MTD: Generating Realistic and Diverse Safety-Critical Scenarios for Autonomous Vehicles

Yuze Wang[1]
[1]University of Electronic Science and Technology of China, China
[2]Yangtze Delta Region Institute(Quzhou), University of Electronic Science and Technology of China, China
{yzwang}@std.uestc.edu.cn

***Abstract*—Autonomous driving is an emerging technology that has developed rapidly over the past decade. In the process of developing autonomous vehicles, evaluating and improving the performance of driving planners requires a large number of realistic and adversarial safety-critical scenarios, such as collisions. However, due to the long-tail effect, sparsity, and low frequency of real-world datasets, obtaining these scenarios is not easy. To address this issue, we propose a method for automatically generating safety-critical scenarios using Motion Transformer Diffusion (MTD). MTD is designed to generate stable, realistic, and diverse safety-critical scenarios along with their corresponding expert solutions. To ensure the realism of the generated scenarios, we designed a Diffusion-based traffic generation model that enhances agent interaction, temporal understanding, and reduces dependency on original trajectories. To make the generated scenarios more adversarial, we developed an adversarial hybrid guidance function, which takes into account adversarial, stability, and realism factors to guide adversary vehicles into collisions. We further designed an expert hybrid guidance function aimed at finding expert solutions within the generated scenarios to help improve data-driven planners. Extensive experimental results demonstrate that MTD advances the state-of-the-art in generating stable, realistic, and diverse safety-critical scenarios. Moreover, MTD not only effectively tests the performance of autonomous driving planners but also enhances the training data distribution through the expert trajectories it generates.**

***Index Terms*—Autonomous Vehicles, Scenario Generation, Diffusion Model**

I. Introduction

In recent years, with the rapid advancement of autonomous driving technology, data-driven algorithms have been widely adopted in the fields of autonomous driving planning and control [1]–[3]. These algorithms, leveraging large amounts of real-world driving data, are capable of learning to handle complex and dynamic traffic scenarios, demonstrating remarkable adaptability. Compared to traditional rule-based algorithms, data-driven models can flexibly manage various driving situations, making them the core technology in modern autonomous driving systems [4].

The safety of autonomous driving is one of the most critical factors in assessing the maturity of this technology [5], [6]. In real-world driving, an autonomous system must be capable of handling a wide range of complex and unpredictable scenarios, particularly in sudden and extreme situations, to ensure safety. Data sparsity is one of the major challenges in achieving this level of safety [7], [8]. Although vast amounts of data have been collected to cover common, everyday driving scenarios, safety-critical scenarios that are low-frequency but high-risk remain extremely scarce. This "long-tail effect" [9], [10] implies that autonomous driving systems may struggle to make accurate decisions when faced with these rare but crucial events. Therefore, it is essential to obtain a diverse set of safety-critical scenarios. This requirement is crucial not only for data-driven autonomous algorithms but also for traditional rule-based planners, which must be tested in such scenarios to verify their safety [11], [12]. We believe that an ideal set of adversarial safety-critical scenarios should meet two key criteria: (1) the scenarios should be rich, realistic, and diverse, and (2) they should effectively test the actual performance of autonomous driving planners in low-frequency, high-risk situations. End-to-end testing is essential, as it provides a direct measure of the quality and effectiveness of the scenarios, aligning with the practical concerns and expectations of users [13], [14].

Currently, there are two primary methods used to generate these scenarios: sampling from real-world data and scenario generation in simulated environments [15].

The first approach involves sampling safety-critical scenarios from real-world driving data. Studies such as [16]–[20] ensure high realism in the generated scenarios. However, since safety-critical scenarios are extremely rare in everyday driving data and do not cover all potential extreme situations, this approach is inefficient and has notable limitations.

A natural alternative solution is to automatically generate safety-critical scenarios in simulated environments, enabling the efficient generation of diverse scenarios. Although significant progress has been made in recent studies, there are still three major limitations. These limitations affect the realism, diversity of the generated scenarios, as well as their role in the actual testing and performance improvement of autonomous driving planners.

L1: Neglecting dynamic interactions in complex traffic environments leads to unrealistic and unreasonable scenario generation. For example, methods such as [21]–[25] optimize pre-selected adversary trajectories to collide with the ego agent in environments with a limited number of agents. However, in complex traffic settings, adversary agents may prematurely collide with other non-target agents, making it difficult to generate the intended adversarial scenarios.

✉Corresponding authors:

L2: Dependency on original trajectories limits scenario diversity. For instance, methods like [26], [27] impose adversarial constraints on original trajectories to trigger collisions and generate target scenarios. While these methods ensure realism, they exhibit certain limitations when it comes to generating more diverse safety-critical scenarios within the same environment. This falls short of the previously mentioned criteria of generating both realistic and diverse scenarios. Moreover, they cannot function without the presence of original trajectories.

L3: Lack of practical application of generated scenarios for testing or improving autonomous planner performance. For example, studies like [21], [28]–[30] do not sufficiently assess the real impact of these scenarios on autonomous planner performance, which users actually care about. Moreover, these studies fail to validate their contribution to improving planner performance, limiting the practical value of these methods in optimizing real-world autonomous driving systems.

To address these three limitations, we propose Motion Transformer Diffusion (MTD), a query-centric multi-agent traffic generation model for safety-critical scenarios. MTD integrates adversarial guidance and data-driven generation, offering improvements in handling dynamic interactions, enhancing scenario diversity, and boosting the performance of autonomous driving planners. Below, we detail how MTD addresses these three limitations.

M1: Improvements in dynamic interactions within complex traffic environments.

We designed the Social Transformer and Decoder to enhance agent interaction and temporal understanding. For guidance, our hybrid loss function includes a Collision Guidance, which helps prevent collisions with non-target agents. Both measures improve the generation of safety-critical scenarios in complex traffic environments.

M2: Overcoming dependence on original trajectories for scenario generation. MTD, a Diffusion-based model, generates future agent trajectories using environmental conditions (initial states, attributes, and map) without relying on original trajectories. This allows us to generate more diverse safety-critical scenarios, improving coverage of sparse cases.

M3: Practical application of adversarial scenario generation for enhancing and testing autonomous driving planners. Our method not only generates safety-critical scenarios but also provides expert trajectories to handle these scenarios. By using these expert trajectories to fine-tune existing pre-trained models, we assess their impact on improving planner performance. Additionally, we evaluate the actual performance of existing planners in these generated scenarios, offering a more comprehensive assessment of the effectiveness of the generated scenarios.

With these improvements, MTD effectively overcomes existing limitations, achieving notable advancements in generating safety-critical scenarios and optimizing autonomous driving planner performance. In summary, our contributions are as follows:

- We propose Motion Transformer Diffusion (MTD), a query-centric multi-agent traffic generation model designed specifically to generate safety-critical scenarios in complex and dynamic traffic environments.
- We introduce two hybrid guidance functions to optimize MTD's generation process: one for generating safety-critical scenarios, and another for providing expert solutions for these scenarios, effectively combining scenario generation with responsive strategies.
- We use the expert solutions to fine-tune existing autonomous driving planners, significantly enhancing their performance and robustness in handling safety-critical scenarios.
- We evaluate and demonstrate the performance of the generated safety-critical scenarios and their corresponding expert trajectories, conducting a clustering analysis on the types of collisions within these scenarios. Additionally, we experimentally verify the value of these scenarios in testing autonomous driving planners and enhancing their performance under adversarial conditions.

II. Preliminaries

A. Problem Definition

We consider an interactive traffic environment where an ego agent V_1 and a set of surrounding agents $\{V_2, \cdots, V_N\}$ are driving on a complex multi-lane road. The ego agent can obtain the states of the surrounding agents (i.e., position, heading angle, and speed, etc.) through its sensors and make decisions at each time step t over the target time horizon T of interest.

Agent. The set of agents $\mathcal{V} = \{V_1, V_2, \cdots, V_N\}$ consists of N agents.

Action. The actions of the agents $\mathcal{V}$ at time t are represented by $m^t = [m_1^t, m_2^t, \cdots, m_N^t]$, where the action $m = [a, \omega]^T$ includes acceleration a and angular velocity ω.

State. Let the state of the agent set $\mathcal{V}$ at time step t be $s^t = [s_1^t, s_2^t, \cdots, s_N^t]$, where the state $s = [x, y, \theta, \nu]^T$ includes the 2D coordinates (x, y), heading angle θ, and speed ν. At time step t, the next state s^{t+1} of agents $\mathcal{V}$ is computed from the current state s^t and current action m^t through the dynamics model f, denoted as $s^{t+1} = f(s^t, m^t)$.

Attribute. Each agent $\mathcal{V}$ also has attributes $Q = [l, w]$, where l and w represent the length and width, respectively.

Vector Map. The local vector map of the agent set $\mathcal{V}$ at time step t is represented as $\mathcal{I}^t \in \mathbb{R}^{N \times Z \times P \times R}$, where Z is the number of polylines in the map, P is the number of points on each polyline, and R is the number of attributes per point (position and angle).

Decision-making Context. The decision-making context of the agents is represented as $C = \left\{\mathcal{S}^{t-T_h:t}, Q, \mathcal{I}^t\right\}$, including the historical states of the agent set $\mathcal{V}$ over the past T_h time steps, $\mathcal{S}^{t-T_h:t} = \left\{s^{t-T_h}, s^{t-T_h+1}, \cdots, s^t\right\}$, their attributes Q, and the local vector map $\mathcal{I}^t$.

Time Step. To simplify the problem, we consider the continuous time horizon as discrete time steps, with a total length of $T_h + T$, where T_h is the historical length, and T is the length

to be generated. The time interval between two consecutive time steps is denoted as Δt, which is set to 0.1 seconds.

Trajectory. The generated trajectory τ includes action and state sequences, $\tau = [\tau_a, \tau_s]^T$, where $\tau_a = \left[m^0, \cdots, m^{T-1}\right]^T$ is the action sequence, and $\tau_s = \left[s^1, \cdots, s^T\right]^T$ is the state sequence. Our model generates only the action trajectory τ_a, and to ensure physical feasibility, the state trajectory τ_s is derived from τ_a: $\tau_s = f(s^0, \tau_a)$.

Objective. Our objective is twofold: first, to select appropriate surrounding agents as adversaries and generate a trajectory that leads to a collision with the ego agent, thus forming a safety-critical scenario. Second, to generate a feasible trajectory for the ego agent in this scenario to avoid the collision, which we call the expert trajectory.

