DiffScene: Diffusion-Based Safety-Critical Scenario Generation for Autonomous Vehicles

Chejian Xu [1] **Ding Zhao** [2] **Alberto Sangiovanni-Vincentelli** [3] **Bo Li** [1]

Abstract

The field of Autonomous Driving (AD) has witnessed significant progress in recent years. Among the various challenges faced, the safety evaluation of autonomous vehicles (AVs) stands out as a critical concern. Traditional evaluation methods are both costly and inefficient, often requiring extensive driving mileage in order to encounter rare safety-critical scenarios, which are distributed on the long tail of the complex real-world driving landscape. In this paper, we propose a unified approach, Diffusion-Based Safety-Critical Scenario Generation (`DiffScene`), to generate high-quality safety-critical scenarios which are both realistic and safety-critical for efficient AV evaluation. In particular, we propose a diffusion-based generation framework, leveraging the power of approximating the distribution of low-density spaces for diffusion models. We design several adversarial optimization objectives to guide the diffusion generation under predefined adversarial budgets. These objectives, such as *safety-based objective*, *functionality-based objective*, and *constraint-based objective*, ensure the generation of safety-critical scenarios while adhering to specific constraints. Extensive experimentation has been conducted to validate the efficacy of our approach. Compared with 6 SOTA baselines, `DiffScene` generates scenarios that are (1) more safety-critical under 3 metrics, (2) more realistic under 5 distance functions, and (3) more transferable to different AV algorithms. In addition, we demonstrate that training AV algorithms with scenarios generated by `DiffScene` leads to significantly higher performance in terms of the safety-critical metrics compared to baselines. These findings highlight the potential of `DiffScene` in addressing the challenges of AV safety evaluation, paving the way for more efficient and effective AV development.

1. Introduction

Innovations driven by recent progress in machine learning (ML) have demonstrated human-competitive performance in various fields (Silver et al., 2018; He et al., 2015; Agostinelli et al., 2019). However, the safety evaluation and guarantees of these ML-based models are still challenging, especially in real-world safety-critical applications such as AV.

To evaluate the safety and robustness of AV systems, the prevailing approaches deploy them in the real world and test them with various traffic scenarios. AV companies also reconstruct safety-critical scenarios collected during their on-road testing in the simulators (Webb et al., 2020) to test. Deviation theories such as importance sampling (IS) and cross-entropy (CE) have been introduced to measure the risk of AVs (Zhao, 2016; O'Kelly et al., 2018; Bucklew & Bucklew, 2004). However, due to the high dimensionality, complexity, and rareness of safety-critical driving scenarios in the real world, it is very challenging and inefficient to test AV safety (CDMV, 2022; Arief et al., 2020).

Recently, with the successes of deep generative models, a promising way is to directly generate such safety-critical scenarios rather than sampling from real-world data (Yang et al., 2020; Chen et al., 2021b; Ehrhardt et al., 2020). The advantages of the generation approaches include improved evaluation efficiency and scenario diversity (Ding et al., 2020b). For example, RELATE (Ehrhardt et al., 2020) use a GAN framework to generate realistic traffic videos with multi-object scene synthesis. STRIVE (Rempe et al., 2022) generates adversarial trajectory by optimizing the latent space of a VAE model. However, most methods focus on only modeling the existing data distribution or applying scenario-specific rules. They fail to generate *controllable* rare events such as safety-critical scenarios *efficiently*.

In this work, to solve these challenges, we propose a diffusion-enabled generation framework `DiffScene`, which is able to generate safety-critical scenarios effectively while preserving its realism, satisfying real-world physical constraints, and can be used to further evaluate and improve the safety and robustness of various AV algorithms. Specifically, we first leverage the powerful diffusion model to capture the low-density spaces in the distribution to generate realistic safety-critical scenarios efficiently. Then we propose a guided adversarial optimization process to modify the generation results. During each diffusion step, we optimize and constrain the generated scenarios using 3 differ-

[1]University of Illinois at Urbana-Champaign [2]Carnegie Mellon University [3]University of California Berkeley. Correspondence to: Chejian Xu <chejian2@illinois.edu>.

2^{nd} *AdvML Frontiers workshop at* 40^{th} *International Conference on Machine Learning*, Honolulu, Hawaii, USA. PMLR 202, 2023.

ent objectives: *safety-based objective*, *functionality-based objective*, and *constraint-based objective*. Extensive experiments on different scenario settings and AV algorithms show that DiffScene is able to generate scenarios that are more safety-critical, realistic, and transferable than baselines. We also demonstrate that DiffScene scenarios achieve higher downstream utility: training AV algorithms with the generated scenarios leads to significantly higher performance in terms of the safety-critical metrics compared to baselines.

Our contributions are summarized as follows: 1) We propose DiffScene, a unified safety-critical scenario generation framework that leverages diffusion models to generate realistic safety-critical traffic scenarios by introducing diverse safety-critical objectives. 2) We propose three different safety-critical objectives, focusing on safety, functionality, and (safe) constraints, respectively, to ensure the effectiveness and naturalness of the generated scenarios. 3) We conduct extensive experiments using Carla under different traffic settings (e.g., different routes and maps) with 3 different reinforcement learning-based (RL) AV algorithms. We show that DiffScene scenarios achieve higher risk scores (i.e., more safety-critical) in terms of 3 safety-critical metrics and smaller distances to benign data distributions (i.e., more realistic) in terms of 5 distance functions compared to existing safety-critical scenario generation algorithms. 4) We also provide comprehensive evaluations under diverse settings to show that existing RL-based AV algorithms are vulnerable to DiffScene scenarios. AV algorithms trained with DiffScene scenarios achieve significantly higher performance in terms of the safety-critical metrics, demonstrating the potential utilities of DiffScene.

2. Related Work

Deep Generative Models. Different generative models have been proposed to advance the ML development. VAE (Kingma & Welling, 2013) is a popular generative model based on autoencoder, which maximizes the variational lower bound of the training samples. GAN (Goodfellow et al., 2020) adopts a generator-discriminator framework to optimize the generated data quality. Recently, diffusion models (Sohl-Dickstein et al., 2015; Song & Ermon, 2019; Ho et al., 2020) have achieved state-of-the-art performance on various generation tasks, which define a Markov chain of diffusion steps to gradually add Gaussian noises to data and then learn to reverse the diffusion process to reconstruct data samples from the noise. Many follow-up works further improve the diffusion models in various aspects. DDPM (Ho et al., 2020) improves the sample quality and proposes a closed form to solve the training objective. An efficient sampling schedule is proposed to improve the generation speed (Nichol & Dhariwal, 2021). By performing the diffusion process in the latent space instead of pixel space, LDMs (Rombach et al., 2022) reduce the training and inference costs. However, it is challenging for DGMs to generate structured data such as dynamic trajectories, and it is even more challenging to control the generated data to satisfy certain safety-critical objectives. In this paper, we design specific trajectory representations and leverage the powerful generation capability of diffusion models to construct realistic and safety-critical scenarios. We also guide the diffusion generation and further optimize and constrain the generated scenarios through guided adversarial optimization.

Safety-critical Scenario Generation. Existing scenario generation algorithms can be divided into three categories. First, *data-driven* algorithms (Scanlon et al., 2021; Knies & Diermeyer, 2020; Ding et al., 2018; 2020b) generate testing scenarios based on real-world data collected by on-track testing. However, the collected data is highly unbalanced regarding safe and risky scenarios, which makes it challenging to train generative models to generate safety-critical scenarios. The second category uses *adversary-based* approaches (Ding et al., 2021a; Zhang et al., 2022; Feng et al., 2021) to generate safety-critical scenarios, which contain safety-critical objects such as adversarial vehicles and traffic signs. These methods fully explore the weakness of AV algorithms, but the scenarios are often less realistic and have limited diversity. Finally, *knowledge-based* scenario generation (Zhong et al., 2022; Ding et al., 2021b; Wang et al., 2021b; Bagschik et al., 2018) integrates knowledge rules, such as safety-critical constraints or traffic rules to guide the generation. However, it is usually hard to represent knowledge rules formally or integrate them with generative models directly. In this paper, we propose a diffusion-guided generation framework with flexible adversarial optimizations designed based on knowledge, which is able to generate diverse safety-critical scenarios and ensure the naturalness.

3. DiffScene

In this section, we first define the problem of safety-critical scenario generation in Section 3.1. Then we describe our scenario generation method based on diffusion models in Section 3.2. Finally, in Section 3.3, we introduce the guided safety-critical adversarial optimization.

