SCORP: Scene-Consistent Multi-agent Diffusion Planning with Stable Online Reinforcement Post-Training for Cooperative Driving

Haojie Bai, Aimin Li, *Member, IEEE*, Ruoyu Yao, Xiongwei Zhao, Tingting Zhang, *Member, IEEE*, Xing Zhang, Caixiong Li, Lin Gao, *Senior Member, IEEE*, and Jun Ma, *Senior Member, IEEE*

***Abstract*—Cooperative driving is a safety- and efficiency-critical task that requires the coordination of diverse, interaction-realistic multi-agent trajectories. Although existing diffusion-based methods can capture multimodal behaviors from demonstrations, they often exhibit weak scene consistency and poor alignment with closed-loop cooperative objectives. This makes post-training necessary for further improvement, yet achieving stable online post-training in reactive multi-agent environments remains challenging. In this paper, we propose SCORP, a scene-consistent multi-agent diffusion planner with stable online reinforcement learning (RL) post-training for cooperative driving. For pre-training, we develop a scene-conditioned multi-agent denoising architecture that couples inter-agent self-attention with a dual-path conditioning mechanism: cross-attention provides direct scene-information injection, while AdaLN-Zero enables additional flexible and stable conditional modulation, thereby improving the scene consistency and road adherence of joint trajectories. For post-training, we formulate a two-layer Markov decision process (MDP) that explicitly integrates the reverse denoising chain with policy–environment interaction. We further co-design dense, well-shaped planning rewards and variance-gated group-relative policy optimization (VG-GRPO) to mitigate advantage collapse and gradient instability during closed-loop training. Extensive experiments show that SCORP outperforms strong open-source baselines on WOMD, with 10.47%–28.26% and 1.70%–7.22% improvements in core safety and efficiency metrics, respectively. Moreover, compared with alternative post-training methods, SCORP delivers significant and consistent gains in both driving safety and traffic efficiency, highlighting stable and sustained advances in closed-loop cooperative driving.**

***Index Terms*—Cooperative driving, reinforcement learning, diffusion policy, multi-agent planning, closed-loop simulation**

I. Introduction

A. Background and Motivation

Advanced *Connected Autonomous Vehicles* (CAVs) have catalyzed the emergence of cooperative multi-agent driving paradigms, opening new possibilities for addressing key transportation challenges and potentially improving both safety and traffic efficiency [1], [2]. By enabling information sharing and coordinated decision-making among vehicles, infrastructure, and cloud platforms, cooperative driving extends the capability of isolated single-vehicle autonomy and supports system-level intelligence in complex traffic environments [3]. As such, cooperative driving is inherently a safety- and efficiency-critical task, especially in dense and interactive scenarios where the behaviors of multiple agents are coupled [4]. Central to this paradigm, *multi-agent behavior modeling* underpins high-level cooperative decision-making by characterizing the interactive behaviors of multiple traffic participants. However, real traffic behavior is inherently stochastic and multimodal: even within the same scene, multiple interaction patterns can be reasonable and effective. This inherent uncertainty and diversity make it difficult to produce cooperative plans that are simultaneously realistic, safe, and efficient. Therefore, generating *well-coordinated cooperative behaviors* while preserving multimodality and realistic interactions remains an open challenge, which in turn limits the deployment potential of cooperative autonomous driving systems [5].

Recent advances in diffusion models have introduced a powerful probabilistic paradigm for modeling complex multi-modal distributions over driving behaviors and trajectories [6], [7]. By leveraging a forward diffusion and reverse denoising process and learning from large-scale human demonstrations via imitation learning, diffusion-based planners can capture dexterous behaviors and joint distributions over interactive multi-agent trajectories. However, existing methods often struggle to balance *multi-agent interaction modeling* with *scene-conditioned modeling*, leading to joint trajectories that may violate scene constraints [8], [9]. Furthermore, these methods suffer from *objective misalignment* and *distribution shift* [10]. Behavior cloning (BC) mainly fits the data distribution and lacks the ability to explicitly enforce human preferences, expectations, or constraints, making it difficult to directly optimize safety and efficiency objectives in multi-agent systems. For example, safety-critical events such as collisions are exceedingly rare in human driving datasets, resulting in sparse supervision for learning safe interactions. Consequently, failures are more likely to occur under closed-loop execution or out-of-distribution scenarios. Collectively, these issues hinder diffusion-based multi-agent planners from achieving safe and efficient closed-loop performance and limit their robustness

The project page is available at: https://zebai9.github.io/SCORP/.

Haojie Bai, Xiongwei Zhao, Tingting Zhang, and Lin Gao are with the School of Information Science and Technology, Harbin Institute of Technology (Shenzhen), Shenzhen 518071, China (e-mail: hjbai@stu.hit.edu.cn, gaol@hit.edu.cn).

Aimin Li is with the Middle East Technology University (METU), Ankara, 06800, Turkiye (e-mail: aimin@metu.edu.tr). Aimin Li contributes equally to this work.

Ruoyu Yao and Jun Ma are with the Robotics and Autonomous Systems Thrust, Hong Kong University of Science and Technology (Guangzhou), Guangzhou 511453, China (e-mail: ryao092@connect.hkust-gz.edu.cn, jun.ma@ust.hk).

Xing Zhang and Caixiong Li are with the School of Computer Science and Technology, Qinghai University, Xining, 810016, China.

and reliability in real-world scenarios [11].

Reinforcement learning (RL) offers a promising avenue for further improving pretrained planning models by coupling sampling-based exploration with reward-driven policy optimization [12], [13]. Recent studies suggest that RL post-training can enhance closed-loop planning performance and shape driving styles, thereby potentially promoting cooperative behaviors beyond what pretraining alone can achieve [14]–[16]. In this paradigm, a pretrained planner serves as the actor, sampling diverse candidate futures that are scored by a reward function and iteratively refined through RL algorithms [17], [18]. However, most existing work focuses on *offline* post-training, which is closer in spirit to reward-augmented supervised fine-tuning [19]. Without online interaction with reactive environments, constrained rollouts and limited exploration often yield only modest performance improvements [12], [16].

More broadly, achieving satisfactory gains with RL post-training primarily depends on (i) online interaction, (ii) well-shaped rewards, and (iii) robust policy optimization. First, online interaction is naturally aligned with closed-loop execution, allowing the policy to explore interaction scenarios beyond the pretraining distribution and to correct behaviors under reward guidance, thereby mitigating performance degradation from distribution shift. Second, well-shaped rewards provide fine-grained optimization signals for safety and efficiency-critical objectives while remaining compatible with realistic, human-like driving. Third, robust policy optimization further improves tolerance to reward noise and gradient variance, enabling stable and sustained performance gains. Nevertheless, realizing these benefits from online RL in fully closed-loop settings remains challenging due to non-stationary interactions, compounding rollout errors, and high-variance gradient estimates, which together impose stringent requirements on the stability and controllability of the post-training pipeline.

B. Solution and Contributions

To address these limitations, we propose SCORP, a scene-consistent multi-agent diffusion planner with *online RL post-training*, tailored for closed-loop cooperative driving. SCORP couples condition-enhanced diffusion pre-training with stable online RL post-training to continually improve closed-loop planning performance and cooperative behaviors.

During pre-training, we build a multi-agent denoising network on the Diffusion Transformer [20] to capture the scene-conditioned joint distribution of multi-agent trajectories while balancing inter-agent interaction modeling and scene conditioning. Specifically, we introduce *AdaLN-Zero* adaptive modulation in conjunction with cross-attention. Through scene-driven feature modulation with zero-initialization, the model strengthens scene consistency and road adherence of multi-agent trajectories while enhancing numerical stability during training, enabling more effective use of scene conditioning than cross-attention alone.

During post-training, we develop a stable online RL post-training framework to strengthen safety- and efficiency-oriented cooperative behaviors. We formulate a *two-layer MDP* to support online optimization, explicitly coupling the denoising chain with policy–environment interaction. To counteract training instability induced by closed-loop interaction, we design dense and well-shaped rewards to characterize cooperative behaviors with respect to safety and efficiency, providing stable and fine-grained optimization signals. We further propose *variance-gated group-relative policy optimization (VG-GRPO)*, which adaptively gates sampled groups and switches normalization schemes based on within-group reward variance, mitigating advantage collapse and gradient instability in standard GRPO [17] and improving the robustness of online training.

Our main contributions are summarized as follows:

- We propose SCORP, a tightly coupled framework that unifies scene-consistent multi-agent diffusion pre-training with stable online RL post-training for cooperative driving. The pre-training stage learns a scene-conditioned multi-modal interaction prior that provides realistic, diverse grouped trajectory samples for closed-loop exploration and group-relative policy optimization. Building on this prior, the post-training stage performs online refinement to progressively promote safety- and efficiency-oriented cooperative behaviors while preserving scene consistency and interaction realism.
- We propose a scene-conditioned multi-agent trajectory diffusion model that couples inter-agent self-attention with a dual-path conditioning mechanism, thereby jointly capturing inter-agent interactions and improving the scene consistency and road adherence of joint trajectories. Meanwhile, we develop a stable online RL post-training pipeline that explicitly integrates the denoising chain with policy–environment interaction through a two-layer MDP formulation. To stabilize closed-loop training, we co-design dense, well-shaped rewards and VG-GRPO, thereby mitigating advantage collapse and gradient instability while achieving continual improvement in closed-loop planning and cooperative behavior.
- Extensive experiments demonstrate that SCORP achieves superior closed-loop planning performance, yielding 10.47%–28.26% and 1.70%–7.22% improvements in safety and efficiency metrics, respectively, over strong representative baselines on WOMD [21]. Furthermore, compared with other post-training methods, the proposed online RL framework delivers larger and more consistent gains across key closed-loop metrics.