B. Conditional Diffusion Probabilistic Models

Diffusion probabilistic models are deep generative models that have achieved state-of-the-art results in fields such as image and video generation [31]–[35]. They generate samples consistent with the original data distribution by adding noise to the samples and learning the reverse denoising process. Diffusion probabilistic models can be formalized as two Markov chain processes [36] of length K, referred to as the diffusion process and the reverse process. Conditional diffusion probabilistic models are similar to unconditional diffusion probabilistic models but consider conditional information C in each step of the diffusion process. Let $\tau^0 \sim p$, where p is the probability distribution of the model trajectory, and τ^k is the sampled latent variable trajectory, where $k = 1, \cdots, K$ are the diffusion steps. $\tau^K \sim \mathcal{N}(0, I)$, where $\mathcal{N}$ denotes the Gaussian distribution. The diffusion process gradually adds Gaussian noise to τ^0 until τ^0 approaches τ^K, while the reverse process denoises τ^K considering the conditional information C to recover τ^0.

III. Methodology

We first introduce the overall framework of our method in Section III-A. In Section III-B, we describe in detail the architecture of MTD and its training process. In Section III-C, we explain the inference stage of MTD, where adversarial agents are selected and guided using a hybrid loss function to collide with the ego agent, generating safety-critical scenarios. In Section III-D, we guide the ego agent to avoid collisions and generate solutions for safety-critical scenarios, namely expert trajectories. In Section III-E, we apply these expert trajectories to fine-tune the autonomous driving planner, improving its safety performance when handling safety-critical scenarios.

A. Overview

Figure 1 presents the overall framework of the proposed MTD, which consists of four components: traffic model, safety-critical scenario generation guidance, expert trajectory optimization guidance, and planner enhancement. MTD generates safety-critical scenarios and expert trajectories to address these scenarios, providing more challenging training data for autonomous driving planners. Additionally, the generated

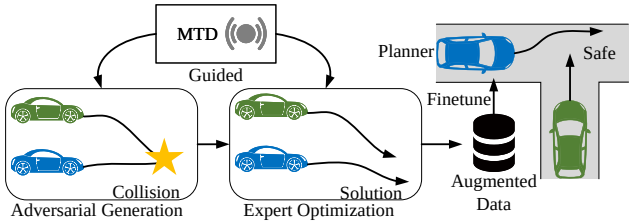

Fig. 1. Overview

safety-critical scenarios are used to evaluate the performance of autonomous driving planners under extreme conditions.

We propose a query-centric multi-agent traffic generation model based on diffusion and trained on real-world datasets. During the inference phase, this model generates realistic and diverse traffic scenarios by inputting initial agent states, attributes, and map information.

For safety-critical scenario generation guidance, we propose a hybrid loss function composed of adversarial, environmental, collision, and initialization guidance. This function ensures the stable and realistic generation of safety-critical scenarios during the inference phase. Specifically, adversarial guidance increases the likelihood of the adversarial agent colliding with the ego agent; environmental guidance penalizes agents that enter non-drivable areas; collision guidance ensures that only the adversary and ego agent collide, preventing other agents from crashing; and initialization loss minimizes deviations from the original trajectory, ensuring the generated trajectories remain as realistic as possible.

For expert trajectory optimization guidance, we propose another hybrid loss function, composed of original, environmental, collision, and initialization guidance, to find solutions—i.e., expert trajectories—within the generated safety-critical scenarios. The original guidance ensures that non-ego agents follow their trajectories from the adversarial scenarios, while the ego agent's trajectory is re-generated to find a stable and realistic expert solution.

For planner enhancement, we design a workflow to evaluate the practical performance of the generated safety-critical scenarios and expert trajectories. Specifically, the generated expert trajectories are used to fine-tune pre-trained planners, allowing us to test and evaluate the improvements in the planner's ability to handle safety-critical scenarios.

B. Conditional Motion Transformer Diffusion Model

In our approach, trajectories are generated through an iterative denoising process, which is learned by reversing a predefined diffusion process. As stated in Section II-A, the state sequence τ^s is derived from the action sequence τ^a through the dynamics model f, expressed as $\tau^s = f(s^0, \tau^a)$. The trajectory input to the model is $\tau = [\tau^a, \tau^s]^T$. Starting with a clean trajectory sampled from the data distribution, $\tau^0 \sim q(\tau^0)$, the forward noising process introduces Gaussian noise at each step k, generating a sequence of trajectories with progressively increasing noise $(\tau^0, \tau^1, \cdots, \tau^K)$:

$$q(\tau^{1:K}|\tau^0) := \prod_{k=1}^{K} q(\tau^k|\tau^{k-1}) \tag{1}$$

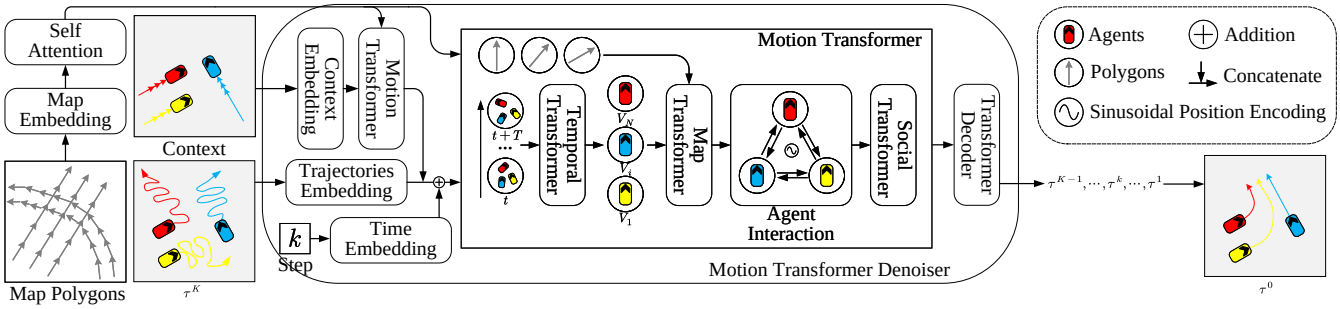

Fig. 2. Denoiser Architectural

$$q(\tau^k|\tau^{k-1}) := \mathcal{N}(\tau^k; \sqrt{1-\beta_k}\tau^{k-1}, \beta_k I) \tag{2}$$

where β_k is the variance at each step, and with sufficiently large K, we approximate $q(\tau^K) \approx \mathcal{N}(\tau^K; 0, I)$.

MTD learns this reverse process, allowing noisy samples to be denoised back into reasonable trajectories. Each step of this reverse diffusion process is conditioned on the agent's decision-making context C:

$$p_\theta(\tau^{0:K}|C) := p(\tau^K) \prod_{k=1}^{K} p_\theta(\tau^{k-1}|\tau^k, C) \tag{3}$$

$$p_\theta(\tau^{k-1}|\tau^k, C) := \mathcal{N}(\tau^{k-1}; \mu_\theta(\tau^k, k, C), \Sigma_\theta(\tau^k, k, C)) \tag{4}$$

where θ are the model parameters. According to [31]–[33], the Gaussian transition variance is fixed as $\Sigma_\theta(\tau^k, k, C) = \Sigma_k = \sigma_k^2 I = \beta_k I$.

A key distinction of our model is that our traffic model is built around query-centric sampling, capable of generating the trajectories of all agents in the scene without extensive preprocessing or coordinate transformations. The architecture of MTD will be detailed further below.

Denoiser Architecture. The structure of the MTD is depicted in Figure 2. Initially, we embed the agent's decision-making context and map its feature dimensions to d dimensions, processed through L_c stacks of the Motion Transformer, resulting in the transformed features $F'_C \in \mathbb{R}^{T_h \times d}$. Additionally, we process the map polygons by embedding them into d dimensions, denoted as $F_M \in \mathbb{R}^{L \times d}$. We utilize a multi-head self-attention mechanism to extract features among the map polygons, helping the agent effectively leverage localized vectorized maps. The self-attention mechanism for map feature encoding can be represented as:

$$F'_M[i] = \mathrm{MHA}\left(\begin{array}{l} \mathrm{Q}: [F_M[i] + \mathrm{PE}] \\ \mathrm{K}: \{[F_M[j] + \mathrm{PE}]\}_{j \in \Omega(i)} \\ \mathrm{V}: \{[F_M[j] + \mathrm{PE}]\}_{j \in \Omega(i)} \end{array}\right) \tag{5}$$

where $i \in \{1, 2, \cdots, L\}$, $\Omega(i)$ is the set of map features i needs to process. $\mathrm{MHA}(\cdot)$ denotes the multi-head attention mechanism. PE represents the sinusoidal encoding [37], [38] for processing the j-th map feature. $F'_M[i] \in \mathbb{R}^d$ is the output feature of the map self-attention mechanism.

We encode the noisy trajectory as $F_\tau \in \mathbb{R}^{T \times d}$ and concatenate it with the agent's decision-making context. Additionally, we combine the result with the encoded denoising step features $F_t \in \mathbb{R}^{(T_h+T) \times d}$, represented as:

$$F_A = \mathrm{Concat}(F_\tau, F'_C) + F_t \tag{6}$$

where $\mathrm{Concat}(\cdot)$ denotes the concatenation operation. The resulting feature $F_A \in \mathbb{R}^{(T_h+T) \times d}$ is input to L_{enc} layers of stacked Motion Transformer, outputting $F'''_A \in \mathbb{R}^{(T_h+T) \times d}$. Finally, through a standard Transformer decoder followed by a three-layer MLP, the denoised trajectory τ^{k-1} is obtained. Specifically, our Transformer decoder's multi-head cross-attention mechanism can be represented as:

$$F_{out}[t] = \mathrm{MHA}\left(\begin{array}{l} \mathrm{Q}: [F'''_A[t] + \mathrm{PE}] \\ \mathrm{K}: \{[F'''_A[\iota] + \mathrm{PE}]\}_{\iota \in \Omega(t)} \\ \mathrm{V}: \{[F'''_A[\iota] + \mathrm{PE}]\}_{\iota \in \Omega(t)} \end{array}\right) \tag{7}$$

where $t \in \{T_h + 1, T_h + 2, \cdots, T_h + T\}$, $\Omega(t)$ is the set of time features that need to be processed at time step t, with the number of elements in $\Omega(t)$ being $T_h + T$. PE represents the sinusoidal encoding for processing the t-th time feature. $F_{out} \in \mathbb{R}^{T \times d}$ is the output of the Transformer decoder, and finally, we output the denoised trajectory τ^{k-1} through a three-layer MLP:

$$\tau^{k-1} = f(s^0, \mathrm{MLP}(F_{out})) \tag{8}$$

Motion Transformer. Figure 2 also presents the detailed structure of the Motion Transformer network, which is composed of three parts: Temporal Transformer, Map Transformer, and Social Transformer. Our network architecture is entirely Transformer-based, similar to the reference, but we have made the following modifications: (1) Traditional methods, such as [39], [40], rely on a global coordinate system centered around a single focal agent, while we use a query-centric approach. This models the relationships between all tokens symmetrically, decoupling from any global coordinate system. Since the relative relationships between agents are symmetric, the query-centric modeling avoids a significant amount of redundant computation, further enabling parallel multi-agent traffic simulation. (2) Inspired by [41], [42], we use a position encoder to apply sinusoidal encoding to the relative relationships between agents, making the similarity of relative positions more consistent.