3.1. Problem Statement

Formally, we define a traffic scenario as $z \in \mathcal{Z} := \{\mathcal{U}, \mathcal{I}, \mathcal{A}\}$. $\mathcal{U}$ represents the participating agents. $\mathcal{I}$ denotes the initial condition and properties of each agent. $\mathcal{A}$ represents the sequential actions. Each action sequence $\boldsymbol{a} \in \mathcal{A}$ is defined for certain agent $u \in \mathcal{U}$ as

$$\boldsymbol{a}(u) := [a_0, a_1, \cdots, a_T], \tag{1}$$

where a_t is the action taken at timestep t, and T is the maximum horizon length. Consider a model M maps the initial condition $\mathcal{I}$ to an initial system state $\boldsymbol{s}_0$ and derives the whole sequence of system states based on action sequences $\mathcal{A}$:

$$\boldsymbol{s}_t = M(\boldsymbol{s}_0, \mathcal{A}, t) \tag{2}$$

Similarly, we define the state sequence for each agent as

$$\boldsymbol{s}(u) := [s_0, s_1, \cdots, s_T], \tag{3}$$

where s_t is the state of agent u at timestep t. The trajectory of u consists of its state and action sequences:

$$\boldsymbol{\tau}_u := \{\boldsymbol{s}(u), \boldsymbol{a}(u)\}. \tag{4}$$

In a safety-critical scenario, we consider the participating agents $\mathcal{U} := \{u_{ego}, u_{sv}\}$, where u_{ego} is the ego vehicle controlled by certain AV algorithm f: $\boldsymbol{a}(u_{ego}) = f(z)$, and u_{sv} is a safety-critical surrounding vehicle (SV) controlled by an adversary. $\mathcal{R}_{adv}(\boldsymbol{\tau}_{sv}, f)$ is an adversarial risk function measuring the risk of the current scenario, e.g., collision rate, where the ego vehicle is controlled by f and the safety-critical SV takes trajectory $\boldsymbol{\tau}_{sv}$. $\mathcal{C}(\boldsymbol{\tau}_{sv})$ is a cost function over the SV trajectory evaluating the naturalness (cost) of the safety-critical trajectory. Given the AV algorithm f, the goal of the safety-critical scenario generator is to create a safety-critical trajectory $\boldsymbol{\tau}_{sv}$ for the safety-critical SV such that the risk of the scenario is maximized while the generated safety-critical trajectory maintains a low naturalness cost:

$$\arg\max_{\boldsymbol{\tau}_{sv}} \mathcal{R}_{adv}(\boldsymbol{\tau}_{sv}, f), \text{ s.t. } \mathcal{C}(\boldsymbol{\tau}_{sv}) < c, \tag{5}$$

where c is a threshold for the naturalness cost budget.

Due to the high dimensionality and rareness of the safety-critical scenarios, we consider a diffusion-based, adversarially guided generation framework to sample and optimize realistic safety-critical traffic scenarios. Specifically, we first leverage a goal-agnostic diffusion model trained on large-scale benign driving data to generate realistic benign traffic scenarios with low naturalness cost $\mathcal{C}(\boldsymbol{\tau}_{sv})$. Then we optimize the generated scenario based on different adversarial objectives at each diffusion step to maximize the risk $\mathcal{R}_{adv}(\boldsymbol{\tau}_{sv}, f)$ and maintain low cost. The detailed pipeline of our method is shown in Figure 1.

3.2. Diffusion-based Scenario Generation

Diffusion models (Ho et al., 2020; Nichol & Dhariwal, 2021) approximate the data distribution by a Markov chain starting from a Gaussian distribution. The model learns to reverse a forward diffusion process and generate data by incrementally denoising the sequence from Gaussian noise. We leverage the reverse diffusion process to generate traffic scenarios with high naturalness, since the model is trained to approximate *natural* traffic distributions.

Trajectory representation A trajectory $\boldsymbol{\tau}$ is composed of a state sequence $\boldsymbol{s}$ and an action sequence $\boldsymbol{a}$. We formulate each trajectory as a matrix:

$$\boldsymbol{\tau} = \begin{bmatrix} \boldsymbol{s} \\ \boldsymbol{a} \end{bmatrix} = \begin{bmatrix} s_0 & s_1 & \dots & s_T \\ a_0 & a_1 & \dots & a_T \end{bmatrix}, \tag{6}$$

where each column consists of a state-action pair at a certain timestep along the horizon of the trajectory.

Trajectory generation with diffusion models We first use a diffusion model to generate the benign trajectory $\boldsymbol{\tau}$ for the SV. The generation process is an iterative denoising procedure starting from the initial data distribution $p_\theta(\boldsymbol{\tau}^K) \approx \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, where K is the total number of diffusion steps. Each denoising transition $\boldsymbol{\tau}^k \to \boldsymbol{\tau}^{k-1}$ from step k to step $k-1$ is parameterized by the diffusion model:

$$p_\theta(\boldsymbol{\tau}^{k-1}|\boldsymbol{\tau}^k) = \mathcal{N}(\boldsymbol{\tau}^{k-1}; \mu_\theta(\boldsymbol{\tau}^k, k), \Sigma_\theta(\boldsymbol{\tau}^k, k)), \tag{7}$$

where θ denotes the parameters of the diffusion model. The covariances in the reverse diffusion process are often fixed and depend on the diffusion step: $\boldsymbol{\Sigma}_\theta(\boldsymbol{\tau}^k, k) = \boldsymbol{\Sigma}^k$, where we adopt a cosine schedule following previous work (Nichol & Dhariwal, 2021; Janner et al., 2022). The distribution of the final generated clean data (i.e., $k = 0$) is represented as

$$p_\theta(\boldsymbol{\tau}^0) = p_\theta(\boldsymbol{\tau}^K) \prod_{k=1}^{K} p_\theta(\boldsymbol{\tau}^{k-1}|\boldsymbol{\tau}^k). \tag{8}$$

To train the diffusion model, we adopt a forward diffusion process starting from the clean trajectory $\boldsymbol{\tau}^0$. We gradually add Gaussian noise to the original trajectory until step K where $\boldsymbol{\tau}^K$ is approximately Gaussian. The forward diffusion process from step $k-1$ to step k is defined as

$$q(\boldsymbol{\tau}^k|\boldsymbol{\tau}^{k-1}) = \mathcal{N}(\boldsymbol{\tau}^k; \sqrt{1-\beta_k}\boldsymbol{\tau}^{k-1}, \beta_k \boldsymbol{I}) \tag{9}$$

where $\beta_1, \beta_2, \cdots \beta_K$ are fixed noise added to the trajectory data at each forward diffusion step. This forward process q contains no trainable parameters, which allows us to construct noisy trajectories from original data. At each training iteration, we train the diffusion model to approximate and reconstruct the natural clean data $\boldsymbol{\tau}^0$ through the denoising process. We use a simplified objective to train the diffusion model (Ho et al., 2020), given by

$$\mathcal{L}(\theta) = \mathbb{E}_{\epsilon,k,\boldsymbol{\tau}^0}[\|\boldsymbol{\tau}^0 - \hat{\boldsymbol{\tau}}\|^2] \tag{10}$$

where ϵ is the noise added to the clean trajectory and $\hat{\boldsymbol{\tau}} = \mu_\theta(\boldsymbol{\tau}^k, k)$ is the reconstructed trajectory.

3.3. Guided Adversarial Optimization

The diffusion model is trained to generate realistic trajectories for SV. To ensure the generated trajectories achieve high risk while maintaining low naturalness cost, we introduce an efficient adversarial optimization process with different optimization objectives. We define an objective function $\mathcal{J}(\boldsymbol{\tau})$ to characterize the risk and the naturalness of a generated trajectory. At each reverse diffusion step k, we modify the denoising process by adding the gradient of $\mathcal{J}$ as guidance:

$$p_\theta(\boldsymbol{\tau}^{k-1}|\boldsymbol{\tau}^k) \approx \mathcal{N}(\boldsymbol{\tau}^{k-1}; \mu + \Sigma g, \Sigma), \tag{11}$$

where $g = \nabla \mathcal{J}(\boldsymbol{\tau})$ specifies the optimization direction. By iteratively optimizing the trajectory towards the desired direction provided by $\mathcal{J}$, the diffusion model will finally generate an SV trajectory satisfying the optimization goals.