The remainder of this paper is organized as follows. Section II reviews the related work. Section III introduces the preliminaries of diffusion models and reinforcement learning. Section IV presents the problem statement. Section V details the proposed multi-agent diffusion pre-training method. Section VI presents the stable online RL post-training pipeline. Section VII reports the simulation results and discussion. Finally, Section VIII concludes this paper.

II. Related Work

This section reviews the studies most relevant to our method, focusing on multi-agent behavior modeling in traffic scenarios and reinforcement post-training for driving planners.

A. Multi-agent Behavior Modeling in Traffic Scenarios

Modeling the joint behavior of multiple interacting agents is essential for advancing autonomous systems, yet it remains challenging because future motion is inherently multimodal, long-horizon, and tightly coupled across participants [5], [22]. Typical distributional regression methods fit parametric continuous distributions such as Gaussian [23] , Laplace [24] to obtain a compact multimodal representation. Other studies incorporate goal anchors [25] or learnable intention queries [23], [26] into the decoding process to generate multi-modal future motions. However, these designs often increase model complexity and memory usage, limiting scalability.

Recently, generative models, notably autoregressive Transformer and diffusion models, have advanced multi-agent planning and simulation [13], [27], [28]. Autoregressive Transformer models cast multi-agent motion modeling as a next-token prediction task [29], [30], with methods such as SMART [27] learning categorical distributions over discrete motion tokens, and MotionLM [28] enabling weighted mode identification via pairwise sampling and a simple rollout aggregation. However, their discrete, sequential decoding can hinder temporal coherence and remain limited to partially joint dependencies.

By contrast, diffusion models offer a compelling alternative, as they excel at modeling complex multimodal distributions while producing temporally consistent trajectories. They have been applied to decision-making, trajectory planning, and traffic simulation [31]–[33]. For example, CTG++ [34] employs a spatiotemporal Transformer that captures the evolving dynamics of multi-agent interactions, and VBD [8] combines a diffusion trajectory model with behavior prediction to produce versatile traffic behaviors. Despite improved behavioral diversity, existing diffusion-based methods often struggle to balance agent interaction modeling with scene-conditioned consistency, leading to scene-inconsistent multi-agent trajectories [8]. Moreover, these methods are susceptible to closed-loop distribution shift and objective misalignment, limiting robustness and reliability in real-world deployment [11]. In contrast, our method is built around this gap: *we retain diffusion's multimodal expressiveness while explicitly coupling scene-conditioned pre-training with RL post-training to optimize closed-loop safety and efficiency.*

B. Reinforcement Post-training for Driving Planners

Reinforcement learning (RL) has been shown to further enhance the capabilities of pretrained models by combining sampling-based exploration with reward-driven policy optimization [11], [35], [36]. Recent studies on reinforcement learning for driving planners generally focus on fine-tuning two major classes of generative models. The first line of work, analogous to RL fine-tuning for large language models, uses autoregressive generation to model each motion token as a continuous distribution and perform policy improvement [14], [37], [38]. However, this approach inherently suffers from conflicts between sequence-level and token-level objectives, and temporal instability induced by sequential decoding. In contrast, diffusion models provide an inherently temporally consistent decision process and produce diverse actions through probabilistic denoising in continuous space, making them particularly well-suited to RL's exploration and exploitation paradigm. For example, TrajHF [15] proposes a human feedback-driven RL fine-tuning scheme that aligns generative trajectory models with diverse human driving preferences. ReCogDrive [16] fine-tunes the diffusion planner using RL with a non-reactive simulator to generate safer and more stable trajectories.

Nevertheless, existing methods are predominantly offline and do not interact with reactive environments; consequently, limited rollouts and exploration often yield marginal performance gains [13], [15], [16]. More importantly, RL training in fully closed-loop settings remains challenging due to non-stationary multi-agent interactions, compounding rollout errors, and high-variance gradient estimates arising from closed-loop environment interaction [12], [16]. Taken together, existing diffusion-based multi-agent planners typically leave two key gaps unresolved: insufficient scene-conditioned denoising and predominantly offline post-training. *Therefore, our method addresses both gaps by coupling a scene-conditioned denoising architecture with analytically tractable reverse-kernel online optimization, further stabilized by dense rewards and variance-gated updates, thereby improving closed-loop cooperative driving performance.*

III. Preliminaries

This section introduces the preliminaries and notation of scene-conditioned diffusion-based planning and reinforcement learning used throughout the paper.

A. Diffusion Model and Diffusion Policies

Denoising Diffusion Probabilistic Models (DDPMs) [39] define a forward noising process together with a learned reverse denoising process for sample generation. In our setting, the clean sample $\mathbf{u}_0$ denotes a future *action chunk*, i.e., a control sequence to be generated for the controlled agents over the planning horizon, conditioned on the scene context $\mathbf{c}$. Given a clean action chunk $\mathbf{u}_0$, the forward diffusion process gradually perturbs it into Gaussian noise through a Markov chain

$$q(\mathbf{u}_{1:K} \mid \mathbf{u}_0) := \prod_{k=1}^{K} q(\mathbf{u}_k \mid \mathbf{u}_{k-1}), \tag{1}$$

where

$$q(\mathbf{u}_k \mid \mathbf{u}_{k-1}) = \mathcal{N}\Big(\mathbf{u}_k; \sqrt{1-\beta_k}\,\mathbf{u}_{k-1}, \beta_k \mathbf{I}\Big), k = 1, \ldots, K. \tag{2}$$

Here, $\beta_k \in (0,1)$ is a predefined noise schedule, $\alpha_k = 1 - \beta_k$, and $\bar{\alpha}_k = \prod_{i=1}^{k} \alpha_i$. The noisy variable at step k admits the closed-form expression

$$\mathbf{u}_k = \sqrt{\bar{\alpha}_k}\,\mathbf{u}_0 + \sqrt{1-\bar{\alpha}_k}\,\boldsymbol{\epsilon}, \qquad \epsilon \sim \mathcal{N}(0, \mathbf{I}). \tag{3}$$

The forward diffusion process is independent of the scene context $\mathbf{c}$ once $\mathbf{u}_0$ is given.

Starting from Gaussian noise $\mathbf{u}_K \sim \mathcal{N}(0, \mathbf{I})$, the reverse process generates actions conditioned on $\mathbf{c}$:

$$p_\theta(\mathbf{u}_{0:K} \mid \mathbf{c}) := p(\mathbf{u}_K) \prod_{k=1}^{K} p_\theta(\mathbf{u}_{k-1} \mid \mathbf{u}_k, \mathbf{c}), \tag{4}$$

where each reverse transition is parameterized as a Gaussian

$$p_\theta(\mathbf{u}_{k-1} \mid \mathbf{u}_k, \mathbf{c}) := \mathcal{N}\big(\mathbf{u}_{k-1}; \mu(\mathbf{u}_k, \mathcal{D}_\theta(\mathbf{u}_k, \mathbf{c}, k)), \sigma_k^2 \mathbf{I}\big) . \tag{5}$$

Here, $\mathbf{u}_k$ denotes the noisy action variable at denoising step k, and $\mathbf{u}_{k-1}$ is its one-step denoised version. The reverse mean $\mu(\cdot)$ is computed by the standard DDPM closed-form expression from the current noisy action chunk $\mathbf{u}_k$ and the denoiser output $\mathcal{D}_\theta(\mathbf{u}_k, \mathbf{c}, k)$, while σ_k^2 is determined by the fixed diffusion schedule. After K reverse steps, the final action chunk $\mathbf{u}_0$ is obtained. The corresponding conditional diffusion policy $\pi_\theta(\mathbf{u}_0 \mid \mathbf{c})$ is defined as the marginal distribution over $\mathbf{u}_0$ induced by the conditioned reverse denoising chain.

B. Markov Decision Process and Policy Optimization

A Markov decision process (MDP) is defined by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P_0, P, R)$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces, P_0 is the initial-state distribution, P is the transition kernel, and R is the immediate reward function. At each time step t, the agent observes a state $s_t \in \mathcal{S}$, samples an action $a_t \in \mathcal{A}$ from the policy $\pi_\theta(\cdot \mid s_t)$, receives reward $R(s_t, a_t)$, and transitions to the next state according to $s_{t+1} \sim P(\cdot \mid s_t, a_t)$.

Starting from $s_0 \sim P_0$, the policy together with the environment dynamics induces a trajectory distribution over state–action sequences, i.e., $\tau = (s_0, a_0, s_1, a_1, \ldots)$. The RL objective is to maximize the expected discounted return

$$\mathcal{J}(\pi_\theta) = \mathbb{E}_{\tau \sim (\pi_\theta, P_0, P)} \left[\sum_{t=0}^{\infty} \gamma^t R(s_t, a_t) \right], \tag{6}$$

where $\gamma \in (0, 1)$ is the discount factor.