Temporal Transformer. The Temporal Transformer module analyzes encoded trajectories to effectively capture and understand dynamic interactions and relationships among agents over time. This process provides crucial temporal information

for traffic simulation, enabling the system to accurately model and respond to changes in complex environments. Specifically, the self-attention mechanism at the encoding layer of the Temporal Transformer can be represented as:

$$F'_A[t] = \text{MHA}\left(\begin{array}{l}\text{Q}: [F_A[t] + \text{PE}] \\ \text{K}: \{[F_A[\iota] + \text{PE}]\}_{\iota \in \Omega(t)} \\ \text{V}: \{[F_A[\iota] + \text{PE}]\}_{\iota \in \Omega(t)}\end{array}\right) \tag{9}$$

where $t \in \{1, 2, \cdots, T_h + T\}$, $\Omega(t)$ is the set of time features that need processing at time step t. $\text{MHA}(\cdot)$ denotes the multi-head attention mechanism. PE represents sinusoidal encoding for processing the t-th time feature. $F'_A[t] \in \mathbb{R}^d$ is the output feature of the Temporal Transformer.

Map Transformer. The Map Transformer's attention mechanism employs a cross multi-head attention mechanism, using the central agent's features as queries and local vectorized map features as keys and values. Map Transformer aims to enable agents to capture surrounding map environments to ensure driving within feasible areas. Specifically, the cross-attention mechanism at the encoding layer of the Map Transformer can be represented as:

$$F''_A[i] = \text{MHA}\left(\begin{array}{l}\text{Q}: F'_A[i] \\ \text{K}: \{[F'_M[j] + \text{PE}]\}_{j \in \Omega(i)} \\ \text{V}: \{[F'_M[j] + \text{PE}]\}_{j \in \Omega(i)}\end{array}\right) \tag{10}$$

where $i \in \{1, 2, \cdots, N\}$, $\Omega(i)$ is the set of map features that need processing at agent V_i. PE represents sinusoidal encoding for processing the j-th map feature. $F''_A[i] \in \mathbb{R}^d$ is the output feature of the Map Transformer.

Social Transformer. Traditional methods rely on agent-centric coordinate systems for inter-agent interactions, significantly impacting the performance of multi-agent traffic models. To address this, we use a query-centric self-attention mechanism to interact with relative agent relationships rather than absolute ones. By leveraging the symmetry and repeatability of these relationships, we avoid redundant calculations and enhance model performance [43], [44]. Additionally, we use position encoder to sinusoidally encode relative agent relationships, ensuring consistency in relative position similarity and avoiding unnecessary dimensional expansion.

To explore the relative relationships between the query agent and other agents in the scene, we transform the coordinate systems of other agents into the query agent's coordinate system using a transformation matrix:

$$R^{pos}[i,j] = (P[j] - P[i]) \begin{bmatrix}\cos H[i] & -\sin H[i] \\ \sin H[i] & \cos H[i]\end{bmatrix} \tag{11}$$

$$R^{\theta}[i,j] = H[j] - H[i] \tag{12}$$

$$R^{\nu}[i,j] = (\nu[j] - \nu[i]) \begin{bmatrix}\cos H[i] & -\sin H[i] \\ \sin H[i] & \cos H[i]\end{bmatrix} \tag{13}$$

$$R[i,j] = \left[R^{pos}[i,j], R^{\theta}[i,j], R^{\nu}[i,j]\right] \tag{14}$$

where $R^{pos}[i,j]$, $R^{\theta}[i,j]$, and $R^{\nu}[i,j]$ represent the 2D relative position, relative heading angle, and relative velocity between agent V_i and agent V_j, respectively. $P[i]$, $H[i]$, and $\nu[i]$ denote the 2D coordinates, heading angle, and velocity of agent V_i in its own coordinate system.

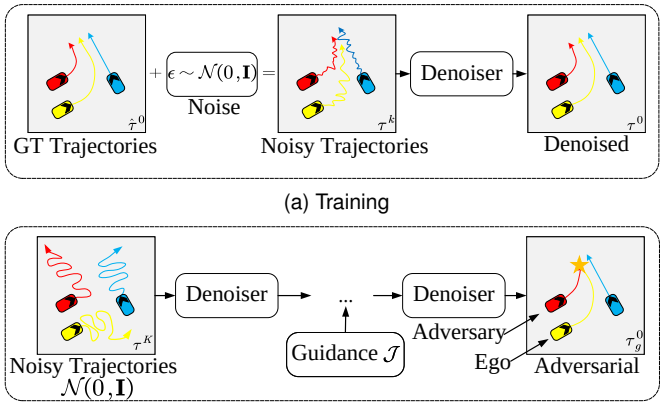

Fig. 3. Training and Sampling

The cross-attention mechanism at the encoding layer of the Social Transformer can be represented as:

$$F'''_A[i] = \text{MHA}\left(\begin{array}{l}\text{Q}: F''_A[i] \\ \text{K}: \{[F''_M[j] + \text{PE}(R[i,j])]\}_{j \in \Omega(i)} \\ \text{V}: \{[F''_M[j] + \text{PE}(R[i,j])]\}_{j \in \Omega(i)}\end{array}\right) \tag{15}$$

where $i \in \{1, 2, \cdots, N\}$, $\Omega(i)$ is the set of other agent features that need processing at agent V_i. $\text{PE}(\cdot)$ represents sinusoidal encoding using the relative position inside parentheses to ensure consistency in relative position similarity [41]. $F'''_A[i] \in \mathbb{R}^d$ is the output feature of the Social Transformer.

After passing through the Temporal, Map, and Social Transformers, we keep the input and output feature dimensions unchanged, allowing us to stack these Transformer structures effectively for enhanced interactions among agents, maps, and agents in traffic scenarios.

Training. During the MTD training phase, the network directly predicts the mean μ of the clean trajectory τ^0, instead of predicting the next noisy trajectory τ^{k-1}. We uniformly sample the denoising step $k = \mathcal{U}(1, K)$ from denoising steps $[1, K]$. We directly add the noise loaded at step τ^k to the real trajectory τ^0 using the formula $\tau^k = \sqrt{\bar{a}_k}\tau^0 + \sqrt{1-\bar{a}_k}\epsilon$, where $\epsilon \sim \mathcal{N}(0, \text{I})$ and $\bar{a}_k = \prod_{l=0}^{k}(1-\beta_l)$. We denote the model's direct output as $\hat{\tau}^0 = D_\theta(\tau^k, k, C)$, as shown in Figure 3a.

Finally, the supervised training loss function is given by:

$$\mathcal{L}_{loss} = \mathbb{E}_{\epsilon,k,\tau^0,C}\left\|\tau^0 - \hat{\tau}^0\right\|^2 \tag{16}$$

C. Safety-Critical Scenario Generation

The purpose of utilizing MTD is to generate safety-critical scenarios that do not exist in the original dataset. Through these scenarios, experts (Section III-D) can identify solutions to tackle adversarial challenges and utilize these expert trajectories to improve the performance of existing planners.

To generate safety-critical scenarios, we incorporate a hybrid guidance function during the MTD inference phase. This function directs selected adversarial agents to collide with the ego agent, thereby creating safety-critical scenarios.

Guided Sampling. Traditional methods for generating safety-critical scenarios typically fall into two categories: one involves generating trajectories using traffic priors and then guiding them with a guidance function; the other directly uses trajectories from datasets, guided by a guidance function. Our approach is distinct in that it incorporates a guidance function into each denoising step of the MTD inference phase, ensuring both the stability and authenticity of the generated safety-critical scenarios. Additionally, through multiple sampling, the inherent randomness of noise allows MTD to generate a diverse set of safety-critical scenarios. A schematic of the guided sampling is shown in Figure 3b.

Inspired by [45]–[47], we have reformulated the reverse diffusion process as:

$$p_\theta\left(\tau_g^{k-1}|\tau_g^k, C\right) := \mathcal{N}\left(\tau_g^{k-1}; \mu + \Sigma\nabla\mathcal{J}(\mu), \Sigma\right) \tag{17}$$

where for simplicity, $\mu = \mu_\theta$ and $\Sigma = \Sigma_\theta$. $\mathcal{J}$ is referred to as the guidance function. τ_g^k denotes the denoised trajectory at step k generated after applying the guidance function $\mathcal{J}$.

In practice, when generating trajectories, we guide sampling from multiple samples produced by MTD and, at the end of the denoising process, select the final output trajectory based on the minimum value of the guidance function $\mathcal{J}$. This method ensures that the generated trajectories more closely align with the guidance requirements, thereby enhancing the quality and reliability of the sample trajectories.

Guidance Function. When generating safety-critical scenarios, our goal is to create situations where an adversarial agent collides with the ego agent. To ensure the rationality and effectiveness of safety-critical scenarios, we do not choose adversaries that are always behind the ego agent. Specifically, our criterion for selecting adversaries is defined as:

$$(i^*, t^*) = \arg\min_{i,t} \|P_i^t - P_0^t\| \tag{18}$$

where the adversarial agent is denoted as V_{i^*} and the ego agent as V_0. P_i^t represents the global coordinates of agent V_i at time step t, and t^* is the time step when the adversarial agent V_{i^*} is closest.