This adversarial optimization process enables flexible control over the generated scenarios. We introduce the following three types of objectives: *safety-based objective* $\mathcal{J}_{safe}(\boldsymbol{\tau})$ provides a safety-critical guarantee for the generated scenarios, *functionality-based objective* $\mathcal{J}_{fun}(\boldsymbol{\tau})$ focuses on interfering the regular operations of ego vehicles, and *constraint-based objective* $\mathcal{J}_{con}(\boldsymbol{\tau})$ controls the generated scenarios to satisfy specific rules or constraints. The

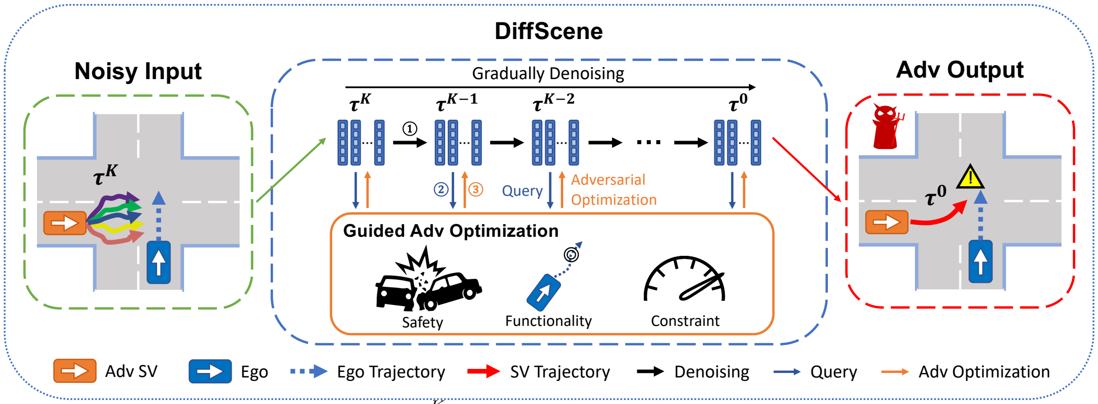


Figure 1: Overview of DiffScene. Given an initial noisy trajectory $\boldsymbol{\tau}^K$, we iteratively perform denoising steps and adversarial optimization steps to obtain the final adversarial SV trajectory $\boldsymbol{\tau}^0$. In each iteration, we first perform a denoising step to calculate the denoised trajectory following Equation (7). Then we perform multiple adversarial optimization steps using different adversarial objectives. The final output maximizes the risk of the generated safety-critical scenarios and maintains a low naturalness cost.

final safety-critical objective $\mathcal{J}(\boldsymbol{\tau})$ is a combination of the three objectives mentioned above:

$$\mathcal{J}(\boldsymbol{\tau}) = \omega_s \mathcal{J}_{safe}(\boldsymbol{\tau}) + \omega_f \mathcal{J}_{fun}(\boldsymbol{\tau}) + \omega_c \mathcal{J}_{con}(\boldsymbol{\tau}) \quad (12)$$

where ω_s, ω_f, and ω_c are three hyper-parameters controlling the weights of three different objectives.

Safety-based objective targets on the safety of the ego vehicle, which tries to maximize the driving risk of the ego vehicle. Specifically, we define safety-based objective as

$$\mathcal{J}_{safe}(\boldsymbol{\tau}) = -D(\boldsymbol{\tau}) + \lambda \mathbb{1}_{collision}(\boldsymbol{\tau}), \quad (13)$$

where $D(\boldsymbol{\tau})$ represents the minimal distance between the ego vehicle and the safety-critical SV in a scenario where SV follows trajectory $\boldsymbol{\tau}$, $\mathbb{1}_{collision}(\boldsymbol{\tau})$ is an indicator function to represent if the trajectory will cause collision between the ego vehicle and the safety-critical SV, and λ is a hyper-parameter. This safety-based objective encourages the SV to stay close to the ego vehicle so that the probability of collisions will increase.

Functionality-based objective targets on the functional ability of the ego vehicle to finish a given driving task. Specifically, in each testing scenario, the ego vehicle is expected to follow and complete a specific pre-defined route and reach the destination. The functionality-based objective controls a safety-critical SV to prevent the ego vehicle from completing its driving task. For example, the SV can stop the ego vehicle by trying to block the road. We define functionality-based objective as

$$\mathcal{J}_{fun}(\boldsymbol{\tau}) = r(\boldsymbol{\tau}), \quad (14)$$

where $r(\boldsymbol{\tau})$ denotes the percentage of the route not completed by the ego vehicle in a scenario with safety-critical SV following trajectory $\boldsymbol{\tau}$.

Constraint-based objective targets on the desired rules and constraints applied on the safety-critical SV in order to keep it realistic. In a real-world scenario, a trajectory must satisfy certain traffic rules or physical constraints. Here, we consider the speed-related constraint focusing on controlling the speed of the SV. We formulate the objective as

$$\mathcal{J}_{con}(\boldsymbol{\tau}) = \sum_{t=0}^{T} -|v_t - v^*|, \quad (15)$$

where v^* is the common driving speed of a vehicle and v_t is the speed of the SV at t. By maximizing this objective, the SV speed will be close to the normal speed v^*. Appendix A shows more detailed process of DiffScene.

4. Experiments

In this section, we conduct comprehensive experiments to evaluate DiffScene in diverse settings. We find that: 1) DiffScene is much more effective in terms of generating safety-critical scenarios compared with baselines. DiffScene scenarios achieve higher scores on safety-critical metrics and better performance on constraint satisfaction. 2) DiffScene achieves lower naturalness cost. DiffScene scenarios are more similar to benign scenarios in terms of both trajectory similarity and action similarity. 3) DiffScene demonstrates better downstream utility. AV algorithms fine-tuned with our safety-critical scenarios achieve lower risk scores than those fine-tuned on scenarios generated by baselines. 4) The transferability of DiffScene is higher than existing scenario generation algorithms. DiffScene scenarios are able to cause higher risks across different AV algorithms. 5) There is a trade-off for the generated scenarios in terms of their safety-critical and naturalness properties, balanced by the number of adversarial optimization steps during each denoising step.

4.1. Experimental Design and Setting

Scenario settings and platform We consider the following 3 scenario settings: Crossing Negotiation (S1), Red-light Running (S2), and Right-turn (S3). We show the detailed explainations and illustrations for all settings in Appendix B.1. We use Carla (Dosovitskiy et al., 2017; Xu et al., 2022) as our simulator. More details can be found in Appendix B.2.

Baselines We mainly consider the following 6 SOTA scenario generation baselines. Adversarial RL (AR), Carla Scenario Generator (CS) (Dosovitskiy et al., 2017), Learning-to-collide (LC) (Ding et al., 2020a), AdvSim (AS) (Wang et al., 2021a), Adversarial Trajectory Optimization (AT) (Zhang et al., 2022), and STRIVE (ST) (Rempe et al., 2022). We provide more details of the baselines in Appendix B.3.

Table 1: **Effectiveness evaluation.** We report *Collision Rate (CR)*, *Incomplete Route (IR)*, and *Speed Satisfaction (SS)* to measure the effectiveness of the generated safety-critical scenarios in terms of safety-level, functionality-level, and constraint-level in 3 different scenario settings. We show the averaged score and standard deviation of the results on 3 different AV algorithms. (All scores are the higher the better).

