Scenario-Based Test Automation for Highly Automated Vehicles: A Review and Paving the Way for Systematic Safety Assurance

Jian Sun, He Zhang, Huajun Zhou, Rongjie Yu, and Ye Tian, *Member, IEEE*

***Abstract*— Highly Automated Vehicles (HAVs) must undergo strict safety testing before being released to the public. Mileage-based on-road testing suffers from unaffordable time costs and high safety risks. Simulated scenario-based testing has been found to be a trustworthy alternative for testing HAVs' built-in algorithms and functionalities. Test automation is typically used to generate target scenarios. This approach facilitates customized testing and avoids wasting time on simple and redundant scenarios. This study aims to review test automation methods and discuss how to accentuate their strengths rather than be trapped in their weaknesses under certain applicable conditions. According to their main purposes, we classify test automation methods into coverage-oriented, unsafe-scenario-oriented, and naturalistic-assessment-oriented categories. To further demonstrate the differences of these methods, we then design numerical experiment to compare the capabilities of seven test automation methods. Finally, we compile our observations to form a comprehensive guide for selecting test automation methods with different test requirements in mind.**

***Index Terms*— Highly automated vehicles, test automation, safety assurance, traffic scenarios.**

I. Introduction

HIGHLY Automated Vehicles (HAVs) are equipped with advanced perception, planning, and control modules that enable them to drive precisely with little to no human involvement. These vehicles have the potential to reduce traffic accidents, alleviate road congestion, improve commutes, and even revolutionize travel pattern [1]–[3]. Despite the rapid development of HAVs, fatal accidents caused by Uber and Tesla in open streets show that the safety of HAVs is still a thorny problem [4], [5]. Systematic safety testing is critical for eliminating or at least reducing possible accidents prior to large-scale deployment. With the goal of ensuring the safety of HAVs, multiple national and continental research projects, such as ENABLE-S3 of the European Union [6] and PEGASUS of Germany [7], and organizations, such as SIP-adus of Japan [8] and SAE International of the United States [9], are currently in the process of proposing new testing methodologies and procedures.

Unlike traditional vehicles testing, the focus of HAV testing is the Safety of the Intended Functionality (SOTIF, [10]) in complex driving environments. The main concern also shifts from vehicle mechanical performance to autonomous driving capability. These changes require a new testing methodology. According to the Working Party on Automated and Connected Vehicles (GRVA) of the United Nations, public road testing, proving ground testing, and simulation testing are the three pillars of safety certification for HAVs [11]. Complementary feedback relationships are present between these three pillars. Testing in the real world has two main disadvantages: the extremely lengthy testing process [12] and potential dangers [13]. Thus, high-fidelity simulation-based testing becomes a necessary step. Instead of simply executing mileage-driven testing in a simulated environment, scenario-based testing is the state-of-the-art in this field. In the context of our paper, we used the definition developed in [14], which stated that a scenario is a temporal sequence of maneuvers and environmental elements, including traffic elements, natural elements, road elements, etc. There are three main reasons for utilizing scenario-based testing. First, driving mileage is composed of a chain of scenarios and HAVs should be able to handle various situations. Second, in scenario-based testing, we can customize test scenarios and avoid wasting time on simple and repeated scenarios. Third, it allows researchers to evaluate rare and extreme scenarios.

However, the core challenges are how to optimize scenario-based testing and improve the efficiency. According to PEGASUS [15], test scenarios can be termed as either functional scenarios, logical scenarios, or concrete scenarios according to their level of abstraction. Functional scenarios contain natural language descriptions. Logical scenarios are the parameterization of functional scenarios, and contain the ranges and distributions of dynamic and static parameters. By applying exact parameter values, a concrete scenario presents a concrete representation of a logical scenario, which is used for scenario testing execution. However, due to the large range of possible scenario parameters and values, parameter combinations explode when we generate concrete scenarios in a simulation [16].

Manuscript received 2 April 2021; revised 26 September 2021 and 25 November 2021; accepted 14 December 2021. Date of publication 28 December 2021; date of current version 12 September 2022. This work was supported in part by the National Key Research and Development Program of China under Grant 2021YFB2501202, in part by the National Natural Science Foundation of China under Grant 52172391, and in part by the Shanghai Automotive Industry Science and Technology Development Foundation under Grant 1912. The Associate Editor for this article was J. Blum. *(Corresponding author: Ye Tian.)*

The authors are with the Key Laboratory of Road and Traffic Engineering, Ministry of Education, Department of Traffic Engineering, Tongji University, Shanghai 201804, China (e-mail: tianye@tongji.edu.cn).

Digital Object Identifier 10.1109/TITS.2021.3136353

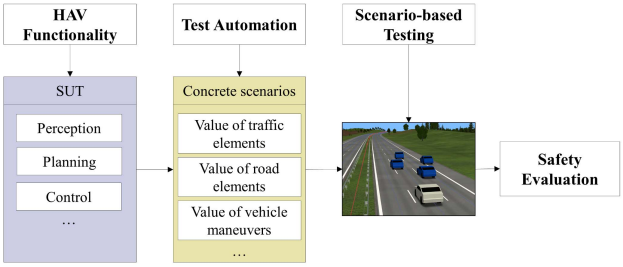


Fig. 1. Technological process of scenario-based simulation testing.

Inspired by software and avionics systems safety verification [17], [18], test automation has been applied. According to PEGASUS [19], test automation automatically generates target concrete scenarios, such that a reliable conclusion of HAV safety performance can be drawn from a smaller number of tests. This approach is one possible workaround for tackling this combination explosion. The functionality to be tested in the scenarios is the System Under Test (SUT). It can be an individual software function or hardware. Here, we do not focus on what the SUT is and its desired level of certification. We instead focus on general methods for automating the testing procedure. The relationships between the SUT, test automation, and scenario-based testing are shown in Fig. 1. The word "scenario" mentioned later mainly refer to concrete scenarios, otherwise we will state clearly.

Test automation methods are evolving rapidly. Some works have summarized the state of the art [19]–[21] based on limited HAV test automation methods. Riedmaier *et al.* [22] carried out a literature review on scenario-based safety assessment. But they paid more attention to steps of scenario-based testing process. The applicability of test automation methods was not fully revealed, mainly due to the following knowledge gaps:

(1) Previous reviews failed to provide a contrastive analysis of different test automation purposes.
(2) Previous reviews ignored the quantitative comparisons between methods about their effectiveness and efficiency.
(3) There is no systematic summary of the applicability of different test automation methods, which hinders the popularization of test automation methods for practical use.

HAV safety testing requires thorough methodology and advanced guidance. To better sort known test automation methods and shed light on test automation methods selection based on these previous works, this paper seeks to make the following three contributions:

(1) We review and classify existing test automation methods according to their purposes. A detailed summary of the 50 mainstream test automation methods is conclued, including their testing purposes, System Under Test, the type of scenarios, scenario parameters, simulation environment and related metrics.
(2) We prove quantitatively that different methods vary in their applicability. Our experiment design makes it possible to intuitively reveal the comparable advantages and disadvantages of various methods.

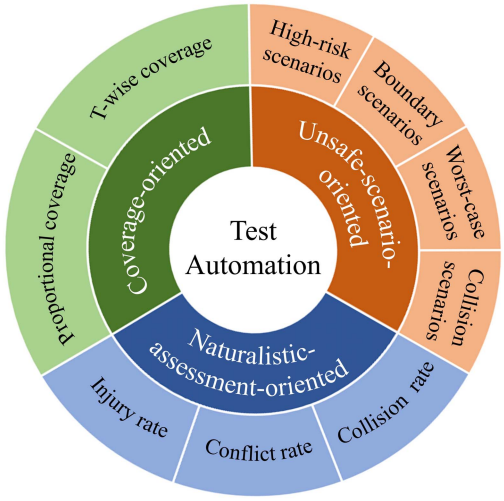


Fig. 2. Classification of test automation methods.

(3) We propose a systematic workflow for test automation methods selection. It aims at the methods selection problem that many HAV testers face.

The rest of this paper is organized as follows. In Section II, we review and classify existing HAV test automation methods. In Section III, we present a detailed quantitative methods comparison. In Section IV, a systematic test automation workflow for HAVs' safety assurance is proposed based on the qualitative and quantitative observations in the previous sections. Section V summarizes the findings and suggests future research directions.

II. Classification and Review of Test Automation Methods

According to their corresponding purposes and applications, we divided test automation methods into coverage-oriented, unsafe-scenario-oriented and naturalistic-assessment-oriented test automation methods (see Fig. 2). The definitions for each category are as follows:

(1) Coverage-oriented test automation method: Methods that maximize testing coverage under a certain coverage metric in a specific Operational Design Domain (ODD).
(2) Unsafe-scenario-oriented test automation method: Methods that generate specific types of scenarios such as high-risk, boundary, collision, and worst-case scenarios to facilitate fault detection.
(3) Naturalistic-assessment-oriented test automation method: Methods that generate scenarios in accordance with naturalistic distributions and can quickly estimate HAV safety indicators such as injury rate, conflict rate and collision rate.

We review the corresponding test automation methods in the order of this classification.

A. Coverage-Oriented Test Automation

One ambition of HAV safety testing is to test as thoroughly as possible with limited time and money resources (i.e. testing resources), such that the successes and failures of HAVs can

be systematically shown. The higher the testing coverage, the more reliable the verification of HAVs' safety. Coverage-oriented test automation can be utilized to maximize testing coverage.

According to Tatar [23], there are five types of possible coverage measures: 1) functional requirement coverage, 2) software source code coverage, 3) operational state coverage, 4) input coverage for a given logical scenario, and 5) coverage across logical scenarios and situations. For HAVs involving many deep learning models, input coverage for a given logical scenario and coverage across logical scenarios and situations are the main focuses.

If testing resources are sufficient, one solution for enlarging testing coverage is to simply generate as many scenarios as possible. Standards, guidelines and expert knowledge can be used for increasing coverage across logical scenarios and situations. Bagschik *et al.* [24] presented an ontology-based, five-layer model for logical scenario representation. It included roads (L1), traffic infrastructure (L2), manipulation of L1 and L2 together (L3), objects (L4) and environment (L5). The model was able to identify comprehensive logical scenarios based on knowledge at hand Jesenski *et al.* [25] divided scenario parameters into vehicle parameters and global parameters such as the traffic light state and weather conditions. To be more realistic, Bayesian networks were used to build the joint distribution of scenario parameters from real-world datasets.

Considering that traffic scenarios and situations are infinite, input coverage is more likely to be measured. Input coverage corresponds to specific logical scenario ODD. If all parameters in the ODD are discrete and have finite values, testing all possible combinations of levels for all factors is the same as achieving 100% input coverage.

A widely-adopted solution for dealing with parameter combination explosion is running thousands of scenarios simultaneously on the cloud. According to some industry reports, companies often use brute force enumeration with cloud computing to save computational time [26]. However, testing on the cloud may still be intractable if the size of the ODD is enormous.

Instead of input coverage and coverage across logical scenarios and situations, the *T-wise* method (i.e. combinatorial testing method) intelligently selects parameters to maximize *T-wise* coverage [27]. It is also adopted to verify software products and communication systems [28], [29]. This method assumes that *T*-parameter combinations can trigger most system circumstances and that the value of other parameters are not related to the results. *T-wise* coverage refers to the ratio of distinct T-parameter combinations covered during testing to the total number of possible T-parameter combinations [30]. Xia *et al.* [31] used a *T-wise* test tool to ensure coverage and test scenarios complexity. This strategy was also applied in [32], where the Analytic Hierarchy Process (AHP) method was adopted to measure the relative importance of scenario parameters. Rocklage *et al.* [33] generated T-parameter combinations for a vehicle merging scenario through non-recursive backtracking. However, the value of T should come from empirical evidence, which is currently absent for HAV testing. As such, it remains unclear whether a *T-wise* method could guarantee that HAVs are fully tested in most circumstances.