Policy-gradient methods optimize this objective using estimators of the form

$$\nabla_\theta \mathcal{J}(\pi_\theta) = \mathbb{E}_{\tau \sim (\pi_\theta, P_0, P)} \left[\sum_{t=0}^{\infty} \nabla_\theta \log \pi_\theta(a_t \mid s_t)\, G_t \right], \tag{7}$$

where

$$G_t := \sum_{\tau \geq t} \gamma^{\tau - t} R(s_\tau, a_\tau) \tag{8}$$

is the *return-to-go* from time t.

In this paper, this generic RL formulation is instantiated through a diffusion-policy optimization problem. The executable control is not sampled in one step; instead, it is generated through a stochastic reverse denoising chain. Accordingly, in Sec. VI-A we reformulate policy optimization as a two-layer MDP, where the inner MDP models the denoising process and the outer MDP models policy–environment interaction.

IV. Problem Statement

We consider the vehicle-cloud cooperative traffic scenario shown in Fig. 1, where a cloud center with powerful computational capability generates future cooperative vehicle trajectories, which are then transmitted to the vehicles for control execution. The core task is multi-agent planning in interactive traffic scenes. A scene is represented as $\mathcal{S} = (\mathbf{x}, \mathbf{u}, \mathbf{c})$, where $\mathbf{x} \in \mathbb{R}^{N_a \times T \times D_x}$ and $\mathbf{u} \in \mathbb{R}^{N_a \times T \times D_u}$ denote the state and control trajectories of N_a agents over a finite horizon T, respectively. D_x and D_u denote the corresponding feature dimensions. The scene context is given by $\mathbf{c} = (\mathbf{c}_a, \mathbf{c}_{mp}, \mathbf{c}_{tl})$, consisting of (i) *agent history* $\mathbf{c}_a$, (ii) *lane graph* $\mathbf{c}_{mp}$, and (iii) *traffic-light states* $\mathbf{c}_{tl}$. We formulate multi-vehicle trajectory planning in a traffic scene as a future trajectory generation task, where planning corresponds to sampling from a target distribution. Specifically, we aim to learn a scene-conditioned *joint* planning policy $\pi_\theta(\cdot \mid \mathbf{c})$ that produces interaction-realistic multi-vehicle behaviors while improving closed-loop *safety* and *efficiency*.

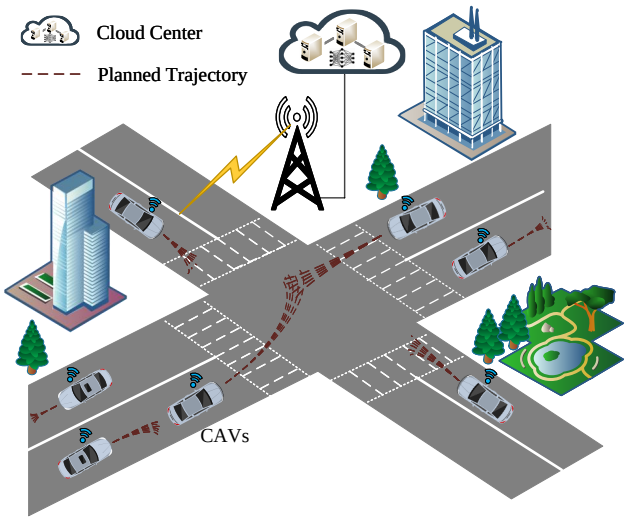


Fig. 1. An illustration of a vehicle-cloud cooperative traffic scenario. A central cloud planner processes scene contexts and generates coordinated multi-agent trajectories using the proposed SCORP framework. Planned trajectories are transmitted to CAVs for execution.

To this end, leveraging diffusion models' strong expressiveness for complex distributions, we first train a multi-agent trajectory diffusion model via imitation learning to capture multimodal interactive behaviors, which jointly generate future trajectories for all vehicles conditioned on the scene context $\mathbf{c}$. However, pretrained diffusion policies π_θ often suffer from distribution shift and objective misalignment. A key question is: *how can we preserve realistic interactive trajectories while promoting safety- and efficiency-oriented cooperative driving?*

Therefore, we leverage reinforcement learning post-training to improve closed-loop planning performance and strengthen cooperative behavior. Specifically, we sample rollouts $\mathbf{x}_0 \sim \pi_\theta(\cdot \mid \mathbf{c})$ and optimize π_θ by maximizing the expected cumulative reward $\mathcal{J}(\pi_\theta)$. Finally, we propose SCORP, a scene-consistent cooperative multi-agent planner that couples condition-enhanced diffusion pre-training with stable online RL post-training.

V. Scene-consistent Multi-Agent Diffusion Pre-Training

This section describes the scene-conditioned multi-agent diffusion planner, the pre-training half of SCORP. It addresses the scene-consistency gap in prior diffusion planners, while Sec. VI leverages the analytically tractable reverse kernel

for stable online optimization. To better balance inter-agent interaction modeling with scene conditioning, we combine cross-attention with AdaLN-Zero modulation to improve the scene consistency and constraint adherence of joint trajectories. We first outline the planner architecture in Fig. 2, and then detail (i) a symmetric scene encoder for comprehensive scene conditioning and (ii) a multi-agent denoising decoder for scene-level interaction modeling.

A. Model Structure of the Multi-agent Diffusion Planner

The planner $\mathcal{P}_\theta \triangleq (\mathcal{E}_{\theta_1}, \mathcal{D}_{\theta_2})$ is composed of a symmetric scene encoder $\mathcal{E}_{\theta_1}$ and a multi-agent denoising decoder $\mathcal{D}_{\theta_2}$, as shown in Fig. 2:

(i) *Symmetric scene encoder* $\mathcal{E}_{\theta_1} : \mathbf{c} \mapsto \tilde{\mathbf{c}}$. The scene context is denoted by $\mathbf{c} = (\mathbf{c}_a, \mathbf{c}_{mp}, \mathbf{c}_{tl})$, where $\mathbf{c}_a$, $\mathbf{c}_{mp}$, and $\mathbf{c}_{tl}$ represent agent history, lane-graph polylines, and traffic-light states, respectively. The encoder uses a query-centric attention Transformer to map the raw scene context into latent scene tokens $\tilde{\mathbf{c}}$. Specifically, scene elements are first transformed into local coordinate frames, and query-centric attention then combines token features with relative geometry to symmetrically capture pairwise relations among heterogeneous scene elements [23]. Detailed descriptions are provided in Sec. V-B.

(ii) *Multi-agent denoising decoder* $\mathcal{D}_{\theta_2} : (\mathbf{u}_k, \tilde{\mathbf{c}}, k) \mapsto \hat{\mathbf{u}}_0$, where $\hat{\mathbf{u}}_0$ is the predicted clean action chunk $\mathbf{u}_0$. Given the encoded scene context $\tilde{\mathbf{c}}$, a noisy action chunk $\mathbf{u}_k$, and the denoising step index k, the decoder $\mathcal{D}_{\theta_2}$ produces a denoising predicted output $\hat{\mathbf{u}}_0$. The decoder first models inter-agent dependencies through self-attention over agent trajectory tokens. It then injects scene information through cross-attention and further modulates the hidden states using AdaLN-Zero, allowing scene conditions to influence the denoising process at multiple layers. This conditioning strategy improves the consistency between generated trajectories and the surrounding scene context. Details of the decoder are given in Sec. V-C.

During pre-training, the scene encoder $\mathcal{E}_{\theta_1}$ and the denoising decoder $\mathcal{D}_{\theta_2}$ are optimized jointly in an end-to-end manner. Algorithm 1 summarizes the loop: sample an expert trajectory and scene context, recover the clean control chunk through inverse dynamics, add noise at a random diffusion step, denoise it conditioned on the encoded scene, roll out the predicted controls through the vehicle dynamics, and optimize a Smooth-ℓ_1 loss in trajectory space. The procedure clarifies that diffusion is defined in action space while supervision is imposed on the induced multi-agent trajectories.

B. Symmetric Scene Context Encoding

Tokenization. The scene context $\mathbf{c}$ consists of three modalities: agent history $\mathbf{c}_a \in \mathbb{R}^{N_a \times T_h \times D_a}$, lane graph polylines $\mathbf{c}_{mp} \in \mathbb{R}^{N_m \times M_w \times D_p}$, and traffic-light states $\mathbf{c}_{tl} \in \mathbb{R}^{N_t \times D_t}$, where N_a, N_m, and N_t denote the numbers of agents, map polylines, and traffic lights, respectively; T_h is the number of observed historical steps; M_w is the number of waypoints in each polyline; and D_a, D_p, and D_t denote the corresponding feature dimensions. For each agent, the historical states are embedded by an MLP and fused with a learnable agent-type embedding to form an agent token. For each map polyline, waypoint-level geometric attributes are first encoded by an MLP and then aggregated by max pooling along the waypoint dimension to obtain a polyline-level feature, which is further fused with a discrete type embedding to form a polyline token. For each traffic light, the stop-point coordinates and signal state are encoded by an MLP to produce a traffic-light token. In this way, heterogeneous scene elements are converted into a unified token representation.

