SAGA: Generating Extreme Scenarios for Autonomous Driving via Adversarial Perturbations

Yan Wang, *Member, IEEE*, Qiong Wu, Xiaoxu Shi, Yishan Li, and Keqin Li, *Fellow, IEEE*

***Abstract*—Evaluating autonomous vehicles' performance in complex, long-tail traffic scenarios, especially under extreme conditions, often highlights the limitations of existing methods in generating realistic and challenging scenarios, which can affect vehicle safety and reliability. To address these gaps, this paper proposes a Scenario-Adaptive Gradient Adjustment (SAGA) network, an adversarial model specifically designed to generate intricate traffic scenarios that closely mimic real-world dynamics. The SAGA network includes a generator and a victim model, where the generator uses adversarial sequences based on the kinematic bicycle model to simulate dynamic vehicle characteristics and calculate precise gradients. These gradients are then used to perturb the victim model, creating safety-critical scenarios essential for evaluating autonomous vehicle performance, such as emergency evasions and complex intersection navigation. Additionally, we use a k-means++ clustering method tailored to categorize ten types of safety-critical scenarios for autonomous driving. The generated scenarios are comprehensive, diverse, and challenging, providing a robust testing environment for autonomous vehicles. Simulation results demonstrate the SAGA network's effectiveness, significantly outperforming traditional non-transparent optimization and the KING method. SAGA achieved a 25% higher success rate than non-transparent optimization and a 2% improvement over the KING method in generating complex scenarios, along with an 8% increase in interpretability, reaching 80%. These findings highlight the capability of SAGA-generated scenarios to thoroughly assess autonomous vehicle performance under diverse traffic conditions, ensuring safer operations.**

***Index Terms*—Autonomous vehicles, traffic scenario generation, adversarial network, safety-critical scenarios.**

I. Introduction

THE safety of autonomous vehicles (AVs) depends on handling complex near-collision scenarios. However, scarce real-world data makes comprehensive assessment challenging, as traditional testing requires hundreds of millions of miles to cover rare critical scenarios.

An effective solution is creating challenging scenarios in simulation rather than relying on real-world data. While scenarios can be manually designed by modifying adversarial agent trajectories, this approach becomes prohibitively expensive at scale. Therefore, actively identifying potential failure modes of autonomous agents is essential for efficiently constructing safety-critical training and assessment scenarios.

Existing work induces failures in driving agents by perturbing the trajectories of dynamic traffic elements in physically feasible ways. This method can be seen as a kinematics-constrained adversarial attack targeted at driving agents, where the volume of safety-critical data generated depends on the success rate of the attack and is limited by a predefined computational budget. Since simulators generally do not support differential operations, non-transparent optimization (NTO) is commonly used. However, we observed in simulators that existing NTO-based attacks [1] suffer from critical reliability issues in triggering collisions for agents trained via imitation learning (IL). Specifically, NTO methods demonstrate poor convergence stability, requiring an average of 150+ iterations to converge, and achieve only 45% success rate in generating collision scenarios consistently. More importantly, the generated scenarios often fail to maintain physical realism and exhibit inconsistent collision patterns across different runs, undermining reliable safety evaluation of autonomous driving systems. This motivated us to explore gradient-based optimization approaches. Meanwhile, the researches on adversarial attacks [2] in image space suggests that the gradient-based optimization methods may be faster and more effective. Additionally, both simulation technologies [3], [4] and driving agents [5], [6], [7], [8], [9], [10] are increasingly adopting end-to-end differentiability. This trend, which allows the entire system to be differentiable, enables the use of gradient-based methods to generate adversarial traffic scenarios more effectively.

In this paper, we define "extreme conditions" as any traffic scenario in which a collision occurs, regardless of the specific environmental settings. These conditions can manifest in various ways, from sparse traffic with sudden maneuvers to highly congested intersections, but they share a common outcome: an unavoidable collision under the current driving agent's policy.

To address the challenges faced by autonomous driving systems in handling such complex and extreme traffic scenarios, we propose an automated extreme scenario generation network named SAGA. Inspired by the adversarial learning princi-

Received 6 March 2025; revised 30 August 2025 and 6 November 2025; accepted 28 November 2025. Date of publication 15 December 2025; date of current version 10 March 2026. This work was supported in part by the Natural Science Foundation of China under Grant 62562048 and Grant 62162047; in part by the Key Research and Development and Technology Transfer Program of Inner Mongolia Autonomous Region under Grant 2025KJHZ0030 and Grant 2025YFHH0110; in part by the Natural Science Foundation of Inner Mongolia under Grant 2023MS06017 and Grant 2023ZD18; in part by the Inner Mongolia Youth Science and Technology Talents Support Project under Grant NJYT24034; in part by the Inner Mongolia Autonomous Region higher Education Carbon Peak and Carbon Neutral Research Project, Ministry of Education, under Grant STZX202322; and in part by the Major Strategic Research and Consulting Project of the Local Institute of the Chinese Academy of Engineering under Grant 2024NMZA-04. The Associate Editor for this article was T. Tettamanti. *(Corresponding author: Yan Wang.)*

Yan Wang, Qiong Wu, Xiaoxu Shi, and Yishan Li are with the College of Computer Science, Inner Mongolia University, Hohhot 010021, China (e-mail: cswy@imu.edu.cn; wq22123@163.com; keviince@keviince.com; 1457950704@qq.com).

Keqin Li is with the Department of Computer Science, State University of New York, New Paltz, NY 12561 USA (e-mail: lik@newpaltz.edu).

Digital Object Identifier 10.1109/TITS.2025.3640578

ples of Generative Adversarial Networks (GAN) [11], SAGA employs gradient-based optimization with a generator and a replaceable victim model to automatically generate safety-critical scenarios. Unlike traditional GANs, SAGA directly optimizes adversarial perturbations for imitation learning (IL) based autonomous driving agents. Through a refined gradient adjustment process using a kinematic bicycle model as the dynamic agent, SAGA achieves critical safety perturbations in non-critical initial scenarios, enhancing the challenge and rigor of testing. The generated scenarios not only enhance the diversity of the training dataset but also significantly improve the autonomous driving systems' capability to handle complex environments.

The main contributions are as follows:

- We propose the SAGA method, which not only dynamically generates challenging scenarios using an improved GAN model but also allows flexible replacement of the victim model to adapt to different configurations of autonomous driving systems, thereby enhancing the applicability and effectiveness of the generated scenarios.
- Through SAGA, we generate challenging, diverse, and highly resolvable test scenarios for driving agents with different input modalities, effectively increasing the breadth and depth of scenario coverage.
- By conducting a detailed analysis of scenarios generated by SAGA, we successfully reveal potential strategic weaknesses of autonomous driving agents, which is crucial for optimizing autonomous driving algorithms and enhancing overall system performance.

II. Related Work

This section investigates and evaluates the research results in two core areas that are directly related to our research, including the generation of safety-critical scenarios and the decision-making models for autonomous driving agents. The theoretical and technical background provided in the literature lays the foundation for the methodologies and experimental of our research.

A. Generation of Safety-Critical Scenarios

Algorithms for generating safety-critical scenarios in autonomous driving can be primarily categorized into three types: data-driven generation, knowledge-based generation, and adversarial generation.

Data-driven generation methods generate various traffic scenarios by sampling collected datasets or using density estimation models to learn scenario distributions [12], [13], [14], [15], [16]. While [12] identified and clustered risky scenarios using factor graphs, direct dataset sampling often faces scarcity challenges in covering rare hazardous scenarios. To address this, [13] developed a data-driven scenario simulator producing LiDAR and trajectory data, though comprehensive coverage remains challenging. Recent learning-based approaches, including SceneGen [14], TrafficSim [4], TrafficGen [17], and DiffScene [18], excel at capturing realistic traffic behaviors from real-world data. In parallel, CDA-SimBoost [19] introduces a digital-twin framework that bridges real and simulated environments for infrastructure-centric cooperative driving, extending scene generation toward real–synthetic integration. Our work focuses on adversarial perturbations that specifically target vulnerabilities in autonomous driving models.

Knowledge-based generation methods create safety-critical scenarios complying with traffic rules and physical laws through domain knowledge integration [20], [21], [22], [23], [24]. However, predefined rules may lack diversity to cover all hazardous situations [20], [23]. Challenges include defining knowledge representation spaces [21] and limited efficiency in constrained optimization frameworks [22], resulting in limited automation and scalability.

Adversarial generation methods generate adversarial perturbations to identify system vulnerabilities in dynamic traffic environments [25], [26], [27], [28], [29]. Recent approaches include optimization-based methods like AdvSim [26] and STRIVE [29], learning-based frameworks [28], and generative approaches using GANs/diffusion [17], [18]. This paradigm efficiently accelerates identification of safety vulnerabilities for testing autonomous driving systems.

Defense mechanisms have also emerged, including adversarially robust object detection through deviation calibration [30] and adversarially-aware training [31]. Sim-to-real adaptation has been addressed through reward-oriented hierarchical learning [32]. Our scenario generation approach provides diverse test cases for evaluating defense mechanisms and enables comprehensive benchmarking of defense effectiveness across different attack types and traffic conditions.

B. Autonomous Driving Models

Traditional autonomous driving systems often adopt a modular design strategy [33], where key functions such as perception, prediction, and planning are developed as independent modules. This approach offers low coupling and high interpretability, facilitating individual module optimization and enhancing system reliability. However, it sacrifices overall optimality due to potential redundancies and efficiency losses from module interactions.

In contrast, end-to-end autonomous driving systems use a single network to directly convert sensor data into vehicle trajectory planning or control signals. This design achieves more direct and optimized decision paths, reducing delays and information loss from multi-module integration. While offering potential advantages in system optimality and reducing human design interventions, end-to-end systems demand higher algorithm performance and generalization capabilities, with notable interpretability challenges that complicate error diagnosis and performance improvement.