TABLE I
FOUR TYPES OF UNSAFE SCENARIOS

Unsafe Scenario Type	Definition
High-risk scenarios	Scenarios that require emergency operations, such as large decelerations and steering, or near-collision scenarios.
Boundary scenarios	Scenarios between the safe and unsafe domains in an ODD.
Collision scenarios	Scenarios where the range between vehicles is zero or negative.
Worst-case scenarios	Extremely unsafe scenario(s) for HAVs in a certain ODD.

For a small-scale ODD, enumeration is possible. For highly complex ODDs, the trade-off between testing resources and test coverage is still fairly ambiguous. Another main obstacle for coverage-oriented testing automation is that traffic scenarios in the real world can never be fully parameterized. Currently, there is no unified and widely recognized standard that relates the coverage of an ODD to the coverage in a naturalistic traffic environment. Achieving full coverage in a naturalistic traffic environment is currently infeasible, but it can remain as an ideal goal.

B. Unsafe-Scenario-Oriented Test Automation

Considering the difficulty of full-coverage testing, exploring unusual, dangerous, and extremely critical scenarios can be seen as low-hanging fruit. All test automation methods that fulfill this purpose are termed as unsafe-scenario-oriented test automation. Unsafe scenarios found in simulation can be reproduced in proving ground or public road testing, which facilitates the efficiency of HAVs' defects detection and safety verification. Unsafe-scenario-oriented test automation methods can be further divided into four sub-categories according to what they try to detect: finding high-risk, boundary, collision, and worst-case scenarios. Their definitions are shown in TABLE I.

Unsafe-scenario-oriented test automation can also be termed as falsification, which seeks to find counterexamples that violate the safety requirements [34]. In the review of Riedmaier *et al.* [22], we noted that falsification methods included three options: (1) using an accident database, (2) taking an exemplary concrete scenario and increasing its criticality, and (3) taking a logical scenario and finding unsafe scenarios within it. Option (1) mainly applies for generating collision scenarios. For Options (2) and (3), they can be widely used for all four unsafe scenario sub-categories.

Different types of unsafe scenarios have different metrics and levels of criticality. Based on existing research (see TABLE II), they are generally measured by relative speed, relative distance, deceleration fluctuation, etc.

Unsafe scenario formulations and searching strategies are hot research topics. A comprehensive introduction of relevant methods is presented below.

TABLE II
SUMMARY OF TEST AUTOMATION METHODS FOR HAVS SAFETY TESTING

	Testing Purpose	Method	System Under Test			Events of Interest	Traffic Scenario	Scenarios Elements		Dataset		Simulation Environment	Metrics
			Perception	Planning	Control			Static	Dynamic	Naturalistic	Simulated		
[24]	Coverage-oriented	Ontology				Input coverage and coverage across logical scenarios	Motorway	*	*	*			
[25]		Bayesian Network					Intersection	*	*	*		CarMaker	
[27]		*T-wise*								*			
[31]		*T-wise*	*			*T-wise* coverage		*	*	*		NI EVS, PreScan	
[32]		*T-wise*		*			Car-following	*	*	*		MATLAB, PreScan and Carsim	
[33]		*T-wise*		*			Merging	*	*		*	VTD	
[38]	Unsafe-scenario oriented	Parameter Variation	*	*	*	Boundary Scenario (Also depicted as Performance bounds/Challenging scenarios/Boundary case failure in literatures)	Entering the highway	*	*	*			TTC, Time to brake, Deceleration
[39]		Online Trajectory Comparison	*	*			Obstacle in the front		*	*			TTC, WTTC Lateral distance
[40]		Binary Search		*			Obstacle in the front	*	*	*			Drivable area
[41]		Evolution Algorithm		*			Intersection Highway scenario		*	*			Drivable area
[42]		Multi-start Optimization, Seed-fill		*	*		Cut-in, Highway exit Car-following		*	*		MATLAB	mnpETTC frequency
[43]		PSO		*			Intersection		*	*		SUMO	Drivable area
[45]		Reasoning on Nonlinear Constraint Formulas			*		Pedestrian vs Vehicle		*		*	OpenDS	Pedestrian hit area
[47]		Bayesian Fault Injection	*	*	*		Freeway and urban driving scenarios		*		*	Carla, DriveSim	Safety envelope[1]
[51]		Analytical derivation	*		*		Car-following		*	*			Brake-threat-number
[52]		Simulated Annealing			*		Lane-change	*	*		*	MATLAB	TTC, speed
[53]		Adaptive Sampling		*			Navigation scenario	*	*				Performance Score
[54]		Adaptive Searching		*			Navigation scenario	*	*				Performance Score
[58]		Simulated Annealing	*				Pedestrian vs Vehicle	*	*		*	Sim-ATAV	Distance, Speed
[59]		Gaussian Process Classification	*				Traffic jam		*		*	CarMaker	TTC
[63]		Real Collisions Reproduction	*			Collision Scenario (Also depicted as Accident scenario/Accident prototypical scenarios/Crash scenarios in literatures)	9 types of collisions			*			Crash severity, Frequency
[64]		Ontology					Collisions described in the police reports	*	*	*		BeamNG	
[65]		Inductive Approach								*			
[66]		Integrated Cause and Context Matching					Intersection	*	*	*			
[69]		Simulated Annealing	*				Pedestrian vs Vehicle, Intersection	*	*		*	Sim-ATAV	Distance
[70]		Genetic Algorithm	*	*	*		5 types of collisions		*		*	LGSVL	Distance
[71]		Alpha-beta Pruning	*	*			Lane change, Intersection turning		*		*		Distance
[72]		Multi-objective Search	*				Pedestrian vs Vehicle	*	*		*	PreScan	TTC, Distance
[73]		Adaptive Stress Testing		*			Pedestrian vs Vehicle		*		*		Distance
[74]		Rapidly-exploring Random Trees		*	*		Car-following		*		*		Distance, Speed
[75]		Adaptive Design of Experiments		*			Car-following		*		*		Distance
[76]		Specific Task Algorithm		*	*		Intersection		*		*	Carla	Distance
[77]		Collision-and-weights Search		*			7 logical scenarios	*	*		*		Distance, Speed
[78]		Sequential Quadratic Programming			*		Steering		*			Simulink	Roll angle, Yaw rate
[79]		Classical Optimal Control, Game Theory			*		Steering		*			ArcSim	Roll angle
[80]		Sequential Quadratic Programming, Mesh Adaptive Direct Search			*		Steering		*			Simulink	Roll angle, Yaw rate
[81]		Genetic Algorithms, GLOBAL Algorithms		*	*	Worst-case Scenarios	Obstacle in the front		*		*		Distance
[82]		Closed-Form Expressions	*		*		Obstacle in the front		*		*		Time of Interventions
[83]		Numerical Approach on Double-worst-case Formulations			*		Obstacle in the front		*		*		Distance and the situation of wheels
[84]		K-means Clustering			*		Car-following	*	*	*			Speed, Headway
[85]		Recurrent Neural Network	*					*	*	*			Performance score
[86]	Indicator-estimation oriented	Monte Carlo	*			Collision rate	Pedestrian vs Vehicle		*	*			Position, speed
[90]		Importance Sampling				Collision rate	Lane-change		*	*			Distance, TTC
[92]		Importance Sampling			*	Collision rate	Highway scenario		*	*			TTC
[93]		Importance Sampling		*	*	Collision rate	Cut-in		*	*			Distance, TTC
[94]		Importance Sampling		*	*	Injury rate, Conflict rate, Collision rate	Car-following		*	*			Critical distance, MAIS2+[1]
[95]		Importance Sampling		*	*	Injury rate, Conflict rate, Collision rate	Lane-change		*	*			Critical distance, MAIS2+
[96]		Importance Sampling		*	*	Conflict rate	Cut-in		*	*			Proximity zone
[97]		Adversarial Adjustments, Importance Sampling		*		Collision rate	5 logical scenarios	*	*	*		Carla	Distance, TTC
[101]		Markov Chain, Monte Carlo		*		Collision rate	Car-following	*	*		*		Distance

[1] Safety envelope: the maximum distance an AV can travel without colliding with any static or dynamic object.

[1] MAIS2+: Maximum Abbreviated Injury Score equal to or larger than 2. It represents moderate-to-fatal injuries.

1) High-Risk Scenarios: High-risk scenarios can pose potential dangers to HAVs or surrounding subjects. If these high-risk scenarios were to occur in the real world, severe accidents may take place. As such, HAVs must be capable of dealing with high-risk scenarios.

There is no universal definition for high-risk scenarios. Generally, these scenarios are viewed as dangerous, emergency situations [35], [36]. The most commonly used metric is "Time To Collision", which is defined as the time until two road users collide if they continue at their present speed along

the same path [37]. For instance, Hallerbach *et al.* [38] set a threshold for high-risk scenarios at TTC = 3.9 s. Wang and Winner [39] used TTC, WTTC (Worst-Time-to-Collision) and the lateral distance between HAVs and other vehicles to determine whether a scenario was high-risk or not.

Some searching methods have emerged in recent years. Althoff and Lutz [40] adopted a binary search method to find high-risk driving situations on straight, non-intersecting roads. Klischat and Althoff [41] used evolutionary algorithms to generate high-risk test scenarios for complex road layouts and dynamics. Feng *et al.* [42] defined high-risk scenarios based on maneuver difficulty and exposure frequency. This criterion differs from most existing studies, which usually don't take frequency into account. Feng *et al.* [42] also designed a search algorithm based on multi-start optimization and seed-fill method. Klischat *et al.* [43] generated high-risk scenarios by increasing the criticality of general traffic scenarios using a nonlinear optimization method. They are now available on the CommonRoad website [44]. Nonnengart *et al.* [45] applied unsafe constraints to transform normal maneuvers of HAVs into emergency situations. The unpredicted risks may cause HAV modules to lose the expected performance. As such, the idea of Fault Injection (FI) is adopted to validates the fault tolerant of systems. Chen *et al.* [46] applied ANN-based classifier to quickly indicate the safety of FI-scenarios. Jha *et al.* [47] presented a machine learning-based fault injection engine named DriveFI. By mining high-risk scenarios, 561 safety-critical faults were found within 4 hours. Considering that multi-agent systems (MASs) have been studied extensively, Wang *et al.* [48] concluded some fault-tolerant strategies for HAVs platoon system.

TestWeaver, originally developed by QTronic, is a test automation software applied to HAV safety testing. It implements game theory to maximize coverage for high-risk scenarios [49], and has been adopted by Daimler for HAV safety testing [50].

Currently, most research can assess the impact of dynamic objects, but few models involve static objects and environmental parameters (see TABLE II). Some high-risk scenarios should correspond to different metric thresholds in different environments. Taking TTC as an example, it is not appropriate to adopt a consistent TTC to judge if a scenario is high-risk no matter how poor the road conditions are. The definition of high-risk scenarios still lacks a uniform standard, and thresholds mainly rely on subjective expert experience.

2) Boundary Scenarios: Boundary scenarios are defined as regions in the parameter space where small changes can lead to transitioning between testing results. It distinguishes between safe domain and unsafe domain in an ODD. Testing these scenarios is vital as they mark the limit of HAVs' safety quality.

There are various methods used to recognize boundary scenarios. Stellet *et al.* [51] used analytical derivation to discover vital transitions in performance over sensor detection ranges and autonomous emergency brake systems. Tuncali *et al.* [52] designed a robustness evaluation function that found boundaries between safe and unsafe behavior. Mullins *et al.* [53], [54] introduced an adaptive search technique. This technique was designed to find performance boundaries for a SUT with higher parameter dimensions. Surrogate Models (SMs) are always a part of adaptive search and help address the searching problem [55]. Using data fed back from the SUT, SMs can approximate the SUT and generate more samples, which provides a fast estimation of the objectives. Successful use of SMs results in significant computational time savings [56]. A framework named Sim-ATAV was developed to specifically detect boundary scenario collisions [57], and was used by Tuncali *et al.* [58] to develop adversarial test scenarios generation. It contributed to evaluating the closed-loop properties of HAV models that include the machine learning components. Batsch *et al.* [59] used Gaussian process classification to find boundary scenarios in sparse data sets.