Algorithm 1: Pre-training of the multi-agent diffusion planner

Input: Planner $\mathcal{P}_\theta = (\mathcal{E}_{\theta_1}, \mathcal{D}_{\theta_2})$, dataset $\mathcal{D}_{\text{pre}}$, diffusion steps K, dynamics $f(\cdot)$.

for *each training iteration* **do**
- Sample expert trajectory $\mathbf{x}_{gt}$ and scene context $\mathbf{c}$ from $\mathcal{D}_{\text{pre}}$;
- Get action trajectory $\mathbf{u}_0 \leftarrow f^{-1}(\mathbf{x}_{gt})$;
- Sample diffusion step $k \sim \mathcal{U}\{1, \dots, K\}$ and Gaussian noise $\epsilon \sim \mathcal{N}(0, \mathbf{I})$;
- Form noisy action trajectory $\mathbf{u}_k \leftarrow \sqrt{\bar{\alpha}_k}\mathbf{u}_0 + \sqrt{1-\bar{\alpha}_k}\epsilon$;
- Predict denoised action chunk $\hat{\mathbf{u}}_0 \leftarrow \mathcal{D}_{\theta_2}(\mathbf{u}_k, \mathcal{E}_{\theta_1}(\mathbf{c}), k)$;
- Roll out denoised trajectory $\hat{\mathbf{x}}_0 \leftarrow f(\hat{\mathbf{u}}_0)$;
- Compute loss $\mathcal{L}_\theta \leftarrow \mathcal{SL}_1(\hat{\mathbf{x}}_0 - \mathbf{x}_{gt})$;
- Update $\theta = (\theta_1, \theta_2)$ by backpropagation;

return *Converged parameters* θ

Local coordinates and query-centric attention. Before applying the Transformer encoder, positional attributes are transformed into local coordinate systems. Specifically, each agent is represented relative to its last observed state, and each polyline is represented relative to its first waypoint. This normalization reduces the burden of modeling absolute positions and emphasizes the relative geometric structure of the scene. The token set is then processed by L query-centric Transformer layers following [23]. For a query token i at layer ℓ, let $P_i^{(\ell)} \in \mathbb{R}^D$ denote its embedding, and let $\Omega(i)$ denote the set of neighboring tokens associated with token i. For each neighbor $j \in \Omega(i)$, we express its pose in the local coordinate frame of token i and compute the relative geometric descriptor $R_{ij} = (\Delta x_{ij}, \Delta y_{ij}, \Delta\psi_{ij})$. Let $\mathrm{PE}(R_{ij})$ be a relative positional encoding and let $[\cdot,\cdot]$ denote the concatenation. The query-centric self-attention is

$$P_i'^{(\ell)} = \mathrm{MHSA}\Big(\mathrm{Q} : \big[P_i^{(\ell)}, \mathrm{PE}(R_{ii})\big], \\ \mathrm{K} : \big\{\big[P_j^{(\ell)}, \mathrm{PE}(R_{ij})\big]\big\}_{j\in\Omega(i)}, \ \mathrm{V} : \big\{P_j^{(\ell)} + \mathrm{PE}(R_{ij})\big\}_{j\in\Omega(i)}\Big), \tag{9}$$

where $\mathrm{MHSA}(\cdot_{\text{query}}, \cdot_{\text{key}}, \cdot_{\text{value}})$ denotes the multi-head self-attention operator. We apply the same computation to all tokens and produce the unified scene encoding, denoted by $\tilde{\mathbf{c}} \in \mathbb{R}^{(N_a+N_m+N_t)\times D}$.

C. Multi-agent Trajectory Denoising Decoding

We define diffusion over the joint control chunks of vehicles in the action space and learn a denoiser $\mathcal{D}_{\theta_2}$ conditioned

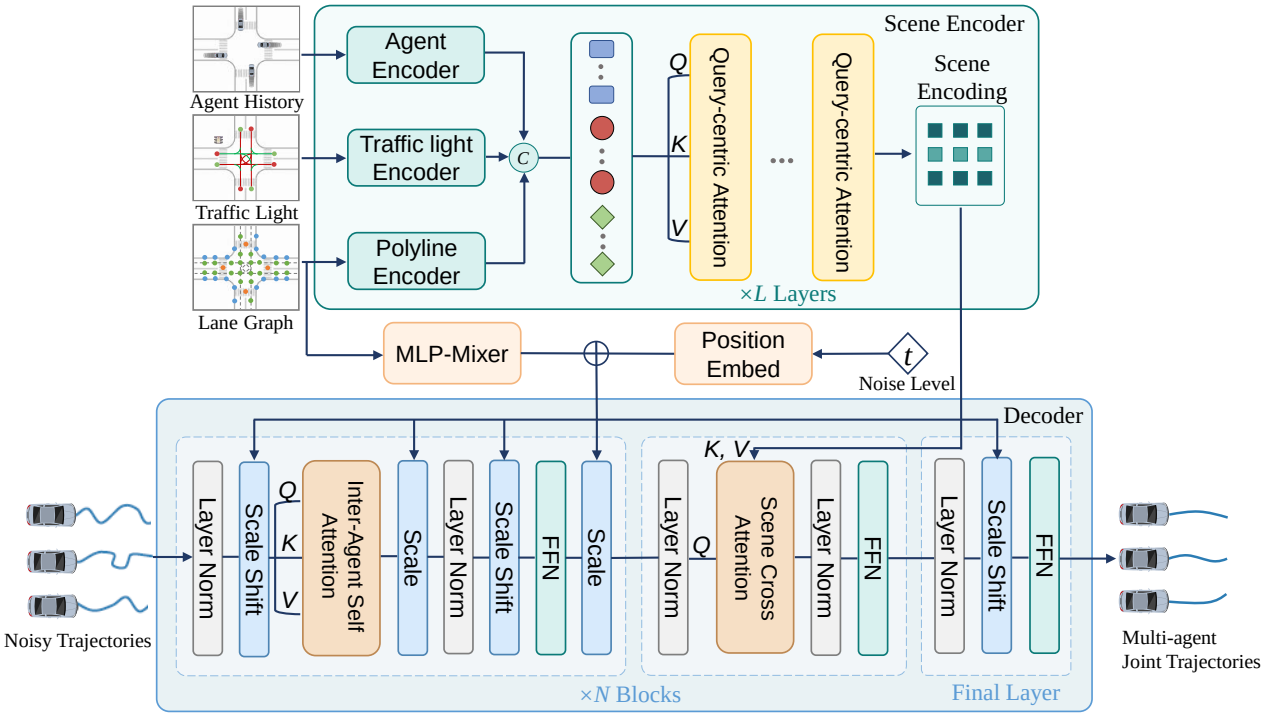


Fig. 2. Architecture of the multi-agent diffusion planner. A symmetric scene encoder models scene elements and their relations in local coordinates through query-centric self-attention. A denoising decoder predicts joint future plans via inter-agent self-attention, scene-conditioned generation, and AdaLN-Zero modulation.

on the scene encoding $\tilde{\mathbf{c}}$ and diffusion step k. The noisy controls are then rolled out through the dynamics $f(\cdot)$ to obtain corresponding noisy state trajectories, which expose vehicle motion and inter-agent interactions more explicitly and better match the trajectory-level supervision used in training. These trajectories are encoded by an MLP and fed into the denoiser, which first models multi-agent dependencies through Transformer blocks with inter-agent self-attention.

Scene-conditioned modeling via cross-attention and AdaLN-Zero. We propose a dual-path scene conditioning mechanism to model the scene-consistent joint trajectory distribution[1]. As shown in Fig. 2, we couple (i) *cross-attention*, which fuses the scene encoding as keys/values to update trajectory tokens, with (ii) *AdaLN-Zero*, which modulates trajectory tokens conditioned on the diffusion timestep and a dense road-feature representation. The former provides a direct information-injection pathway from the scene encoder, while the latter provides a flexible and stable constraint-enforcement pathway that remains active throughout the denoising stack. Specifically, since the road-graph embedding $\tilde{\mathbf{c}}_{mp}$ can be relatively sparse, we first densify it with a lightweight MLP-Mixer block. Here, $\mathrm{MLP}_{\mathrm{tok}}$ denotes a token-mixing MLP that propagates information across tokens, and $\mathrm{MLP}_{\mathrm{ch}}$ denotes a channel-mixing MLP that mixes feature dimensions. The update is written as

$$\begin{aligned}\tilde{\mathbf{c}}_{mp} &\leftarrow \tilde{\mathbf{c}}_{mp} + \mathrm{MLP}_{\mathrm{tok}}\big(\tilde{\mathbf{c}}_{mp}^{\top}\big)^{\top},\\ \tilde{\mathbf{c}}_{mp} &\leftarrow \tilde{\mathbf{c}}_{mp} + \mathrm{MLP}_{\mathrm{ch}}(\tilde{\mathbf{c}}_{mp}).\end{aligned} \tag{10}$$

AdaLN then modulates each trajectory-token feature $\mathbf{x}$ by

$$\mathrm{AdaLN}(\mathbf{x}) = \big(1+\gamma(k,\tilde{\mathbf{c}}_{mp})\big)\odot \mathbf{x} + \beta(k,\tilde{\mathbf{c}}_{mp}), \tag{11}$$

where $\gamma(\cdot)$ and $\beta(\cdot)$ are the scale and shift factors regressed from the sum of the diffusion-step embedding and the dense road features. We further regress a gating factor $\alpha(k,\tilde{\mathbf{c}}_{mp})$ applied to the residual branch. We initialize all α to zero so that the full decoder starts as an identity function and gradually learns to introduce conditioning, which strengthens conditional modulation and improves training stability.