Our guidance function consists of Adversarial Guidance, Environmental Guidance, Collision Guidance, and Initialization Guidance. Thus, our optimization objective is:

$$\min_\mu \left(\mathcal{J}_{adv} + \mathcal{J}_{env} + \mathcal{J}_{coll} + \mathcal{J}_{init}\right) \tag{19}$$

Each guidance function has a corresponding weight, denoted as w_{adv}, w_{env}, w_{coll}, and w_{init}, and we use the Adam optimizer to optimize the minimum objective function o times.

Adversarial Guidance. Adversarial guidance aims to make the selected adversary more likely to collide with the ego agent by minimizing the distance at each time step t in two-dimensional position. Specifically, the adversarial guidance function is defined as:

$$\mathcal{J}_{adv} = \sum_t \xi^t \|P_{i^*}^t - P_0^t\|^2 \tag{20}$$

$$\xi^t = \frac{(-\|P_{i^*}^t - P_0^t\|)}{\sum_t \exp(-\|P_{i^*}^t - P_0^t\|)} \tag{21}$$

where ξ^t is defined as the softmin of the two-dimensional distance between the adversarial agent V_{i^*} and the self-driving agent V_0 at time step t.

Environmental Guidance. Environmental guidance penalizes agents for driving in non-drivable areas. It detects this by checking the overlap between gridded non-drivable map layers and agent boundaries [48]. The collision point is assumed at point c, the average pixel value of the overlapping area. Specifically, the environmental loss can be represented as:

$$\mathcal{L}_{env}(i) = \begin{cases} 1 - \frac{d}{r_i} & \text{if overlap} \\ 0 & \text{otherwise} \end{cases} \tag{22}$$

where d is the distance between the collision point c and the agent center, and r_i is half of the diagonal of agent V_i's bounding box.

The environmental guidance is the sum of the environmental losses of all agents under time decay:

$$\mathcal{J}_{coll} = \frac{1}{N}\sum_i \sum_t \gamma^t \mathcal{L}_{env}(i) \tag{23}$$

where γ is a decay factor.

Collision Guidance. The goal of collision guidance is to ensure that only the adversary and the ego agent collide in the generated safety-critical scenarios, avoiding collisions between other agents. Collision guidance ensures that other agents maintain a safe distance during trajectory generation, thereby preventing collisions. Specifically, we use pairwise collision loss and efficient differentiable relaxation to simplify optimization. We approximate each agent with two circles and calculate the L2 distance between the nearest centers of each pair of agents. It can be represented as:

$$\mathcal{L}_{coll}(i,j,t) = \begin{cases} 1 - \frac{\|P_i^t - P_j^t\|}{r_i + r_j} & \|P_i^t - P_j^t\| \leqslant r_i + r_j \\ 0 & \text{otherwise} \end{cases} \tag{24}$$

where r_i represents the radius of agent V_i's approximating circle.

The collision guidance is the sum of collision losses of all agents under time decay:

$$\mathcal{J}_{coll} = \frac{1}{N^2}\sum_{i\neq j} \sum_t \gamma^t \mathcal{L}_{coll}(i,j,t) \tag{25}$$

To not compute the collision guidance between the adversary and the self-driving agent, we use masking to shield the corresponding values in experiments.

Initialization Guidance. The setup for initialization loss aims to minimize the impact of the guiding trajectory on the original trajectory when generating safety-critical scenarios. This setup ensures that the naturalness and rationality of the original trajectory are preserved to the greatest extent during the creation of adversarial scenes. Specifically, the initialization loss is represented as:

$$\mathcal{J}_{init} = \frac{1}{N}\sum_i \sum_t \|P_i^t - P_{i_{init}}^t\|^2 \tag{26}$$

where $P_{i_{init}}^t$ is the global coordinate of agent V_i at the initial state.

D. Expert Trajectory Optimization

Unlike the generation of safety-critical scenarios, the purpose of expert trajectory optimization is to generate solution trajectories for the ego agent in collision scenarios, focusing on optimizing the performance of the ego agent. In contrast, safety-critical scenario generation enhances the behavior of the adversarial agents by causing them to collide with the self-driving agent. The key difference is that expert trajectory optimization targets the ego agent, while safety-critical scenario generation targets the adversarial agents.

Specifically, expert trajectory optimization guides the MTD to generate solution trajectories for the ego agent, while other agents continue to follow their original trajectories from the safety-critical scenarios. This method ensures that optimized trajectories for the ego agent are obtained in safety-critical scenarios, thus enhancing its capability to handle such scenarios.

The guidance function for expert trajectory optimization consists of Original Guidance, Collision Guidance, Environmental Guidance, and Initialization Guidance, with the optimization objective defined as:

$$\min_{\mu} \left(\mathcal{J}_{ori} + \mathcal{J}_{ego} + \mathcal{J}_{env} + \mathcal{J}_{init} \right) \tag{27}$$

Similarly, each guidance function has corresponding weights, denoted as w_{ori}, w_{ego}, w_{env}, and w_{init}, and we use the Adam optimizer to optimize the minimum objective function o times. The definition of $\mathcal{J}_{ego}$ is similar to $\mathcal{J}_{coll}$ from Section III-C, but $\mathcal{J}_{ego}$ calculates only the collision loss between the ego agent and other agents, not among all agents, and it carries a greater weight. $\mathcal{J}_{env}$ and $\mathcal{J}_{init}$ are similar to their counterparts in Section III-C but are only directed towards the ego agent, while other agents are guided by $\mathcal{J}_{ori}$.

Original Guidance. Original guidance directs non-ego agents to follow the trajectories from the original safety-critical scenarios. Specifically, it is defined as:

$$\mathcal{J}_{ori} = \frac{1}{N} \sum_{i} \sum_{t} \gamma^t \| P_i^t - P_{i_{adv}}^t \|^2 \tag{28}$$

where $P_{i_{adv}}^t$ is the global coordinate of agent V_i in the safety-critical scenario. Similarly, to not compute original guidance for the ego agent, we use masking in experiments to shield the corresponding values.

E. Planner Enhancement

The ultimate goal of generating safety-critical scenarios is to enhance the planner's ability to respond in such scenarios [49]. To achieve this, we employed the following strategies: (1) Using MTD to generate a large number of safety-critical scenarios. (2) Identifying expert solutions within these adversarial scenarios. (3) Utilizing these solutions to fine-tune the original planner, thereby enhancing its safety and robustness in safety-critical scenarios.

To demonstrate the effectiveness of the MTD-generated safety-critical scenarios and the performance of the expert solution trajectories, we designed a simple yet effective planner. Specifically, we utilize existing trajectory prediction algorithms (e.g., [50]–[52]) to generate future trajectories for the self-driving agent, and then employ a dynamics model [53] to translate these trajectories into actions to drive the motion of the ego agent. Through this approach, we are able to assess the planner's ability and effectiveness in handling safety-critical scenarios after fine-tuning.

IV. Experiments

A. Experimental Settings

Datasets. nuScenes [54] is a widely used dataset in autonomous driving research, covering diverse scenes and dense traffic conditions, providing 5.5 hours of manually annotated trajectories. The dataset contains 1,000 scenes, each consisting of 20-second traffic segments annotated at 2 Hz. We interpolated the nuScenes dataset to increase its sampling frequency to 10 Hz and divided it into training and validation sets. All models were trained on the training set and evaluated on the validation set. Our primary focus is on simulating moving agents as they are most relevant to control.

Simulation Environment. Our simulation environment is initialized based on the positions and historical states of agents in real driving data. Specifically, all scenes are from the validation set, ensuring that these scenes have not been seen by the trained models. The simulation runs at a frequency of 10 Hz. In practical experiments, collisions between agents in safety-critical scenarios typically occur within 10 seconds, so we set the simulation timestep to 10 seconds. To comprehensively evaluate the diversity of generated scenarios and the long-term performance of the model, the traffic simulation duration is set to 20 seconds. All experiments are conducted in a closed-loop simulation environment, with a decision frequency of 2 Hz.

Implementation Details. The MTD model is trained to generate 5.2 seconds of future motion based on 3 seconds of historical motion (both sampled at 10 Hz, with the historical trajectory including the current state). The specific parameters are as follows: the timestep for generating future motion is $T = 52$, the timestep for historical motion is $T_h = 31$, and the diffusion steps are $K = 100$. Regarding the details of the MTD network structure, the number of stacked layers of the Motion Transformer mentioned in Section III-B is $L_c = L_{enc} = 2$, and the feature dimension is $d = 128$. The local vector map covers a 50-meter radius centered on each agent, with $Z = 15$ polylines and $P = 80$ points per polyline, and the attribute dimension is $P = 3$ (including 2D coordinates and polyline orientation). The dynamics model f adopts the unicycle model [55]. During training, we consider the $N = 20$ nearest agents to the sampled agent, while during inference, the number of nearby agents is not limited. We use the AdamW [56] optimizer for training with a learning rate of 0.0001.

Baselines. Due to the different focuses of previous related works, they are not directly comparable. For example, [5], [57] focus only on generating specific scenarios, while [58], [59] attack the entire autonomous driving stack during adversarial scenario generation, rather than just the planning module. Additionally, [60], [61] primarily focus on generating videos of adversarial scenarios. Therefore, we selected several

state-of-the-art safety-critical scenario generation methods for comparison:

• CTG++(Adv) [46]. A diffusion-based method for generating safety-critical scenarios.

• STRIVE [26]. A graph-based CVAE incorporating traffic priors for generating safety-critical scenarios.

Additionally, we conducted evaluations on traffic simulation performance, aiming to assess the diversity of the generated scenarios and the long-term performance of the model. Therefore, in addition to the previously mentioned methods, we also compared our approach with the following advanced traffic simulation methods:

• SimNet [62]. A deterministic behavior cloning approach for traffic simulation.

• TrafficSim [48]. A traffic simulation method utilizing isotropic Gaussian CVAE.

• BITS [63]. A traffic simulation method based on a two-layer imitation learning framework.

B. Evaluation of Safety-Critical Scenario Generation

Metrics. In the evaluation of safety-critical scenario generation and expert trajectory optimization, we primarily focus on the **effectiveness**, **realism**, and **stability** of the motion.

Effectiveness is assessed through the generated collision rate and the expert resolution rate. During the safety-critical scenario generation phase, effectiveness is measured by the generated collision rate, while in the expert trajectory optimization phase, it is evaluated by the expert resolution rate.

(1) Generated Collision Rate (GCR) is determined by calculating the proportion of scenarios where the ego agent collides with the adversary agent, obtained by dividing the number of collision scenarios by the total number of scenarios.