If a test ODD already exists, the primary advantage of boundary scenario evaluation is that it can be used as a benchmark to quickly categorize safe and unsafe scenarios. Analyzing performance changes in these boundary scenarios helps to improve HAV behaviors.

3) Collision Scenarios: Collision scenario evaluation is one of the most important topics for researchers, as it reveals the fairly dangerous aspects of HAVs.

There are two main methods used to extract collision scenarios. One involves a Test Matrix, which is a compilation of test scenarios based on accumulated collision data and expert knowledge. Simulation platforms can be used to reconstruct these collisions based on recorded speed, acceleration, road curvature, etc. HAVs can then be placed in these dangerous scenarios to see if they can perform better than human drivers.

Many projects and institutions have explored collision data and defined test scenarios [60]–[62]. The ASSESS projects [63] used representative collision datasets from four countries to build collision scenarios. Scenarios were selected based on a combination of injury severity and frequency. Gambi *et al.* [64] generated collision scenarios from police accident reports. They also invited 34 participants to assess the accuracy of reconstructed collision scenarios. To generalize test scenarios, several researchers have also explored the connection between collision data and collision causes [65]–[67]. Better understanding of the underlying reasons for collisions can let researchers create prototype collision scenarios. The Test Matrix method has typically been used to verify HAV performance in recorded human accidents. However, it is time-consuming and costly to collect sufficiently detailed collision data.

To overcome these problems, another idea involves intelligent search algorithms [68]–[70]. Masuda *et al.* [71] proposed a rule-based method to search for collision scenarios on a three-lane highway and a signalized intersection. Abdessalem *et al.* [72] provided a test approach that combined a multi-objective search with SMs. It mainly tried to detect and predict pedestrian-vehicle collisions. Koren *et al.* [73] applied an adaptive stress testing methodology. Monte Carlo Tree Search (MCTS) and Deep Reinforcement Learning (DRL) were implemented to find collision scenarios caused by stochastic element perturbance. Koschi *et al.* [74] presented two novel falsification methods to reveal safety flaws in the

ACC systems of HAVs. Rapidly-exploring random tree methods have also been used to generate rear-end collisions. They found that a backward search algorithm was able to find targeted scenarios much faster than a forward search approach. Sun *et al.* [75] proposed an Adaptive Search method to quickly search for collision scenarios and also compared the performance of six SMs. Ding *et al.* [76] combined a specific task algorithm with a generative model to specifically learn the rare event distribution, which helped to overcome the inefficiency of collisions searching. Calò *et al.* [77] used jMetal 5.7 as a search framework to identify dangerous collisions, and then searched for collisions that could be avoided by changing the parameters of the path planner algorithm.

Unlike the Test Matrix method, these approaches do not restrict researchers to vague and limited information mined from existing collision data. These methods also generalize potential collision scenarios to a larger range.

4) Worst-Case Scenarios: The worst-case scenarios generally indicate the most destructive situations HAVs can encounter in a specific ODD. Investigating the functionality of HAVs in worst-case scenarios also helps to improve safety in other scenarios to some extent. Once worst-case scenarios are available, they can be used as a benchmark for the evaluation of various development schemes.

Worst-Case Scenario Evaluation (WCSE) allows researchers to test a wide variety of dangerous scenarios, including ones that are not feasible or too costly to try in the field. For example, Jung *et al.* [78] generated worst-case scenarios for Integrated Chassis Control (ICC) systems about rolling over.

Traditional WCSE of vehicles focused on mechanical performance [79], [80]. For autonomous systems, black-box or gray-box attributes are taken into consideration, such as stochastic error, perturbation and probability uncertainty. Srikanthakumar and Chen [81] used a genetic algorithm and GLOBAL algorithm to assess worst-case scenarios of moving obstacle avoidance systems. Nilsson *et al.* [82] analyzed collision avoidance systems, including measurement errors, nonideal state prediction, sensor delays, and actuator delays. They derived closed-form expressions for worst-case scenarios caused by systematic errors and unexpected future object motion. Liu *et al.* [83] developed a double-worst-case formulation, which considered both the most likely worst-case scenario and the less likely one. This approach improved the robustness of the obstacle avoidance algorithm when dealing with parametric uncertainty. Chelbi *et al.* [84] identified worst-case scenarios with an unsupervised classification technique, where the prediction model of vehicle's driving state contained 18 machine learning techniques. In the study of Xu *et al.* [85], worst-case scenarios were mined by a single layer recurrent neural network with Long Short-Term Memory (LSTM) neurons. They also found similar perceiving defects among several vehicle detectors. In PEGASUS [19], they mentioned that Particle Swarm and Simulated Annealing approaches can also be chosen to generate worst-case scenarios.

A common criticism of "worst-case" analysis is that worst-case scenarios may never occur and designs taking these cases into account are very costly under most operating conditions. However, WCSE is crucial for increasing public trust in HAVs. For instance, in 1992, Consumer Reports rated the safety of the Isuzu Trooper as "unacceptable" after it rolled over in a test maneuver. Isuzu countered that the maneuver performed was extreme and rarely occurred in the real world. Nevertheless, its sales plunged 53% in one year [79].

C. Naturalistic-Assessment-Oriented Test Automation

Testing HAVs in specific scenarios contributes to micro-level fault detection. However, the ultimate goal of safety testing is to validate the performance of HAVs under real driving conditions and to ensure HAVs are safer than average human drivers. Naturalistic-assessment-oriented test automation methods have been proposed to derive safety indicators such as risk rate or collision rate under naturalistic distributions.

Monte Carlo approaches can be used to generate scenarios for naturalistic assessment via the Law of Large Numbers. It samples scenarios from naturalistic driving data. With reliable data and statistical tools, this approach ensures that the distribution of generated test scenarios is consistent with the real traffic environment. Sampling won't finish until the collision rate or risk rate converges within a set confidence interval.

Monte Carlo has been widely used to assess threats in driving scenarios. Nicolao *et al.* [86] used Monte Carlo to develop risk assessment on DaimlerChrysler PROTECTOR. In Danielsson *et al.* [87], they used a Monte Carlo approach to estimate the safety improvements with advanced vehicle dynamic models. Eidehall and Petersson [88] promoted this method and claimed that Monte Carlo can both assist with online safety applications in a vehicle and offline data analysis. However, ()Lam [89] noticed that randomly sampling of Monte Carlo introduced input uncertainty. To assess this problem, Huang *et al.* [90] provided a solution to construct valid confidence intervals for evaluation results.

One problem that cannot be ignored is the efficiency of Monte Carlo. Because vehicle collisions are naturally rare, the estimated collision rate can be an extremely small number. Accurate assessments using Monte Carlo thus require massive samples, making it extremely time-consuming.

Therefore, Importance Sampling (IS), a classical variance reduction technique [91], can be used to accelerate rare-event probability estimation [92]. The core idea of Importance Sampling is to replace the original probability density function with a new one where rare events are more likely to occur. Huang *et al.* [93] and Zhao *et al.* [94], [95] applied this approach to study HAV behavior in traffic scenarios like cut-in, lane changing and car-following. Simulation results show that using Importance Sampling can reduce the required simulation time by 300 to 100,000 times. Xu *et al.* [96] used a genetic algorithm to calculate Importance Sampling parameters, which further improved the efficiency of testing. Feng *et al.* [97] made adversarial adjustments to the naturalistic driving environment, which can significantly reduce the required testing time of Importance Sampling and guarantee the unbiasedness of estimation. Other than Importance Sampling, rare event simulation methods can be found in Juneja and Shahabuddin [98], Estecahandy *et al.* [99], Straub *et al.* [100], etc. It can be

further applied to HAV naturalistic-assessment-oriented test automation.

However, naturalistic-assessment-oriented test automation does have limitations. First, since empirical expectations are estimated from field data, the authenticity of data has a direct effect on result reliability. Second, Monte Carlo and Importance Sampling suffer from probabilistic errors due to the random sampling in the initial conditions. As such, distributions and results may differ from execution to execution [101]. Third, sampling considering the joint distribution of parameters in high-dimensional scenarios has not yet been realized.

D. Format of Test Scenarios

Many of the test automation methods mentioned above are based on numerical simulations. For theoretical research, this is a convenient simplification. However, the formal test scenarios need to be digitized and formalized in a standard format before they can be loaded and executed by test automation software.

There exist brief reviews on scenario specification language in [102] and [103]. The OpenX series, developed by German company VIRES, is the mainstream file format standard for HAV simulated-based testing. It includes OpenDRIVE [104], OpenCRG [105], and OpenSCENARIO [106]. Additionally, scenario specification languages such as Scenic [107], Traffic Sequence Charts [108] and Paracosm [109] also prompt the development of test automation [102], [110].

E. Summary of Literature Review

The contents of 50 well-received works on test automation are summarized in TABLE II. These methods have all contributed to the development of test automation. We can easily see that the unsafe-scenario-oriented test automation research accounts for the largest proportion, followed by naturalistic-assessment-oriented test automation research. The research on coverage is relatively sparse.

Existing test automation methods for HAV safety have been verified in individual experiment designs. It is hard to intuitively distinguish the applicability of these categories of test automation methods. In addition, although many methods have been designed to achieve the same test purpose, different experiment designs make it impossible to perform any quantitative comparison of their efficiency.

Furthermore, quantitative comparison results may vary for different logical scenarios. This may instead mislead future implementations of different logical scenarios if not interpreted properly. Therefore, methods comparison and generalized conclusion are necessary in order to facilitate method selection out of all these choices.

III. Quantitative Method Comparison

After considering existing methods' purposes and applications, we classified them into three categories. To verify the correctness of this classification and the necessity of test methods selection, we sought to quantitatively prove the superiority of various test automation methods. We designed the method comparison experiment with three aspects in mind:

(1) Comparing the capabilities of test automation methods from different categories for achieving the same test purpose.
(2) Comparing the capabilities of test automation methods from the same category for achieving the same test purpose.
(3) Comparing the capabilities of test automation methods when testing in different ODDs.

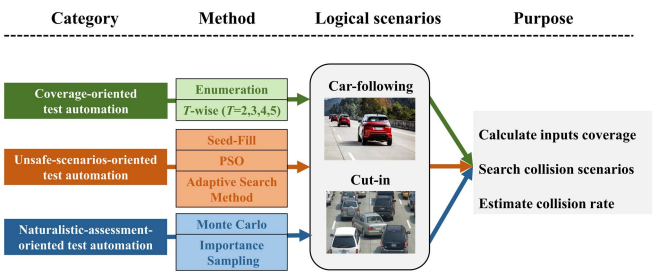


Fig. 3. Procedure of the numerical experiment.

Experiment platform can be simulation tools such as Vires VTD and Cognata [111], [112]. The specific procedure used for comparison is shown in Fig. 3.

We chose seven methods to compare. Enumeration and T-wise are two representatives of coverage-oriented category. Enumeration is the most basic and straightforward test automation method, and we treated it as a baseline. Monte Carlo and Importance Sampling were chosen to represent the naturalistic-assessment-oriented category. Although these two approaches have been compared in many past studies, a full analysis focusing on the three aforementioned aspects has never been done. For the unsafe-scenario-searching category, we selected Seed-Fill, Particle Swarm Optimization (PSO) and the Adaptive Search Method, due to their similarities in output and differences in searching strategy.

The SUT in our experiment was the rear-end collision avoidance functionality fulfilled by a motion planning algorithm. We chose Car-following and Cut-in as the logical scenarios. Not only is this scenario universal in a naturalistic driving environment, the designed low-dimensional Car-following and high-dimensional Cut-in scenarios can provide two distinct ODD spaces. As such, the performance of a method with different ODD spaces can be examined.

A. Methods Description

1) Enumeration: Enumeration tests scenarios within the test ODD space one by one. If only part of the scenarios are enumerated, they should be randomly selected.

2) T-Wise: T-wise is performed by AllPairs, an open source test combinations generator written in Python. If the logical scenario has m scenario parameters, T can be $2, 3, \ldots, m-1$. A candidate scenario is qualified only if its T-parameter combinations were not included in previously sampled scenarios. When the maximum *T-wise* coverage rate is reached, testing terminates.