End-to-end autonomous driving primarily includes imitation learning (IL) and reinforcement learning (RL). Imitation learning trains models through supervised learning to mimic human driving behaviors [5], [7], [10], [34], [35]. IL methods rely on dataset $D = \{\xi_i\}$ collected under expert policy π_β, training a proxy policy π to match π_β. Main IL approaches include behavior cloning (BC) and inverse optimal control

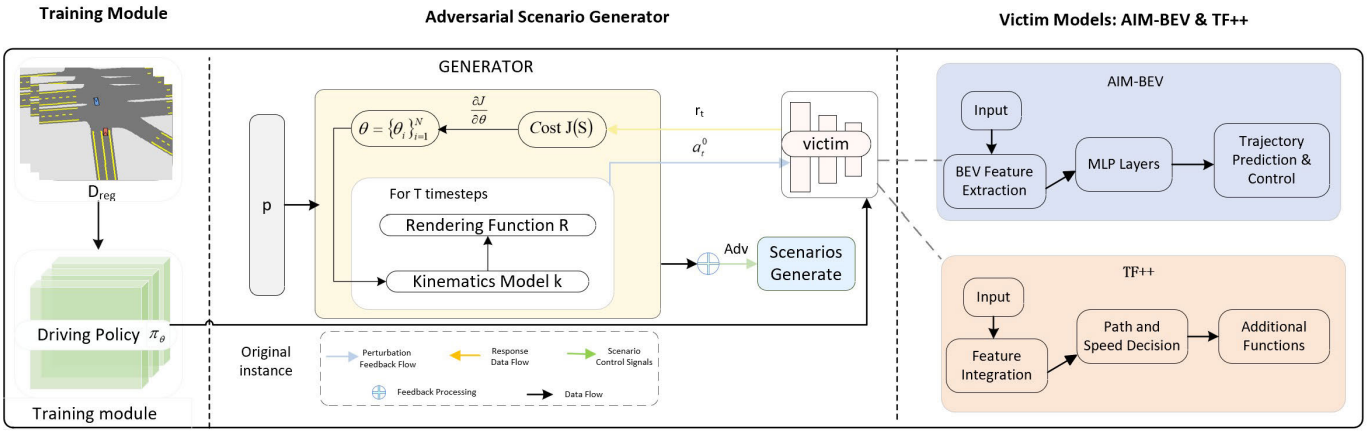

Fig. 1. SAGA Framework Architecture. The rendering function R converts the simulation state x_t to observation O_t using either differentiable rasterization (AIM-BEV) or point cloud processing (TF++). The feedback j_t represents collision indicators and safety metrics from the simulation. Data flows from initial scenario p through the generator $G(z_t;\theta_g)$, which outputs action sequences a_t, then through dynamics model K and rendering R to create observations for the victim model M. The gradient $\nabla_{\theta_g} J$ flows backward for optimization.

(IOC). Early BC applications used end-to-end neural networks to generate control signals from camera inputs [34], with improvements through multi-sensor inputs [36], auxiliary tasks [37], and enhanced expert designs [38]. BC offers simplicity and efficiency without requiring manually designed rewards but faces issues with spurious correlations [10]. IOC infers reward functions from expert demonstrations, optimizing cost functions through methods like GAIL to enhance decision-making safety and interpretability [39], though it faces optimization challenges in high-dimensional scenarios requiring significant data and computational resources.

Reinforcement learning is a trial-and-error method. Deep Q-Networks (DQN) brought deep RL [40] to human-level control on Atari benchmarks. RL requires environments permitting potentially unsafe actions and significantly more training data than IL, posing substantial real-world challenges. Hence, RL research in autonomous driving primarily occurs in simulations. RL has been successfully applied to end-to-end driving and performs better when combined with supervised learning [41]. However, RL's end-to-end training performance remains inferior to IL, possibly due to insufficient gradients for training deep perceptual architectures. RL goals expressed as reward functions often require dense feedback, and existing methods using simple objectives like progress and collision avoidance may encourage risky behaviors [42]. Designing better reward functions and developing algorithms handling sparse rewards remain open challenges despite RL's effective world model integration [43].

Given IL's advantages in enhancing autonomous driving safety and reducing trial-and-error data dependency, we chose IL as our primary learning strategy. IL employs supervised learning to directly mimic human driving behavior through strategies like behavior cloning and inverse optimal control, effectively replicating expert decision-making and enhancing model interpretability and safety. These advantages make IL particularly suitable for practical driving scenarios, especially in high-dimensional continuous environments. Our goal is developing adversarial scenario generation specifically tailored for IL-based systems, which presents unique challenges compared to RL-based approaches.

III. SAGA Generates Safety-Critical Scenarios

In this section, we aim to generate safety-critical scenarios by utilizing driving agents trained using the IL strategy. While SAGA's architecture draws inspiration from GANs with its generator-victim structure, it fundamentally differs by using direct gradient optimization rather than adversarial training. In our framework, imitation learning (IL) serves as the training methodology for victim models (AIM-BEV and TF++), which act as the target systems being tested by our adversarial scenario generator. This design specifically addresses the computational complexity concerns by avoiding the expensive discriminator training process while maintaining scenario generation quality.

In SAGA framework, knowledge integration is achieved through three synergistic approaches: (1) encoding traffic regulations and physical constraints into the objective function, (2) applying K-means++ clustering to historical collision data for identifying typical risk scenarios, and (3) adaptively optimizing generation strategies based on real-time victim model feedback.The SAGA network treats scenario generation as an optimization problem that identifies and exploits vulnerabilities in the victim model. It perturbs attacker agent trajectories in initial scenarios to automatically generate safety-critical scenarios tailored to specific victim models. This method significantly increases the diversity of the training data and expands the distribution of the original dataset.

The overall architecture of safety-critical scenario generation is shown in Figure 1, mainly consisting of three parts: the training module, the adversarial scenario generator, and the victim model. The training module utilizes a broad regular traffic dataset D_{reg} for baseline training of the autonomous driving strategy π_θ, ensuring that the vehicle can make accurate driving decisions in varied traffic conditions. As illustrated in Figure 1, this module serves as the initial stage of the framework, enabling the victim model to first learn standard driving behaviors before encountering adversarial perturbations. This

provides a stable baseline, allowing for fair performance comparisons between regular and adversarial scenarios.

The adversarial scenario generator serves as the core of the framework, implemented through the innovative SAGA network. The SAGA network is composed of two primary components: (1) the generator, which is responsible for creating adversarial traffic scenarios to test and challenge the perception and decision-making capabilities of the victim model; and (2) the victim model, which acts as the autonomous driving agent, tasked with interpreting and responding to the adversarial scenarios produced by the generator, thereby aiding in the adjustment and optimization of subsequent scenarios.

The process begins with an initial instance p, which is fed into the generator. This initial instance defines a specific configuration of the traffic scenario, including the positions, velocities, and states of all agents in the simulation environment. The generator employs a dynamics model k to simulate the physical movement of vehicles, and through a rendering function R, converts the simulated data into visual outputs to evaluate the performance of the victim model. The adversarial output comprises a sequence of actions a_t, which, when combined with the original scenario p, forms a new perturbed scenario $p+a_t$. This perturbed scenario, along with the current state of the victim model, is then input into the scenario generation module to construct the adversarial scenario.

The objective function $J(S)$ uses feedback j_t to evaluate the adversarial agent's action sequence, effectively guiding the attack on the victim model. As the target of these attacks, the victim model responds within the adversarial scenario, providing reactions r_t that, along with the feedback, determine the next sequence of adversarial actions, thereby iteratively updating the scenario. This iterative process continues, with new perturbation sequences a_t being generated and the scenario updated in each iteration, until the adversarial agent's sequence results in a collision with the victim model in the simulation, at which point the optimization process halts. This method ensures that each generated scenario is based on the most recent adversarial actions and the state of the victim model, progressively approaching the collision termination condition.

A. Adversarial Scenario Generator Module

The adversarial scenario generator module iteratively generates adversarial scenarios by optimizing perturbation sequences, effectively testing and challenging the driving strategies of the victim model.The specific design of the module is given below.

To generate these scenarios, the SAGA framework takes structured representations of traffic scenarios as inputs and produces optimized scenarios with the following key features:

Inputs:

- **Scene Layout:** Road geometry, lane boundaries, intersection types, and traffic signal status.
- **Agent State:** Vehicle positions, velocities, orientations, and future trajectory predictions.
- **Environmental Conditions:** Weather and lighting conditions.

Outputs:

- **Collision Rate (CR):** Percentage of scenarios resulting in collisions.
- **Driving Score (DS):** A combined metric that reflects overall driving performance.
- **Time-to-Collision (TTC):** Estimated time before a collision occurs in each generated scenario.

1) Traffic Simulation State and Action: First, we define the agent's state variables as $S_t^i \in \mathbb{R}^2$, representing the planar position, $\psi_t^i \in [0, 2\pi]$ for orientation, and $v_t^i \in \mathbb{R}$ represents the speed magnitude. Here, the index $i = 0$ denotes the autonomous driving agent associated with the victim model. The traffic state is represented as $x_t = \{(S_t^i, \psi_t^i, v_t^i)\}_{i=0}^N$, where N is the number of agents. The state of the victim model is denoted by x_t^i, and the generated scenarios are instantiated as a sequence of these states $X = \{x_t\}_{t=0}^T$, where T is a fixed simulation time. The initial state X is initialized as a regular non-critical dynamic traffic scene. To advance the simulation, we compute the next time step state x_{t+1} using the kinematic bicycle model $K(x_t, a_t)$, given the current state x_t and all agents' actions $a_t = \{a_t^i\}_{i=0}^N$. The control action a_t^i is a two-dimensional vector. This model, due to its robust prior and differentiability on the physical motion of non-omnidirectional vehicles, allows for backpropagation. In the simulation, the victim model adjusts its actions based on the observational data O_t received from the environment.