(2) Expert Resolution Rate (ERR) measures the expert's ability to find collision-avoidance trajectories in the generated safety-critical scenarios, calculated by dividing the number of successful expert solutions by the number of collision scenarios.

Realism is evaluated by comparing the statistical driving features of the generated motion with those of real-world driving data. According to [63], this comparison is performed by calculating the Wasserstein distance between the normalized histograms of the driving features of the generated motion and those of real-world motion.

(3) Realism Bias (RB) is defined as the average Wasserstein distance of the distributions for longitudinal acceleration magnitude, lateral acceleration magnitude, and jerk.

Stability is assessed by reporting the failure rate, collision rate, and off-road rate in the scenario. A critical failure is defined as an agent either colliding with another agent or driving off the road for more than one second.

(4) Failure Rate (FR) is the average proportion of agents experiencing critical failures in a scenario.

(5) Collision Rate (CR) is the average proportion of agents colliding with another agent.

(6) Off-road Rate (OR) represents the proportion of timesteps an agent spends outside the drivable area, averaged across all agents in the scenario.

TABLE I
EVALUATION OF SAFETY-CRITICAL SCENARIO GENERATION

Method	GCR ↑	RB ↓	AOR ↓	OFR ↓	OCR ↓
STRIVE	43.00	0.088	**0.00**	9.28	8.79
CTG++(Adv)	71.00	0.190	1.31	6.73	5.92
MTD	**73.00**	**0.060**	0.01	**5.18**	**4.65**

During the safety-critical scenario generation phase, we primarily report the FR, CR, and OR for the adversary agent in relation to other agents. In the expert trajectory optimization phase, the focus shifts to the ego agent, where we evaluate its FR, CR, and OR in relation to other agents.

Table I presents the quantitative results for safety-critical scenario generation. We compared the performance of various methods on metrics such as GCR, RB, Adv OR (AOR), Other FR (OFR), and Other CR (OCR). Except for Adv OR, which is only 0.01% lower than STRIVE, GCR, RB, Other FR, and Other CR all outperform the baseline methods, demonstrating the exceptional ability of our approach to generate stable and realistic safety-critical scenarios. One reason for this is that MTD is a generative model trained on real-world datasets, and the output trajectories conform to real-world distributions. Additionally, the inclusion of the Social and Decoder components allows for better handling of complex dynamic traffic environments (see Section IV-G for details). Moreover, our adversarial guidance function is a hybrid loss function, balancing the task of generating adversarial scenarios with ensuring realism and stability. Admittedly, as a generative model, MTD exhibits a certain off-road rate when generating trajectories, whereas STRIVE reduces this issue by matching real-world trajectories from the test dataset before generating safety-critical scenarios. However, STRIVE's reliance on test dataset significantly limits the diversity of its generated scenarios. Across all metrics, CTG++(Adv) performs worse than MTD, likely because MTD is more focused on generating safety-critical scenarios compared to the more general approach of CTG++(Adv). Overall, these performance metrics highlight the ability of our method to generate stable and realistic safety-critical scenarios, as further supported by the visualized results in Figure 4.

In Figure 4, we illustrate the advantages of MTD's adversarial guidance function through two examples. Our adversarial guidance function focuses on optimizing the adversary most likely to collide with the ego agent, unlike STRIVE and CTG++(Adv), which optimize all potential adversaries simultaneously. This approach effectively reduces interference with other agents, leading to lower failure and collision rates in complex scenarios. In Figure 4a, we see that CTG++(Adv)'s Agent 29 and STRIVE's Agent 30 both tend to approach the ego agent, which compromises the realism of the generated scenarios. In Figure 4b, due to STRIVE and CTG++(Adv) optimizing all potential adversaries, STRIVE takes longer to generate adversarial scenarios, and CTG++(Adv) even fails to find a reasonable adversarial solution. MTD successfully avoids these issues. In Figure 4c, we illustrate the correspondence between trajectory color and timesteps. Unless

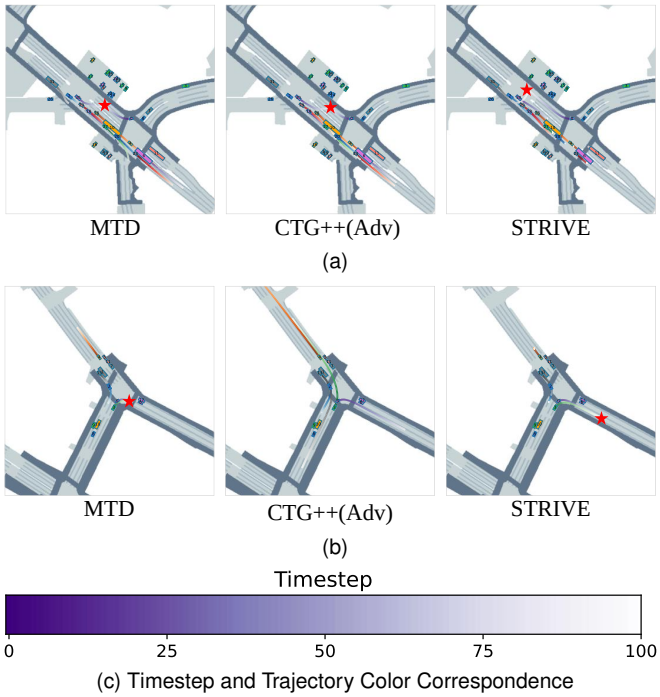

Fig. 4. Comparison of MTD, CTG++(Adv), and STRIVE across Different Scenarios

TABLE II
EVALUATION OF EXPERT TRAJECTORIES GENERATION

Method	EER ↑	RB ↓	EOR ↓	OFR ↓	OCR ↓
STRIVE	48.84	0.075	**0.65**	11.34	11.65
CTG++(Adv)	59.15	0.234	6.54	15.26	14.08
MTD	**75.34**	**0.060**	6.14	**8.29**	**8.17**

otherwise specified, all trajectories plotted in this paper follow this mapping.

Table II presents the quantitative metrics for expert solutions generated in safety-critical scenarios. MTD outperforms the baseline methods in the EER, RB, Ego OR, Other FR (OCR), and Other CR (OCR) metrics, and while its performance on the Ego OR metric is better than CTG++(Adv), it is slightly below STRIVE. This demonstrates that our method not only generates stable and realistic safety-critical scenarios but also finds feasible and realistic expert trajectory solutions within these scenarios. High-quality expert trajectories provide more comprehensive training data for data-driven autonomous driving planners, effectively reducing the issue of data sparsity in safety-critical scenarios.

In Figure 5, we present three types of cases where MTD successfully found expert solutions in the generated safety-critical scenarios. Based on the statistics from [64], we selected the most common cases in daily traffic: from left to right, a cut-in collision, a head-on collision, and a rear-end collision. MTD is able to generate stable and realistic expert solutions for all three types of safety-critical scenarios.

C. Evaluating Adversarial Critical Security Scenarios Generated by the Planner

To further illustrate the effectiveness of the safety-critical scenarios generated by MTD, we evaluated the performance of

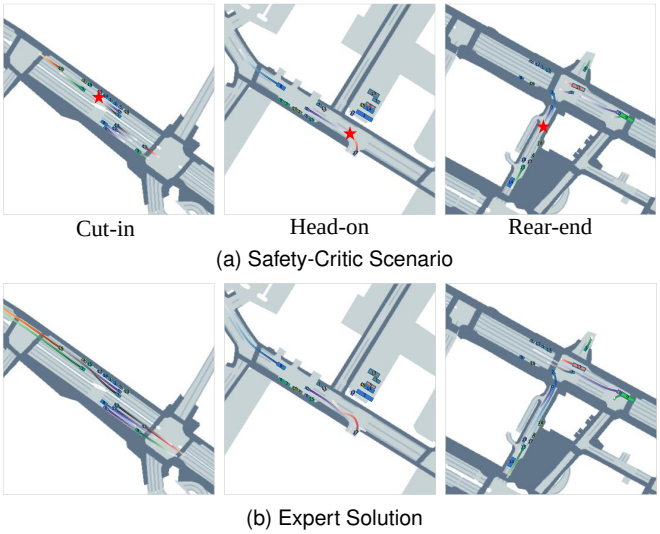

Fig. 5. Visualization of Expert Trajectories in Safety-Critical Scenarios

TABLE III
PLANNER PERFORMANCE IN REG. AND ADV. SCENARIOS

Planner	Scenario	PCR ↓	OR ↓	AV ↑	Jerk ↓
Pre-Trained	Regular	20.00	3.54	4.77	1.05
	Adversarial	44.00	2.34	4.75	1.04
Fine-Tune	Regular	19.00	9.70	6.04	0.65
	Adversarial	39.00	8.85	6.20	0.62

the autonomous driving planner AutoBot [50] in both safety-critical scenarios generated by MTD and regular original scenarios. We measured the effectiveness by recording the planner's collision rate. **Planner's collision rate (PCR)** is defined as the number of collision scenarios divided by the total number of test scenarios, similar to the GCR metric described in Section IV-B.

In Table III, the Pre-Trained row shows the test performance of AutoBot in both regular and adversarial scenarios. In the safety-critical scenarios generated by MTD, the Pre-Trained AutoBot exhibited a 24.00% increase in PCR, indicating that the scenarios generated by MTD effectively test the safety performance of the planner in extreme environments. Compared to the GCR during scenario generation, the planner's PCR in adversarial scenarios was significantly lower. However, the metrics during scenario generation only reflect offline evaluations, while the actual impact of the generated scenarios when applied to the planner is what truly matters to the user. We conducted a comprehensive evaluation of the planner's actual performance in generated scenarios, an aspect that has not been thoroughly explored in other works such as [21], [28], [46], [65]. Our results, focusing on real-world application, provide a more accurate reflection of how generated scenarios influence the planner's performance, bridging the gap left by purely offline evaluation studies.