3) Seed-Fill: Seed-Fill is a classic algorithm from computer graphics [113]. Its key idea is to exhaustively explore the adjacent points of a known space [114]. In the test ODD, each scenario can be treated as a point. We used Seed-Fill to check the unknown scenario points surrounding the tested scenario points.

4) Adaptive Search Method: Measured by an acquisition function [115], the Adaptive Search Method gives testing priority to scenarios predicted to be collisions. In our experiment, we selected KNN (K-Nearest Neighbor) as a surrogate model due to its simplicity, effectiveness and intuitiveness in the field of fault detection [116].

5) Particle Swarm Optimization (PSO): The PSO algorithm randomly samples particles in the test ODD. Particles were set to move towards the collision scenarios in the test ODD space. They continuously adjusted their speeds and positions according to the collision scenarios collected in their own path and the collision scenarios collected by the entire particle swarm.

6) Monte Carlo: To obtain the occurring of collision scenarios, Monte Carlo tests scenarios within the test ODD space using random sampling. Sampling scenarios are selected based on the original probability of each scenario in the test ODD.

7) Importance Sampling: Importance Sampling works by replacing the original probability of scenarios with a new one which increases the occurrences of rare events. The testing process is the same as for Monte Carlo, but sampling scenarios need to be selected based on the Importance Sampling probability of each scenario.

All seven methods are recorded in TABLE II. A more detailed description of these methods can be found in corresponding literatures.

B. Logical Scenario Design

1) Car-Following: Car-following is the most basic functional scenario that a HAV is required to handle. The rear-end collision avoidance functionality of an HAV is characterized by the Intelligent Driver Model (IDM, [117]). IDM is a popular continuous, microscopic, single-lane Car-following model, and has been widely used in adaptive cruise control system simulation [118], [119]. It involves an acceleration process in a free flow state and a deceleration process within congested flow. However, the IDM can decelerate at a rate greater than desired deceleration if the gap between vehicles becomes too small. This braking strategy makes IDM collision-free [120]. In order to capture more failures, we introduced an additional parameter $b_m = 5m/s^2$ as a hard-coded cap on the deceleration. The IDM parameter calibration is shown in Appendix. A.

Car-following is defined as "following the same vehicle ahead within 50 meters in the same lane for more than ten seconds." We considered a case with only one HAV involved. We used the initial speed of the HAV (v_s), the initial speed of the leading vehicle (v_l), and the initial distance between them (S) as the key input parameters (see Fig. 4).

HighD [121] was applied as the data source. A total of 30,292 Car-following scenarios were extracted. The upper and lower boundaries of the three key input parameters were rounded to construct a test ODD via Full Factorial Design [122] (see TABLE III). According to all the possible levels of each parameter, we can produce $43 \times 41 \times 46 = 81,098$ concrete scenarios in all. The probability of each scenario was determined using Gaussian Mixed Model (GMM) joint distribution of v_s, v_l and S from naturalistic Car-following scenarios. For a Gaussian Mixture Model with K components, the k^{th} component has a mean of $\vec{\mu}_k$ and covariance matrix of $\sum_k$ for the multivariate case. The mixture component weights are defined as Φ_k for component C_k, with the constraint that $\sum_{i=1}^{K} \Phi_i = 1$. The scenario probabilities $p(\vec{x})$ can be formulated as (1):

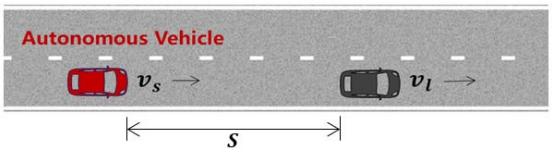


Fig. 4. Car-following logical scenario.

TABLE III
PARAMETER RANGES FOR THE CAR-FOLLOWING SCENARIOS

Inputs	Unit	Range	Value Interval	Levels
v_s	m/s	[1, 43]	1	43
v_l	m/s	[1, 41]	1	41
S	m	[5, 50]	1	46
Products				81,098

$$p(\vec{x}) = \sum_{i=1}^{K} \Phi_i N(\vec{x}|\vec{\mu}_k, \sum\nolimits_i)$$
$$N\left(\vec{x} \mid \vec{\mu}_k, \sum\nolimits_i\right) = \frac{1}{\sqrt{2\pi^k |\sum\nolimits_i|}} \exp\left(-\frac{1}{2}(\vec{x} - \vec{\mu}_k)^T \times \sum\nolimits_i^{-1} (\vec{x} - \vec{\mu}_k)\right) \quad (1)$$

2) Cut-In: Cut-in is commonly seen and may cause severe collisions. When a Cut-in maneuver is performed in real traffic, related elements and influencing factors are often complicated. Traffic participants in surrounding lanes can all impact the outcome of a cut-in maneuver. In order to simplify the model and ensure simulation accuracy, we considered three participants in a Cut-in scenario: the HAV, the Cut-in HAV, and the leading vehicle. We used the initial speeds of three vehicles (v_1, v_2, v_3), the initial longitudinal gaps (S_{1x}, S_{2x}) and the initial lateral gap (dis_{1y}) as the key input parameters (see Fig. 5).

We decoupled the Cut-in motion into longitudinal and lateral motion. In the longitudinal direction, the motion between the three cars can be regarded as two separate Car-following behaviors. As such, we still applied the IDM and b_m for motion planning. In the lateral direction, we treated the Cut-in motion trajectory as a third-order Bezier curve. The testing results recorded if any two cars collided.

We only considered the Cut-in maneuver within 50 meters in the front of a HAV. A total of 993 Cut-in scenarios were

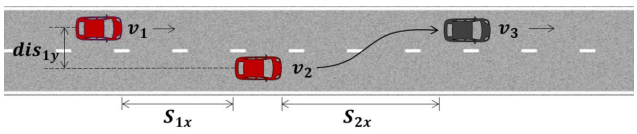
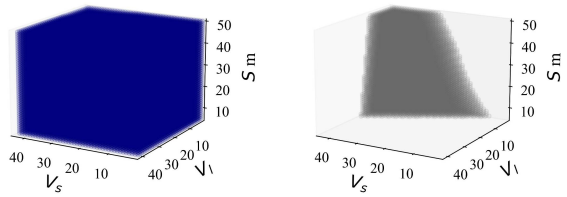


Fig. 5. Cut-in logical scenario.



(a) Parameters Orthogonal Combinations (b) Unavoidable Collisions

Fig. 6. Unavoidable collisions caused by irrational parameter combinations.

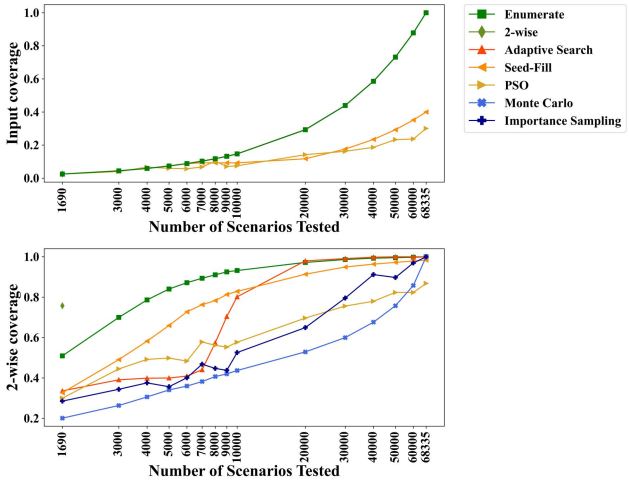


Fig. 7. Comparison results for coverage in Car-following test ODD.

extracted from HighD. The upper and lower boundaries of the six key input parameters were rounded to construct a test ODD via Full Factorial Design (see TABLE IV). The probability of each scenario was determined from the GMM joint distribution of $v_1, v_2, v_3, S_{1x}, S_{2x}$ and dis_{1y} from naturalistic Cut-in scenarios.

3) Elimination of Unavoidable Collisions: Constructing the test ODD through Full Factorial Design has one shortcoming. This approach causes it to contain many parameter combinations that obviously result in unavoidable collisions just from the initial state. These unavoidable collisions are actually meaningless for testing and need to be removed due to their irrational existence. In our experiment, they were simply defined as collisions that could not be avoided even if the following vehicle braked at the highest deceleration b_m at the very start. It represented situations where the initial distance between the two vehicles was too close or the initial speed of the following vehicle was too high. Take Car-following scenario as an example, the deceleration time t_d can be formulated as v_s/b_m, and unavoidable collisions were formulated as (2):

$$S \leq \frac{v_s^2 - 2v_s v_l}{2b_m} \tag{2}$$

After removing these unavoidable collisions, there were 68,335 scenarios left in the Car-following test ODD and 1,778,745 scenarios left in the Cut-in test ODD. Since the Car-following scenarios can be described by a three-dimensional matrix, we visualize the unavoidable collisions within the Car-following scenarios in Fig. 6.

C. Comparison Criteria

We executed several rounds of testing and used different numbers of scenarios for each round (e.g., 10,000 scenarios or 20,000 scenarios). That being said, the scenarios being sampled in later rounds were not an accumulation of the previous rounds. If fewer scenarios were needed to achieve the testing purpose, the method was determined to be more capable.

As stated in Fig. 3, we have three purposes for testing. "Calculating coverage" means evaluating the input coverage and T-wise combination coverage for the sampled scenarios across the entire ODD. "Searching collision scenarios" means prioritizing collision scenarios searching instead of safe scenarios. "Estimating collision rate" means estimating the collision rate of the tested motion planner for the two logical scenarios.

D. Comparison of Results for Car-Following Test ODD

We use green lines to represent coverage-oriented methods, orange lines for unsafe-scenario-oriented methods, and blue lines for naturalistic-assessment-oriented methods. We executed 15 rounds of testing for each testing purpose.

The performance of each method for achieving coverage is shown in Fig. 7. For most methods, the same number of sampled scenarios corresponded with the same input coverages. However, for some unsafe-scenario-oriented methods, such as Seed-Fill and PSO, repeated samples were involved in the searching process, so it was difficult to achieve the ideal input coverage.

The *2-wise* method achieved *2-wise* coverage of 75.7% with 1,690 scenarios, which required far less time when compared with enumeration, which only achieved 70.0% *2-wise* coverage with 3,000 scenarios. Theoretically, *2-wise* can achieve 100% coverage of 2-parameter orthogonal combinations as shown in Fig. 6 (a) with 2,006 scenarios. Once unavoidable collision scenarios were removed, the test ODD was no longer orthogonal and some 2-parameter orthogonal combinations were accordingly removed (same for the Cut-in test ODD). The other methods lagged far behind *T-wise* and enumeration in terms of achieving coverage.

The performance of each method when searching for collision scenarios is shown in Fig. 8. Using enumeration to explore the test ODD, we found a total of 6,320 collision scenarios. If a method was able to find all 6,320 collisions, that method achieved 100% collision scenario coverage.

One can easily see that unsafe-scenario-oriented methods have a notable advantage over other methods in terms of

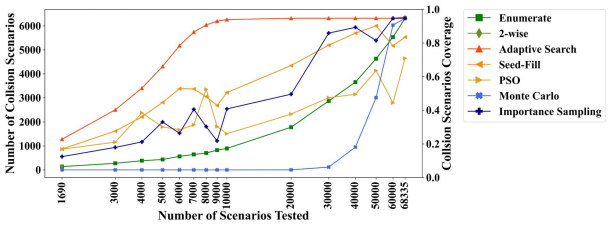


Fig. 8. Comparison results for collision searching in Car-following test ODD.