Scenario	Metric	AR	CS	LC	AS	AT	ST	DiffScene
S1	Collision Rate	0.19 ± 0.03	0.60 ± 0.14	0.58 ± 0.52	0.57 ± 0.33	0.62 ± 0.49	0.72 ± 0.11	**0.85** ± 0.08
	Incomplete Route	0.14 ± 0.10	0.27 ± 0.05	0.27 ± 0.24	0.26 ± 0.16	0.28 ± 0.22	0.32 ± 0.04	**0.39** ± 0.05
	Speed Satisfaction	0.09 ± 0.01	0.26 ± 0.01	0.33 ± 0.02	0.14 ± 0.01	0.30 ± 0.04	0.31 ± 0.05	**0.43** ± 0.01
S2	Collision Rate	0.38 ± 0.09	0.63 ± 0.15	0.71 ± 0.43	0.57 ± 0.14	0.71 ± 0.50	0.73 ± 0.08	**0.87** ± 0.10
	Incomplete Route	0.18 ± 0.03	0.29 ± 0.06	0.33 ± 0.20	0.25 ± 0.07	0.33 ± 0.23	0.34 ± 0.04	**0.40** ± 0.05
	Speed Satisfaction	0.12 ± 0.00	0.26 ± 0.01	0.27 ± 0.02	0.24 ± 0.01	0.30 ± 0.05	0.32 ± 0.06	**0.47** ± 0.01
S3	Collision Rate	0.34 ± 0.22	0.68 ± 0.16	0.59 ± 0.27	0.29 ± 0.30	0.59 ± 0.50	0.53 ± 0.40	**0.79** ± 0.15
	Incomplete Route	0.13 ± 0.09	0.22 ± 0.04	0.21 ± 0.10	0.09 ± 0.09	0.19 ± 0.16	0.23 ± 0.18	**0.27** ± 0.08
	Speed Satisfaction	0.08 ± 0.00	0.19 ± 0.01	0.21 ± 0.01	0.20 ± 0.02	0.34 ± 0.00	0.34 ± 0.01	**0.38** ± 0.00

AV algorithms and models To evaluate the effectiveness and transferability of the scenario generation algorithms, we test the generated scenarios against different AV algorithms: *SAC*, *PPO*, and *TD3*. We train 3 target RL models using the 3 different RL algorithms in benign driving scenarios and evaluate them in the generated safety-critical scenarios. More model details can be found in Appendix B.4. We also show more training details in Appendix B.5.

4.2. Evaluation Metrics

In terms of effectiveness, we calculate 3 different metrics: *Collision Rate (CR)*, *Incomplete Route (IR)*, and *Speed Satisfaction (SS)*. For naturalness, we calculate 5 distance functions. We report *Symmetric Segment-Path Distance (SSPD)*, *Fréchet Distance (Fréchet)*, and *Dynamic Time Warping (DTW)* to measure trajectory similarity, and we report *Wasserstein Distance (WD)* and *Kullback–Leibler Divergence (KL)* to measure action similarity. All definitions of the evaluation metrics can be found in Appendix B.6.

4.3. Effectiveness of DiffScene

The quantitative results are shown in Table 1, and qualitative comparisons are shown in Appendix D.2. From the scenario generation algorithm perspective, we observe that DiffScene achieves the best scores among all the methods, demonstrating its advantage of creating more safety-critical scenarios while satisfying rules and constraints. From the scenario setting perspective, *Red-light Running (S2)* is the most safety-critical scenario setting, with the highest collision rate of 87% achieved by DiffScene. The Right-turn (S3) is the safest scenario setting, where DiffScene achieves 79% collision rate. From the collision rate perspective, we notice that DiffScene achieves over 75% average collision rate in all the 3 scenario settings, showing that existing RL-based AV algorithms are vulnerable to DiffScene scenarios. Finally, from the speed satisfaction perspective, we find that the generated scenarios are hard to achieve higher scores. This is due to the physical constraints of the vehicles: the limited acceleration. It will always take some time to increase the speed from 0 to v^* even with the highest acceleration.

4.4. Naturalness of DiffScene

Trajectory similarity We show the results in Table 2a, where we only report the scores for AR, LC, AT, ST, and DiffScene since the paths generated by CS and AS are pre-defined as a fixed straight line. We note that our method achieves the lowest scores among the baselines, which shows that the DiffScene trajectories are the closest to the benign ones. Among the 3 scenario settings, DiffScene has the lowest similarity score in S2, which again demonstrates that S2 is more safety-critical: easier to achieve high collision rate with a low cost. ST has the highest similarity scores since it has weak restrictions on the trajectory similarity of the generated scenarios to the benign ones. We omit scores greater than 100 caused by ST.

Table 2: **Naturalness evaluation.** For trajectory similarity evaluation, we report the *SSPD*, *Fréchet*, and *DTW* to measure the similarity between the SV paths in the generated and real collected scenarios. For action similarity evaluation, we report the *WD* and *KL* scores to measure the similarity between the behaviors of the SV in the generated and real collected scenarios. We evaluate the scenarios on 3 different target AD algorithms and report the averaged scores. (All scores are the lower the better).

(a) Trajectory similarity evaluation

Scenario	Metric	AR	LC	AT	ST	DiffScene
S1	SSPD	1.07	0.36	0.35	94.78	**0.19**
	Fréchet	6.51	1.45	1.12	>100	**1.04**
	DTW	69.10	57.80	21.16	>100	**12.96**
S2	SSPD	0.54	0.48	0.29	>100	**0.17**
	Fréchet	3.38	1.64	1.11	>100	**1.04**
	DTW	34.74	81.85	18.62	>100	**11.96**
S3	SSPD	0.38	0.33	0.40	>100	**0.25**
	Fréchet	2.80	2.40	2.14	>100	**1.99**
	DTW	30.65	65.55	35.44	>100	**24.58**

(b) Action similarity evaluation

Scenario	Metric	AR	CS	LC	AS	AT	ST	DiffScene
S1	WD	1.74	0.53	0.62	0.47	0.96	0.95	**0.37**
	KL	>10	>10	>10	>10	2.17	1.77	**1.43**
S2	WD	1.78	0.55	0.56	0.59	0.92	0.88	**0.38**
	KL	>10	>10	>10	>10	2.39	2.17	**1.34**
S3	WD	1.24	0.59	0.63	0.59	1.03	1.07	**0.48**
	KL	>10	>10	>10	>10	**0.99**	1.38	1.41

Action similarity We present the results in Table 2b. The action similarity scores of the scenarios generated by DiffScene are almost the lowest, meaning that the action distribution of the SV is more similar to the benign distribution. Since KL can be very large when the two distributions are extremely different, we omit the scores greater than 10.

4.5. Downstream Utility of DiffScene

We evaluate the downstream utility of the generated safety-critical scenarios by measuring the safety improvements of AV algorithms after being finetuned on these scenarios.

Table 3: **Downstream utility evaluation.** We report *Collision Rate (CR)* and *Incomplete Route (IR)* after training with generated scenarios to measure the downstream utility of corresponding generation algorithms. We finetune the target SAC model in the generated *S1* scenarios using 3 different random seeds and show the averaged testing results. (All scores are the lower the better).

Metric	SAC	AR	CS	LC	AS	AT	ST	DiffScene
CR	0.90	0.82	0.37	0.32	0.75	0.33	0.76	**0.26**
IR	0.36	0.27	0.15	0.32	0.30	0.13	0.29	**0.11**

We use the *Crossing Negotiation (S1)* scenario setting as an example. For the scenarios generated by each generation algorithm, we use 80% of them as the training set. The remaining 20% scenarios from all algorithms together form a standard test set. We finetune the target SAC model in the different training sets using 3 different random seeds, each for 500 episodes, and report the averaged testing result on the standard test set. The results are shown in Table 3, where we report the *Collision Rate* and *Incomplete Route* scores of the ego vehicle after finetuning. We also show the performance of the target SAC model on the standard testing dataset before finetuning it as a reference.

According to Table 3, SAC finetuned on the DiffScene scenarios achieves the lowest *collision rate* and *incomplete route*, which also means that the DiffScene is more useful in terms of improving the robustness of the AV algorithms. Among the baselines, LC is the most helpful algorithm in terms of reducing the collision rate, while AT is the most helpful algorithm to improve route completion. However, they are still not as effective as DiffScene.

4.6. Ablation Studies

Transferability In our experiments, we perform a transferability-based black-box attack, where we generate and optimize safety-critical scenarios against a surrogate SAC model and evaluate the generated scenarios using 3 different RL-based AV algorithms. We show the standard deviation of the testing results on 3 different algorithms in Table 1. We also show the heatmap of *collision rate* for each AV algorithm achieved by each generation algorithm in 3 different scenario settings in Figure 2.

The numbers in Table 1 show that in many cases, DiffScene has the lowest standard deviation across 3 different algorithms, meaning that the scenarios generated by DiffScene can be easily transferred to other AV algorithms. Baselines with low standard deviations usually suffer from limited effectiveness, e.g., AR and CS. The detailed results in Figure 2 also verify our conclusions. In the heatmap, our DiffScene shows little difference across 3 different AV algorithms. In practice, safety-critical scenarios with higher transferability can be used to detect vulnerabilities of other AV algorithms and help to improve their robustness, which is more useful in real-world applications.