D. Evaluation of Planner Enhancement

In this subsection, we analyze the effectiveness of the generated expert trajectories in enhancing the original training data distribution. Specifically, we evaluate the performance of the autonomous driving planner, AutoBot, before and after fine-

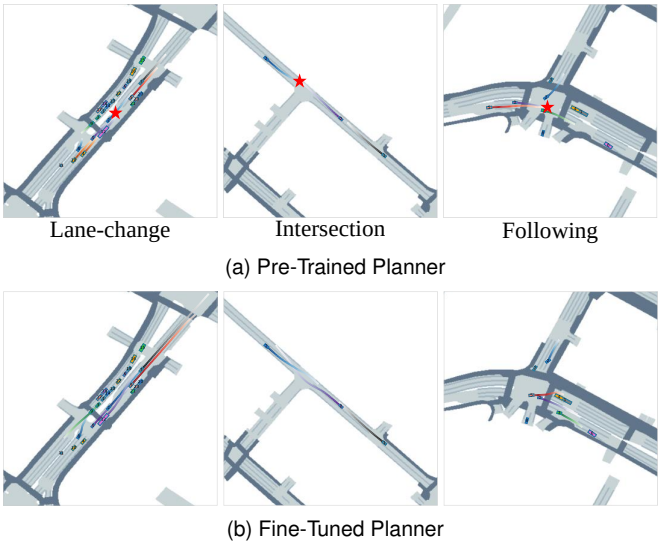

Fig. 6. Comparison of Pre-Trained and Fine-Tuned Planner in Safety-Critical Scenarios

tuning with expert trajectories, to assess the effectiveness of the generated safety-critical scenarios and their corresponding expert trajectories.

Metrics. We measure the planner's performance by evaluating the PCR, OR, average velocity, and jerk. The PCR and OR have already been defined in the previous sections and will not be repeated here.

(1) Average Velocity (AV), is used to assess the efficiency of the autonomous vehicle [66], [67]. A higher average velocity indicates better driving efficiency of the planner.

(2) Jerk, defined as the rate of change of acceleration, is used to measure driving comfort, as it significantly impacts passenger comfort [68]. The higher the jerk, the more uncomfortable the passengers will feel. We evaluate the planner's comfort by calculating the average jerk for each scenario.

The Pre-Trained row in Table III shows the test performance of AutoBot in both regular and adversarial scenarios, as previously introduced in Section IV-C. Subsequently, we utilized the expert trajectories generated in the safety-critical scenarios, performed data cleaning, and fine-tuned the pre-trained planner. The fine-tuned test results are presented in the Fine-Tuned row of Table III. The off-road rate has slightly increased compared to the pre-trained planner, likely due to the fine-tuning planner with a large number of expert trajectories generated in safety-critical scenarios, leading to a more aggressive driving style. The fine-tuned model achieved lower PCR, higher AV, and lower jerk in both regular and adversarial scenarios. This demonstrates that the generated expert trajectories help improve the safety, efficiency, and comfort of existing autonomous driving planners, further validating the effectiveness of MTD in enhancing the original training data distribution.

Figure 6 compares the performance of the pre-trained planner and the fine-tuned planner in adversarial safety-critical scenarios, with lane-change, intersection, and following scenarios shown from left to right. In the lane-change scenario, the pre-trained planner collides with another agent during the lane change, while the fine-tuned planner successfully avoids the collision through better trajectory planning. In the intersection scenario, the pre-trained planner fails to respond properly in the complex intersection environment, resulting in a collision between the ego agent and other agents. In contrast, the fine-tuned planner handles the intersection more effectively, ensuring the ego agent passes through safely. In the following scenario, the pre-trained planner fails to detect that agent 5 accelerates uncontrollably, leading to a rear-end collision with the ego agent. The fine-tuned planner, however, reacts quickly and accelerates in time to avoid the collision, allowing the ego agent to safely cruise in its lane (by the time agent 5 reaches the collision point, the ego agent has already passed through). These comparisons demonstrate that the fine-tuned planner, by learning from expert trajectories in safety-critical scenarios, significantly improves its ability to handle complex traffic situations, reduces the planner's collision rate, and enhances safety, driving efficiency, and comfort. This further validates the effectiveness of our method in augmenting the original training distribution.

E. Analyzing Safety-Critical Scenario

In this subsection, we conducted a comprehensive analysis of the safety-critical scenarios generated by MTD. First, we filtered out scenarios where MTD failed to generate collisions. Then, to further categorize and analyze the types of collisions, we applied K-Means [69] clustering, setting the number of clusters to $k = 7$, and visualized the clustering results as shown in Figure 7.

Based on the clustering results, and following the classification of collision types from [64], we manually assigned semantic labels to the clusters. The examples of each collision type are displayed at the bottom of Figure 7. The green bars (left side) represent the proportion of each collision type among the total successfully generated safety-critical scenarios, while the orange bars (right side) represent the proportion of each collision type for which an expert trajectory solution was found to avoid the collision. Specifically, we identified seven types of collisions: rear-end collisions (by adversary), head-on collisions, rear-end collisions (by ego), head-side collisions, T-bone collisions, sideswipe collisions, and oblique cut-in collisions. MTD is capable of generating a diverse range of collision types, with head-on collisions being the most frequent at 29.67%, and oblique cut-in collisions being the least frequent at 4.75%. Rear-end collisions (by ego) had the highest rate of finding an expert trajectory solution, reaching 88.10%, while head-side collisions were the most challenging to resolve, with only 42.31% of cases finding a solution.

F. Evaluating the Diversity and Long-Term Performance

In this subsection, we evaluate the diversity and long-term performance of scenarios generated by MTD and baseline methods, extending the simulation timestep to 20 seconds to assess both aspects.

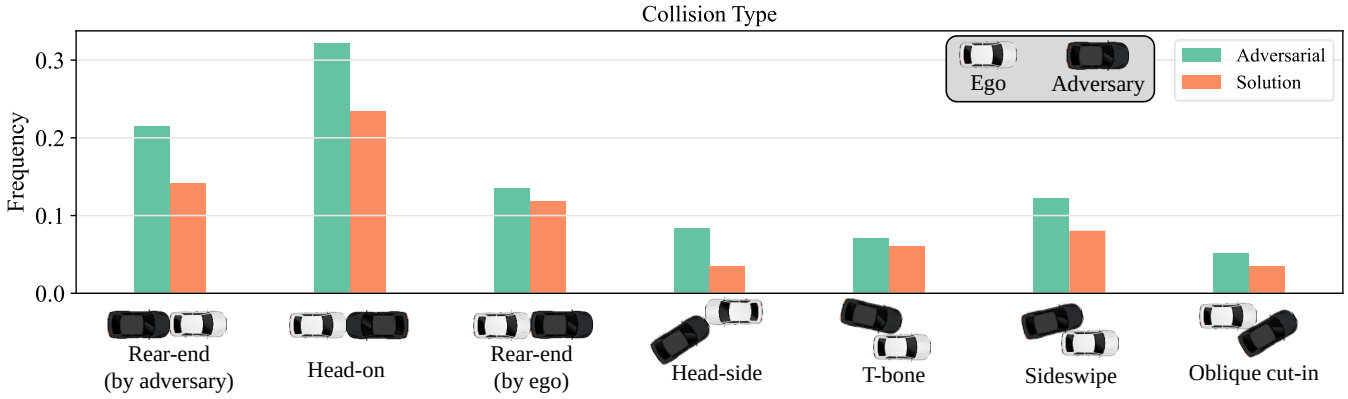

Fig. 7. Frequency and Solution Rate of Different Collision Types in Safety-Critical Scenarios

TABLE IV
EVALUATION OF DIV. AND LONG-TERM PERFORMANCE

Method	FR ↓	CR ↓	OR ↓	Cov ↑	TD ↑	RB ↓
SimNet	24.58	15.80	3.05	395.2	0.00	0.098
TrafficSim	27.08	20.39	3.42	861.0	4.28	0.119
BITS	14.82	8.62	**0.95**	1078.7	4.66	0.120
CTG++	14.35	9.17	1.45	**1518.9**	**7.13**	**0.069**
STRIVE	15.27	10.55	1.20	937.4	5.97	0.096
MTD	**11.21**	**6.16**	1.71	1089.8	6.54	0.086
Dataset	17.09	15.95	0.50	539.3	0.00	0.000

Metrics. Similar to the evaluation of safety-critical scenarios, we assess stability by reporting the Failure Rate (FR), Collision Rate (CR), and Off-road Rate (OR), while Realism Bias (RB) is used to measure the realism of the generated scenarios. **Diversity** is evaluated using two metrics: coverage and trajectory diversity.

(1) Coverage (Cov) evaluates how well the agents cover the scene. We estimate the trajectory distribution density using Gaussian Kernel Density Estimation (KDE) across all timesteps, focusing on the 2D spatial distribution. Coverage is defined as the number of grid points where the KDE estimate exceeds a threshold. In our experiments, the map is discretized with a 2m x 2m resolution, and we accumulate the results over 5 trials to calculate coverage density following [63].

(1) Trajectory Diversity (TD) is calculated by measuring the Wasserstein distance between the density distributions of trajectories from different trials. Based on the previous study [46], we unfold all grid points with non-zero density, calculate the Euclidean distance between each pair, and generate a distance matrix. The density distributions are normalized to sum to 1, and the Wasserstein distance is calculated. For n trials of the same scene, the average Wasserstein distance between $n(n-1)/2$ pairs of density distributions is used as the diversity metric, expressed as:

$$\text{Diversity} = \frac{2}{n\,(n-1)} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \text{Wass}\,(\rho_i, \rho_j) \tag{29}$$

where $\text{Wass}(\cdot,\cdot)$ is the Wasserstein distance, and ρ_i is the density distribution of the i-th trial.

Table IV presents the quantitative results of long-term closed-loop simulations for generating regular driving scenarios. Due to differences in environmental settings, we re-tested the performance of all baseline methods in the same environment. Additionally, we report the performance of the real-world dataset (Dataset), which contains some noise reflected in a non-zero failure rate. We suspect that the main errors are caused by agent collisions due to inaccurate bounding box annotations. Notably, MTD achieved a lower failure rate and generated more diverse scenarios compared to the real-world dataset.

In our comparisons, MTD outperformed SimNet and TrafficSim across all metrics. Although CTG++ slightly surpassed MTD in diversity and realism, it fell short in terms of stability. BITS showed slight advantages in Offroad and Coverage metrics, while STRIVE performed marginally better in Offroad but lagged behind MTD in other metrics. Importantly, MTD achieved the best results across all baseline methods in the FR and CR metrics, highlighting its superior stability, particularly in preventing agent collisions during long-term scenario generation. In summary, MTD not only excels in generating diverse scenarios but also demonstrates strong performance in long-term simulations.