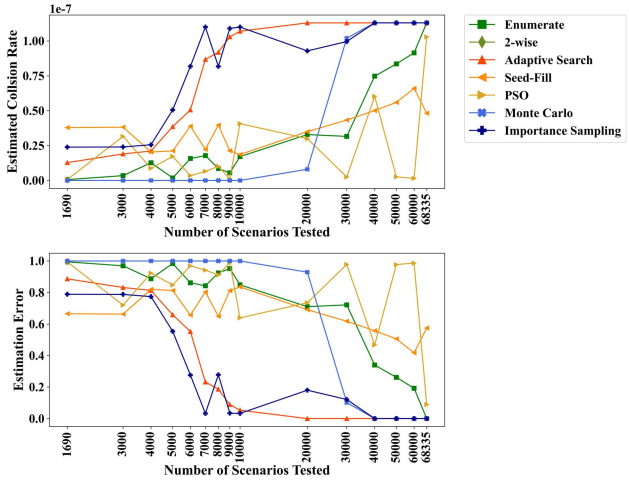


Fig. 9. Comparison results for collision rate estimation in Car-following test ODD.

finding collision scenarios. To identify more than 90% of the collision scenarios, the Adaptive Search Method only needed 7,000 sampled scenarios. Essentially, one collision scenario was captured by every 1.1 samples on average. However, it took more than 60,000 and 55,000 sampled scenarios for Enumeration and Monte Carlo to reach the same collision scenarios coverage. Seed-Fill and PSO also had good efficiency for the first several rounds. However, due to the stronger randomness of these two methods, the number of collision scenarios being captured did not strictly increase with the number of sampled scenarios.

Importance Sampling also found more collision scenarios than Enumeration and Monte Carlo since it sampled collision scenarios with higher probability. However, it was still not as capable as unsafe-scenario-oriented methods in general. In addition, the *2-wise* method only found 141 collision scenarios, which means that collisions in our experiment could not be explained by only 2 parameters.

The performance of each method when estimating collision rate is shown in Fig. 9. The exact collision rate calculated by Monte Carlo was 1.13^{-7}. This was also used as the benchmark to evaluate the collision rate estimation error of the other methods.

Between the two naturalistic-assessment-oriented methods, Importance Sampling was much more efficient than Monte Carlo. Importance Sampling only needed 7,000-10,000 sampled scenarios to achieve an estimation error below 5%. Although the speed of Monte Carlo was slower than Importance Sampling, it was still reliable enough to obtain the

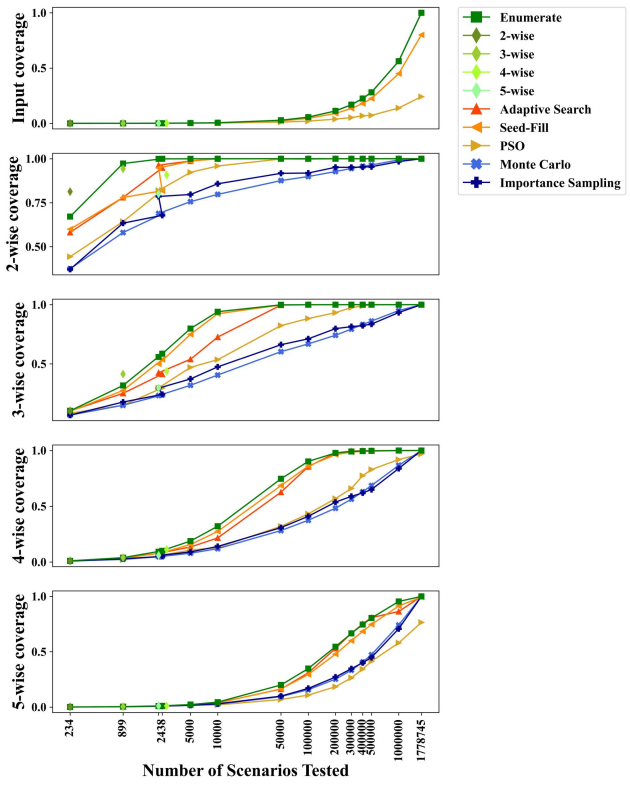


Fig. 10. Comparison results for coverage in Cut-in test ODD.

exact collision rate. Its estimation error dropped to 10% after 30,000 sampled scenarios. Unsafe-scenario-oriented methods such as Seed-Fill and PSO in our experiments were even hard to achieve an estimation error below 40%. In contrast, Adaptive Search showed great promise. By capturing a large proportion of collisions with high efficiency, the collision rate was easily calculated by considering the GMM joint distribution in HighD. Enumeration was also able to obtain the exact collision rate, but it required a total of 68,335 sampled scenarios to achieve so.

E. Comparison of Results in Cut-in Test ODD

We executed 14 rounds of testing for each test purpose. The performance of each method to achieve coverage is shown in Fig. 10. To make the it more concise, we unified the horizontal axis of Fig. 10. Since there were six key input parameters for the Cut-in scenarios, we implemented *2-wise*, *3-wise*, *4-wise* and *5-wise* methods. Due to the limitations of our local machine, we were only able to achieve 41.4% 3-wise coverage over 899 sampled scenarios, 11.2% 4-wise coverage over 2,438 sampled scenarios and 1.1% 5-wise coverage over 2,223 sampled scenarios. As stated by Amersbach and Winner [27], the *2-wise* method is still tractable, whereas the computational burden quickly grows in higher-dimensional *T-wise* algorithms.

However, we were still able to see that the *2-wise* method achieved 81.3% *2-wise* coverage with only 234 scenarios. This means fewer sampled scenarios were necessary since each 6-paramater Cut-in scenario contained many more pairwise parameter combinations.

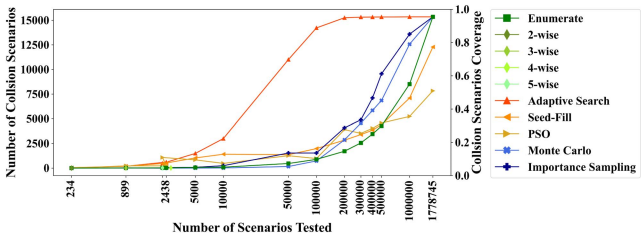


Fig. 11. Comparison results for collision searching in Cut-in test ODD. scenarios.

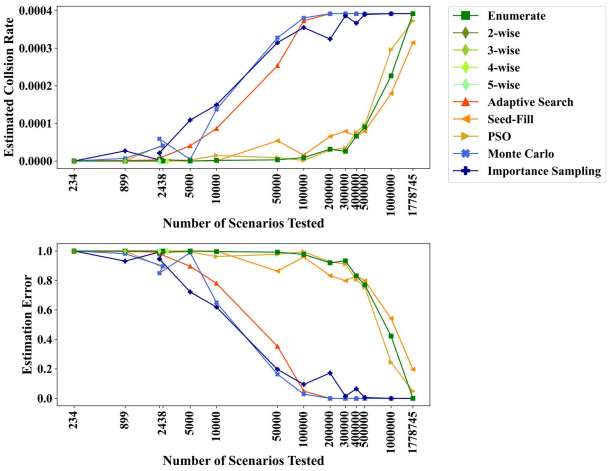


Fig. 12. Comparison results for collision rate estimation in Cut-in test ODD.

In general, coverage-oriented methods dominated in terms of input coverage and *2-wise* coverage. The overall performance trend for each method was the same as that in the Car-following test ODD.

The performance of each method to search for collision scenarios is shown in Fig. 11. Using enumeration to explore the test ODD, we found a total of 15,344 collision scenarios. Since the order of magnitude of the test ODD was large, the efficiency of each method dropped when compared to that of the Car-following test ODD in general.

Nevertheless, Adaptive Search still outperformed the others. It took 100,000 sampled scenarios to find 85.5% of the collision scenarios. One collision scenario was captured by every 7 samples on average. This approach benefited from the dispersed sampling strategy. The other two unsafe-scenario-oriented methods only had a slight advantage in the first few rounds of testing. When searching the test ODD, Seed-Fill merely gave priority to unknown scenarios adjacent to known collision scenarios, resulting in limited searching ability. PSO was easily trapped by local optimal solutions. In this complex and large-scale test ODD, the advantages of Seed-Fill and PSO were weakened.

It is also worth pointing out that 1.1% *5-wise* coverage was able to cover 3.4% of the collision scenarios, which is more effective than *2-wise*, *3-wise*, and *4-wise*. As such, we speculate that collisions in Cut-in scenarios can be explained using five parameter combinations.

The performance of each method to evaluate collision rate is shown in Fig. 12. The exact collision rate calculated by Monte Carlo was 3.9^{-4}. This is higher than that of the Car-following test ODD, which indicates that Cut-in collisions have a higher probability of occurrence.

It is interesting to note that the performances of the two naturalistic-assessment-oriented test automation methods were similar, which means that the efficiency of Importance Sampling was less significant when collisions were not extremely rare. If so, it is not necessary to use Importance Sampling to estimate the collision rate since the solution for the Importance Sampling probability requires additional computational time. Just like in the Car-following test ODD, the capability of Adaptive Search to estimate collision rate was demonstrated in the Cut-in test ODD.

F. Further Discussion

During the method comparison, each test automation method is applied to attempt at three testing purposes respectively, even included the purposes they are not designed to achieve. The comparison results proved that bypassing the boundaries of different method categories would cause potential performance deterioration.

Despite that some methods, such as Adaptive Search, show potentials to deal with multiple purposes, most methods are only appropriate for one certain purpose. For instance, we can also find some collision scenarios using Monte Carlo. However, these collision scenarios were not found explicitly by searching for unsafe scenarios. So, the efficiency was not guaranteed. Therefore, distinguishing methods with different purposes is crucial. This step can help researchers avoid unnecessary wastes during the testing process and go straight to the point.

Comparing different methods with the same purpose is also crucial on a practical level. As our comparison results show, the capability of methods can vary greatly for ODDs with different sizes and collision rates.

In our numerical experiment, we tested automated vehicles' local motion planning algorithms, but in the future, such analysis can be implemented to test any functionality of HAVs as long as they can be parameterized and regulated in a scenario.

IV. A Systematic Safety Assurance Test Automation Workflow

Our ultimate purpose is not limited to demonstrating the distinctions between methods. Based on the decentralized review methods in Section II and comparison results in Section III, we create a tree-like workflow diagram (see Fig. 13). Compared to individual test automation methods, the advantage of our workflow is that we suggest different test automation methods after considering testing purposes, the size of test ODD, testing resources, and the test stage.

Most parts of the test automation workflow come from the comparison results in Section III. For example, if the required testing resources are affordable, testers can adopt enumeration to conduct full-coverage testing. As the testing process evolves, the test ODD expands exponentially. *T-wise* can be applied if testers need to get a rough idea of the HAVs'

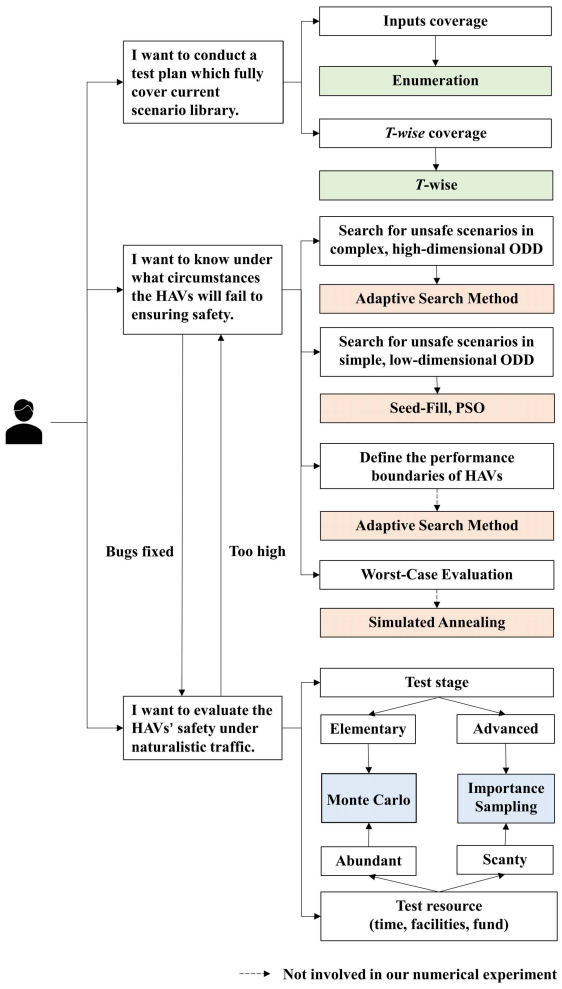


Fig. 13. Workflow conceptual diagram.

