, compared to random-based seed generation?
- **RQ2:** How effectively do the proposed occlusion-aware metrics guide the fuzzing process towards generating critical and diverse blind-spot scenarios?
- **RQ3:** How does our fuzzing framework perform in terms of fault detection effectiveness and scenario diversity compared to state-of-the-art ADS fuzzing techniques?
- **RQ4:** What is the relative impact and difficulty of static environmental occlusions versus dynamic occlusions on ADS perception and planning modules?

B. Overall Approach

The core of our approach is developing a unified fuzzing framework that can generate scenarios exploiting both dynamic occlusions (such as a bus blocking a pedestrian) and static occlusions (like a building corner hiding a cyclist). We argue that static occlusions create a persistent and uniquely challenging combinatorial state space that traditional fuzzing methods struggle to address. Our method, summarized in Figure 1, consists of four integrated phases.

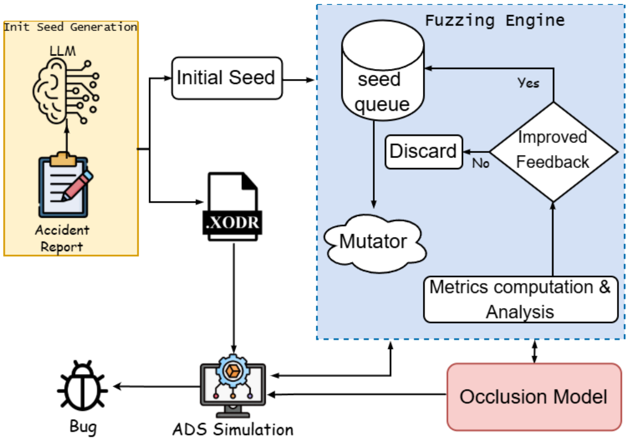

Fig. 1. An overview architecture of OS-Fuzz.

Phase 1: LLM-Driven Seed Generation This phase addresses RQ1. We will develop a seed generation module that uses a Large Language Model (e.g., GPT-4) to analyze real-world accident reports. For dynamic occlusion seeds, the LLM will extract features like interacting objects and motion paths. Importantly, for static occlusion seeds, the LLM will be prompted to identify and extract descriptions of environmental structures, which will then be mapped to parameters for environmental models in simulators. This approach enables us to programmatically generate worlds with specific hazardous static occlusion points.

Phase 2: Multi-Objective Test Case Evolution with an Occlusion Model This phase addresses RQ2 and RQ4. We will implement a fuzzing engine using the NSGA-II algorithm [16]. The key innovation is a generalized Occlusion Model that calculates risk for both occlusion types. For dynamic Occlusions, the model uses quaternions and relative positioning to calculate transient blind spots. For static occlusions, the model analyzes the persistent geometry of the environment to identify static blind zones. The fitness function for the evolutionary algorithm will aim to optimize: *Occlusion Severity*, which measures the "persistent risk" of a location in static occlusions; *Scenario Diversity*, to ensure exploration of both dynamic and static occlusion scenarios; and *Realism*, maintaining physically plausible environments and object behaviors.

Phase 3: The Integrated Fuzzing Framework This phase involves integrating all components into a cohesive framework, which we propose to call Occlusion-Sensitive Fuzzing (OS-Fuzz), comprising the BSF for dynamic occlusions and the StaticOccluFuzz module for static ones. The framework will utilize several core metrics to guide the process and evaluate outputs: *Severity of Occlusion (SO)* Measures the potential risk of a scenario; *Evolutionary Coverage (EC)* Tracks diversity over generations; *Scenario Change Detection (SCD)* Identifies significant shifts; *Coverage of Blind Spot (CBS)* Quantifies exploration of the occlusion space; *Static Occlusion Persistence (SOP)* Quantifies the duration and spatial consistency of a blind spot created by an environmental object.

Phase 4: Iterative Development and Comparative Validation The development will follow an iterative action research cycle [15]. We will first design and implement the framework (*Diagnosing/Planning*), then conduct comparative experiments on ADS platforms such as Apollo and Pylot (*Action Taking*). Importantly, the evaluation will be structured to answer RQ3 and RQ4 by comparing: OS-Fuzz (dynamic) vs. State-of-the-Art Fuzzers (e.g., DoppelTest); OS-Fuzz (static) vs. State-of-the-Art Fuzzers; OS-Fuzz (dynamic) vs. OS-Fuzz (static) to directly analyze the relative impact and difficulty of each occlusion type (RQ4). The insights from each iteration (*Learning*) will be used to refine the models and algorithms, ensuring the framework is robust against both classes of occlusion challenges.

IV. Evaluation Plan

To expand on our initial findings and thoroughly validate the OS-Fuzz framework, we have developed a detailed, multi-phase evaluation plan. This strategy is specifically designed to address the research questions presented in Section III.