2) Adversarial Agent Action Sequences and Scenario Generation: To discover safety-critical perturbations, this paper optimizes the adversarial agent's action sequences $\{a_t^i\}_{i>0}^N$. The adversarial agent sequences are continually adjusted based on the victim model's feedback and the evaluation of the objective function $J(S)$, to generate and refine scenarios. This learning process iterates until safety-critical scenarios are generated. Here, θ represents the entire search space of possible actions and scenarios, with a dimension of $N \times T \times 2$, formalized as:

$$\theta = \{\theta_i\}_{i=1}^N, \theta_i = \{a_t^i\}_{t=0}^T \tag{1}$$

This formulation ensures that the adversarial agent can modify both the lateral and longitudinal control of the vehicle during adversarial scenario generation. We find a perturbation sequence by minimizing the objective function $J(S)$, formalized as:

$$\theta^* = \arg\min_{\theta} J(S) \tag{2}$$

3) Cost function:$J(S)$ is a composite function consisting of four parts, each representing different evaluation metrics, all with consistent units, formalized as:

$$J(S) = C_{\text{dist}}^U(S) + \alpha C_{\text{vel}}^U(S) + \beta C_{\text{adv}}^A(S) + \gamma C_{\text{bound}}^A(S) \tag{3}$$

To ensure the stability of the optimization process, each cost function component in $J(S)$ is normalized to maintain consistent numerical scales across different scenarios, preventing a single cost term from dominating the learning process. It can be viewed as a risk metric,

conceptually similar to reinforcement learning reward functions, aiming to reduce collision risk and improve trajectory smoothness.

This normalization technique ensures that all loss terms contribute meaningfully to the overall objective while avoiding numerical instability caused by large differences in magnitude between different cost functions.

- **Collision Cost Metrics for Autonomous Driving Agent $C^U_{\text{dist}}(S)$:** This cost calculates the Euclidean distance between the autonomous driving agent and the nearest adversarial agent to induce collisions, measured in meters (m), and is formalized as:

$$C^U_{\text{dist}}(S) = \min_{i\in\{1,\ldots,N\}} \frac{1}{T}\sum_{t=0}^{T} D(x_t^0, x_t^i) \tag{4}$$

  where $D(x_t^0, x_t^i)$ denotes the Euclidean distance between the bounding boxes of the nearest point of agent i and the autonomous driving agent at time t.
- **Velocity Collision Cost $C^U_{\text{vel}}(S)$:** This cost item evaluates the speed difference at the approach to the collision point and its impact on collision risk. It guides the occurrence of collisions and is formalized as:

$$C^U_{\text{vel}}(S) = \min_{i\in\{1,\ldots,N\}} \frac{1}{T}\sum_{t=0}^{T} v_{\text{bb}}(x_t^0, x_t^i) \times T \tag{5}$$

  where $v_{\text{bb}}(x_t^0, x_t^i)$ represents the speed difference between agent i and the autonomous driving agent at time t.
- **Adversarial Agent Collision Constraint $C^A_{\text{adv}}(S)$:** This metric sets a safety threshold τ to prevent collisions among adversarial agents and is measured in meters (m). It is formalized as:

$$C^A_{\text{adv}}(S) = \min(\min_{\substack{i,j\in\{1,\ldots,N\}\\ t\in\{0,\ldots,T\}}} D(x_t^i, x_t^j), \tau) \tag{6}$$

  The threshold τ defines the range of a repulsion potential that prevents agents from getting too close, thus reducing the risk of collisions.
- **Out-of-Bounds Constraint $C^A_{\text{bound}}(S)$:** This measure penalizes adversarial agents for straying outside the drivable area. It is measured in dimensionless units and is formalized as:

$$C^A_{\text{bound}}(S) = \frac{1}{T}\sum_{i=0}^{N}\sum_{t=0}^{T}\sum_{m=0}^{L} f(B_t^i(p_m)) \circledast \mathcal{M} \tag{7}$$

  where $f(B_i^t(p_m))$ represents the Gaussian potential at the corner points p_m of the bounding rectangle B_i^t of adversarial agent i. The convolution operation $\circledast\mathcal{M}$ with the non-drivable area map $\mathcal{M}$ integrates this Gaussian potential to ensure that the adversarial agents stay within drivable areas. The symbol L denotes the number of corner points of the bounding rectangle B_t^i.

We have ensured that all cost functions are dimensionally consistent, allowing for reasonable comparisons and integrations in the weighted sum to optimize the overall objective function $J(S)$. We have particularly emphasized the inclusion of collision costs for the autonomous driving agent, $C^U_{\text{dist}}(S)$ and $C^U_{\text{vel}}(S)$, while also ensuring that the behavior of adversarial agents remains reasonable within the simulation environment. If any adversarial agent deviates from the drivable area of the map or collides with another hostile agent, the simulation will terminate. To detect collisions, we perform cross-checks on the agents' bounding boxes. For boundary violations, we prevent collisions among adversarial agents by setting thresholds $C^A_{\text{adv}}(S)$, and prevent adversarial agents from straying from the drivable area with boundary regularization $C^A_{\text{bound}}(S)$. The weights of these cost functions are determined by the hyperparameters α, β, and γ to ensure a comprehensive evaluation and to address all related risks and challenges.

B. Driving Agents

We explore the training of autonomous driving agents using the Imitation Learning (IL) strategy on a standard traffic dataset D_{reg} to enhance their adaptability and decision-making capabilities. This training approach uses the D_{reg} dataset, collected from CARLA simulator using the privileged expert agent described in Appendix B. The dataset encompasses a wide range of road types across Town03-Town06 maps, diverse traffic densities (10-50 vehicles), and various weather conditions (clear, rain, fog) with different times of day (dawn, day, dusk, night), ensuring the autonomous systems can effectively learn to drive under real-world conditions. Notably, we introduce two high-performance autonomous driving models, AIM-BEV and TransFuser++ (TF++), which demonstrate exceptional ability in handling complex long-tail traffic scenarios. These models not only undergo regular training to enhance basic navigation and obstacle avoidance capabilities but are also integrated into the SAGA network framework as key victim models to test and respond to generated safety-critical scenarios. SAGA generates challenging scenarios by simulating real vehicle dynamics and complex traffic dynamics, to assess and improve these models' performance under extreme conditions. While existing approaches [44] typically rely on iterative optimization frameworks that demand extensive UAV-based remote sensing data, our SAGA method attains superior performance through efficient adversarial generation without requiring UAV data.

The SAGA framework requires robust victim models that can serve as realistic targets for adversarial scenario generation. To this end, we employ two categories of driving agents: (1) victim models that act as the target systems being tested, and (2) a privileged expert model that serves as the baseline for dataset generation and performance comparison. The selection and configuration of these models is crucial for the effectiveness of our adversarial scenario generation approach.

1) Victim Model Selection and Overview
   Our SAGA framework requires differentiable victim models to enable gradient-based optimization. We specifically selected two state-of-the-art IL-based autonomous driving models: AIM-BEV (Adaptive Instance Mixing with Bird's Eye View) and TransFuser++ (TF++).

AIM-BEV is a model that employs differentiable rasterization for efficient Bird's Eye View representation, enabling direct gradient computation essential for our adversarial optimization process. TransFuser++ represents the enhanced version of the TransFuser architecture [35] with additional perception modules for improved multimodal fusion.

Both models follow the end-to-end learning paradigm, consisting of three core components: perception (processing sensor inputs), decision-making (trajectory planning), and control (action generation). The detailed architectures and training procedures for these models are provided in Appendix A.

2) Model Integration in SAGA Framework
The victim models are integrated into SAGA as differentiable components within the adversarial optimization loop. During scenario generation, these models receive perturbed observations from the generator and produce driving decisions that are evaluated against our safety-critical objective function J(S). The gradient information flows back through the models to guide the adversarial perturbation process.

3) Performance Baseline Establishment
To validate the effectiveness of our generated scenarios, we establish performance baselines using a privileged expert agent. This rule-based system has access to complete simulation state information and serves as both a dataset generator for training and a performance upper bound for evaluation. The expert system implementation details are described in Appendix B.

The diverse scenarios generated by SAGA targeting these victim models require systematic categorization to understand their characteristics and ensure comprehensive coverage of safety-critical situations. This categorization process is essential for both validating the quality of generated scenarios and providing interpretable insights for autonomous driving system improvement.

C. Scene Classification

Building upon the victim models described above, accurately understanding and classifying the collision scenarios generated by SAGA is crucial for optimizing the vehicle's response strategies and validating the comprehensiveness of our approach. For this purpose, this study adopts a K-means++ based scene clustering method to deeply analyze and understand the structure and characteristics of safety-critical scenarios. The clustering serves three purposes: (1) identifying recurring collision patterns in the generated scenarios to provide prior knowledge for generation, (2) ensuring diversity by guiding the optimization to cover different collision types, and (3) providing interpretable categories for safety analysis and validation against real-world collision taxonomies. We utilized the K-means++ method to effectively improve the quality of clustering and accelerate the convergence speed of the algorithm when dealing with datasets with complex feature spaces. The specific implementation steps of the method are detailed below.

1) Data Preprocessing: Initially, for each scene, we calculate key geometric features related to collisions. We define the relative position rel_{pos} and the relative direction coll_{ang} as follows:

$$\text{rel}_{\text{pos}} = \text{pos}_2 - \text{pos}_1 \tag{8}$$

$$\text{distance} = \|\text{rel}_{\text{pos}}\| \tag{9}$$

$$\text{coll}_{\text{ang}} = \tan^{-1}\left(\frac{\text{rel}_{\text{pos}}[1]}{\text{rel}_{\text{pos}}[0]}\right) \tag{10}$$

These features are used to describe the geometric layout of the collision scenarios.

2) Initialization of Cluster Centers: The step is adopted to enhance the stability and efficiency of clustering. An initial center C_1 is randomly selected from the safety-critical dataset, and then the probability of each data point becoming a new center is calculated based on the square of the distance $D(x)$ to the nearest existing center:

$$P(x) = \frac{D(x_i)^2}{\sum_{x \in X} D(x)^2} \tag{11}$$

where $D(x)$ is the distance from the point x to its nearest center.

3) Clustering Iteration: During the iterations, each data point is assigned to the nearest cluster center:

$$\arg\min_{c_i \in C} \|x - c_i\| \tag{12}$$

Subsequently, each cluster center is updated to the mean of the points assigned to it:

$$c_i = \frac{1}{|S_i|} \sum_{x \in S_i} x \tag{13}$$

where S_i is the set of data points assigned to cluster center c_i. The iteration continues until the clustering results stabilize.