TABLE IV
PARAMETER RANGES FOR THE CUT-IN SCENARIOS

Inputs	Unit	Range	Value Interval	Levels
v_1	m/s	[1,48]	3	16
v_2	m/s	[5,43]	3	13
v_3	m/s	[1,43]	3	15
S_{1x}	m	[6,50]	3	15
S_{2x}	m	[5,50]	3	16
dis_{1y}	m	[2,7]	2	3
Products				2,246,400

TABLE V
MODEL PARAMETERS OF IDM

Parameters	Typical value
Desired velocity v_d	29.08 m/s
Safe time headway T	1.6 s
Maximum acceleration a	2.62 m/s2
Desired deceleration b	2.67 m/s2
Acceleration exponent δ	4
Jam distance s_0	1 m
Jam distance s_1	2 m
Vehicle length l	5 m

safety performance. When the purpose is to search for specific unsafe scenarios, Seed-Fill and PSO can be used, whereas the Adaptive Search Method can handle a similar task for complex test ODDs. Unsafe-scenario-oriented test automation also benefits bug-fixing. When HAVs have progressed to a relatively mature stage, naturalistic-assessment-oriented test automation can be carried out. Monte Carlo is applicable when the collision rate is relatively high. However, during later stages of development, collisions will become rarer due to more sophisticated built-in algorithms. At this time, the probability of rare events can be quickly evaluated using Importance Sampling. HAVs' safety performance needs to be compared with a certain safety standard. If it is inferior to said safety standard, it needs to be examined for bug-fixing again.

To ensure that the workflow covered all testing purposes discussed in Section II, the rest of it was derived from literature review. The dotted line marks this distinction in Fig. 13. The Adaptive Search Method can also be used to recognize boundary scenarios if they exist [54]. Simulated Annealing can also be chosen if the test focuses is on worst-case scenarios [19].

By choosing the most appropriate test automation methods or combinations of methods, the efficiency and effectiveness of the entire testing process can be greatly optimized. The workflow is still not fully comprehensive, and some options are still open for discussion. With future supplements, this customized testing scheme will play a significant role for the future development of HAVs.

V. CONCLUSION

In this study, we classified and reviewed related works on test automation methods. Then the distinctions of coverage-oriented, unsafe-scenario-oriented, and naturalistic-assessment-oriented test automation methods were revealed through numerical experiment. A systematic safety assurance test automation workflow was developed to help guide method selection for testers.

There also remain some gaps in this research. First, among the test automation methods with the same purpose, further classifications should be made according to their algorithm design such that relevant methods can be solidly interpreted, compared, and applied. Second, the motion planner and inputs of scenarios are simplified. The efficiency and effectiveness of test automation methods in more complex scenarios need to be discussed. Third, it would be very informative to have many local minima across the test ODD to see if test automation methods could address such case. We are now trying more searching methods and advanced commercialized products such as Apollo and Autoware, which will potentially shed more light on the final guidance proposed.

As HAV technology evolves, machine learning methods such as deep learning and deep reinforcement learning have become prominent. This leaves us with plenty of black boxes

where we are not able to clearly see why failures occur and how we can avoid them. One possible workaround is to enlarge the test coverage and eliminate failures as much as possible. However, coverage-oriented test automation methods are still limited. In the future, there will be more end-to-end applications, and many functions are difficult to parameterize. Therefore, the generalization of test automation remains a challenge.

APPENDIX

A. Model Parameters of the IDM

The specific explanations and values of IDM parameters are shown in TABLE V:

REFERENCES

[1] Z. Wadud, D. MacKenzie, and P. Leiby, "Help or hindrance? The travel, energy and carbon impacts of highly automated vehicles," *Transp. Res. A, Policy Pract.*, vol. 86, pp. 1–18, Apr. 2016, doi: 10.1016/j.tra.2015.12.001.

[2] M. W. Levin and S. D. Boyles, "Effects of autonomous vehicle ownership on trip, mode, and route choice," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2493, no. 1, pp. 29–38, Apr. 2015, doi: 10.3141/2493-04.

[3] M. Martínez-Díaz, F. Soriguera, and I. Pérez, "Autonomous driving: A bird's eye view," *IET Intell. Transp. Syst.*, vol. 13, no. 4, pp. 563–579, Dec. 2018, doi: 10.1049/iet-its.2018.5061.

[4] D. Bissell, "Automation interrupted: How autonomous vehicle accidents transform the material politics of automation," *Political Geogr.*, vol. 65, pp. 57–66, Jul. 2018, doi: 10.1016/j.polgeo.2018.05.003.

[5] National Safety Transportation Board. (2018). *Preliminary Report Highway HWY18FH011*. [Online]. Available: https://www.ntsb.gov/investigations/AccidentReports/Reports/HWY18FH011-preliminary.pdf

[6] A. Leitner, "ENABLE-S3: Project introduction," in *Validation and Verification of Automated Systems*. Berlin, Germany: Springer, 2020, pp. 13–23.

[7] H. Winner, K. Lemmer, T. Form, and J. Mazzega, "PEGASUS—First steps for the safe introduction of automated driving," in *Road Vehicle Automation 5*. Berlin, Germany: Springer, 2019, pp. 185–195.

[8] Y. Sugimoto and S. Kuzumaki, "SIP-adus: An update on Japanese initiatives for automated driving," in *Road Vehicle Automation 5*. Berlin, Germany: Springer, 2019, pp. 17–26.

[9] K. Groh, S. Wagner, T. Kuehbeck, and A. Knoll, "Simulation and its contribution to evaluate highly automated driving functions," *SAE Int. J. Adv. Current Pract. Mobility*, vol. 1, no. 2, pp. 539–549, Apr. 2019, doi: 10.4271/2019-01-0140.

[10] *Road Vehicles—Safety of the Intended Functionality*, Standard ISO/PAS 21448, 2019.

[11] UNECE. (2019). *Proposal for the Future Certification of Automated/Autonomous Driving Systems*. [Online]. Available: https://unece.org/fileadmin/DAM/trans/doc/2019/wp29grva/GRVA-02-09e.pdf

[12] N. Kalra and S. M. Paddock, "Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability?" *Transp. Res. A, Policy Pract.*, vol. 94, pp. 182–193, Dec. 2016, doi: 10.1016/j.tra.2016.09.010.

[13] F. M. Favarò, N. Nader, S. O. Eurich, M. Tripp, and N. Varadaraju, "Examining accident reports involving autonomous vehicles in California," *PLoS ONE*, vol. 12, no. 9, pp. 1–20, Sep. 2017, doi: 10.1371/journal.pone.0184952.

[14] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, "Defining and substantiating the terms scene, situation, and scenario for automated driving," in *Proc. IEEE 18th Int. Conf. Intell. Transp. Syst. (ITSC)*, Gran Canaria, Spain, Sep. 2015, pp. 982–988.

[15] PEGASUS. (2017). *Scenario Description*. [Online]. Available: https://www.pegasusprojekt.de/files/tmpl/PDF-Symposium/04_Scenario-Description.pdf

[16] F. Gao, J. Duan, Y. He, and Z. Wang, "A test scenario automatic generation strategy for intelligent driving systems," *Math. Problems Eng.*, vol. 2019, pp. 1–10, Jan. 2019, doi: 10.1155/2019/3737486.

[17] E. M. A. Rauf and E. M. Reddy, "Software test automation: An algorithm for solving system management automation problems," *Proc. Comput. Sci.*, vol. 46, pp. 949–956, Jan. 2015, doi: 10.1016/j.procs.2015.01.004.

[18] A. K. Jha, "Development of test automation framework for testing avionics systems," in *Proc. 29th Digit. Avionics Syst. Conf.*, Salt Lake City, UT, USA, Oct. 2010, pp. 1–11.

[19] PEGASUS. Test Automation. (2019). *Test-Case Generation & Contribution to the Safety Argument*. [Online]. Available: https://www.pegasusprojekt.de/files/tmpl/Pegasus-Abschlussveranstaltung/17_Test_Automation.pdf

[20] F. Batsch, S. Kanarachos, M. Cheah, R. Ponticelli, and M. Blundell, "A taxonomy of validation strategies to ensure the safe operation of highly automated vehicles," *J. Intell. Transp. Syst.*, pp. 1–20, Mar. 2020, doi: 10.1080/15472450.2020.1738231.

[21] T. Ponn, F. Diermeyer, and C. Gnandt, "An optimization-based method to identify relevant scenarios for type approval of automated vehicles," in *Proc. 26th Int. Tech. Conf. Enhanced Saf. Vehicles (ESV)*, Eindhoven, The Netherlands, 2019, pp. 10–13.

[22] S. Riedmaier, T. Ponn, D. Ludwig, B. Schick, and F. Diermeyer, "Survey on scenario-based safety assessment of automated vehicles," *IEEE Access*, vol. 8, pp. 87456–87477, 2020, doi: 10.1109/ACCESS.2020.2993730.

[23] M. Tatar. (2018). Chasing critical situations in large parameter spaces. QTronic. [Online]. Available: https://www.pegasusprojekt.de/files/tmpl/pdf/Tatar_AVTD_Symposium_2018.pdf

[24] G. Bagschik, T. Menzel, and M. Maurer, "Ontology based scene creation for the development of automated vehicles," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Changshu, China, Jun. 2018, pp. 1813–1820.

[25] S. Jesenski, J. E. Stellet, F. Schiegg, and J. M. Zollner, "Generation of scenes in intersections for the validation of highly automated driving functions," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Paris, France, Jun. 2019, pp. 502–509.

[26] Y. Zhang and Z. Zhong, *China Autonomous Driving Simulation Blue Book 2020*. Beijing, China: ChinaEV, 2020. [Online]. Available: https://case.valuepr.net/file/1012_blue_paper.pdf

[27] C. Amersbach and H. Winner, "Defining required and feasible test coverage for scenario-based validation of highly automated vehicles," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Auckland, New Zealand, Oct. 2019, pp. 425–430.

[28] D. R. Kuhn, D. R. Wallace, and A. M. Gallo, "Software fault interactions and implications for software testing," *IEEE Trans. Softw. Eng.*, vol. 30, no. 6, pp. 418–421, Jun. 2004, doi: 10.1109/TSE.2004.24.

[29] A. W. Williams and R. L. Probert, "A measure for component interaction test coverage," in *Proc. ACS/IEEE Int. Conf. Comput. Syst. Appl.*, Beirut, Lebanon, Jun. 2001, pp. 304–311.

[30] S. R. Dalal and C. L. Mallows, "Factor-covering designs for testing software," *Technometrics*, vol. 40, no. 3, pp. 234–243, Aug. 1998.

[31] Q. Xia, J. Duan, F. Gao, Q. Hu, and Y. He, "Test scenario design for intelligent driving system ensuring coverage and effectiveness," *Int. J. Automot. Technol.*, vol. 19, no. 4, pp. 751–758, Jun. 2018, doi: 10.1007/s12239-018-0072-6.

[32] F. Gao, J. Duan, Z. Han, and Y. He, "Automatic virtual test technology for intelligent driving systems considering both coverage and efficiency," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 14365–14376, Dec. 2020, doi: 10.1109/TVT.2020.3033565.

[33] E. Rocklage, H. Kraft, A. Karatas, and J. Seewig, "Automated scenario generation for regression testing of autonomous vehicles," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, Yokohama, Japan, Oct. 2017, pp. 476–483.

[34] J. Kapinski, J. V. Deshmukh, X. Jin, H. Ito, and K. Butts, "Simulation-based approaches for verification of embedded control systems: An overview of traditional and advanced modeling, testing, and verification techniques," *IEEE Control Syst.*, vol. 36, no. 6, pp. 45–64, Dec. 2016, doi: 10.1109/MCS.2016.2602089.

[35] T. D. Son, A. Bhave, T. Geluk, and H. V. Auweraer, "Autonomous driving control in safety critical scenarios," presented at the JSAE Annu. Congr., Yokohama, Japan, May 2019.