Impact of the number of adversarial optimization steps We generate the safety-critical scenarios with different numbers of adversarial optimization steps N, ranging from

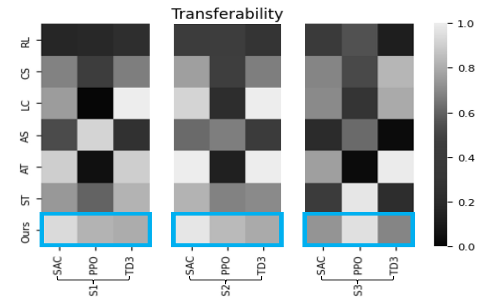


Figure 2: **Transferability**. We report the *CR* for each AV algorithm in 3 scenarios. DiffScene demonstrates the highest transferability across AV algorithms.

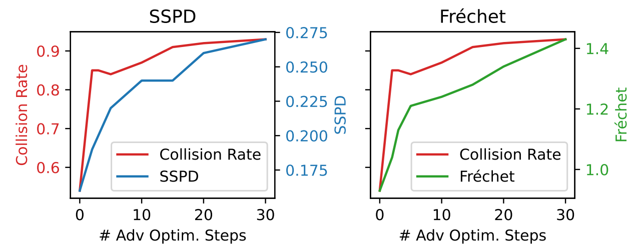


Figure 3: **Effects of adversarial optimization**. We show *Collision Rate*, *SSPD*, and *Fréchet* of scenarios generated by DiffScene under different numbers of adversarial optimization steps, indicating the tradeoff between safety-critical and naturalness. (Higher *Collision Rate* means more safety-critical, and lower *SSPD* and *Fréchet* indicate higher naturalness).

$N = 0$ to $N = 30$. Due to the space limit, we plot the lines for *collision rate*, *SSPD*, and *Fréchet* in Figure 3 and leave the *DTW* results in Appendix D.3.

We find that as N increases, the collision rate will also increase, meaning that the adversarial optimization steps do help to generate more safety-critical scenarios. However, when applying a larger N, SSPD and Fréchet will also be larger, showing that more adversarial optimization steps will lead to more naturalness cost. From this result, we can clearly see a trade-off between the effectiveness and naturalness of the generated scenarios. We can easily control and balance them by choosing a proper number of guided adversarial optimization steps in DiffScene.

5. Conclusion

In this paper, we propose DiffScene, a diffusion-based, safety-critical guided generation framework to generate realistic and safety-critical scenarios. Extensive experiments in Carla show that our framework is able to generate safety-critical scenarios against different AV algorithms under various settings. We show that our generated scenarios are more effective, natural, and transferable, and have higher downstream utilities. We also show that current RL-based AV algorithms are vulnerable to the generated safety-critical scenarios. In the meantime, we need to control DiffScene to make sure that the generated safety-critical scenarios are not used for adversarial purposes (see Appendix E for more discussion). We hope this study will shed light on future research on identifying weaknesses in existing AVs, thus facilitating more efficient and effective AV development.

References

Agostinelli, F., McAleer, S., Shmakov, A., and Baldi, P. Solving the rubik's cube with deep reinforcement learning and search. *Nature Machine Intelligence*, 1(8):356–363, 2019.

Arief, M., Huang, Z., Kumar, G. K. S., Bai, Y., He, S., Ding, W., Lam, H., and Zhao, D. Deep probabilistic accelerated evaluation: A certifiable rare-event simulation methodology for black-box autonomy. *arXiv preprint arXiv:2006.15722*, 2020.

Bagschik, G., Menzel, T., and Maurer, M. Ontology based scene creation for the development of automated vehicles. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1813–1820. IEEE, 2018.

Bucklew, J. A. and Bucklew, J. *Introduction to rare event simulation*, volume 5. Springer, 2004.

CDMV. California Department of Motor Vehicle Disengagement Report. `https://www.dmv.ca.gov/portal/vehicle-industry-services/autonomous-vehicles/disengagement-reports/`, 2022. [Online].

Chen, J., Yuan, B., and Tomizuka, M. Model-free deep reinforcement learning for urban autonomous driving. In *2019 IEEE intelligent transportation systems conference (ITSC)*, pp. 2765–2771. IEEE, 2019.

Chen, J., Li, S. E., and Tomizuka, M. Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, 2021a.

Chen, Y., Rong, F., Duggal, S., Wang, S., Yan, X., Manivasagam, S., Xue, S., Yumer, E., and Urtasun, R. Geosim: Realistic video simulation via geometry-aware composition for self-driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7230–7240, 2021b.

Ding, W., Wang, W., and Zhao, D. A new multi-vehicle trajectory generator to simulate vehicle-to-vehicle encounters. *arXiv preprint arXiv:1809.05680*, 2018.

Ding, W., Chen, B., Xu, M., and Zhao, D. Learning to collide: An adaptive safety-critical scenarios generating method. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2243–2250. IEEE, 2020a.

Ding, W., Xu, M., and Zhao, D. Cmts: A conditional multiple trajectory synthesizer for generating safety-critical driving scenarios. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4314–4321. IEEE, 2020b.

Ding, W., Chen, B., Li, B., Eun, K. J., and Zhao, D. Multimodal safety-critical scenarios generation for decision-making algorithms evaluation. *IEEE Robotics and Automation Letters*, 6(2):1551–1558, 2021a.

Ding, W., Li, B., Eun, K. J., and Zhao, D. Semantically controllable scene generation with guidance of explicit knowledge. *arXiv preprint arXiv:2106.04066*, 2021b.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pp. 1–16, 2017.

Ehrhardt, S., Groth, O., Monszpart, A., Engelcke, M., Posner, I., Mitra, N., and Vedaldi, A. Relate: Physically plausible multi-object scene synthesis using structured latent spaces. *Advances in Neural Information Processing Systems*, 33:11202–11213, 2020.

Feng, S., Yan, X., Sun, H., Feng, Y., and Liu, H. X. Intelligent driving intelligence test for autonomous vehicles with naturalistic and adversarial environment. *Nature communications*, 12(1):1–14, 2021.

Fujimoto, S., Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pp. 1587–1596. PMLR, 2018.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. PMLR, 2018.

He, K., Zhang, X., Ren, S., and Sun, J. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Janner, M., Du, Y., Tenenbaum, J., and Levine, S. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, 2022.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Al Sallab, A. A., Yogamani, S., and Pérez, P. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

Knies, C. and Diermeyer, F. Data-driven test scenario generation for cooperative maneuver planning on highways. *Applied Sciences*, 10(22):8154, 2020.

Najm, W. G., Smith, J. D., Yanagisawa, M., et al. Pre-crash scenario typology for crash avoidance research. Technical report, United States. National Highway Traffic Safety Administration, 2007.

Nichol, A. Q. and Dhariwal, P. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, pp. 8162–8171. PMLR, 2021.

O'Kelly, M., Sinha, A., Namkoong, H., Tedrake, R., and Duchi, J. C. Scalable end-to-end autonomous vehicle testing via rare-event simulation. *Advances in neural information processing systems*, 31, 2018.

Polack, P., Altché, F., d'Andréa Novel, B., and de La Fortelle, A. The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles? In *2017 IEEE intelligent vehicles symposium (IV)*, pp. 812–818. IEEE, 2017.

Poli, R., Kennedy, J., and Blackwell, T. Particle swarm optimization. *Swarm intelligence*, 1(1):33–57, 2007.

Rempe, D., Philion, J., Guibas, L. J., Fidler, S., and Litany, O. Generating useful accident-prone driving scenarios via a learned traffic prior. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17305–17315, 2022.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10684–10695, 2022.

Ru, B., Cobb, A., Blaas, A., and Gal, Y. Bayesopt adversarial attack. In *International Conference on Learning Representations*, 2019.

Sallab, A. E., Abdou, M., Perot, E., and Yogamani, S. Deep reinforcement learning framework for autonomous driving. *Electronic Imaging*, 2017(19):70–76, 2017.

Scanlon, J. M., Kusano, K. D., Daniel, T., Alderson, C., Ogle, A., and Victor, T. Waymo simulated driving behavior in reconstructed fatal crashes within an autonomous vehicle operating domain. *Accident Analysis & Prevention*, 163:106454, 2021.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., and Hassabis, D. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018. ISSN 0036-8075. doi: 10.1126/science.aar6404. URL https://science.sciencemag.org/content/362/6419/1140.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pp. 2256–2265. PMLR, 2015.

Song, Y. and Ermon, S. Generative modeling by estimating gradients of the data distribution. *Advances in Neural Information Processing Systems*, 32, 2019.

Srinivas, N., Krause, A., Kakade, S. M., and Seeger, M. Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*, 2009.