A. Evaluation Overview and Baselines

Our evaluation will target two open-source, production-grade ADS platforms to ensure generalizability: Baidu Apollo [17]: A full-stack ADS widely used in research. Pylot [18]: A modular ADS framework ideal for testing perception and planning components in isolation. We will benchmark OS-Fuzz against state-of-the-art ADS fuzzing techniques, which serve as our baselines: DoppelTest [9]: For general scenario generation. BehAVExplor [5]: For behavior diversity-guided testing. AV-FUZZER [10]: As a representative of earlier fuzzing work.

B. Experimental Design to Address Research Questions

The evaluation will be conducted in three phases, each corresponding to a core part of our methodology.

Phase 1: Evaluating Seed Generation (Addressing RQ1)

- *Objective:* Quantify the improvement in relevance and efficiency gained by using LLM-generated seeds from accident reports.
- *Method:* We will generate two sets of initial scenarios: (1) using our LLM-driven method, and (2) through random generation (the baseline). Both sets will be evolved using the same NSGA-II configuration for a fixed time period (e.g., 12 hours).
- *Metrics:*
  - *Relevance:* Percentage of initial seeds that lead to a safety violation (collision, comfort violation).
  - *Efficiency:* Time to first failure (TTFF) and number of unique failures found per unit time.
- *Expected Outcome:* We expect the LLM-generated seed set to yield a higher proportion of relevant seeds and a significantly lower TTFF, demonstrating superior efficiency in finding critical blind-spot scenarios.

Phase 2: Evaluating Framework Effectiveness (Addressing RQ2, RQ3, and RQ4)

- *Objective:* Compare the overall fault-finding capability and diversity of OS-Fuzz against baseline methods, and analyze the differences between static and dynamic occlusion failures.
- *Method:* We will run each fuzzing tool (OS-Fuzz-Dynamic, OS-Fuzz-Static, DoppelTest, BehAVExplor, AV-FUZZER) for an extended period on both ADS platforms across diverse environments (urban, highway, intersection).
- *Metrics:*
  - *Fault Detection:* Total number of unique collisions, comfort violations, and unsafe lane changes detected.
  - *Scenario Diversity:* Measured using the five metrics from our framework (SO, EC, SCD, CBS, SOP) and entropy-based measures.
  - *Occlusion Type Analysis (RQ4):* We will categorize all found failures by their root cause: Dynamic Occlusion, Static Occlusion, or Other. We will then analyze the relative proportion and severity for failures in each category.
- *Expected Outcome:* We expect OS-Fuzz (in both modes) to significantly outperform all baselines in finding unique failures, especially those related to occlusions. We further expect that failures caused by static occlusions will be

uniquely elusive to baseline methods and may reveal more fundamental planning flaws.

Phase 3: Case Studies

- *Objective:* Demonstrate the practical relevance of the failures found by our framework.
- *Method:* We will select the top-most severe failures discovered by OS-Fuzz and conduct a qualitative analysis. This will involve inspecting the ADS's perception and planning modules to identify the root cause and assess the real-world criticality of the scenario.
- *Expected Outcome:* This qualitative analysis will provide strong evidence that the failures found by OS-Fuzz are not just simulation artifacts but represent tangible, high-severity safety risks.

V. Progress and Future Work

To date, we have successfully implemented a proof-of-concept for the core BSF framework, with a focus on dynamic occlusions. We have conducted the preliminary evaluation against Baidu Apollo, as reported in Section IV, which validates our core approach and provides a baseline for further development.

The next step in our research will involve executing the evaluation plan and expanding the framework as follows: First, develop and integrate the SOF module into the unified OS-Fuzz framework. This includes implementing the static occlusion model and the SOP metric. Next, carry out Phases 1 and 2 of the evaluation plans. This involves conducting large-scale fuzzing campaigns on both Apollo and Pylot, collecting extensive data on failures and metrics to answer RQ1 through RQ4. Following that, execute Phase 3 of the evaluation plan. Perform in-depth qualitative analysis and case studies of the most critical failures identified. Finally, complete the thesis writing and dissemination process by finalizing the dissertation and preparing manuscripts for top-tier conferences and journals.

This plan ensures a thorough and systematic validation of our research contributions, leading to a successful PhD dissertation and a notable scholarly impact.

Acknowledgment

This work is partially supported by the National Natural Science Foundation of China (Grant No. 52131204)

References

[1] S. Tang, Z. Zhang, Y. Zhang, J. Zhou, Y. Guo, S. Liu, S. Guo, Y.-F. Li, L. Ma, Y. Xue, and Y. Liu, "A Survey on Automated Driving System Testing: Landscapes and Trends," *ACM Trans. Softw. Eng. Methodol.*, vol. 32, no. 5, pp. 1–62, 2023. [Online]. Available: https://dl.acm.org/doi/10.1145/3579642

[2] Z. Zhong, Y. Tang, Y. Zhou, V. d. O. Neves, Y. Liu, and B. Ray, "A Survey on Scenario-Based Testing for Automated Driving Systems in High-Fidelity Simulation," *arXiv*, 2021. [Online]. Available: http://arxiv.org/abs/2112.00964

[3] Z. Wan, J. Shen, J. Chuang, X. Xia, J. Garcia, J. Ma, and Q. A. Chen, "Too afraid to drive: systematic discovery of semantic dos vulnerability in autonomous driving planning under physical-world attacks," *arXiv preprint arXiv:2201.04610*, 2022.