[36] Z. Xinxin, L. Fei, and W. Xiangbin, "CSG: Critical scenario generation from real traffic accidents," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Las Vegas, NV, USA, Oct. 2020, pp. 1330–1336.

[37] J. C. Hayward, "Near-miss determination through use of a scale of danger," *Highway Res. Rec.*, vol. 384, no. 384, pp. 24–34, 1972.

[38] S. Hallerbach, Y. Xia, U. Eberle, and F. Köester, "Simulation-based identification of critical scenarios for cooperative and automated vehicles," *SAE Int. J. Connected Automated Vehicles*, vol. 1, no. 2, pp. 93–106, Apr. 2018, doi: 10.4271/2018-01-1066.

[39] C. Wang and H. Winner, "Overcoming challenges of validation automated driving and identification of critical scenarios," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Auckland, New Zealand, Oct. 2019, pp. 2639–2644.

[40] M. Althoff and S. Lutz, "Automatic generation of safety-critical test scenarios for collision avoidance of road vehicles," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Changshu, China, Jun. 2018, pp. 1326–1333.

[41] M. Klischat and M. Althoff, "Generating critical test scenarios for automated vehicles with evolutionary algorithms," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Paris, France, Jun. 2019, pp. 2352–2358.

[42] S. Feng, Y. Feng, C. Yu, Y. Zhang, and H. X. Liu, "Testing scenario library generation for connected and automated vehicles. Part I: Methodology," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 3, pp. 1573–1582, Mar. 2021, doi: 10.1109/TITS.2020.2972211.

[43] M. Klischat, E. I. Liu, F. Holtke, and M. Althoff, "Scenario factory: Creating safety-critical traffic scenarios for automated vehicles," in *Proc. IEEE 23rd Int. Conf. Intell. Transp. Syst. (ITSC)*, Rhodes, Greece, Sep. 2020, pp. 1–7.

[44] M. Althoff, M. Koschi, and S. Manzinger, "CommonRoad: Composable benchmarks for motion planning on roads," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Los Angeles, CA, USA, Jun. 2017, pp. 719–726.

[45] A. Nonnengart, M. Klusch, and C. Müller, "CriSGen: Constraint-based generation of critical scenarios for autonomous vehicles," in *Proc. Int. Symp. Formal Methods*, Porto, Portugal, 2020, pp. 233–248.

[46] Q. Chen, H. Zhang, H. Zhou, Y. Tian, and J. Sun, "Adaptive design of experiments for fault injection testing of highly automated vehicles," Transp. Res. Board, Washington, DC, USA, Tech. Rep. 22-04554, 2021.

[47] S. Jha *et al.*, "ML-based fault injection for autonomous vehicles: A case for Bayesian fault injection," in *Proc. 49th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw. (DSN)*, Portland, OR, USA, Jun. 2019, pp. 112–124.

[48] W. Wang, B. Han, Y. Guo, X. Luo, and M. Yuan, "Fault-tolerant platoon control of autonomous vehicles based on event-triggered control strategy," *IEEE Access*, vol. 8, pp. 25122–25134, 2020, doi: 10.1109/ACCESS.2020.2967830.

[49] A. Junghanns, J. Mauss, and M. Tatar, "TestWeaver: A tool for simulation-based test of mechatronic designs," in *Proc. 6th Int. Modelica Conf.*, Bielefeld, Germany, 2008, pp. 341–348.

[50] PEGASUS. (2017). *Software-in-the-Loop*. [Online]. Available: https://www.pegasusprojekt.de/files/tmpl/PDF-HZE/16_Software-in-the-Loop.pdf

[51] J. E. Stellet, P. Vogt, J. Schumacher, W. Branz, and J. M. Zöllner, "Analytical derivation of performance bounds of autonomous emergency brake systems," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Gothenburg, Sweden, Jun. 2016, pp. 220–226.

[52] C. E. Tuncali, T. P. Pavlic, and G. Fainekos, "Utilizing S-TaLiRo as an automatic test generation framework for autonomous vehicles," in *Proc. IEEE Intell. Conf. Intell. Transp. Syst. (ITSC)*, Rio de Janeiro, Brazil, Nov. 2016, pp. 1470–1475.

[53] G. E. Mullins, P. G. Stankiewicz, and S. K. Gupta, "Automated generation of diverse and challenging scenarios for test and evaluation of autonomous vehicles," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Singapore, May 2017, pp. 1443–1450.

[54] G. E. Mullins, P. G. Stankiewicz, R. C. Hawthorne, and S. K. Gupta, "Adaptive generation of challenging scenarios for testing and evaluation of autonomous vehicles," *J. Syst. Softw.*, vol. 137, pp. 197–215, Mar. 2018, doi: 10.1016/j.jss.2017.10.031.

[55] H. Beglerovic, M. Stolz, and M. Horn, "Testing of autonomous vehicles using surrogate models and stochastic optimization," in *Proc. IEEE 20th Int. Conf. Intell. Transp. Syst. (ITSC)*, Yokohama, Japan, Oct. 2017, pp. 1–6.

[56] A. Bhosekar and M. Ierapetritou, "Advances in surrogate based modeling, feasibility analysis, and optimization: A review," *Comput. Chem. Eng.*, vol. 108, no. 4, pp. 250–267, Jan. 2018, doi: 10.1016/j.compchemeng.2017.09.017.

[57] C. E. Tuncali, G. Fainekos, H. Ito, and J. Kapinski, "Sim-ATAV: Simulation-based adversarial testing framework for autonomous vehicles," presented at the 21st Int. Conf. Hybrid Syst. Comput. Control, Porto, Portugal, Apr. 2018.

[58] C. E. Tuncali, G. Fainekos, H. Ito, and J. Kapinski, "Simulation-based adversarial test generation for autonomous vehicles with machine learning components," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Changshu, China, Jun. 2018, pp. 1555–1562.

[59] F. Batsch, A. Daneshkhah, M. Cheah, S. Kanarachos, and A. Baxendale, "Performance boundary identification for the evaluation of automated vehicles using Gaussian process classification," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Auckland, New Zealand, Oct. 2019, pp. 419–424.

[60] NHTSA, "Crashes," in *Traffic Safety Facts 2007*. Washington, DC, USA: Nat. Highway Traffic Saf. Admin., 2007, ch. 2, pp. 43–57.

[61] W. G. Najm, S. Toma, and J. Brewer, *Depiction of Priority Light-Vehicle Pre-Crash Scenarios for Safety Applications Based on Vehicle-to-Vehicle Communications*. Washington, DC, USA: National Highway Traffic Safety Administration, 2013. [Online]. Available: https://www.nhtsa.gov/sites/nhtsa.gov/files/811732.pdf

[62] EuroNCAP. (2017). *Test Protocol—AEB Systems*. [Online]. Available: https://cdn.euroncap.com/media/32278/euro-ncap-aeb-c2c-test-protocol-v201.pdf

[63] H. Fagerlind *et al.*, "Analysis of accident data for test scenario definition in the ASSESS project," Transp. Res. Board, Washington, DC, USA, Tech. Rep., 2010. [Online]. Available: https://trid.trb.org/view/1147419

[64] A. Gambi, T. Huynh, and G. Fraser, "Generating effective test cases for self-driving cars from police reports," presented at the 27th ACM Joint Meeting ESEC/FSE, Tallinn, Estonia, 2019.

[65] D. Fleury and T. Brenac, "Accident prototypical scenarios, a tool for road safety research and diagnostic studies," *Accident Anal. Prevention*, vol. 33, no. 2, pp. 267–276, Mar. 2001, doi: 10.1016/S0001-4575(00)00041-5.

[66] M. L. Aust, "Generalization of case studies in road traffic when defining pre-crash scenarios for active safety function evaluation," *Accident Anal. Prevention*, vol. 42, no. 4, pp. 1172–1183, Jul. 2010, doi: 10.1016/j.aap.2010.01.006.

[67] W. G. Najm and D. L. Smith, "Definition of a pre-crash scenario typology for vehicle safety research," in *Proc. 20th Int. Tech. Conf. Enhanced Saf. Vehicles*, Lyon, France, 2007, pp. 1–10.

[68] F. Hauer, A. Pretschner, and B. Holzmüller, "Fitness functions for testing automated and autonomous driving systems," in *Proc. Int. Conf. Comput. Saf. Rel. Secur.*, Turku, Finland, 2019, pp. 69–84.

[69] C. E. Tuncali, G. Fainekos, D. Prokhorov, H. Ito, and J. Kapinski, "Requirements-driven test generation for autonomous vehicles with machine learning components," *IEEE Trans. Intell. Vehicles*, vol. 5, no. 2, pp. 265–280, Jun. 2020, doi: 10.1109/TIV.2019.2955903.

[70] G. Li *et al.*, "AV-FUZZER: Finding safety violations in autonomous driving systems," in *Proc. IEEE 31st Int. Symp. Softw. Rel. Eng. (ISSRE)*, Coimbra, Portugal, Oct. 2020, pp. 25–36.

[71] S. Masuda, H. Nakamura, and K. Kajitani, "Rule-based searching for collision test cases of autonomous vehicles simulation," *IET Intell. Transp. Syst.*, vol. 12, no. 9, pp. 1088–1095, Aug. 2018, doi: 10.1049/iet-its.2018.5335.

[72] R. B. Abdessalem, S. Nejati, L. C. Briand, and T. Stifter, "Testing advanced driver assistance systems using multi-objective search and neural networks," in *Proc. 31st IEEE/ACM Int. Conf. Autom. Softw. Eng.*, Singapore, Aug. 2016, pp. 63–74.

[73] M. Koren, S. Alsaif, R. Lee, and M. J. Kochenderfer, "Adaptive stress testing for autonomous vehicles," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Changshu, China, Jun. 2018, pp. 1–7.

[74] M. Koschi, C. Pek, S. Maierhofer, and M. Althoff, "Computationally efficient safety falsification of adaptive cruise control systems," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Auckland, New Zealand, Oct. 2019, pp. 2879–2886.

[75] J. Sun, H. Zhou, H. Zhang, Y. Tian, and Q. Ji, "Adaptive design of experiments for accelerated safety evaluation of automated vehicles," in *Proc. IEEE 23rd Int. Conf. Intell. Transp. Syst. (ITSC)*, Rhodes, Greece, Sep. 2020, pp. 1–7.

[76] W. Ding, B. Chen, M. Xu, and D. Zhao, "Learning to collide: An adaptive safety-critical scenarios generating method," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Las Vegas, NV, USA, Oct. 2020, pp. 2243–2250.

[77] A. Calo, P. Arcaini, S. Ali, F. Hauer, and F. Ishikawa, "Generating avoidable collision scenarios for testing autonomous driving systems," in *Proc. IEEE 13th Int. Conf. Softw. Test., Validation Verification (ICST)*, Porto, Portugal, Oct. 2020, pp. 375–386.

[78] D. Jung, D. Jung, C. Jeong, Y. Kou, and H. Peng, "Worst case scenarios generation and its application on driving," SAE Tech. Paper 2007-01-3585, 2007, doi: 10.4271/2007-01-3585.

[79] W.-H. Ma and H. Peng, "A worst-case evaluation method for dynamic systems," *J. Dyn. Syst., Meas. Control*, vol. 121, no. 2, pp. 191–199, Jun. 1999.

[80] Y. Kou, "Development and evaluation of integrated chassis control systems," Ph.D. dissertation, Dept. Mech. Eng., Univ Michigan, Ann Arbor, MI, USA, 2010.

[81] S. Srikanthakumar and W. Chen, "Worst-case analysis of moving obstacle avoidance systems for unmanned vehicles," *Robotica*, vol. 33, pp. 1–21, Jan. 2014, doi: 10.1017/S0263574714000642.

[82] J. Nilsson, A. C. E. Ödblom, and J. Fredriksson, "Worst-case analysis of automotive collision avoidance systems," *IEEE Trans. Veh. Technol.*, vol. 65, no. 4, pp. 1899–1911, Apr. 2016, doi: 10.1109/TVT.2015.2419196.