G. Ablation Study

Impact of Adversarial Hybrid Guidance. We studied how the components of the guidance functions affect the generated safety-critical scenarios. Specifically, we considered the following variations:

(1) MTD-w/o-I. We removed the initialization guidance function.

(2) MTD-w/o-A. We removed our adversarial guidance function and adopted an adversarial guidance function similar to STRIVE.

(3) MTD-w/o-C. We removed the guidance function that avoids collisions with other vehicles.

(4) MTD-w/o-E. We removed the guidance function that avoids collisions with the environment.

Through the ablation study on the adversarial hybrid guidance function in Table V, we can draw the following conclusions: The adversarial guidance is crucial for generating adversarial scenarios with a high collision rate, as it increases the likelihood of collisions and reduces other failure rates and unrelated collisions. The environment guidance plays

TABLE V
ABLATION OF ADVERSARIAL HYBRID GUIDANCE

Method	GCR ↓	RB ↓	AOR ↓	OFR ↓	OCR ↓
MTD	**73.00**	0.060	**0.01**	**5.18**	**4.65**
MTD-w/o-I	72.00	0.080	0.02	5.67	5.20
MTD-w/o-A	70.00	0.071	0.04	6.30	5.70
MTD-w/o-C	72.00	**0.051**	0.07	7.74	7.27
MTD-w/o-E	72.00	0.077	1.82	9.16	5.83

TABLE VI
ABLATION OF MTD COMPONENTS

	FR ↓	CR ↓	OR ↓	Cov ↑	TD ↑	RB ↓
MTD	**11.21**	**6.16**	1.71	1089.8	6.54	**0.086**
MTD-w/o-D	12.44	7.90	**1.65**	1021.2	6.04	0.093
MTD-w/o-S	14.61	9.64	1.77	**1639.6**	**7.03**	0.103

an important role in ensuring the realism of the scenarios, reducing off-road incidents, and lowering the failure rate. The initialization guidance function has a significant impact on the initial setup of the model and the reasonableness of agent behavior, particularly in preventing off-road incidents and reducing other collision rates. Therefore, the complete adversarial hybrid guidance function performs best across multiple metrics, particularly in maintaining realism and generating effective adversarial scenarios with a well-balanced performance.

Impact of Components. We evaluated the effectiveness of MTD components in traffic simulation, specifically assessing our designed Social Transformer and Decoder. Specifically, we considered the following variations:

(1) MTD-w/o-D: MTD without the Decoder component.

(2) MTD-w/o-S: MTD without the Social Transformer component.

As shown in the long-term traffic simulation ablation study in Table VI, the Social Transformer is crucial for significantly reducing the failure rate, collision rate, and maintaining traffic realism. Removing the Decoder has a smaller impact on several metrics but still contributes to both coverage and diversity. Overall, the full MTD model demonstrates the best balance across all metrics, generating stable, realistic, and diverse scenarios.

V. RELATED WORK

Methods for obtaining safety-critical scenarios can be divided into two categories: real-world data collection and simulation-based generation.

The real-world data collection approach samples and extracts safety-critical scenarios from daily driving data. Works like [16]–[20] use historical analysis and clustering to identify sparse, low-frequency scenarios. However, safety-critical scenarios are rare in daily driving data, making extraction difficult and time-consuming. Moreover, the limited diversity of real-world scenes means sampled data may not cover all extreme cases, leaving gaps in autonomous system training.

Another approach is to generate safety-critical scenarios in a simulated environment. Using simulation tools like CARLA [70], it is possible to manually design the paths of autonomous vehicles to create such scenarios [71]–[73]. However, when large numbers of complex scenarios need to be generated, manually designing paths becomes both time-consuming and labor-intensive, making it difficult to meet the growing demand for diverse scenarios in autonomous driving systems [74].

Consequently, recent research has increasingly focused on automating the generation of safety-critical scenarios to efficiently generate diverse scenes. [21], [22] generate safety-critical scenarios by initializing opponents' states. [23] applies adversarial optimization in each step of a Diffusion model with a U-Net [75] architecture. [24] uses kinematic gradients to modify vehicle trajectories, improving imitation learning robustness. [25] leverages LLMs to generate safety-critical scenarios by converting language into control codes. [26] combines traffic priors and CVAE [76] to optimize adversarial trajectories for realistic scenarios. [27] transforms historical trajectory replays into safety-critical scenarios using resampling. [59] uses Bayesian optimization to disrupt vehicle paths by modifying trajectories. [29] models human driving strategies with GAIL [77] and designs reward functions to generate safety-critical scenarios.

However, these methods have several limitations. [21]–[25] overlook dynamic factors in complex traffic environments, particularly in multi-vehicle interaction scenarios. Adversarial vehicles may prematurely collide with non-autonomous vehicles in these environments, leading to unrealistic and implausible generated scenarios. [26], [27] heavily rely on original trajectories, which limits their diversity when generating safety-critical scenarios. Furthermore, without these original trajectories, they are unable to produce meaningful data. Lastly, [21], [28]–[30] do not adequately assess the actual impact of safety-critical scenarios on the performance of autonomous driving planners, which is a primary concern for users. These methods lack end-to-end testing on autonomous driving planners. Additionally, they fail to validate the contribution of the generated scenarios to the improvement of planner performance. Advancements in generative models have made controllable traffic simulation possible. CTG guides Diffusion to generate controllable traffic simulations using STL [78], [79] rule analysis loss functions [39] or LLMs. While CTG can generate goal-oriented trajectories, it struggles with efficiently handling specific safety-critical scenarios.

VI. CONCLUSION

In this work, we propose Motion Transformer Diffusion (MTD), a novel method for generating safety-critical scenarios. MTD is built on a query-centric multi-agent traffic generation model, capable of generating stable, realistic, and diverse safety-critical scenarios, along with expert solutions. Extensive experiments show that MTD outperforms state-of-the-art methods in stability, realism, and diversity. Additionally, MTD not only effectively tests the performance of autonomous driving planners but also enhances the training data distribution through its generated expert trajectories.

REFERENCES

[1] K. Xu, X. Xiao, J. Miao, and Q. Luo, "Data driven prediction architecture for autonomous driving and its application on apollo platform," in *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2020, pp. 175–181.

[2] S. Teng, X. Hu, P. Deng, B. Li, Y. Li, Y. Ai, D. Yang, L. Li, Z. Xuanyuan, F. Zhu *et al.*, "Motion planning for autonomous driving: The state of the art and future perspectives," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 6, pp. 3692–3711, 2023.

[3] W. Schwarting, J. Alonso-Mora, and D. Rus, "Planning and decision-making for autonomous vehicles," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 187–210, 2018.

[4] L. Chen, P. Wu, K. Chitta, B. Jaeger, A. Geiger, and H. Li, "End-to-end autonomous driving: Challenges and frontiers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[5] B. Chen, X. Chen, Q. Wu, and L. Li, "Adversarial evaluation of autonomous vehicles in lane-change scenarios," *IEEE transactions on intelligent transportation systems*, vol. 23, no. 8, pp. 10 333–10 342, 2021.

[6] Z. Ghodsi, S. K. S. Hari, I. Frosio, T. Tsai, A. Troccoli, S. W. Keckler, S. Garg, and A. Anandkumar, "Generating and characterizing scenarios for safety testing of autonomous vehicles," in *2021 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2021, pp. 157–164.

[7] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, "Class-balanced loss based on effective number of samples," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 9268–9277.

[8] Y. Zhang, B. Kang, B. Hooi, S. Yan, and J. Feng, "Deep long-tailed learning: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 9, pp. 10 795–10 816, 2023.

[9] W. Zhou, Z. Cao, Y. Xu, N. Deng, X. Liu, K. Jiang, and D. Yang, "Long-tail prediction uncertainty aware trajectory planning for self-driving vehicles," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2022, pp. 1275–1282.

[10] O. Makansi, Ö. Cicek, Y. Marrakchi, and T. Brox, "On exposing the challenging long tail in future prediction of traffic actors," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 13 147–13 157.

[11] H. Tian, G. Wu, J. Yan, Y. Jiang, J. Wei, W. Chen, S. Li, and D. Ye, "Generating critical test scenarios for autonomous driving systems via influential behavior patterns," in *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, 2022, pp. 1–12.

[12] C. E. Tuncali and G. Fainekos, "Rapidly-exploring random trees for testing automated vehicles," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2019, pp. 661–666.

[13] A. Calò, P. Arcaini, S. Ali, F. Hauer, and F. Ishikawa, "Generating avoidable collision scenarios for testing autonomous driving systems," in *2020 IEEE 13th International Conference on Software Testing, Validation and Verification (ICST)*. IEEE, 2020, pp. 375–386.

[14] J. Sun, H. Zhang, H. Zhou, R. Yu, and Y. Tian, "Scenario-based test automation for highly automated vehicles: A review and paving the way for systematic safety assurance," *IEEE transactions on intelligent transportation systems*, vol. 23, no. 9, pp. 14 088–14 103, 2021.

[15] W. Ding, C. Xu, M. Arief, H. Lin, B. Li, and D. Zhao, "A survey on safety-critical driving scenario generation—a methodological perspective," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 7, pp. 6971–6988, 2023.

[16] C. Knies and F. Diermeyer, "Data-driven test scenario generation for cooperative maneuver planning on highways," *Applied Sciences*, vol. 10, no. 22, p. 8154, 2020.

[17] M. Arief, P. Glynn, and D. Zhao, "An accelerated approach to safely and efficiently test pre-production autonomous vehicles on public streets," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 2006–2011.

[18] F. Kruber, J. Wurst, and M. Botsch, "An unsupervised random forest clustering technique for automatic traffic scenario categorization," in *2018 21st International conference on intelligent transportation systems (ITSC)*. IEEE, 2018, pp. 2811–2818.

[19] F. Kruber, J. Wurst, E. S. Morales, S. Chakraborty, and M. Botsch, "Unsupervised and supervised learning with the random forest algorithm for traffic scenario clustering and classification," in *2019 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2019, pp. 2463–2470.

[20] W. Wang and D. Zhao, "Extracting traffic primitives directly from naturalistically logged data for self-driving applications," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1223–1229, 2018.

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

[22] W. Ding, B. Chen, B. Li, K. J. Eun, and D. Zhao, "Multimodal safety-critical scenarios generation for decision-making algorithms evaluation," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1551–1558, 2021.