Finally, the noise embedding is propagated to the final layer, which outputs the denoised action sequence $\mathbf{u}_0$. We train the encoder $\mathcal{E}_{\theta_1}$ and the denoising decoder $\mathcal{D}_{\theta_2}$ by minimizing the loss:

$$\mathcal{L}_\theta = \mathbb{E}_{\mathbf{u}_0,k,\mathbf{u}_k\sim p_k(\cdot|\mathbf{u}_0)}\big[\mathcal{SL}_1(\mathbf{x}_0(\mathcal{D}_{\theta_2}(\mathbf{u}_k,\mathcal{E}_{\theta_1}(\mathbf{c}),k))-\mathbf{x}_{gt})\big], \tag{12}$$

where $\mathcal{SL}_1(\cdot)$ is the Smooth-ℓ_1 loss between the expert trajectory $\mathbf{x}_{gt}$ and the trajectory $\mathbf{x}_0$ induced by executing the denoised controls $\mathbf{u}_0$ through the dynamics $f(\cdot)$.

VI. Stable Online Reinforcement Learning Post-Training

A multi-agent diffusion planner trained via imitation learning can capture general driving capabilities, but it often degrades under closed-loop execution due to distribution shift and the

[1]In multi-agent denoising with strong inter-agent interactions, relying on cross-attention alone can underutilize scene context, especially when road-graph features are relatively sparse, thereby increasing the risk of scene inconsistency and scene-constraint violations [8].

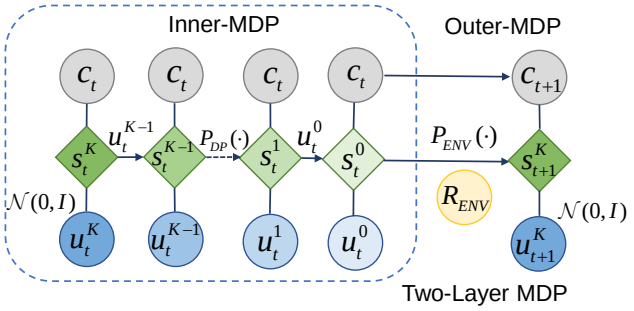


Fig. 3. Two-layer MDP formulation for reinforcement learning. We construct a two-layer MDP consisting of an inner denoising MDP and an outer environment-interaction MDP, and use Gaussian likelihoods at each denoising step so that the denoising process can be optimized via policy gradients.

scarcity of safety-critical interaction data. Moreover, it fails to explicitly promote safety- and efficiency-oriented cooperative behaviors, relying solely on the BC loss.

To address these limitations, we develop a stable *online* RL post-training framework for closed-loop cooperative driving, as shown in Fig. 4. We first formulate a two-layer MDP to enable online optimization. We then co-design dense, well-shaped rewards and VG-GRPO to stabilize online training during closed-loop interaction. Finally, we present the overall RL post-training pipeline.

A. Two-layer MDP Formulation

We now tailor the MDP to diffusion-policy optimization. Unlike standard policies that sample a_t in one shot, a diffusion policy produces the executed action $\mathbf{u}_t^0$ through a *K-step stochastic denoising chain*. Treating the entire denoising chain as a black-box sampler in a single-level MDP leads to poor credit assignment across denoising steps and high-variance gradients, which is exacerbated in *multi-vehicle online* rollouts due to severe non-stationarity from coupled agent interactions. We therefore model denoising as an inner MDP $\mathcal{M}_{\mathrm{DP}}$ with *analytically tractable Gaussian likelihoods* at each step, coupled with an outer environment-interaction MDP $\mathcal{M}_{\mathrm{ENV}}$ that provides closed-loop rewards and constraints. Following [40], their composition yields the two-layer MDP in Fig. 3. We use subscript t for the outer environment step and superscript k for the inner denoising step.

- **State.** $\bar{\mathbf{s}}_t^k = (\mathbf{c}_t, \mathbf{u}_t^k)$, where $\mathbf{c}_t$ is the scene observation at outer time step t and $\mathbf{u}_t^k$ is the intermediate denoising output at denoising step k.
- **Action.** $\bar{\mathbf{a}}_t^k = \mathbf{u}_t^{k-1}$, sampled from the step-wise reverse transition $p_\theta(\mathbf{u}_t^{k-1} \mid \mathbf{u}_t^k, \mathbf{c}_t)$; the final denoised action is executed in the outer MDP.
- **Initial distribution.** $\bar{\mathbf{s}}_0^K = (\mathbf{c}_0, \mathbf{u}_0^K) \sim \bar{\mathbf{P}}_0$, where $\mathbf{c}_0$ is drawn from the outer MDP's initial distribution and $\mathbf{u}_0^K \sim \mathcal{N}(0, \mathbf{I})$.
- **Transition.** The kernel $\bar{\mathbf{P}}(\bar{\mathbf{s}}_{t'}^{k'} \mid \bar{\mathbf{s}}_t^k, \bar{\mathbf{a}}_t^k)$ is defined as

$$\bar{\mathbf{s}}_{t'}^{k'} = \begin{cases} (\mathbf{c}_t, \mathbf{u}_t^{k-1}) & (t', k') = (t, k-1),\ k > 0, \\ (\mathbf{c}_{t+1}, \mathbf{u}_{t+1}^K) & (t', k') = (t+1, K),\ k = 0, \end{cases} \tag{13}$$

  For $k > 0$, transitions occur within the inner denoising chain: $\mathbf{c}_t$ is fixed and the action is updated to $\mathbf{u}_t^{k-1}$. For $k = 0$, we execute $\mathbf{u}_t^0$ in the environment to obtain $\mathbf{c}_{t+1}$ and re-initialize $\mathbf{u}_{t+1}^K \sim \mathcal{N}(0, \mathbf{I})$.
- **Reward.** $\bar{\mathbf{R}}(\bar{\mathbf{s}}_t^k, \bar{\mathbf{a}}_t^k)$ is assigned only when denoising reaches $k = 0$:

$$\bar{\mathbf{R}}(\bar{\mathbf{s}}_t^k, \bar{\mathbf{a}}_t^k) = \begin{cases} 0 & k > 0, \\ R_{\mathrm{ENV}}(\mathbf{c}_t, \mathbf{u}_t^0) & k = 0, \end{cases} \tag{14}$$

  where R_{ENV} is specified in Sec. VI-B.
- **Step-wise likelihood.** In the inner MDP, the policy is identified with the reverse diffusion kernel. $\bar{\pi}_\theta(\bar{\mathbf{a}}_t^k \mid \bar{\mathbf{s}}_t^k)$ can be evaluated by computing the diffusion-policy likelihood of each denoising step along the sampled denoising chain.

$$\begin{aligned} &\bar{\pi}_\theta(\bar{\mathbf{a}}_t^k \mid \bar{\mathbf{s}}_t^k) = p_\theta(\mathbf{u}_t^{k-1} \mid \mathbf{u}_t^k, \mathbf{c}_t) \\ &= \mathcal{N}\big(\mathbf{u}_t^{k-1}; \mu\left(\mathbf{u}_t^k, \mathcal{D}_{\theta_2}\left(\mathbf{u}_t^k, \mathcal{E}_{\theta_1}(\mathbf{c}_t), k\right)\right), \sigma_k^2 \mathbf{I}\big) \end{aligned} \tag{15}$$

  where $p_\theta(\mathbf{u}_t^{k-1} \mid \mathbf{u}_t^k, \mathbf{c}_t)$ is a Gaussian transition with mean $\mu(\cdot)$ calculated from $\mathbf{u}_t^k$ and denoiser output $\mathcal{D}_{\theta_2}\left(\mathbf{u}_t^k, \mathcal{E}_{\theta_1}(\mathbf{c}_t), k\right)$, and variance σ_k^2 specified by the fixed noise schedule. Hence, the step-wise log-likelihood is analytically tractable.
- **Objective.** The objective of the two-layer MDP is

$$\bar{\mathcal{J}}\left(\bar{\pi}_\theta\right) = \mathbb{E}_{\bar{\pi}_\theta, \bar{\mathbf{P}}, \bar{\mathbf{P}}_0}\left[\sum_{t \geq 0} \sum_{\tau \geq t} \gamma^{\tau - t}\, \bar{\mathbf{R}}\left(\bar{\mathbf{s}}_\tau^k, \bar{\mathbf{a}}_\tau^k\right)\right]. \tag{16}$$

B. Dense Planning Reward Design

Rewards determine both the optimization direction and the driving preference. We use simple but fine-grained rule-based rewards that generalize across diverse large-scale scenarios and follow a safety-first principle, with explicit incentives for efficiency. The overall reward is defined as

$$R_{\mathrm{ENV}} = \sum_h w_c\, R_{\mathrm{coll}}(h) + w_o\, R_{\mathrm{offroad}}(h) + w_e\, R_{\mathrm{eff}}(h). \tag{17}$$

where h is the step index within the reward horizon.