[83] J. Liu, P. Jayakumar, J. L. Stein, and T. Ersal, "Improving the robustness of an MPC-based obstacle avoidance algorithm to parametric uncertainty using worst-case scenarios," *Vehicle Syst. Dyn.*, vol. 57, no. 6, pp. 874–913, Jun. 2019, doi: 10.1080/00423114.2018.1492141.

[84] N. E. Chelbi, D. Gingras, and C. Sauvageau, "Worst-case scenarios identification approach for the evaluation of advanced driver assistance systems in intelligent/autonomous vehicles under multiple conditions," *J. Intell. Transp. Syst.*, pp. 1–28, Dec. 2020, doi: 10.1080/15472450.2020.1853538.

[85] L. Xu, C. Zhang, Y. Liu, L. Wang, and L. Li, "Worst perception scenario search for autonomous driving," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Las Vegas, NV, USA, Oct. 2020, pp. 1702–1707.

[86] G. De Nicolao, A. Ferrara, and L. Giacomini, "Onboard sensor-based collision risk assessment to improve pedestrians' safety," *IEEE Trans. Veh. Technol.*, vol. 56, no. 5, pp. 2405–2413, Sep. 2007, doi: 10.1109/TVT.2007.899209.

[87] S. Danielsson, L. Petersson, and A. Eidehall, "Monte Carlo based threat assessment: Analysis and improvements," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Istanbul, Turkey, Jun. 2007, pp. 233–238.

[88] A. Eidehall and L. Petersson, "Statistical threat assessment for general road scenes using Monte Carlo sampling," *IEEE Trans. Intell. Transp. Syst.*, vol. 9, no. 1, pp. 137–147, Mar. 2008, doi: 10.1109/TITS.2007.909241.

[89] H. Lam, "Advanced tutorial: Input uncertainty and robust analysis in stochastic simulation," in *Proc. Winter Simulation Conf. (WSC)*, Arlington, VA, USA, Dec. 2016, pp. 178–192.

[90] Z. Huang, M. Arief, H. Lam, and D. Zhao, "Evaluation uncertainty in data-driven self-driving testing," 2019, *arXiv:1904.09306*.

[91] P. W. Glynn and D. L. Iglehart, "Importance sampling for stochastic simulations," *Manage. Sci.*, vol. 35, no. 11, pp. 1367–1392, Nov. 1989, doi: 10.1287/mnsc.35.11.1367.

[92] M. O'Kelly, A. Sinha, H. Namkoong, J. Duchi, and R. Tedrake, "Scalable end-to-end autonomous vehicle testing via rare-event simulation," in *Proc. Adv. Neural Inf. Process. Syst.*, Montreal, QC, Canada, 2018, pp. 9827–9838.

[93] Z. Huang, D. Zhao, H. Lam, D. J. LeBlanc, and H. Peng, "Evaluation of automated vehicles in the frontal cut-in scenario—An enhanced approach using piecewise mixture models," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, Singapore, May 2017, pp. 197–202.

[94] D. Zhao, X. Huang, H. Peng, H. Lam, and D. J. LeBlanc, "Accelerated evaluation of automated vehicles in car-following maneuvers," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 3, pp. 733–744, May 2017, doi: 10.1109/TITS.2017.2701846.

[95] D. Zhao *et al.*, "Accelerated evaluation of automated vehicles safety in lane-change scenarios based on importance sampling techniques," *IEEE Trans. Intell. Transp. Syst.*, vol. 18, no. 3, pp. 595–607, Mar. 2017, doi: 10.1109/TITS.2016.2582208.

[96] Y. Xu, Y. Zou, and J. Sun, "Accelerated testing for automated vehicles safety evaluation in cut-in scenarios based on importance sampling, genetic algorithm and simulation applications," *J. Intell. Connected Vehicles*, vol. 1, no. 1, pp. 28–38, Oct. 2018, doi: 10.1108/JICV-01-2018-0002.

[97] S. Feng, X. Yan, H. Sun, Y. Feng, and H. X. Liu, "Intelligent driving intelligence test for autonomous vehicles with naturalistic and adversarial environment," *Nature Commun.*, vol. 12, no. 1, pp. 1–14, Feb. 2021, doi: 10.1038/s41467-021-21007-8.

[98] S. Juneja and P. Shahabuddin, "Rare-event simulation techniques: An introduction and recent advances," in *Handbooks in Operations Research and Management Science*. Amsterdam, The Netherlands: Elsevier, 2006, pp. 291–350.

[99] M. Estecahandy, L. Bordes, S. Collas, and C. Paroissin, "Some acceleration methods for Monte Carlo simulation of rare events," *Rel. Eng. Syst. Saf.*, vol. 144, pp. 296–310, Dec. 2015, doi: 10.1016/j.ress.2015.07.010.

[100] D. Straub, I. Papaioannou, and W. Betz, "Bayesian analysis of rare events," *J. Comput. Phys.*, vol. 314, pp. 538–556, Jun. 2016, doi: 10.1016/j.jcp.2016.03.018.

[101] M. Althoff and A. Mergel, "Comparison of Markov chain abstraction and Monte Carlo simulation for the safety assessment of autonomous cars," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 4, pp. 1237–1247, Dec. 2011, doi: 10.1109/TITS.2011.2157342.

[102] D. J. Fremont *et al.*, "Formal scenario-based testing of autonomous vehicles: From simulation to the real world," in *Proc. IEEE 23rd Int. Conf. Intell. Transp. Syst. (ITSC)*, Rhodes, Greece, Sep. 2020, pp. 1–8.

[103] S. C. Schnelle, M. K. Salaani, S. J. Rao, F. S. Barickman, and D. Elsasser, "Review of simulation frameworks and standards related to driving scenarios," Nat. Highway Traffic Saf. Admin., Washington, DC, USA, Tech. Rep. DOT HS 812 815, 2019. [Online]. Available: https://rosap.ntl.bts.gov/view/dot/43621

[104] (2020). *OpenDRIVE*. [Online]. Available: https://www.asam.net/standards/detail/opendrive/

[105] (2020). *OpenCRG*. [Online]. Available: https://www.asam.net/standards/detail/opencrg/

[106] (2020). *OpenSCENARIO*. [Online]. Available: https://www.asam.net/standards/detail/openscenario/

[107] D. J. Fremont, T. Dreossi, S. Ghosh, X. Yue, A. L. Sangiovanni-Vincentelli, and S. A. Seshia, "Scenic: A language for scenario specification and scene generation," in *Proc. 40th ACM SIGPLAN Conf. Program. Lang. Design Implement.*, Phoenix, AZ, USA, Jun. 2019, pp. 63–78.

[108] W. Damm, E. Möhlmann, T. Peikenkamp, and A. Rakow, "A formal semantics for traffic sequence charts," in *Principles of Modeling*. Berlin, Germany: Springer, 2018, pp. 182–205.

[109] R. Majumdar, A. Mathur, M. Pirron, L. Stegner, and D. Zufferey, "Paracosm: A test framework for autonomous driving simulations," in *Proc. 24th Int. Fundam. Approaches Softw. Eng. Conf. (FASE)*, Luxembourg City, Luxembourg, 2021, pp. 172–195.

[110] A. Eggers, M. Stasch, T. Teige, T. Bienmüller, and U. Brockmeyer, "Constraint systems from traffic scenarios for the validation of autonomous driving," in *Proc. Symbolic Comput. Satsfiability Checking*, 2018, pp. 1–15, doi: 10.29007/x3v9.

[111] K. Franke. (2018). Volkswagen group: Leveraging vires VTD to design a cooperative driver assistance system. Volkswagen. [Online]. Available: https://www.mscsoftware.com/sites/default/files/Volkswagen-Group.pdf

[112] C. Pilz, G. Steinbauer, M. Schratter, and D. Watzenig, "Development of a scenario simulation platform to support autonomous driving verification," in *Proc. IEEE Int. Conf. Connected Vehicles Expo (ICCVE)*, Graz, Austria, Nov. 2019, pp. 1–7.

[113] E.-M. Nosal, "Flood-fill algorithms used for passive acoustic detection and tracking," in *Proc. New Trends Environ. Monit. Using Passive Syst.*, Hyeres, France, Oct. 2008, pp. 1–5.

[114] M. Kalisiak and M. van de Panne, "RRT-blossom: RRT with a local flood-fill behavior," in *Proc. IEEE Int. Conf. Robot. Autom.*, Orlando, FL, USA, May 2006, pp. 1237–1242.

[115] J. Sun, H. Zhou, H. Xi, H. Zhang, and Y. Tian, "Adaptive design of experiments for safety evaluation of automated vehicles," *IEEE Trans. Intell. Transp. Syst.*, early access, Dec. 3, 2021, doi: 10.1109/TITS.2021.3130040.

[116] B. Song, S. Tan, H. Shi, and B. Zhao, "Fault detection and diagnosis via standardized k nearest neighbor for multimode process," *J. Taiwan Inst. Chem. Eng.*, vol. 106, pp. 1–8, Jan. 2020, doi: 10.1016/j.jtice.2019.09.017.

[117] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and microscopic simulations," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 62, no. 2, pp. 1805–1824, Feb. 2000, doi: 10.1103/PhysRevE.62.1805.

[118] A. Kesting, M. Treiber, M. Schönhof, and D. Helbing, "Adaptive cruise control design for active congestion avoidance," *Transp. Res. C, Emerg. Technol.*, vol. 16, no. 6, pp. 668–683, Dec. 2008, doi: 10.1016/j.trc.2007.12.004.

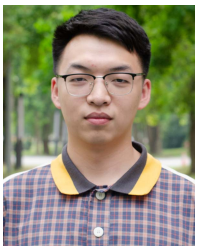
[119] Y. Li, H. Wang, W. Wang, S. Liu, and Y. Xiang, "Reducing the risk of rear-end collisions with infrastructure-to-vehicle (I2V) integration of variable speed limit control and adaptive cruise control system," *Traffic Injury Prevention*, vol. 17, no. 6, pp. 597–603, Aug. 2016, doi: 10.1080/15389588.2015.1121384.

[120] A. Kesting and M. Treiber, "Calibrating car-following models by using trajectory data: Methodological study," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2088, no. 1, pp. 148–156, Jan. 2008, doi: 10.3141/2088-16.

[121] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein, "The highD dataset: A drone dataset of naturalistic vehicle trajectories on German highways for validation of highly automated driving systems," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Maui, HI, USA, Nov. 2018, pp. 2118–2125.
[122] J. Antony, *Full Factorial Designs*. Oxford, U.K.: Elsevier, 2014, pp. 63–85.

Jian Sun received the Ph.D. degree in transportation engineering from Tongji University, Shanghai, China. He is currently a Professor of transportation engineering with Tongji University. His research interests include intelligent transportation systems, traffic flow theory, AI in transportation, and traffic simulation.

He Zhang received the B.S. degree in transportation engineering from Southwest Jiaotong University, Chengdu, Sichuan, China. She is currently pursuing the Ph.D. degree with the Department of Traffic Engineering, Tongji University, Shanghai, China. Her main research interest includes safety test on highly automated vehicle.



Huajun Zhou received the B.S. degree in transportation engineering from Tongji University, Shanghai, China, where he is currently pursuing the master's degree with the Department of Traffic Engineering. His main research interests include AI in transportation and traffic simulation.

Rongjie Yu received the B.Sc. degree in civil engineering from Tongji University, Shanghai, China, in 2010, and the M.Sc. degree in transportation engineering and the Ph.D. degree from the University of Central Florida, Orlando, FL, USA, in 2012 and 2013, respectively. He is currently an Associate Professor at the School of Transportation Engineering, Tongji University. His research interests include traffic safety analysis, driving behavior modeling, proactive safety management, and autonomous vehicle safety evaluation.

Ye Tian (Member, IEEE) received the Ph.D. degree in transportation engineering from The University of Arizona, Tucson, AZ, USA, in 2015. He is currently an Associate Professor of transportation engineering with Tongji University, Shanghai, China. His research interests include safety assurance of automated vehicles, active demand management, dynamic traffic assignment, and mesoscopic traffic simulation.