Wang, J., Pun, A., Tu, J., Manivasagam, S., Sadat, A., Casas, S., Ren, M., and Urtasun, R. Advsim: Generating safety-critical scenarios for self-driving vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9909–9918, 2021a.

Wang, X., Krasowski, H., and Althoff, M. Commonroad-rl: a configurable reinforcement learning environment for motion planning of autonomous vehicles. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pp. 466–472. IEEE, 2021b.

Webb, N., Smith, D., Ludwick, C., Victor, T., Hommes, Q., Favaro, F., Ivanov, G., and Daniel, T. Waymo's safety methodologies and safety readiness determinations. *arXiv preprint arXiv:2011.00054*, 2020.

Xu, C., Ding, W., Lyu, W., Liu, Z., Wang, S., He, Y., Hu, H., Zhao, D., and Li, B. Safebench: A benchmarking platform for safety evaluation of autonomous vehicles. In *Advances in Neural Information Processing Systems*, 2022.

Yang, Z., Chai, Y., Anguelov, D., Zhou, Y., Sun, P., Erhan, D., Rafferty, S., and Kretzschmar, H. Surfelgan: Synthesizing realistic sensor data for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11118–11127, 2020.

Zhang, Q., Hu, S., Sun, J., Chen, Q. A., and Mao, Z. M. On adversarial robustness of trajectory prediction for autonomous vehicles. *arXiv preprint arXiv:2201.05057*, 2022.

Zhao, D. *Accelerated Evaluation of Automated Vehicles.* PhD thesis, 2016.

Zhong, Z., Rempe, D., Xu, D., Chen, Y., Veer, S., Che, T., Ray, B., and Pavone, M. Guided conditional diffusion for controllable traffic simulation. *arXiv preprint arXiv:2210.17366*, 2022.

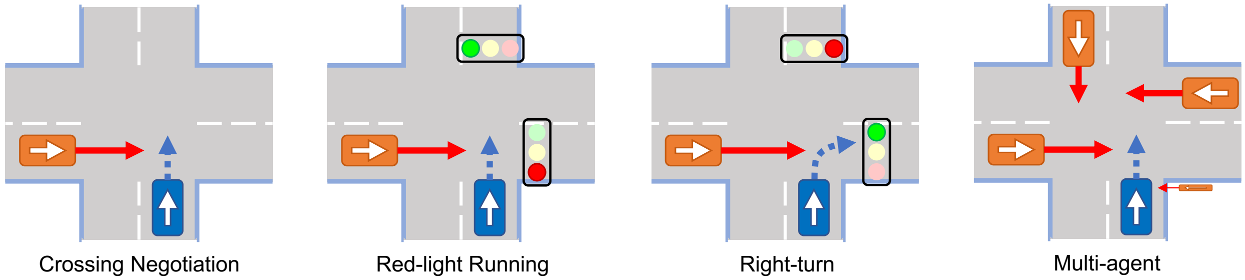


Figure 4: Details of each scenario setting.

A. DiffScene Details

The detailed process of DiffScene is shown in Algorithm 1. We first use the benign driving data to train a diffusion model μ_θ approximating the real trajectory distribution and a separate model $\mathcal{J}_\phi$ predicting the safety-critical objective $\mathcal{J}(\boldsymbol{\tau})$. At each reverse diffusion step, the diffusion model first predicts the denoised clean trajectory $\hat{\boldsymbol{\tau}}$ following Equation (7). Then we perform a multi-step optimization using the gradient of safety-critical objective $\mathcal{J}_\phi(\boldsymbol{\tau})$. The multi-step optimization process provides flexible control over the trade-off between the goal of being safety-critical and staying close to the real data distribution. At the end of each denoising step, we calibrate the generated trajectory using the ground truth initial system state calculated by model M. We align the initial state s_0 in the generated SV trajectory with the real initial SV state to make sure every trajectory starts from the same true state. After the initial state calibration, the generated trajectory is then used as the noisy input for the next denoising step until we get the final safety-critical trajectory $\boldsymbol{\tau}_{sv} = \boldsymbol{\tau}^0$. Different from CTG (Zhong et al., 2022) and Diffuser (Janner et al., 2022), Algorithm 1 generates the whole safety-critical trajectory using only one reverse diffusion process. Since the reverse process is time-consuming, our DiffScene is much more efficient and enables real-time scenario generation in practice.

Algorithm 1 Guided Adversarial Trajectory Generation

Input: Model M, initial condition $\mathcal{I}$, diffusion model μ_θ, adversarial objective model $\mathcal{J}_\phi$, scale α, number of diffusion steps K, number of guided steps N, covariances Σ^k
Output: Adversarial SV trajectory $\boldsymbol{\tau}_{sv}$
$\boldsymbol{s}_0 = M(\mathcal{I})$ ▷ observe initial state
$\boldsymbol{\tau}^K \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ ▷ sample initial trajectory
$\boldsymbol{\tau}^K_{s_0} \leftarrow \boldsymbol{s}_0$ ▷ initial state calibration
for $k = K$ **to** 1 **do**
  $\hat{\boldsymbol{\tau}} \leftarrow \mu_\theta(\boldsymbol{\tau}^k)$ ▷ reverse diffusion
  for $j = 1$ **to** N **do**
    $\hat{\boldsymbol{\tau}} = \hat{\boldsymbol{\tau}} + \alpha \nabla \mathcal{J}_\phi(\hat{\boldsymbol{\tau}})$ ▷ adversarial optimization
  end for
  $\boldsymbol{\tau}^{k-1} \sim \mathcal{N}(\hat{\boldsymbol{\tau}}, \Sigma^k)$ ▷ sampling
  $\boldsymbol{\tau}^{k-1}_{s_0} \leftarrow \boldsymbol{s}_0$ ▷ initial state calibration
end for
Return: $\boldsymbol{\tau}_{sv} \leftarrow \boldsymbol{\tau}^0$

B. Experimental Design and Setting

B.1. Scenario settings

We consider the three most representative and challenging scenario settings of pre-crash traffic (Najm et al., 2007) summarized by NHTSA. **Crossing Negotiation (S1)**: the ego vehicle meets a crossing SV when passing an intersection with no traffic lights. The ego vehicle should negotiate with the SV to cross the unsignalized intersection. **Red-light Running (S2)**: a crossing SV runs a red light while the ego vehicle is going straight at an intersection. Collision avoidance actions must be taken to keep safe. **Right-turn (S3)**: the ego vehicle is performing a right turn at an intersection, with a crossing SV in front. The ego vehicle should take action to avoid collisions. In addition, we also consider a multi-agent scenario setting.

We show the details of each scenario setting in Figure 4. In each scenario, the ego vehicle is supposed to drive along a pre-defined route and react to emergencies that occur on the road while driving. In a safety-critical scenario, the SV tries to attack the ego vehicle while behaving like a benign vehicle. The ego vehicle should avoid potential car accidents and reach its destination. In addition to the single-SV settings, we also consider a multi-agent setting where multiple SVs are involved in the scenario, including vehicles and pedestrians/cyclists. We calculate the effectiveness metrics and naturalness metrics according to the testing results on the 100 testing scenarios under each scenario setting.

B.2. Simulation platform

We use Carla (Dosovitskiy et al., 2017; Xu et al., 2022) as our simulator, which provides realistic simulations of traffic scenarios. We consider 10 different routes in each scenario setting and use 10 different seeds to generate different testing scenarios in each route, obtaining 100 testing scenarios in total for each scenario generation algorithm.

B.3. Baselines

We mainly consider the following 6 state-of-the-art scenario generation baselines. **Adversarial RL (AR)** leverages an RL-based SV to generate safety-critical scenarios. We train an *SAC* (Haarnoja et al., 2018) as our safety-critical vehicle. **Carla Scenario Generator (CS)** (Dosovitskiy et al., 2017) uses rule-based methods to construct scenarios. Following the standard process, we adopt the rules and use grid search to search for the optimal safety-critical testing scenarios in 3 different scenario settings. **Learning-to-collide (LC)** (Ding et al., 2020a) uses a Bayesian network to describe the relationship between traffic participants. Following the default setting, we generated scenarios by sampling from the joint distribution of a series of auto-regressive building blocks. **AdvSim (AS)** (Wang et al., 2021a) manipulates the trajectory of the SV to attack the ego vehicle using Bayesian optimization (Srinivas et al., 2009; Ru et al., 2019). They use the kinematic bicycle model (Polack et al., 2017) to represent and calculate the entire trajectory of SV. **Adversarial Trajectory Optimization (AT)** (Zhang et al., 2022) improves the scenario optimization process using explicit knowledge as constraints. We adopt the same constraints and apply the default PSO-based (Poli et al., 2007) optimization to generate safety-critical scenarios. **STRIVE (ST)** (Rempe et al., 2022) learns a traffic model for the trajectories first and then performs adversarial optimization based on the given planners and the prediction of the traffic model. We adapt STRIVE to SafeBench following the same hyper-parameter settings in the official codebase.