IV. Experiments

This research rigorously validates the efficacy of the SAGA network through an extensive series of experiments. In this section, we employ the SAGA network to generate tailored scenarios for the AIM-BEV agent and conduct comprehensive performance comparisons across multiple baseline methods, including non-transparent optimization techniques, rule-based expert models, adversarial generation methodologies, and our proposed SAGA. Subsequently, we delve further into the scenarios generated for both the AIM-BEV and TF++ agents, examining the impact of various scenario generation strategies on the complexity and authenticity of the scenarios. Finally, an ablation study is undertaken, focusing on the assessment of the coefficients of the four cost functions integrated within the objective function. By modulating these coefficients, the individual contributions of each cost function to the model's performance are analyzed, thereby elucidating the significance of each parameter within the scenario generation process. These experiments are designed to provide a comprehensive evaluation of SAGA's practical application efficacy by comparing its performance against multiple baseline methods, thus demonstrating its robustness and potential value within real-world settings.

Fig. 2. Macro Views of Four Virtual Towns.

A. Experimental Setup

For clarity and reproducibility, we summarize the environment, routes/initialization, and evaluation metrics used in our experiments below.

1) Benchmark Testing and Scenario Initialization: We utilize various town scenes (Town03-Town06) provided by CARLA, as shown in Figure 2, to filter from dense routes and initialize adversarial traffic scenarios, thereby creating challenging scenarios tailored for specific driving agents. The simulation environment encompasses several towns, each presenting unique challenges for autonomous driving systems. Town03 is the most complex, featuring five-lane intersections, roundabouts, uneven terrain, and tunnels. Town04 consists of an infinite loop that connects a highway with a small town, making it suitable for continuous testing of driving behaviors. Town05 exhibits a grid layout with intersections and bridges, facilitating evaluations of lane change maneuvers. Finally, Town06 comprises a lengthy highway with multiple entrance and exit ramps, including a Michigan left turn, which enables the examination of merging and diverging traffic behaviors.
   While Town03 offers the most complex road topology, we selected Town10 intersection routes for our controlled experiments because: (1) its grid structure provides reproducible collision scenarios essential for fair comparison, (2) intersection collisions account for approximately 40% of real-world accidents, making it a practically relevant testbed, (3) the focused 80-100m routes enable more iterations within computational constraints, yielding statistically significant results, and (4) previous works (KING, AdvSim) used similar intersection scenarios, facilitating direct comparison.
2) Metric used in this paper:
   - **Route Completion (RC):** The percentage of routes completed by the agent before being blocked or deviating from the route.
   - **Infraction Score (IS):** The cumulative product penalty for each infraction (such as running red lights, violating stop signs, collisions, and lane infractions).
   - **Driving Score (DS):** Calculated as the final metric of RC multiplied by IS for each route.
   - **Collision Rate (CR):** The percentage of routes in which the agent collides while traversing intersections.
   - **Time to Reach 50% Collision Rate ($t_{50\%}$):** The time required for the collision rate to reach 50% during optimization.
   - **Seconds per Iteration (s/it):** The average computation time per optimization iteration.

B. Evaluation Protocol

Building on the above setup, we specify the evaluation protocol for constructing and running scenarios.

Temporal configuration. Our experimental setup uses a 20-second simulation (4 frames per second, 80 frames total), providing sufficient time for autonomous vehicles to travel from start to end while approaching adversarial agents.

Route sampling and scenario construction. We compared adversarial optimization techniques across 80 scenarios by sampling 20 autonomous-vehicle (AV) routes from each of four CARLA maps (Town03–Town06). For each AV route, we densely sampled candidate positions at intersections and enumerated all adversarial-vehicle (AdvV) routes passing close to the AV trajectory, selecting those positioning AdvVs around the AV as start and end positions.

Control, density, and success criterion. A privileged expert drove these adversarial vehicles along designated routes, creating non-critical initial scenarios that simulate CARLA traffic with explicit control over vehicle numbers. We tested three traffic densities: one, two, and four vehicles. Collision Rate (CR)—the proportion of routes resulting in collisions under behavioral constraints—assessed adversarial scenarios. A search succeeds only if all adversarial vehicles remain on drivable map areas without inter-vehicle collisions.

This protocol underpins the results in subsequent subsections.

C. Scenario Generation Assessment

For readability, the baseline driving agent assessment and the detailed results and analysis have been moved to Appendix C. In the remainder of this section, we use AIM-BEV and TF++ as the main victim models for scenario generation and comparisons, since their superior performance

TABLE I
COMPARISON OF VARIOUS METHODS ACROSS DIFFERENT SCENARIOS WITH VARYING NUMBER OF AGENTS AND COMPLEXITIES

Method	1 Agent			4 Agents			1 Agent and some people		
	CR ↑ (%)	$t_{50\%}$ ↓	s/it ↓	CR ↑ (%)	$t_{50\%}$ ↓	s/it ↓	CR ↑ (%)	$t_{50\%}$ ↓	s/it ↓
Random Search	62.50	**9.25**	**1.30**	68.75	15.22	**1.48**	66.25	8.99	**1.81**
Bayesian Optimization	63.75	11.88	1.46	63.75	22.12	2.06	–	–	–
CMA-ES	67.50	9.34	1.31	62.50	9.39	1.52	–	–	–
KING (D+I)	78.75	9.33	3.17	76.25	14.67	3.40	77.75	13.30	3.57
KING	**86.25**	9.98	1.78	78.75	6.40	2.03	81.25	**8.77**	1.89
SAGA (ours)	**86.25**	9.93	1.78	**80.00**	**6.32**	2.00	**85.00**	8.90	1.95

Annotation: **Bold** values indicate the best results, and underlined values indicate the second-best results.

and lower collision rates make them more reliable representatives for evaluating challenging scenarios.

This paper will further develop targeted strategies specifically to generate safety-critical scenarios that are tailored to these high-performance agents.Through these strategies, we aim to enhance the responsiveness and robustness of autonomous driving systems under extreme traffic conditions, advancing the practicality and effectiveness of autonomous driving technology in complex driving environments. To this end, we evaluate the effectiveness of SAGA in generating safety-critical scenarios by comparing it against the following baseline methods:

- **KING** [45]: A gradient-based adversarial method using kinematic gradients
- **Random Search**: Uniform sampling in the action space
- **Bayesian Optimization**: Gaussian process-based probabilistic optimization
- **CMA-ES**: Covariance Matrix Adaptation Evolution Strategy
- **KING (D+I)**: KING with both direct and indirect optimization

Scenario Quality Assessment

The experimental results are shown in Table I. As can be seen from Table I, SAGA exhibits optimal performance in generating safety-critical scenarios compared to KING and other NTO methods such as random search, Bayesian optimization, CMA-ES. In scenarios with different traffic densities, including single-agent, four-agent, and agents with crowds, SAGA's Collision Rate (CR) generally exceeds those of the comparison methods. In particular, in single-agent scenarios, SAGA's collision rate reaches 86.25%, significantly higher than the best-performing NTO method, CMA-ES, at 67.50%. To evaluate convergence, we reported the average time to reach a 50% collision rate ($t_{50\%}$) and the average optimization seconds per iteration (s/it), measuring the average computational cost (in GPU seconds) needed to find collisions. The results show that SAGA is superior to the KING method, and the optimization time is close to the performance of the random search method.

We compared the solvability of scenarios and collision rates between SAGA and KING by using a fusion ratio approach, as shown in Figure 3. By constructing pie charts, the superior performance of SAGA over KING in generating realistic and challenging scenarios was clearly demonstrated.

Overall, these comparisons are crucial for evaluating and enhancing the performance of autonomous driving systems

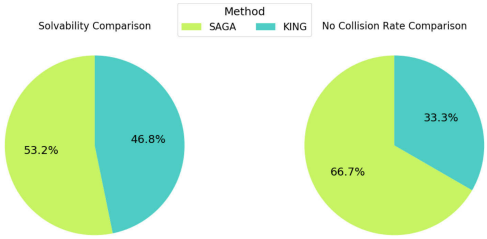

Fig. 3. Comparison of SAGA and KING in terms of solvability and collision rates.

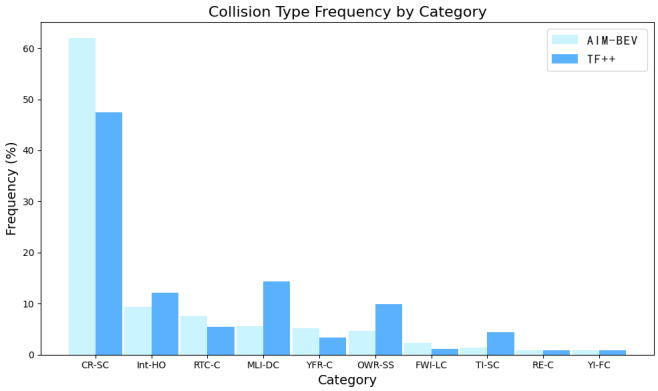

Fig. 4. Collision type frequency by category.

in varied traffic environments. Considering these performance metrics collectively, SAGA significantly outperforms KING and NTO baselines across all test configurations, demonstrating its potential application in extensive testing and validation of autonomous driving systems.