Collision. We define the collision reward through SAT-based oriented-box overlap checking, i.e., coll_overlaps$(\cdot)$:

$$R_{\mathrm{coll}}(h) = \mathbb{1}[\text{coll\_overlaps}(h)]. \tag{18}$$

Off-road. We define the off-road reward through signed distances of the four vehicle-box corners to lane boundaries, i.e., off_road$(\cdot)$:

$$R_{\mathrm{offroad}}(h) = \mathbb{1}[\text{off\_road}(h)]. \tag{19}$$

Efficiency. We define the efficiency reward by measuring normalized progress along a road centerline:

$$R_{\mathrm{eff}}(h) = \max\left(\frac{s_{h+1} - s_h}{s_{\max}}, 0\right), \tag{20}$$

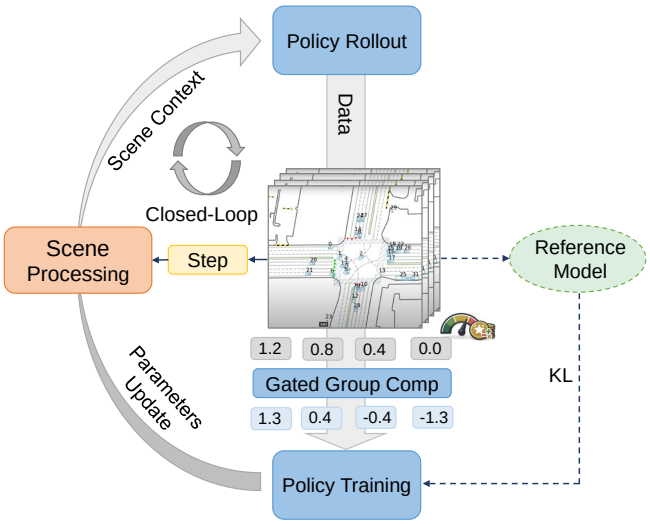


Fig. 4. Online RL post-training framework. The pipeline consists of three components: (1) closed-loop policy rollout, where the policy samples multimodal multi-agent trajectories conditioned on the updated scene context; (2) rule-based reward evaluation, where each trajectory group is scored based on safety and efficiency metrics to compute variance-gated group-relative advantages; and (3) policy training, where mini-batches are randomly sampled from the rollout buffer to optimize the actor network with policy loss and KL regularization.

where s_h is the arc-length projection of the vehicle position onto the centerline and $s_{\max}$ normalizes the progress to $[0, 1]$. In multi-vehicle rollouts, we evaluate rewards at each step over the horizon and average them across the controlled vehicles. This formulation provides dense and well-shaped reward evaluations that enable finer-grained discrimination among sampled trajectories, thus yielding stable learning signals. To avoid *trajectory* dynamic-quality collapse when emphasizing safety, we use three built-in safeguards: dynamics-aware denoising with rollout through $f(\cdot)$ to ensure trajectory feasibility, KL anchoring to the pre-training prior, and short-horizon execute-then-replan in closed loop.

C. Variance-gated Group Relative Policy Optimization

With the two-layer MDP (Sec. VI-A) and reward (Sec. VI-B) in place, we optimize the pretrained diffusion policy by maximizing the objective $\bar{\mathcal{J}}(\bar{\pi}_\theta)$. We propose VG-GRPO for online refinement of the multi-agent diffusion policy. Our design is built on the critic-free GRPO paradigm [17] and extends it with variance-gated advantages and denoising-aware optimization to stabilize training under multi-agent non-stationarity. This is a functional coupling rather than a loose combination: diffusion provides grouped multimodal candidates each step, and VG-GRPO directly consumes their relative ranking signal for online updates.

At each outer environment step t, we sample a group of G candidate rollouts from the old policy $\bar{\pi}_{\theta_{\text{old}}}$. Let $i \in \{1, \ldots, G\}$ index samples in the group. Each sample yields a scalar reward $r_{t,i} := \bar{\mathbf{R}}_{t,i}$, forming the within-group reward set $\{r_{t,1}, \ldots, r_{t,G}\}$.

VG-GRPO objective. We maximize the following objective

$$\bar{\mathcal{J}}(\theta) = \mathbb{E}_{\left(\{\bar{\mathbf{a}}^k_{t,i}\}_{i=1}^G, \bar{\mathbf{s}}_t\right) \sim \bar{\pi}_{\theta_{old}}} \frac{1}{G} \sum_{i=1}^{G} -\beta \mathbb{D}_{KL}\left(\bar{\pi}_\theta || \bar{\pi}_{ref}\right) + \left(\min\left(\rho^k_{t,i}, \text{clip}\left(\rho^k_{t,i}, 1-\varepsilon^-, 1+\varepsilon^+\right)\right) \gamma^k_{\text{denoise}} A_{t,i}\right), \tag{21}$$

where $\rho^k_{t,i}$ is the importance ratio, computed from step-wise likelihoods under the two-layer MDP in Sec. VI-A.

$$\rho^k_{t,i} := \frac{\bar{\pi}_\theta\left(\bar{\mathbf{a}}^k_{t,i} \mid \bar{\mathbf{s}}^k_t\right)}{\bar{\pi}_{\theta_{old}}\left(\bar{\mathbf{a}}^k_{t,i} \mid \bar{\mathbf{s}}^k_t\right)}, \tag{22}$$

and $[\varepsilon^-, \varepsilon^+]$ is the DAPO-style clipping ratio that stabilizes training and encourages exploration, and $\gamma^k_{\text{denoise}} \in (0, 1]$ is a denoising-step discount factor that downweights the policy-gradient contributions of noisier steps, thereby improving training stability [41]. To prevent mode collapse during exploration and preserve general driving capabilities, we regularize post-training with a KL term to the reference policy (set to the pretrained policy here), using the non-negative unbiased estimator [42] and weighting it by β to control the regularization strength:

$$\mathbb{D}_{KL}\left(\bar{\pi}_\theta || \bar{\pi}_{ref}\right) = \frac{\bar{\pi}_{ref}\left(\bar{\mathbf{a}}^k_{t,i} \mid \bar{\mathbf{s}}^k_t\right)}{\bar{\pi}_\theta\left(\bar{\mathbf{a}}^k_{t,i} \mid \bar{\mathbf{s}}^k_t\right)} - \log \frac{\bar{\pi}_{ref}\left(\bar{\mathbf{a}}^k_{t,i} \mid \bar{\mathbf{s}}^k_t\right)}{\bar{\pi}_\theta\left(\bar{\mathbf{a}}^k_{t,i} \mid \bar{\mathbf{s}}^k_t\right)} - 1, \tag{23}$$

Variance-gated advantages. Standard within-group normalization can induce training instability. In simple cases (e.g., short rollouts or near-stationary vehicles), sampled rewards become nearly identical; after GRPO's mean–std normalization, the resulting advantages collapse toward zero, yielding vanishing policy gradients. This amplifies the sensitivity of minibatch gradients to noise and directly destabilizes training. To address this issue, we introduce a variance-gated mechanism, inspired by [18] and adapted to fine-grained selection over group-based trajectory rollouts. Let $\sigma_t \triangleq \text{std}(\{r_{t,1}, \ldots, r_{t,G}\})$, the group relative advantage for sample i is defined as

$$A_{t,i} = \begin{cases} \text{drop this group}, & \sigma_t \le \text{std}_1, \\ r_{t,i} - \text{mean}(\{r_{t,1}, \ldots, r_{t,G}\}), & \text{std}_1 < \sigma_t \le \text{std}_2, \\ \dfrac{r_{t,i} - \text{mean}(\{r_{t,1}, \ldots, r_{t,G}\})}{\text{std}(\{r_{t,1}, \ldots, r_{t,G}\})}, & \sigma_t > \text{std}_2, \end{cases} \tag{24}$$

where std_1 and std_2 denote the tunable gating parameters. During policy sampling, we measure group diversity via the within-group standard deviation of rewards. (i) If the within-group standard deviation is near zero, rollouts are essentially identical, so we discard the group to avoid vanishing gradients. (ii) If it is small but nonzero, the differences may be noise-driven. Normalization would amplify this noise and corrupt the update direction, so we keep the raw differences to retain sample utility while reducing instability. (iii) If it is large, rollouts exhibit meaningful quality gaps and provide informative gradients; we then apply standard group-relative normalization to compute advantages. This variance gate retains informative samples and reduces gradient variance, thereby stabilizing typical GRPO for online RL training of multi-agent diffusion policies.

D. Online RL Post-training Framework

We implement the online RL framework in Fig. 4 with three stages: closed-loop policy rollout, reward evaluation, and policy training. In rollout, the policy repeatedly samples *groups* of multi-agent action chunks from the latest scene context, which is refreshed after each executed action. In reward evaluation, rule-based metrics score collision, off-road behavior, and progress, producing a scalar reward for each sampled chunk. In policy update, grouped samples are used to optimize the diffusion-policy denoiser via **VG-GRPO** (Sec. VI-C); the variance gate is critical for stabilizing multi-vehicle online refinement under non-stationary interactions.

At each outer environment step t, the planner samples G candidate chunks by running the inner denoising chain under the previous behavior policy $\bar{\pi}_{\theta_{\text{old}}}$ conditioned on the current context $\bar{\mathbf{s}}_t^K = (\mathbf{c}_t, \mathbf{u}_t^K)$. Following [41], we use a rolling horizon: plan over T_p steps but execute only the first T_a steps. For group-relative optimization, we compute a scalar reward $\bar{\mathbf{R}}_{t,i}^k$ for each sample $i \in \{1, \dots, G\}$. The environment executes the best-performing sample for interaction, while *all* group samples are retained to compute variance-gated group-relative advantages and update the policy.