B.4. AV algorithms and models

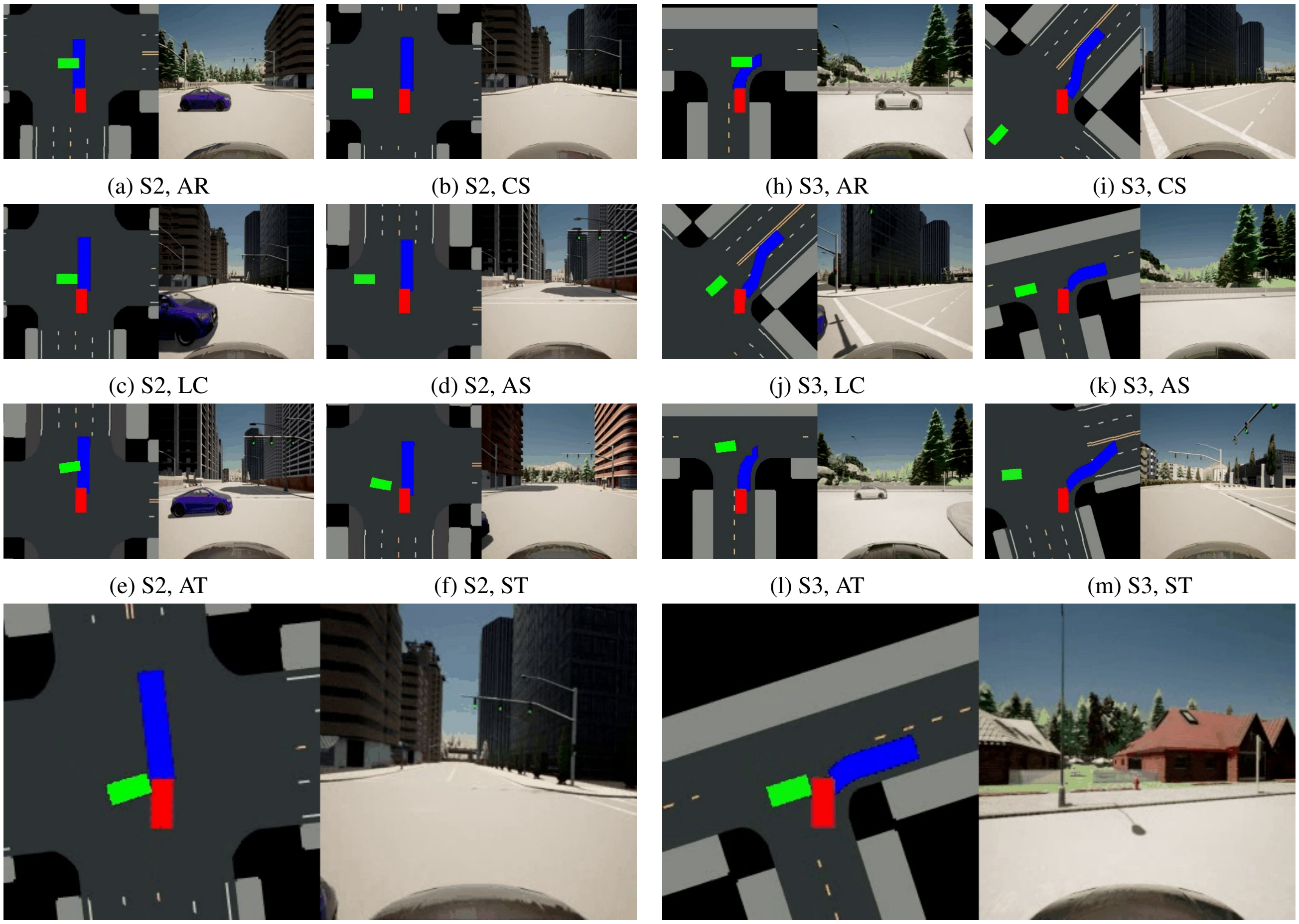
To evaluate the effectiveness and transferability of the scenario generation algorithms, we test the generated scenarios against different AV algorithms. We mainly focus on RL-based AV algorithms, since they require minimum domain knowledge of the overall system and driving scenarios (Sallab et al., 2017; Chen et al., 2019; Kiran et al., 2021). Specifically, we control the ego vehicle using 3 representative deep RL algorithms: *SAC*, *PPO* (Schulman et al., 2017), and *TD3* (Fujimoto et al., 2018). We train 3 target RL models using the 3 different RL algorithms in benign driving scenarios and evaluate them in the generated safety-critical scenarios.

To better evaluate the performance of different scenario generation algorithms, we also consider the transferability-based black-box attack in our experiments. Therefore, we additionally train a surrogate SAC model with the same configuration but using a different initialization. When evaluating a scenario generation algorithm, we first use it to generate safety-critical scenarios against the surrogate model. Then the generated scenarios are tested on the 3 target models.

Model input and output We design the state spaces for each RL algorithm based on previous works (Chen et al., 2019; 2021a) as a 4-dimensional observation: distance to the waypoint, longitude speed, angular speed, and a front-vehicle detection signal. The reward function is given by a weighted sum of the route following bonus, the collision penalty, the speeding penalty, and the energy consumption penalty. The action space is a 2-dimensional vector specifying the steering and throttle of the vehicle.

Model architecture and hyperparameters We use MLPs as our deep RL-based AV models. The size of the hidden layer is [256, 256]. The detailed hyperparameters for each algorithm are specified as follows.

- **SAC hyperparameters.** The policy learning rate and Q-value learning rate are both 0.001. The entropy regularization



(a) S2, AR (b) S2, CS (h) S3, AR (i) S3, CS

(c) S2, LC (d) S2, AS (j) S3, LC (k) S3, AS

(e) S2, AT (f) S2, ST (l) S3, AT (m) S3, ST

(g) S2, `DiffScene` (n) S3, `DiffScene`

Figure 5: Qualitative Results. We show examples of the generated scenarios obtained by different baseline algorithms and our `DiffScene`.

coefficient is 0.1. The discount factor is 0.99, and the number of models in the Q-ensemble critic is 2.

- **PPO hyperparameters.** The policy learning rate is 0.0003, and the Q-value learning rate is 0.001. The clipping ratio of the policy object is 0.2. The target KL divergence is 0.01. The discount factor is 0.99, and the number of interaction steps is 1000.
- **TD3 hyperparameters.** The policy learning rate and Q-value learning rate are both 0.001. The standard deviation for Gaussian noise added during training is 0.1. The standard deviation for smoothing noise is 0.2. The discount factor is 0.99. The number of models in the Q-ensemble critic is 2.

Model training We train all RL algorithms in Carla town03, since the environment of town03 is complicated and diverse. In each episode, we place the ego agent at a random starting point and create random benign surrounding traffic around it where all the SVs are auto-piloted. The agent is trained to follow its route and avoid potential collisions.

We train our RL models on NVIDIA GeForce RTX 3090 GPUs, and the training usually takes 24 hours. For each trained model, we achieve a stable reward value of around 1500 in one episode.

B.5. Data collection

To train the diffusion model μ_θ, we construct a benign trajectory dataset in Carla. Specifically, we adopt similar configurations in RL training and train several RL models in benign scenarios from scratch. We collect the trajectories of all episodes during training. Finally, we collect $6,995$ trajectories as the benign driving dataset to train the diffusion model. Once the diffusion model is trained, it can generate trajectories in all scenario settings. To train the safety-critical objective model $\mathcal{J}_\phi$, we collect $5,000$ trajectories under each scenario setting using the trained diffusion model and calculate the safety-critical objective $\mathcal{J}(\boldsymbol{\tau})$ for each trajectory as ground truth. In each scenario setting, we use $4,000$ trajectories as the training set and $1,000$ trajectories as the testing set. We train 3 different $\mathcal{J}_\phi$ models separately using datasets collected from 3 different scenario settings.