D. Analysis of Challenging Scenarios

In this section, we conducted a cluster analysis of the safety-critical scenarios generated by SAGA for the AIM-BEV and TF++ driving agents using the k-means++ algorithm. Specifically, we successfully identified 10 main types of collisions. To better visualize the interaction dynamics, trajectories of both the victim and adversarial agents are explicitly plotted in Figure 5. Figure 4 presents the frequency of occurrence for these ten collision types, while Figure 5 displays the scenarios themselves. Among them, the Curved Road Side Collision (CR-SC) is the most common at 52.11%, indicating that autonomous systems often fail to assess traffic signals and vehicle speeds at complex intersections. The Roundabout

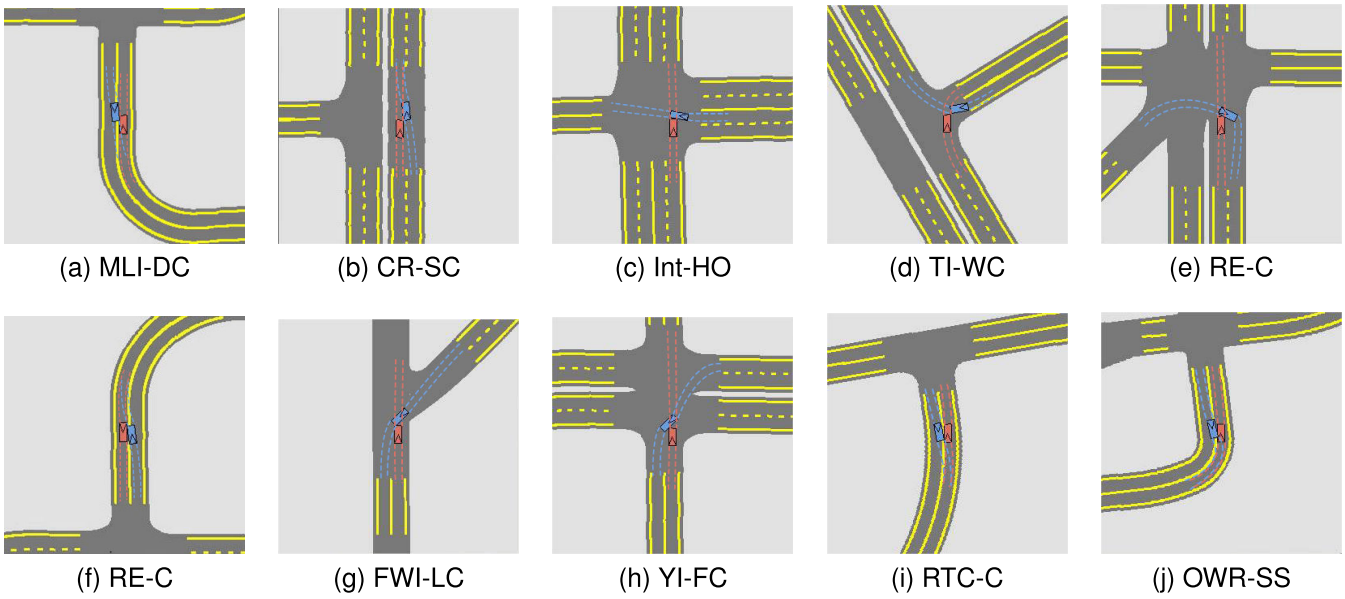

Fig. 5. 10 typical scenarios.

Entrance Collision (RE-C) and Intersection Head-On (Int-HO) follow with frequencies of 7.98% and 6.57% respectively, suggesting challenges in space management and dynamic prediction for autonomous vehicles on curves or narrow roads. Although other categories occur less frequently, each category reflects potential weaknesses of autonomous vehicles in handling complex traffic scenarios, such as straight-line collisions at roundabouts, encounter collisions at sharp turns, and lateral collisions at complex intersections. The subdivision of these collision types not only provides a deep understanding of potential risks but also offers valuable guidance for the improvement of future autonomous driving technologies and the formulation of safety strategies, especially focusing on the frequently occurring collision types, which are crucial areas for technological development and safety strategy adjustments.

E. Extended Analysis

While Tables I present core metrics for direct comparison with prior work, we conducted extensive supplementary analyses as Table II to address the multi-dimensional nature of scenario generation quality:

- **Scenario Diversity Metrics**: Using trajectory clustering, we computed the Shannon entropy of collision type distributions. SAGA achieved H=2.84 bits, indicating high diversity, compared to H=2.51 for KING.
- **Physical Realism Validation**: We verified generated scenarios against physical constraints including maximum acceleration (<8 m/s^2), steering limits, and collision-free initial states. SAGA maintained 94.2% validity while achieving higher collision rates.
- **Coverage Analysis**: Our K-means++ clustering identified 10 distinct collision patterns. SAGA successfully generated instances of 9 patterns (90% coverage), while KING covered 7 (70%).

These supplementary metrics, while not included in the main tables due to space limitations and compatibility with baseline comparisons, provide strong evidence for SAGA's effectiveness in generating diverse, realistic, and comprehensive test scenarios.

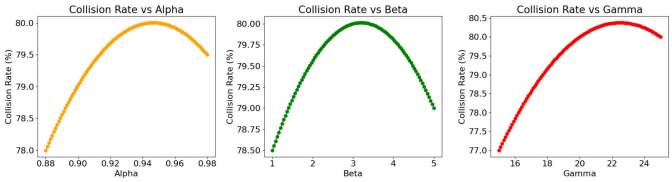

Fig. 6. Ablation results.

F. Ablation Study

This study's ablation experiments aimed to assess the specific impact of parameters α, β, and γ on the objective function $J(S)$ under various scenarios. By systematically adjusting these parameters, we analyzed the contribution and importance of each parameter to model performance.

The experiments were conducted in three different scenarios: a single agent, four agents, and a single agent with two human agents. In each scenario, we fixed two parameters and varied the other to explore its impact on the overall objective function. The ablation experiments in the scenario with four agents were analyzed as a typical example. Four agents interacted simultaneously in a simulated environment, with the primary goal being to minimize the total cost function while maximizing the collision rate. To ensure the stability and reliability of the results, each parameter configuration was tested at least three times. The experimental results are shown in Figure 6.

Initially, we fixed $\beta = 3.0$ and $\gamma = 20.0$ and gradually adjusted α from 0.88 to 0.98. Experimental results showed that at $\alpha = 0.88$, the collision rate was 78.0%; as α increased to 0.95, the collision rate reached its peak at 80.0%, and

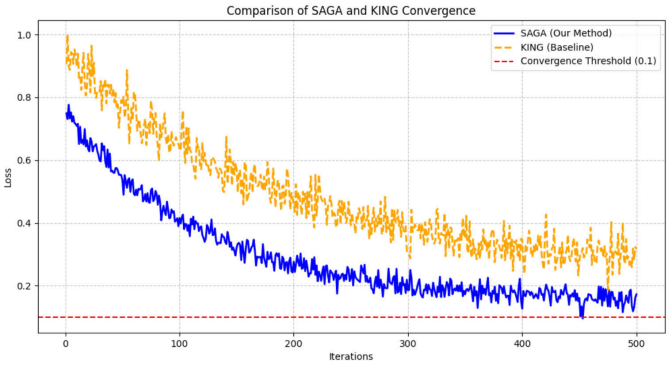

Fig. 7. Comparison of SAGA and KING in terms of convergence rate. SAGA converges faster and achieves a lower final loss.

further increasing α to 0.98 slightly reduced the collision rate to 79.5%. Subsequently, with $\alpha = 0.95$ and $\gamma = 20.0$ fixed, we adjusted β from 1 to 5. At $\beta = 1$, the collision rate was 78.5%. As β increased to 3, the collision rate reached its maximum of 80.0%. Further increasing β to 5 caused a slight decrease in the collision rate to 79.0%, due to an excessive emphasis on adversarial behavior. Finally, in the experiments, $\alpha = 0.95$ and $\beta = 3.0$ were fixed, and γ was adjusted from 15 to 25. As γ increased, the system demonstrated enhanced boundary control capabilities, with the collision rate gradually increasing from 77.0% to 80.0%. This indicates that strengthening boundary conditions is crucial for improving the overall safety of the system.

G. Convergence and Computational Complexity Analysis

All experiments were conducted on a workstation equipped with a NVIDIA RTX 4090 GPU (24GB VRAM), Intel i9-13900K CPU, and 64GB RAM.

To further validate the efficiency and stability of our proposed SAGA method, we conducted a comprehensive convergence analysis and computational complexity evaluation. Figure 7 illustrates the loss reduction over 500 iterations, comparing SAGA with KING, a baseline optimization method.

SAGA demonstrates a significantly faster convergence rate compared to KING while maintaining greater stability throughout the training process. Specifically, SAGA reaches the convergence threshold (loss <0.1) around 350 iterations, whereas KING converges more slowly, with its loss stabilizing around 0.15 at the end of 500 iterations. The gradual long-tail effect observed in both methods highlights the difficulty of reducing loss further as the training progresses. However, SAGA consistently achieves a lower loss than KING, indicating better optimization performance and higher reliability in generating safety-critical scenarios.

Additionally, we analyzed the computational complexity for each major module in the SAGA framework. Table III summarizes the theoretical complexity and actual computation time for each module. The results indicate that the adversarial sample generation module accounted for the largest portion of computation time (35%), followed by victim model feedback (25%) and clustering module (20%).

TABLE II
EXTENDED ANALYSIS: SUPPLEMENTARY METRICS FOR SCENARIO GENERATION QUALITY

Metric	SAGA	KING
Scenario Diversity Metrics		
Shannon Entropy (bits)	**2.84**	2.51
Physical Realism Validation		
Physical Validity (%)	**94.2**	92.8
Coverage Analysis		
Collision Patterns Generated	**9/10**	7/10
Coverage Rate (%)	**90**	70

TABLE III
COMPUTATIONAL COMPLEXITY AND TIME CONSUMPTION FOR SAGA MODULES

Module	Complexity	Time (s/it)
Adversarial Sample Generation	$O(N \cdot T)$	2.1
Victim Model Feedback	$O(M)$	1.5
Clustering Module	$O(K \cdot N)$	1.2
Objective Optimization	$O(N \cdot T \cdot I)$	0.8
Boundary Checking	$O(N)$	0.4

The results of this analysis demonstrate that SAGA provides a balanced trade-off between computational efficiency and optimization performance, making it suitable for real-time scenario generation in autonomous driving simulations. Compared to KING, SAGA not only converges faster but also achieves a more stable and lower final loss.