[23] C. Xu, D. Zhao, A. Sangiovanni-Vincentelli, and B. Li, "Diffscene: Diffusion-based safety-critical scenario generation for autonomous vehicles," in *The Second Workshop on New Frontiers in Adversarial Machine Learning*, 2023.

[24] N. Hanselmann, K. Renz, K. Chitta, A. Bhattacharyya, and A. Geiger, "King: Generating safety-critical driving scenarios for robust imitation via kinematics gradients," in *European Conference on Computer Vision*. Springer, 2022, pp. 335–352.

[25] J. Zhang, C. Xu, and B. Li, "Chatscene: Knowledge-enabled safety-critical scenario generation for autonomous vehicles," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 15 459–15 469.

[26] D. Rempe, J. Philion, L. J. Guibas, S. Fidler, and O. Litany, "Generating useful accident-prone driving scenarios via a learned traffic prior," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 17 305–17 315.

[27] L. Zhang, Z. Peng, Q. Li, and B. Zhou, "Cat: Closed-loop adversarial training for safe end-to-end driving," in *Conference on Robot Learning*. PMLR, 2023, pp. 2357–2372.

[28] W.-J. Chang, F. Pittaluga, M. Tomizuka, W. Zhan, and M. Chandraker, "Controllable safety-critical closed-loop traffic simulation via guided diffusion," 2023.

[29] K. Hao, W. Cui, Y. Luo, L. Xie, Y. Bai, J. Yang, S. Yan, Y. Pan, and Z. Yang, "Adversarial safety-critical scenario generation using naturalistic human driving priors," *IEEE Transactions on Intelligent Vehicles*, 2023.

[30] M. Klischat, E. I. Liu, F. Holtke, and M. Althoff, "Scenario factory: Creating safety-critical traffic scenarios for automated vehicles," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2020, pp. 1–7.

[31] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[32] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, "High-resolution image synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695.

[33] W. Peebles and S. Xie, "Scalable diffusion models with transformers," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4195–4205.

[34] F. Bao, C. Xiang, G. Yue, G. He, H. Zhu, K. Zheng, M. Zhao, S. Liu, Y. Wang, and J. Zhu, "Vidu: a highly consistent, dynamic and skilled text-to-video generator with diffusion models," *arXiv preprint arXiv:2405.04233*, 2024.

[35] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach, "Sdxl: Improving latent diffusion models for high-resolution image synthesis," *arXiv preprint arXiv:2307.01952*, 2023.

[36] C. C. White III, "A survey of solution techniques for the partially observed markov decision process," *Annals of Operations Research*, vol. 32, no. 1, pp. 215–230, 1991.

[37] A. Vaswani, "Attention is all you need," *Advances in Neural Information Processing Systems*, 2017.

[38] J. Gehring, M. Auli, D. Grangier, D. Yarats, and Y. N. Dauphin, "Convolutional sequence to sequence learning," in *International conference on machine learning*. PMLR, 2017, pp. 1243–1252.

[39] Z. Zhong, D. Rempe, D. Xu, Y. Chen, S. Veer, T. Che, B. Ray, and M. Pavone, "Guided conditional diffusion for controllable traffic simulation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 3560–3566.

[40] S. Shi, L. Jiang, D. Dai, and B. Schiele, "Motion transformer with global intention localization and local movement refinement," *Advances in Neural Information Processing Systems*, vol. 35, pp. 6531–6543, 2022.

[41] S. Liu, F. Li, H. Zhang, X. Yang, X. Qi, H. Su, J. Zhu, and L. Zhang, "Dab-detr: Dynamic anchor boxes are better queries for detr," in *International Conference on Learning Representations*, 2022.

[42] S. Shi, L. Jiang, D. Dai, and B. Schiele, "Mtr++: Multi-agent motion prediction with symmetric scene modeling and guided intention querying," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[43] Z. Zhou, L. Ye, J. Wang, K. Wu, and K. Lu, "Hivt: Hierarchical vector transformer for multi-agent motion prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 8823–8833.

[44] Z. Zhou, J. Wang, Y.-H. Li, and Y.-K. Huang, "Query-centric trajectory prediction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 17 863–17 873.

[45] P. Dhariwal and A. Nichol, "Diffusion models beat gans on image synthesis," *Advances in neural information processing systems*, vol. 34, pp. 8780–8794, 2021.

[46] Z. Zhong, D. Rempe, Y. Chen, B. Ivanovic, Y. Cao, D. Xu, M. Pavone, and B. Ray, "Language-guided traffic simulation via scene-level diffusion," in *Conference on Robot Learning*. PMLR, 2023, pp. 144–177.

[47] L. Zhang, A. Rao, and M. Agrawala, "Adding conditional control to text-to-image diffusion models," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 3836–3847.

[48] S. Suo, S. Regalado, S. Casas, and R. Urtasun, "Trafficsim: Learning to simulate realistic multi-agent behaviors," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 10 400–10 409.

[49] J. Fu, X. Zhang, Z. Jian, S. Chen, J. Xin, and N. Zheng, "Efficient safety-enhanced velocity planning for autonomous driving with chance constraints," *IEEE Robotics and Automation Letters*, vol. 8, no. 6, pp. 3358–3365, 2023.

[50] R. Girgis, F. Golemo, F. Codevilla, M. Weiss, J. A. D'Souza, S. E. Kahou, F. Heide, and C. Pal, "Latent variable sequential set transformers for joint multi-agent motion prediction," in *International Conference on Learning Representations*, 2022.

[51] S. Liu, X. Chen, Z. Wu, L. Deng, H. Su, and K. Zheng, "Hega: heterogeneous graph aggregation network for trajectory prediction in high-density traffic," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2022, pp. 1319–1328.

[52] Y. Xia, S. Liu, Q. Yu, L. Deng, Y. Zhang, H. Su, and K. Zheng, "Parameterized decision-making with multi-modality perception for autonomous driving," in *2024 IEEE 40th International Conference on Data Engineering (ICDE)*. IEEE, 2024, pp. 4463–4476.

[53] H. Marzbani, H. Khayyam, . V. Quoc, and R. N. Jazar, "Autonomous vehicles: Autodriver algorithm and vehicle dynamics," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 4, pp. 3201–3211, 2019.

[54] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, "nuscenes: A multimodal dataset for autonomous driving," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 621–11 631.

[55] K. Lynch, *Modern Robotics*. Cambridge University Press, 2017.

[56] I. Loshchilov, "Decoupled weight decay regularization," *arXiv preprint arXiv:1711.05101*, 2017.

[57] Y. Abeysirigoonawardena, F. Shkurti, and G. Dudek, "Generating adversarial driving scenarios in high-fidelity simulators," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8271–8277.

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

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

[60] V. Voleti, A. Jolicoeur-Martineau, and C. Pal, "Mcvd-masked conditional video diffusion for prediction, generation, and interpolation," *Advances in neural information processing systems*, vol. 35, pp. 23 371–23 385, 2022.

[61] A. Blattmann, R. Rombach, H. Ling, T. Dockhorn, S. W. Kim, S. Fidler, and K. Kreis, "Align your latents: High-resolution video synthesis with latent diffusion models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 22 563–22 575.

[62] L. Bergamini, Y. Ye, O. Scheel, L. Chen, C. Hu, L. Del Pero, B. Osiński, H. Grimmett, and P. Ondruska, "Simnet: Learning reactive self-driving simulations from real-world observations," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 5119–5125.

[63] D. Xu, Y. Chen, B. Ivanovic, and M. Pavone, "Bits: Bi-level imitation for traffic simulation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 2929–2936.

[64] R. J. Mitchell, M. Bambach, and B. Toson, "Injury risk for matched front and rear seat car passengers by injury severity and crash type: An exploratory study," *Accident Analysis & Prevention*, vol. 82, pp. 171–179, 2015.

[65] Z. Huang, Z. Zhang, A. Vaidya, Y. Chen, C. Lv, and J. F. Fisac, "Versatile scene-consistent traffic scenario generation as optimization with diffusion," *arXiv preprint arXiv:2404.02524*, 2024.

[66] Y. Xia, S. Liu, X. Chen, Z. Xu, K. Zheng, and H. Su, "Rise: A velocity control framework with minimal impacts based on reinforcement learning," in *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2022, pp. 2210–2219.

[67] S. Liu, H. Su, Y. Zhao, K. Zeng, and K. Zheng, "Lane change scheduling for autonomous vehicle: A prediction-and-search framework," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, 2021, pp. 3343–3353.

[68] M. Zhu, Y. Wang, Z. Pu, J. Hu, X. Wang, and R. Ke, "Safe, efficient, and comfortable velocity control based on reinforcement learning for autonomous driving," *Transportation Research Part C: Emerging Technologies*, vol. 117, p. 102662, 2020.

[69] J. Macqueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability/University of California Press*, 1967.

[70] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on robot learning*. PMLR, 2017, pp. 1–16.

[71] Z. Wei, H. Huang, G. Zhang, R. Zhou, X. Luo, S. Li, and H. Zhou, "Interactive critical scenario generation for autonomous vehicles testing based on in-depth crash data using reinforcement learning," *IEEE Transactions on Intelligent Vehicles*, 2024.

[72] Z. Xinxin, L. Fei, and W. Xiangbin, "Csg: Critical scenario generation from real traffic accidents," in *2020 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2020, pp. 1330–1336.

[73] A. Gambi, V. Nguyen, J. Ahmed, and G. Fraser, "Generating critical driving scenarios from accident sketches," in *2022 IEEE International Conference On Artificial Intelligence Testing (AITest)*. IEEE, 2022, pp. 95–102.

[74] Y. Guan, H. Liao, Z. Li, J. Hu, R. Yuan, Y. Li, G. Zhang, and C. Xu, "World models for autonomous driving: An initial survey," *IEEE Transactions on Intelligent Vehicles*, 2024.

[75] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*. Springer, 2015, pp. 234–241.

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

[77] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Advances in neural information processing systems*, vol. 29, 2016.

[78] O. Maler and D. Nickovic, "Monitoring temporal properties of continuous signals," in *International symposium on formal techniques in real-time and fault-tolerant systems*. Springer, 2004, pp. 152–166.

[79] K. Leung, N. Aréchiga, and M. Pavone, "Backpropagation through signal temporal logic specifications: Infusing logical structure into gradient-based methods," *The International Journal of Robotics Research*, vol. 42, no. 6, pp. 356–370, 2023.