B.6. Evaluation Metrics

In this section, we introduce the evaluation metrics used in our experiments. Specifically, we evaluate the effectiveness of the generated scenarios from 3 different levels: safety level, functionality level, and constraint level. We evaluate the naturalness of the generation algorithm by measuring the similarity between the generated and benign scenarios.

Effectiveness In order to identify the weakness of the AV algorithms, a good safety-critical scenario generation algorithm is supposed to cause more safety concerns to the ego vehicle, interfere with the regular operation of the ego vehicle, and satisfy physical constraints in the meantime. We use the following 3 metrics to evaluate the effectiveness of a scenario generation algorithm. **Collision Rate (CR)** calculates the average collision rate of the generated scenarios, which can be calculated as $\mathbb{E}_{\boldsymbol{\tau}\sim\mathcal{P}}[\mathbb{1}_{collision}(\boldsymbol{\tau})]$, where $\mathcal{P}$ is the generated trajectory distribution. **Incomplete Route (IR)** evaluates the average percentage of the route not completed by the ego vehicle given the generated safety-critical SV trajectory $\boldsymbol{\tau}$: $\mathbb{E}_{\boldsymbol{\tau}\sim\mathcal{P}}[r(\boldsymbol{\tau})]$. **Speed Satisfaction (SS)** measures the satisfaction of the generated scenario in terms of keeping the normal driving speed. It can be calculated as $\mathbb{E}_{\boldsymbol{\tau}\sim\mathcal{P}}[\mathbb{E}_t[\mathbb{1}(|v_t - v^*| < \delta_v)]]$, where $\mathbb{1}$ is an indicator function and δ_v is a velocity threshold. In our experiments, we set the speed threshold $\delta_v = 1$.

Naturalness Besides being effective and safety-critical, the generated scenarios are also supposed to be highly realistic and naturalistic. We use 5 metrics in total to measure 2 different kinds of similarities between the generated scenarios and the benign scenarios. **Trajectory Similarity** evaluates how similar the actual path traveled by the SV is to the benign SV path, where the path is represented by a sequence of coordinates: $(x_i, y_i), i \in [0, \cdots, T]$. We consider 3 different metrics measuring the trajectory similarity: *Symmetric Segment-Path Distance (SSPD)*, *Fréchet Distance (Fréchet)*, and *Dynamic Time Warping (DTW)*. Since trajectory similarity metrics are strongly affected by the length of the traveled path, we preprocess the generated trajectories by cutting the end of the paths to so that they are longer than the benign path by a maximum of δ_τ, where δ_τ is a length threshold. To accurately eliminate the effect of length on the similarity results, we set $\delta_\tau = 0.5$ when calculating trajectory similarity. **Action Similarity** measures how similar the actual behavior taken by the SV is to the benign SV behavior, where the behavior is represented by the distribution of the acceleration in the horizontal plane: (acc_x, acc_y). We use 2 metrics to calculate the action similarity: *Wasserstein Distance (WD)* and *Kullback–Leibler Divergence (KL)*. Action similarity metrics evaluate the distance between the acceleration distribution of the generated scenarios and the benign ones, which are barely affected by the path length. Therefore, we directly calculate the action similarity without limiting the length threshold.

To evaluate naturalness, we calculate different kinds of similarity scores between the generated scenarios and benign scenarios. Specifically, we first use the surrogate SAC model to control the SV in the 3 different scenario settings and collect the output trajectories from the simulation results as benign trajectories since the SAC model is trained on normal traffic data and represents the benign driving behavior. Then we calculate the similarities between these benign trajectories and the generated trajectories.

Implementation We use public code repository[1] to calculate the trajectory similarity scores (*Symmetric Segment-Path Distance (SSPD)*, *Fréchet Distance (Fréchet)*, and *Dynamic Time Warping (DTW)*). For action similarities, since it's hard to calculate the distance between two-dimensional distributions efficiently, we adopt 2 different strategies when calculating *Wasserstein Distance (WD)* and *Kullback–Leibler Divergence (KL)*, respectively. For *Wasserstein Distance*, we decouple the accelerations in the two directions $\{acc_x, acc_y\}$. We first calculate the WD scores in the two directions separately, then take the average of the two scores as the final result. For *Kullback–Leibler Divergence*, we assume the distribution of the accelerations is a multivariate Gaussian distribution. We then calculate the approximate result as the KL between two multivariate Gaussian distributions.

[1]Publicly available at `https://github.com/bguillouet/traj-dist`

Table 4: **Evaluation in multi-agent scenarios.** We use `DiffScene` to generate multi-agent scenarios and compared them with previous results in *Crossing Negotiation*. We report the *Collision Rate (CR)*, *Incomplete Route (IR)*, and *Speed Satisfaction (SS)*. We include the averaged score and standard deviation of the evaluation results on 3 different target AV algorithms. (MA: multi-agent. All scores are the higher the better).

Metric	`DiffScene`	`DiffScene-MA`
Collision Rate	0.85 ± 0.08	**0.90** ± 0.08
Incomplete Route	0.39 ± 0.05	**0.43** ± 0.07
Speed Satisfaction	**0.43** ± 0.01	0.41 ± 0.02

C. `DiffScene` model details

We adopt a U-Net type architecture with one-dimensional temporal convolutions as our diffusion model which shows better performance in sequence-based diffusion models (Janner et al., 2022; Zhong et al., 2022). The maximum length of each trajectory is $T = 32$, and the total number of diffusion steps is $K = 100$.

For the adversarial objective model $\mathcal{J}_\phi$, we adopt similar architecture but modify the output layer to output only one value. In our experiments, we use the same weight for different objectives: $\omega_s = \omega_f = \omega_c = 1$. The weight λ to calculate the safety-based objective is set to $\lambda = 5$. The common driving speed of a vehicle in SafeBench is $v^* = 8$.

D. Additional experimental results

D.1. Quantitative results

We increase the scenario complexity and use `DiffScene` to generate safety-critical scenarios under the multi-agent setting. Results are shown in Table 4. In the multi-agent setting, `DiffScene` achieves higher *collision rate* and *incomplete route* than single-agent setting *Crossing Negotiation*, demonstrating that `DiffScene` can generalize well into the multi-agent setting.

D.2. Qualitative results

We provide qualitative results in Figure 5. For each scenario generation algorithm, we show two examples of the generated scenarios in two different scenario settings. Results show that `DiffScene` is more effective in optimizing the trajectory of the surrounding vehicle and generating safety-critical scenarios. Due to the space limit, we provide more qualitative results at this URL.

D.3. Impact of the number of adversarial optimization steps

We generated safety-critical scenarios using different number of adversarial optimization steps N, and evaluated the *collision rate* and *DTW* of the generated scenarios. The results are shown in Figure 6. Similar to *SSPD* and *Fréchet*, we find that when with larger N, the collision rate of the scenarios will be higher, and the DTW score will also be larger, which means the generated scenarios will have larger naturalness cost.

E. Limitations and potential negative societal impacts

E.1. Limitations

Although simulation is a useful tool for evaluating the effectiveness of scenario generation algorithms, it cannot exactly reflect real-world conditions. Real-world data and on-track testing are necessary before using `DiffScene` in real-world applications.

E.2. Potential negative societal impacts

As we will open-source our framework, attackers may leverage our code and data to perform real-world adversarial attacks against existing AV systems. We suggest evaluating the safety and robustness of AV systems in various scenarios before

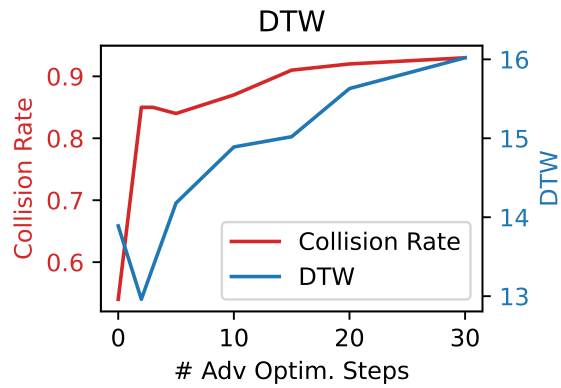


Figure 6: Effects of adversarial optimization steps. We show *Collision Rate* and *DTW* of the scenarios generated by DiffScene under different numbers of adversarial optimization steps. The *Collision Rate* is the higher the better. The *DTW* is the lower the better.

deploying them to the real world. Our generated scenarios can also be used to finetune existing AV algorithms to further improve safety and reliability.