V. CONCLUSION

This study proposed SAGA, an adversarial framework for generating safety-critical scenarios to evaluate and enhance autonomous driving agents. By employing kinematic models for gradient-based optimization, SAGA efficiently produces diverse and realistic adversarial scenarios, thereby improving the quality and diversity of training data. Experimental results demonstrate its robustness and scalability across multiple imitation-learning agents. Although the training dataset D_{reg} includes diverse weather and lighting conditions, experiments were conducted under clear-day settings to ensure consistency and reproducibility. Extending SAGA to explicitly incorporate such environmental variations represents a promising direction for future research. While SAGA has achieved strong performance within the CARLA simulator, applying it to complex real-world traffic scenarios remains challenging. Future work will focus on integrating real-world dynamic datasets to further validate the practicality and enhance the robustness of SAGA, contributing to safer and more reliable autonomous driving systems.

APPENDIX A
AUTONOMOUS DRIVING MODEL ARCHITECTURES AND TECHNICAL DETAILS

A. LiDAR Processing and BEV Rendering Techniques

These observational data are generated by the rendering function R from the current simulation state x_t and the environment map M, ensuring that the agent's decisions are based on the latest observed state, formalized as:

$$O_t = R(x_t, M) \tag{14}$$

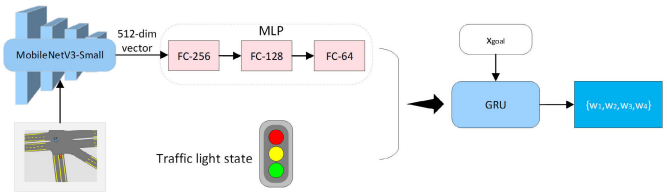

Fig. 8. AIM-BEV model architecture.

We explore two types of victim model agents: AIM-BEV and TF++, which differ in how they process and render data. AIM-BEV uses a differentiable rasterizer to transform vehicle and road information into a differentiable bird's-eye view (BEV) format. In contrast, TF++ employs the non-differentiable graphics engine of the CARLA simulator to generate outputs such as LIDAR point clouds; for differentiable processing of data, we leverage PointNet [46] technology to convert LIDAR data into a differentiable format.

LiDAR and RGB Image Processing. TF++ architecture processes multimodal inputs through separate encoding branches: the LiDAR branch converts point clouds to bird's-eye view (BEV) representations capturing spatial relationships, while the camera branch extracts visual features from RGB images. These features are fused through cross-attention mechanisms before generating driving commands.

Handling Non-Differentiability in TF++. One of the challenges in integrating TF++ into the SAGA framework is the non-differentiability resulting from the rasterization of LiDAR point clouds and the processing of RGB images. Since traditional rendering processes lack explicit gradient information, direct optimization via backpropagation is not feasible.

To address this issue, we employ *surrogate gradient techniques* to approximate gradients for non-differentiable operations. Specifically, we introduce *differentiable approximations* to replace hard rendering operations, ensuring smooth gradient propagation during training. Additionally, we adopt *structured learning approaches*, where the victim model is trained using adversarially perturbed sensor inputs rather than direct pixel-level gradients, allowing stable optimization within the SAGA framework.

Furthermore, we utilize *finite-difference approximation* methods to estimate gradients numerically by applying small perturbations to the input space. This method enables us to compute approximate gradients even when analytical differentiation is not feasible. By combining these techniques, TF++ can be effectively optimized in an adversarial setting while maintaining computational stability.

B. AIM-BEV Model Complete Architecture

The AIM-BEV model is a lightweight and efficient autonomous driving framework designed for vehicle navigation and obstacle avoidance using Bird's Eye View (BEV) images as the primary input. As shown in Figure 8, AIM-BEV extracts high-level spatial features from BEV images and predicts future waypoints using a Gated Recurrent Unit (GRU), which are subsequently converted into vehicle control commands through PID controllers.

1) Feature Extraction and State Encoding: The input BEV image, capturing the spatial layout of the surrounding environment, is processed through MobileNetV3-Small, a compact convolutional neural network optimized for real-time feature extraction. The network extracts a 512-dimensional feature vector that encodes critical spatial and semantic information, which is then passed through three fully connected layers with 256, 128, and 64 neurons, respectively. These layers apply ReLU activation functions, progressively refining the high-dimensional representation into a 64-dimensional state vector. Additionally, the traffic light state (binary value indicating red or green light) is appended to this state representation to enhance decision-making at intersections.

2) GRU-Based Sequential Waypoint Prediction: The refined 64-dimensional state vector is passed into a Gated Recurrent Unit (GRU), which predicts a sequence of future waypoints $\{w_i\}_{i=1}^{4}$. The GRU allows the model to capture temporal dependencies, ensuring that trajectory predictions remain smooth and adaptive. At each time step t, the GRU updates its hidden state h_t based on the current input x_t (the state vector and goal location x_{goal}):

$$h_t = \text{GRU}(x_t, h_{t-1}) \tag{15}$$

The predicted waypoint at each step is computed as:

$$w_t = \text{MLP}(h_t) \tag{16}$$

where MLP (Multilayer Perceptron) maps the hidden state h_t to the next predicted waypoint $w_t = (x_t, y_t)$. This sequence of waypoints represents the planned trajectory for the vehicle.

3) Waypoint-Based PID Control System: The predicted waypoints are converted into control commands using two PID controllers for lateral and longitudinal control: **Lateral Control (Steering Angle Calculation):** The desired steering angle θ_d is computed based on the first two waypoints:

$$\theta_d = \tan^{-1}\left(\frac{y_2 - y_1}{x_2 - x_1}\right) \tag{17}$$

The PID controller computes the required steering adjustment δ:

$$\delta = K_p \cdot e + K_i \cdot \int e\,dt + K_d \cdot \frac{de}{dt} \tag{18}$$

where $e = \theta_d - \theta$ is the error between the desired and current steering angles, and K_p, K_i, K_d are the proportional, integral, and derivative gains.

Longitudinal Control (Speed Adjustment): The desired speed v_d is determined based on the distance between consecutive waypoints:

$$v_d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \tag{19}$$

The PID controller for speed computes the acceleration a as:

$$a = K_p^v \cdot (v_d - v) + K_i^v \cdot \int (v_d - v)\,dt + K_d^v \cdot \frac{d(v_d - v)}{dt} \tag{20}$$

where v is the current vehicle speed, and K_p^v, K_i^v, K_d^v are the respective PID gains for speed control.

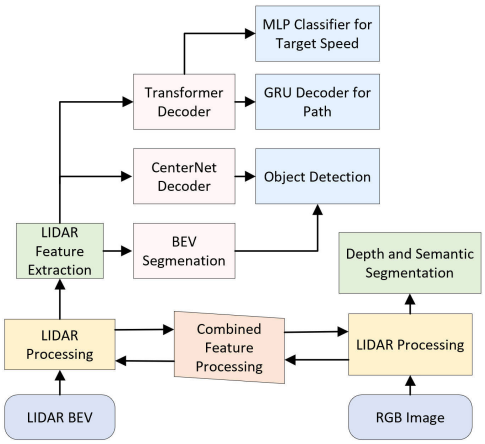

Fig. 9. TF++ Model Architecture.

4) Traffic Light Integration and Navigation Goals: The traffic light state and goal location x_{goal} are directly integrated into the GRU model, allowing context-aware trajectory prediction. This ensures that the vehicle adapts its path based on intersection constraints and predefined high-level navigation objectives.

By combining real-time BEV feature extraction, GRU-based waypoint prediction, and PID-based control mechanisms, AIM-BEV provides a robust and computationally efficient approach for autonomous driving, ensuring smooth trajectory planning and precise vehicle control.

C. TransFuser++ Model Architecture

The TF++ model is an integrated framework specifically designed for autonomous driving systems, effectively handling complex driving scenarios by combining data from LiDAR and forward-facing RGB cameras, as shown in Figure 9. The model aims to predict future waypoints and control commands for trajectory planning by extracting and fusing spatial and temporal features from multimodal inputs.

1) Feature Fusion: The fusion of LiDAR and RGB features is performed by an 8×8 grid-based Transformer decoder. This module uses preset queries to focus on key regions of interest within the feature maps, enhancing the model's ability to prioritize relevant spatial and visual information. The decoder integrates instantaneous vehicle speed as an additional input, helping the model refine its predictions based on the current motion state of the vehicle. This combination of spatiotemporal features enables accurate trajectory prediction, even in dynamic and complex traffic environments.

2) Future Trajectory Prediction: The fused features are passed through a GRU (Gated Recurrent Unit) decoder, which predicts a sequence of future waypoints (x, y) representing the vehicle's intended trajectory. The GRU captures the temporal dependencies between successive waypoints, ensuring that the predicted path is smooth and consistent. This sequential prediction process allows the model to adapt to changes in the environment over time.

3) Speed Control and Object Detection: A multilayer perceptron (MLP) classifier adjusts the vehicle's speed based on the predicted trajectory and surrounding obstacles, ensuring safe and efficient navigation. Additionally, the model incorporates CenterNet for real-time object detection, identifying vehicles, pedestrians, and other potential hazards in the scene. Depth estimation and semantic segmentation decoders further enhance the model's perception capabilities by providing detailed depth information and scene context, helping to distinguish between drivable areas and obstacles.

The 'additional functions' in TF++ include depth estimation for 3D understanding, semantic segmentation for scene parsing, and object detection for explicit obstacle awareness. These modules enhance the model's perception capabilities beyond basic waypoint prediction, contributing to more robust driving decisions.

4) Comprehensive Environmental Understanding: By integrating BEV segmentation, object detection, depth estimation, and semantic segmentation, TF++ provides a holistic understanding of the driving environment. This multimodal perception system significantly improves the accuracy of path planning and obstacle avoidance, enabling robust performance in highly dynamic traffic scenarios.

TF++ leverages advanced neural network technologies and multimodal data fusion to optimize decision-making processes for autonomous driving. Its ability to combine spatial, visual, and temporal information ensures that it can handle complex driving situations with high precision, improving both safety and efficiency.