[4] Y. Huai, S. Almanee, Y. Chen, X. Wu, Q. A. Chen, and J. Garcia, "scenoRITA: Generating Diverse, Fully Mutable, Test Scenarios for Autonomous Vehicle Planning," *IEEE Transactions on Software Engineering*, vol. 49, no. 10, pp. 4656–4676, 2023. [Online]. Available: https://ieeexplore.ieee.org/document/10234383

[5] M. Cheng, Y. Zhou, and X. Xie, "BehAVExplor: Behavior Diversity Guided Testing for Autonomous Driving Systems," in *Proceedings of the 32nd ACM SIGSOFT International Symposium on Software Testing and Analysis*, 2023, pp. 488–500. [Online]. Available: http://arxiv.org/abs/2307.07493

[6] A. Gambi, M. Mueller, and G. Fraser, "Automatically testing self-driving cars with search-based procedural content generation," in *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*. ACM, 2019, pp. 318–328. [Online]. Available: https://dl.acm.org/doi/10.1145/3293882.3330566

[7] A. Guo, Y. Feng, Y. Cheng, and Z. Chen, "Semantic-guided fuzzing for virtual testing of autonomous driving systems," *Journal of Systems and Software*, vol. 212, p. 112017, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0164121224000608

[8] Z. Hu, S. Guo, Z. Zhong, and K. Li, "Coverage-based Scene Fuzzing for Virtual Autonomous Driving Testing," *ArXiv*, 2021. [Online]. Available: http://arxiv.org/abs/2106.00873

[9] Y. Huai, Y. Chen, S. Almanee, T. Ngo, X. Liao, Z. Wan, Q. A. Chen, and J. Garcia, "Doppelgänger Test Generation for Revealing Bugs in Autonomous Driving Software," in *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, 2023, pp. 2591–2603. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10172903

[10] G. Li, Y. Li, S. Jha, T. Tsai, M. Sullivan, S. K. S. Hari, Z. Kalbarczyk, and R. Iyer, "AV-FUZZER: Finding Safety Violations in Autonomous Driving Systems," in *2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 2020, pp. 25–36. [Online]. Available: https://ieeexplore.ieee.org/document/9251068/

[11] S. Lin, F. Chen, L. Xi, G. Wang, R. Xi, Y. Sun, and H. Zhu, "TM-fuzzer: Fuzzing autonomous driving systems through traffic management," *Autom Softw Eng*, vol. 31, no. 2, p. 61, 2024. [Online]. Available: https://doi.org/10.1007/s10515-024-00461-w

[12] S. Kim, M. Liu, J. J. Rhee, Y. Jeon, Y. Kwon, and C. H. Kim, "DriveFuzz: Discovering Autonomous Driving Bugs through Driving Quality-Guided Fuzzing," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '22. Association for Computing Machinery, 2022, pp. 1753–1767. [Online]. Available: https://dl.acm.org/doi/10.1145/3548606.3560558

[13] S. Tang, Z. Zhang, J. Zhou, Y. Zhou, Y.-F. Li, and Y. Xue, "EvoScenario: Integrating Road Structures into Critical Scenario Generation for Autonomous Driving System Testing," in *2023 IEEE 34th International Symposium on Software Reliability Engineering (ISSRE)*, 2023, pp. 309–320. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/10301222

[14] A. Zeller, R. Gopinath, M. Böhme, G. Fraser, and C. Holler, *The Fuzzing Book*. CISPA Helmholtz Center for Information Security, 2024. [Online]. Available: https://www.fuzzingbook.org/

[15] S. Mallissery and Y.-S. Wu, "Demystify the Fuzzing Methods: A Comprehensive Survey," *ACM Comput. Surv.*, vol. 56, no. 3, pp. 1–38, 2023. [Online]. Available: https://dl.acm.org/doi/10.1145/3623375

[16] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002. [Online]. Available: https://ieeexplore.ieee.org/document/996017

[17] Baidu. (2022) ApolloAuto/apollo: An open autonomous driving platform. GitHub. [Online]. Available: https://github.com/ApolloAuto/apollo

[18] I. Gog, S. Kalra, P. Schafhalter, M. A. Wright, J. E. Gonzalez, and I. Stoica, "Pylot: A modular platform for exploring latency-accuracy tradeoffs in autonomous vehicles," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 8806–8813.