We store grouped tuples in the rollout buffer $\mathcal{D}_{\text{itr}}$:

$$\Big(\{\bar{\mathbf{s}}_{t,i}^k\}_{i=1}^G,\ \{\bar{\mathbf{a}}_{t,i}^k\}_{i=1}^G,\ \{\bar{\mathbf{R}}_{t,i}^k\}_{i=1}^G,\ k\Big), \tag{25}$$

where $\bar{\mathbf{s}}_{t,i}^k = (\mathbf{c}_t, \mathbf{u}_{t,i}^k)$ and $\bar{\mathbf{a}}_{t,i}^k = \mathbf{u}_{t,i}^{k-1}$ follow the two-layer MDP in Sec. VI-A. During training, we sample mini-batches from $\mathcal{D}_{\text{itr}}$ by randomly sampling B denoising steps k and group indices i, compute the variance-gated group-relative advantages (Eq. (24)) and likelihoods (Eq. (15)), and optimize the planner parameters by maximizing the VG-GRPO objective (Eq. (21)).

The complete post-training procedure and implementation details are summarized in Algorithm 2.

Algorithm 2: Closed-loop online RL post-training

Input: Current policy $\bar{\pi}_\theta$, dataset $\mathcal{D}_{\text{inter}}$, group size G, denoising steps K, clip ε, KL weight β, thresholds $\text{std}_1, \text{std}_2$.

for *each iteration* **do**
- $\theta_{\text{old}} \leftarrow \theta$; $\mathcal{D}_{\text{itr}} \leftarrow \emptyset$;
- initialize outer state $\mathbf{c}_1$ from $\mathcal{D}_{\text{inter}}$;
- **for** *each outer step* t **do**
  - initialize $\mathbf{u}_{t,i}^K \sim \mathcal{N}(0, \mathbf{I}),\ i = 1, \dots, G$;
  - set $\bar{\mathbf{s}}_{t,i}^K = (\mathbf{c}_t, \mathbf{u}_{t,i}^K),\ i = 1, \dots, G$;
  - **for** $k = K, K-1, \dots, 0$ **do**
    - **if** $k > 0$ **then**
      - sample denoising actions $\bar{\mathbf{a}}_{t,i}^k = \mathbf{u}_{t,i}^{k-1} \sim p_{\theta_{\text{old}}}(\cdot \mid \mathbf{u}_{t,i}^k, \mathbf{c}_t),\ i = 1, \dots, G$;
      - set next inner states $\bar{\mathbf{s}}_{t,i}^{k-1} = (\mathbf{c}_t, \mathbf{u}_{t,i}^{k-1})$ and $\bar{\mathbf{R}}_{t,i}^k = 0,\ i = 1, \dots, G$;
    - **else**
      - compute terminal rewards $\bar{\mathbf{R}}_{t,i}^0 \leftarrow R_{\text{ENV}}(\mathbf{c}_t, \mathbf{u}_{t,i}^0),\ i = 1, \dots, G$;
      - select $i^\star = \arg\max_i \bar{\mathbf{R}}_{t,i}^0$ and execute $\mathbf{u}_{t,i^\star}^0$ in the simulator to obtain $\mathbf{c}_{t+1}$;
    - store $\Big(\{\bar{\mathbf{s}}_{t,i}^k\}_{i=1}^G, \{\bar{\mathbf{a}}_{t,i}^k\}_{i=1}^G, \{\bar{\mathbf{R}}_{t,i}^k\}_{i=1}^G, k\Big)$ in the buffer $\mathcal{D}_{\text{itr}}$;
  - **for** *each update epoch* **do**
    - **for** *each mini-batch* **do**
      - randomly sample B denoising steps k and group indices i;
      - sample the corresponding grouped tuples $(\bar{\mathbf{s}}_{t,i}^k, \bar{\mathbf{a}}_{t,i}^k, \bar{\mathbf{R}}_{t,i}^k, k)$ from $\mathcal{D}_{\text{itr}}$;
      - compute variance-gated advantages $A_{t,i}$ and likelihoods via Eq. (24) and (15);
      - update $\bar{\pi}_\theta$ by maximizing Eq. (21);

return *Converged policy* $\bar{\pi}_\theta$;

VII. Experiments and Discussion

This section reports the closed-loop evaluation of the proposed method. We first describe the datasets, metrics, and implementation details, and then present benchmark comparisons and ablation studies. The evaluation addresses three questions: whether SCORP improves closed-loop safety and efficiency; whether the proposed online post-training framework yields robust gains; and how key modules and design choices, such as AdaLN-Zero, variance gating, and post-training data distribution, contribute to the overall performance.

A. Experimental Setup

Dataset. For pre-training, we use the Waymo Open Motion Dataset (WOMD) [21], containing 486,995 training scenarios and 44,097 validation scenarios; each scenario covers 9 s of real traffic with all participant trajectories and map topology. We perform closed-loop evaluation and ablation studies across 41,590 Testing Interactive scenarios. For post-training, we uniformly sample 20,756 scenarios from the WOMD Validation Interactive split. We then evaluate all sampled scenarios with the pretrained diffusion planner and stratify them into three subsets according to their performance scores: (i) a low-score set with 2,504 failure scenarios involving either inter-vehicle collisions or off-road events; (ii) a high-score set with 5,857 scenarios whose planned trajectories achieve high overall scores; and (iii) a full set containing all available scenarios, which mixes the low-score, high-score, and regular scenarios.

Implementation Details. Scene inputs include up to $N_a = 32$ agents, $N_m = 256$ polylines with $M_w = 30$ waypoints each, and $N_t = 16$ traffic lights. The planner generates a future control sequence of length $T_p = 80$ with a 0.1 s time step. We discard the entire history and condition on the current state to mitigate potential causal confusion and improve closed-loop performance. In closed-loop testing, we execute $T_a = 10$ steps.

The scene encoder uses $L = 6$ query-centric Transformer layers with hidden size $D = 256$. The denoising decoder alternates two block types for three rounds (6 Transformer layers total). MLP-Mixer token/channel dimensions are 64/128. Pre-training uses the log noise schedule in [8] with $\bar{\alpha}_{\min} = 10^{-9}$, scaling 0.0031, and $K = 20$. We pretrain the model with AdamW using a weight decay of 0.01. The learning rate is set to 2×10^{-4}, warmed up for 3000 steps, and decayed by a factor of 0.02 every 3000 steps. We use gradient clipping at 1.0 and BF16 precision, and train for 30 epochs on 4 RTX 4090 GPUs with a global batch size of 32.

Post-training uses group size $G = 10$. Each rollout batch is

TABLE I
CLOSED-LOOP BENCHMARK RESULTS ON THE WOMD TESTING INTERACTIVE SPLIT (PRIMARY: CR, OR, AS; SECONDARY: ADE, KIN)

Method	Primary closed-loop objectives			Secondary diagnostics	
	CR (%)↓	OR (%)↓	AS (m/s)↑	ADE (m)↓	Kin (%)↓
TrafficBotsV1.5 [45]	2.74±0.21	1.79±0.14	8.03±0.48	1.68±0.09	0.26±0.02
SMART-large [27]	2.22±0.09	1.58±0.10	8.34±0.30	1.30±0.01	**0.21±0.01**
VBD [8]	2.46±0.14	1.92±0.18	8.08±0.52	1.41±0.02	0.24±0.01
SMART-tiny-CLSFT [46]	2.10±0.10	1.53±0.12	8.47±0.44	**1.23±0.03**	0.25±0.02
SCORP	**1.89±0.12**	**1.36±0.08**	**8.61±0.46**	1.36±0.04	0.32±0.03

TABLE II
CLOSED-LOOP COMPARISON OF POST-TRAINING METHODS (PRIMARY: CR, OR, AS; SECONDARY: ADE, KIN)

Method	Primary closed-loop objectives			Secondary diagnostics	
	CR (%)↓	OR (%)↓	AS (m/s)↑	ADE (m)↓	Kin (%)↓
Pre-trained only	2.04±0.11	1.68±0.10	8.36±0.42	1.28±0.02	**0.25±0.02**
SFT	2.01±0.07	1.64±0.06	8.37±0.36	**1.15±0.015**	**0.25±0.01**
DPO	1.97±0.13	1.58±0.09	8.15±0.39	1.33±0.04	0.27±0.01
Offline RL	2.18±0.09	1.82±0.14	**8.98±0.68**	1.37±0.05	0.26±0.02
SCORP (Online RL)	**1.89±0.12**	**1.36±0.08**	8.61±0.46	1.36±0.04	0.32±0.03

optimized for 1 epoch with mini-batch size 16 for 10M fine-tuning steps. Following [11], we use a reward horizon of 4 s, reward weights (collision/off-road/progress) $8/1/4$, KL weight $\beta = 0.1$, and learning rate 10^{-5}. For effective exploration, we use DAPO-style clipping $[-0.15, 0.2]$ [18], and clamp a minimum sampling-time standard deviation of $\sigma_{\min}^{\text{sam}} = 0.2$. For training stability, we clamp the per-step log-likelihood standard deviation to $\sigma_{\min}^{\text{prob}} = 0.1$ and use $\gamma_{\text{denoise}} = 0.9$. Moreover, we set the gating thresholds to $\text{std}_1/\text{std}_2 = 0.03/0.06$, corresponding to approximately 10%/20% of the empirical standard deviation of group rewards, as informed by signal-to-noise considerations [43] and advantage-collapse analysis [44]. Post-training runs in BF16 on one NVIDIA 5090 GPU.