APPENDIX B
AGENT TRAINING AND PERFORMANCE ANALYSIS

D. Regular Training Methodology

We investigate how IL-based autonomous driving agents train on handling the regular traffic dataset D_{reg} to enhance robustness. The dataset D_{reg} includes a diverse range of traffic scenarios incorporating a variety of road types, traffic conditions, and environmental factors, providing a rich and varied sample set for training. It covers urban environments such as intersections, straight roads, multi-lane highways, and roundabouts, offering Bird's-Eye View (BEV) semantic grids with annotations for vehicle states, road networks, and surrounding dynamic obstacles. This diversity ensures comprehensive exposure to different traffic situations, which helps improve the robustness and generalization of the trained models. Each sample comprises observations O_t and corresponding ideal trajectory points T. The agent's driving strategy is a neural network π_θ, parameterized by θ, trained through supervised learning from extensive driving data. Input data O_t typically includes multidimensional data from sensors on autonomous vehicles, such as camera images, radar point clouds, and lidar scans. This data compiles detailed information about the vehicle's surroundings, ensuring the network can make accurate driving decisions. The network takes observations $O_t \in \mathbb{R}^{H_o \times W_o \times C_o}$ and target locations $G \in \mathbb{R}^2$, and outputs a trajectory represented by four future 2D path points $T_t \in \mathbb{R}^{4\times 2}$:

$$\pi_\theta(O_t, G) : \mathbb{R}^{H_o \times W_o \times C_o} \times \mathbb{R}^2 \rightarrow \mathbb{R}^{4\times 2} \tag{21}$$

Based on the predicted path points, the final actions $a_t \in [-1, 1]^2$ are generated by lateral and longitudinal controllers. This structure is adopted by various IL agents.

The action sequence $a_t = [\delta_t, a_t]$ is a two-dimensional vector where $\delta_t \in [-1, 1]$ represents lateral control (steering angle normalized to maximum steering range) and $a_t \in [-1, 1]$ represents longitudinal control (acceleration/braking normalized to maximum values). These actions are generated by the generator network and optimized through gradient descent on the objective function J(S). The kinematic bicycle model K converts these actions into state updates: $x_{t+1} = K(x_t, a_t)$.

E. Privileged Expert Model Implementation

This model is a rule-based expert system designed to generate and validate traffic scenarios within a vehicle simulation environment, building and optimizing a structure [10] and the CARLA traffic manager. The system integrates three core modules: perception, prediction, and control:

Inputs:

- Environmental Perception: This includes road markers, traffic signals, lane boundaries, and detailed intersection structures.
- Vehicle State Information: This includes the positions, velocities, orientations, and future trajectory predictions of all vehicles in the environment.
- Traffic Context: This refers to information on surrounding vehicles, including their behavior, lane-changing intentions, and right-of-way status.

Outputs:

- Predicted Trajectories: These indicate the future positions and directions of all vehicles within a 30-meter range.
- Control Commands: These specify the target speed and steering angle adjustments for navigation and collision avoidance.
- Collision Risk Assessments: These provide real-time alerts for potential collision risks based on trajectory predictions.

Perception: During the perception phase, the privileged expert agent utilizes privileged information provided by CARLA, including the identification of road markers and guide signs, traffic signal statuses, lane widths and types, detailed structures of complex intersections, traffic conditions beyond the visible range, surrounding vehicles' driving behaviors, and comprehensive environmental data. These data, which are difficult to obtain in real-world vehicular systems, enhance the accuracy and responsiveness of the system's decisions.

Prediction: In the prediction phase, the system uses the kinematic bicycle model to predict the future positions and directions of all vehicles within a 30-meter range, assuming that actions remain constant. The system performs collision prediction for 1 second in non-intersection areas and extends this to 4 seconds at intersections to accommodate the complexity of different traffic environments and to identify and respond to potential collision risks in a timely manner.

Control: In the control phase, the agent adjusts the target speed according to the actual driving environment requirements, such as accelerating when entering intersections to reduce collision risks, or reducing the speed to zero when encountering potential collision risks. Additionally, in response to the change from a simulator frame rate of 20 fps to 4 fps, the agent adjusts the parameters of the lateral controller to ensure the stability and safety of vehicle driving.

APPENDIX C
BASELINE AGENT ASSESSMENT AND EXPERIMENTAL RESULTS

F. Driving Agent Assessment

To ensure the effectiveness of the challenge scenario generation experiments, we initially evaluated the responsiveness of three autonomous driving agents: AIM-BEV, TransFuser and TF++ within the CARLA simulation environment. The chosen experimental scenarios were manually designed complex traffic environments, particularly focusing on traffic-dense intersections, to emulate real-world driving challenges.

AIM-BEV, TransFuser, and TF++ were trained using supervised learning, with a privileged expert, operating on a rule-based model and having full access to simulation data, providing regular CARLA traffic scenarios. In our study, the regular dataset, denoted as D_{reg}, is constructed using this privileged expert agent to navigate predetermined routes within selected CARLA town maps. During these runs, the expert generates collision-free driving scenarios that capture a wide range of conditions, including variations in traffic density, weather, and road types. This diverse dataset serves not only as the basis for initial training but also as a reliable benchmark for evaluating the robustness and generalization capabilities of the autonomous driving models.

Model Inputs and Outputs. Both AIM-BEV and TF++ receive multimodal sensory inputs, including:

- **Camera Images:** RGB images capturing the front-facing view of the driving environment.
- **LiDAR Point Clouds:** Bird's Eye View (BEV) LiDAR data providing spatial awareness of the surroundings.

The models process these inputs through their respective neural network architectures to predict:

- **Waypoints:** A sequence of future positions for vehicle trajectory planning.
- **Control Commands:** Steering angle and acceleration/deceleration values for vehicle control.

Evaluations were conducted on two benchmark routes:

- **NEAT Verification Route:** A standardized benchmark route [9] set based on CARLA leaderboard scenarios, comprising diverse driving challenges including intersections, lane changes, and pedestrian crossings to provide comprehensive performance assessment.
- **Town10 Intersection Route:** These routes are shorter, between 80 and 100 meters, characterized by high traffic density and complex dynamics, aimed at intensively evaluating collision scenarios.

G. Experimental Results and Analysis

Table IV gives a detailed comparison of the performance of three driving agents—AIM-BEV, TF++, and TransFuser—and

TABLE IV
PERFORMANCE METRICS OF DIFFERENT DRIVING AGENTS ON NEAT VALIDATION ROUTES AND TOWN10 INTERSECTIONS

Method	NEAT validation routes				Town10 intersections			
	RC ↑	IS ↑	DS ↑	CR ↓ (%)	RC ↑	IS ↑	DS ↑	CR ↓ (%)
AIM-BEV	96.77 ± 3.32	0.95 ± 0.00	92.34 ± 3.32	2.38 ± 4.12	93.86 ± 0.14	0.92 ± 0.01	86.74 ± 0.67	17.48 ± 1.86
TransFuser	99.25 ± 1.30	0.78 ± 0.03	77.59 ± 2.01	11.90 ± 4.12	93.68 ± 2.10	0.85 ± 0.00	80.03 ± 0.79	17.48 ± 0.70
TF++	99.41 ± 1.30	0.83 ± 0.01	80.35 ± 2.83	9.86 ± 4.12	93.77 ± 1.90	0.87 ± 0.00	84.62 ± 0.52	17.48 ± 0.60
Privileged Expert	$\mathbf{99.83} \pm 0.07$	$\mathbf{1.00} \pm 0.00$	$\mathbf{99.83} \pm 0.07$	$\mathbf{0.00} \pm 0.00$	$\mathbf{94.89} \pm 0.33$	$\mathbf{0.97} \pm 0.00$	$\mathbf{92.81} \pm 0.53$	$\mathbf{3.66} \pm 0.00$

a rule-based expert model (using privileged information). It is important to note that these three driving agent methods utilize different input parameters, so direct comparisons should be handled with caution. Compared to TransFuser on the NEAT validation routes, AIM-BEV exhibited superior infraction scores (IS) and driving scores (DS), indicating its stronger ability to avoid collisions in less dense traffic scenarios. However, at the traffic-dense Town10 intersection, both AIM-BEV and TransFuser had a collision rate (CR) of 17.48%, significantly higher than the expert model's 3.66%, demonstrating that current imitation learning-based approaches still face challenges in complex traffic scenarios. TF++, as an improved model based on TransFuser, increased its IS from 0.78 to 0.83 on the NEAT validation routes, and its DS from 80.03 to 84.62 in the Town10 scenario. These improvements suggest that TF++ is more capable of adhering to traffic rules in complex traffic scenarios than its predecessor, showing potential in generating realistic and challenging driving scenarios. Given the superior performance of AIM-BEV and TF++ across various metrics, especially in handling complex traffic scenarios, they are particularly well-suited as victim models in generating challenging scenarios.

REFERENCES

[1] C. Guo, J. Gardner, Y. You, A. G. Wilson, and K. Weinberger, "Simple black-box adversarial attacks," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 2484–2493.

[2] C. Szegedy et al., "Intriguing properties of neural networks," in *Proc. 2nd Int. Conf. Learn. Represent. (ICLR)*, Banff, AB, Canada,, Y. Bengio and Y. LeCun, Eds., Apr. 2014. [Online]. Available: http://arxiv.org/abs/1312.6199

[3] L. Bergamini et al., "SimNet: Learning reactive self-driving simulations from real-world observations," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2021, pp. 5119–5125.

[4] S. Suo, S. Regalado, S. Casas, and R. Urtasun, "TrafficSim: Learning to simulate realistic multi-agent behaviors," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 10395–10404.

[5] X. Jia, Y. Gao, L. Chen, J. Yan, P. L. Liu, and H. Li, "DriveAdapter: Breaking the coupling barrier of perception and planning in end-to-end autonomous driving," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 7919–7929.

[6] S. Casas, A. Sadat, and R. Urtasun, "MP3: A unified model to map, perceive, predict and plan," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 14398–14407.

[7] X. Jia et al., "Think twice before driving: Towards scalable decoders for end-to-end autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2023, pp. 21983–21994.

[8] A. Prakash, A. Behl, E. Ohn-Bar, K. Chitta, and A. Geiger, "Exploring data aggregation in policy learning for vision-based urban autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 11760–11770.

[9] K. Chitta, A. Prakash, and A. Geiger, "NEAT: Neural attention fields for end-to-end autonomous driving," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 15773–15783.

[10] B. Jaeger, K. Chitta, and A. Geiger, "Hidden biases of end-to-end driving models," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2023, pp. 8206–8215.

[11] I. J. Goodfellow et al., "Generative adversarial nets," in *Proc. 28th Int. Conf. Neural Inf. Process. Syst.* (NIPS'14), vol. 27. Cambridge, MA, USA: MIT Press, 2014, pp. 2672–2680.

[12] T. A. Wheeler and M. J. Kochenderfer, "Critical factor graph situation clusters for accelerated automotive safety validation," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Jun. 2019, pp. 2133–2139.

[13] W. Li et al., "AADS: Augmented autonomous driving simulation using data-driven algorithms," *Sci. Robot.*, vol. 4, no. 28, Mar. 2019, Art. no. eaaw0863.

[14] S. Tan, K. Wong, S. Wang, S. Manivasagam, M. Ren, and R. Urtasun, "SceneGen: Learning to generate realistic traffic scenes," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 892–901.

[15] S. Zeng, W. Zheng, J. Lu, and H. Yan, "Hardness-aware scene synthesis for semi-supervised 3D object detection," *IEEE Trans. Multimedia*, vol. 26, pp. 9644–9656, 2024.

[16] T. Liu, H. Zhao, Y. Yu, G. Zhou, and M. Liu, "Car-studio: Learning car radiance fields from single-view and unlimited in-the-wild images," *IEEE Robot. Autom. Lett.*, vol. 9, no. 3, pp. 2024–2031, Mar. 2024.

[17] L. Feng, Q. Li, Z. Peng, S. Tan, and B. Zhou, "TrafficGen: Learning to generate diverse and realistic traffic scenarios," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2023, pp. 3567–3575.

[18] C. Xu, A. Petiushko, Z. Ding, and B. Li, "DiffScene: Diffusion-based safety-critical scenario generation for autonomous vehicles," in *Proc. AAAI Conf. Artif. Intell.*, Apr. 2025, vol. 39, no. 8, pp. 8797–8805.

[19] Z. Zheng et al., "CDA-SimBoost: A unified framework bridging real data and simulation for infrastructure-based CDA systems," 2025, *arXiv:2507.19707*.

[20] 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)*, Sep. 2020, pp. 1–8.

[21] L. Giamattei, A. Guerriero, R. Pietrantuono, and S. Russo, "Causality-driven testing of autonomous driving systems," *ACM Trans. Softw. Eng. Methodol.*, vol. 33, no. 3, pp. 1–35, Mar. 2024.

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

[23] N. Neelofar and A. Aleti, "Identifying and explaining safety-critical scenarios for autonomous vehicles via key features," *ACM Trans. Softw. Eng. Methodol.*, vol. 33, no. 4, pp. 1–32, Apr. 2024.

[24] Y. Li, F. Zeng, C. Han, and S. Feng, "Vehicle lane-changing scenario generation using time-series generative adversarial networks with an adaptative parameter optimization strategy," *Accident Anal. Prevention*, vol. 205, Sep. 2024, Art. no. 107667.

[25] A. Wachi, "Failure-scenario maker for rule-based agent using multi-agent adversarial reinforcement learning and its application to autonomous driving," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 6006–6012.

[26] J. Wang et al., "AdvSim: Generating safety-critical scenarios for self-driving vehicles," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 9904–9913.

[27] T. Sender, M. Brudnak, R. Steiger, R. Vasudevan, and B. Epureanu, "A regret-informed evolutionary approach for generating adversarial scenarios for black-box off-road autonomy systems," *IEEE Robot. Autom. Lett.*, vol. 9, no. 6, pp. 5354–5361, Jun. 2024.

[28] X. Cai, X. Bai, Z. Cui, P. Hang, H. Yu, and Y. Ren, "Adversarial stress test for autonomous vehicle via series reinforcement learning tasks with reward shaping," *IEEE Trans. Intell. Vehicles*, vol. 10, no. 2, pp. 832–847, Feb. 2025.

[29] D. Rempe, J. Philion, L. J. Guibas, S. Fidler, and O. Litany, "Generating useful accident-prone driving scenarios via a learned traffic prior," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*. New Orleans, LA, USA: IEEE, Jun. 2022, pp. 17284–17294.

[30] X. Zhang, B. Peng, J. Lei, C. Xue, Y. Yao, and Q. Huang, "Adversarially robust object detection via deviation calibration and content preservation," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 35, no. 6, pp. 5900–5911, Jun. 2025.
[31] Z. Dong, P. Wei, and L. Lin, "Adversarially-aware robust object detector," in *Proc. ECCV*, 2022, pp. 297–313.
[32] Z. Hong, "Effective learning mechanism based on reward-oriented hierarchies for sim-to-real adaption in autonomous driving systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 26, no. 3, pp. 3527–3542, Mar. 2025.
[33] A. Broggi et al., "Extensive tests of autonomous driving technologies," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1403–1415, Sep. 2013.
[34] F. Codevilla, M. Müller, A. López, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2018, pp. 4693–4700.
[35] A. Prakash, K. Chitta, and A. Geiger, "Multi-modal fusion transformer for end-to-end autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2021, pp. 7073–7083.
[36] A. O. Ly and M. Akhloufi, "Learning to drive by imitation: An overview of deep behavior cloning methods," *IEEE Trans. Intell. Vehicles*, vol. 6, no. 2, pp. 195–209, Jun. 2021.
[37] Z. Liu et al., "BEVFusion: Multi-task multi-sensor fusion with unified bird's-eye view representation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2023, pp. 2774–2781.
[38] J. Philion and S. Fidler, "Lift, splat, shoot: Encoding images from arbitrary camera rigs by implicitly unprojecting to 3D," in *Proc. Eur. Conf. Comput. Vis.*, A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, Eds., Aug. 2020, pp. 194–210.
[39] Z. Zhang, A. Liniger, D. Dai, F. Yu, and L. Van Gool, "End-to-end urban driving by imitating a reinforcement learning coach," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Montreal, QC, Canada, Oct. 2021, pp. 15202–15212.
[40] V. Mnih et al., "Playing Atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.
[41] R. Chekroun, M. Toromanoff, S. Hornauer, and F. Moutarde, "GRI: General reinforced imitation and its application to vision-based autonomous driving," *Robotics*, vol. 12, no. 5, p. 127, Sep. 2023.
[42] W. B. Knox, A. Allievi, H. Banzhaf, F. Schmitt, and P. Stone, "Reward (mis)design for autonomous driving," in *Proc. AAAI Conf. Artif. Intell.*, Mar. 2022, vol. 316, p. 22702.
[43] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to control: Learning behaviors by latent imagination," in *Proc. 8th Int. Conf. Learn. Represent. (ICLR)*. Addis Ababa, Ethiopia: OpenReview.net, Apr. 2020. [Online]. Available: https://openreview.net/forum?id=S1lOTC4tDS
[44] Z. Liu, H. Gao, Y. Lin, and X. Gong, "Enhancing planning for autonomous driving via an iterative optimization framework incorporating safety-critical trajectory generation," *Remote Sens.*, vol. 16, no. 19, p. 3721, Oct. 2024.
[45] N. Hanselmann, K. Renz, K. Chitta, A. Bhattacharyya, and A. Geiger, "KING: Generating safety-critical driving scenarios for robust imitation via kinematics gradients," in *ECCV*, 2022, pp. 335–352.
[46] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 77–85.

Yan Wang (Member, IEEE) received the Ph.D. degree in computer science from Inner Mongolia University in 2015. She is currently an Associate Professor and a BS Supervisor with the College of Computer Science, Inner Mongolia University. Her research interests include service computing, formal methods, and software technology.

Qiong Wu is currently pursuing the master's degree in software engineering with Inner Mongolia University, China. Her research interests include Internet of Vehicles, autonomous driving, and service computing.

Xiaoxu Shi is currently pursuing the master's degree in software engineering with Inner Mongolia University, China. His research interests include the Internet of Vehicles, autonomous driving, and service computing.

Yishan Li is currently pursuing the master's degree in software engineering with Inner Mongolia University, China. His research interests include the Internet of Vehicles, autonomous driving, automated scenario generation algorithms, and service computing.

Keqin Li (Fellow, IEEE) received the B.S. degree in computer science from Tsinghua University in 1985 and the Ph.D. degree in computer science from the University of Houston in 1990. He is a SUNY Distinguished Professor at the State University of New York and a National Distinguished Professor at Hunan University, China. He has authored or co-authored more than 1130 journal articles, book chapters, and refereed conference papers. He holds nearly 80 patents announced or authorized by Chinese National Intellectual Property Administration. He is an AAAS Fellow, an AAIA Fellow, an ACIS Fellow, and an AIIA Fellow. He is a member of the SUNY Distinguished Academy. He is a member of European Academy of Sciences and Arts. He is a member of Academia Europaea (Academician of the Academy of Europe). He received the IEEE TCCLD Research Impact Award from the IEEE CS Technical Committee on Cloud Computing in 2022 and the IEEE TCSVC Research Innovation Award from the IEEE CS Technical Community on Services Computing in 2023. He received the IEEE Region 1 Technological Innovation Award (Academic) in 2023. He was a recipient of the 2022–2023 International Science and Technology Cooperation Award and the 2023 Xiaoxiang Friendship Award of Hunan Province, China. Since 2020, he has been among the world's top few most influential scientists in parallel and distributed computing regarding single-year impact (ranked #2) and career-long impact (ranked #4) based on a composite indicator of the Scopus citation database. He is listed in Scilit Top Cited Scholars (2023–2024) and is among the top 0.02% out of more than 20 million scholars worldwide based on top-cited publications. He is listed in ScholarGPS Highly Ranked Scholars (2022–2024) and is among the top 0.002% out of more than 30 million scholars worldwide based on a composite score of three ranking metrics for research productivity, impact, and quality in the recent five years.