Metrics. We report collision rate (CR), off-road rate (OR), average speed (AS), average displacement error (ADE), and kinematic infeasibility (Kin). We treat CR, OR, and AS as *primary* closed-loop objectives reflecting safety and traffic efficiency, and use ADE and Kin as *secondary* diagnostics for imitation fidelity and physical feasibility. CR checks SAT-based oriented-box overlap; OR checks drivable-area boundary crossing; AS is mean per-step displacement; ADE is the ℓ_2 position error to ground truth; and Kin counts violations of acceleration and curvature bounds with limits $6\,\text{m/s}^2$ and $0.3\,\text{m}^{-1}$. For fairness, all methods are evaluated with the same simulator configuration, including scenario split, agent count, horizon, and replanning frequency; values with $\pm$ report mean and standard deviation over repeated closed-loop evaluations in the main comparison tables, while ablation tables report point estimates under the same protocol for compactness.

B. Performance Evaluation

This subsection evaluates interactive trajectory generation using both quantitative benchmark results and representative qualitative cases.

Performance Evaluation. We compare four strong open-source baselines: VBD [8], SMART-large [27], SMART-tiny-CLSFT [46], and TrafficBotsV1.5 [45]. They cover autoregressive, behavior-cloning, and diffusion paradigms. Each scenario runs for 8 s in closed loop with replanning at 1 Hz. Table I shows that SCORP leads on all three primary closed-loop objectives and delivers superior closed-loop planning performance, yielding 10.47%–28.26% and 1.70%–7.22% improvements in safety-related metrics (CR and OR) and efficiency metrics, respectively, over strong representative baselines on WOMD. Compared with SMART-tiny-CLSFT, a state-of-the-art open-source baseline that uses targeted closed-loop supervised fine-tuning, our method reduces CR from 2.10 to 1.89 (-10.0%) and OR from 1.53 to 1.36 (-11.1%), while increasing AS from 8.47 to 8.61 (+1.66%). Relative to the pretrained policy in Table II, online post-training yields 7.4% lower CR, 19.0% lower OR, and 3.0% higher AS. These results demonstrate the effectiveness of the proposed scene-conditioned pre-training and closed-loop RL post-training paradigm.

Post-training Methods. We further compare four post-training strategies, SFT, DPO, offline RL, and online RL, as shown in Table II. All methods improve upon the pre-training-only policy to some extent, but with different objective alignment. SFT mainly improves imitation-oriented diagnostics (ADE). DPO and offline RL improve selected indicators but can degrade the balance between safety and efficiency. SCORP with online RL provides the most consistent co-improvement on the primary closed-loop objectives CR, OR, and AS, reflecting the benefit of coupling diffusion group sampling with VG-GRPO.

Online RL shows a mild ADE increase, indicating a distributional trade-off: closed-loop exploration can reduce one-step imitation accuracy while improving interaction robustness. Kin remains low across methods, all below 0.4%; compared with the pretrained policy, it increases by at most 0.07 percentage points from 0.25 to 0.32, indicating no dynamic-quality collapse under safety-oriented optimization. In summary, through closed-loop policy sampling, online RL continually exposes the model to more out-of-distribution scenarios, thereby improving generalization and robustness under complex interactions and offering greater potential for further gains in closed-loop performance.

Qualitative Visualization. Fig. 5 first illustrates the ability of the proposed planner to generate realistic and interactive trajectories in a real intersection scenario involving about 20 vehicles. For clarity, each vehicle is indexed, and the colored traces behind the vehicles show their 1-second motion history. The time-ordered rollout shows that the vehicles can pass through the scene in an orderly and safe manner under the planned trajectories. For example, when traversing a narrow road segment, Vehicles 0, 1, 20, and 28 adjust yielding and acceleration behaviors to maintain safe spacing and complete cooperative passing, indicating that the planner captures multi-vehicle coordination effectively.

We next demonstrate the ability of SCORP to generate diverse cooperative behaviors. Fig. 6 presents different interaction outcomes obtained from repeated inference in the same scenario. For clarity, Vehicles 0 and 1 are highlighted with red circles, and their historical trajectories are shown. As illustrated in Fig. 6 (a), Vehicle 1 enters the main road before Vehicle 0 because, in this inference instance, Vehicle 0 is relatively far away,

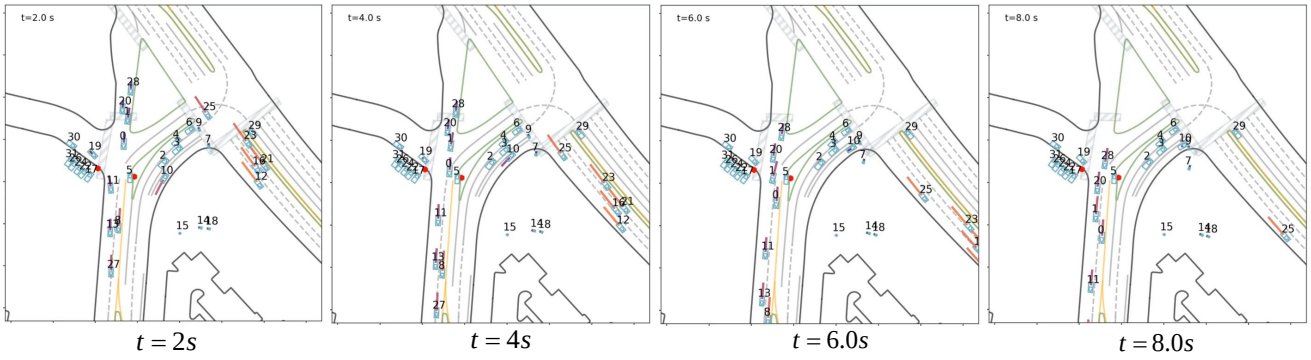


Fig. 5. Closed-loop planning visualizations in real traffic scenarios, showing interactive trajectories. We perform closed-loop simulation over an 8-second horizon using a viewpoint centered on Vehicle 1. The 1-second history trajectory is marked behind the vehicles.

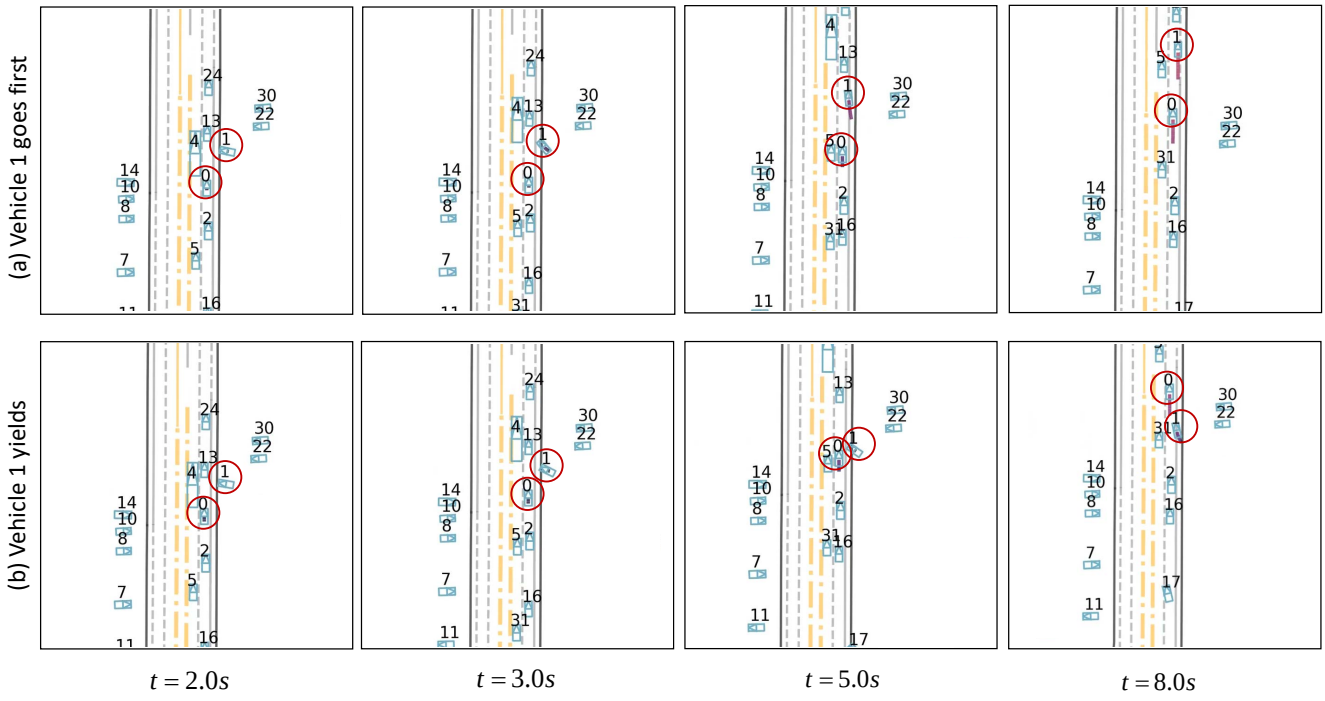


Fig. 6. Visualization of diverse interaction behaviors under repeated inference in the same scenario, where Vehicles 0 and 1 exhibit different interaction outcomes: (a) Vehicle 1 goes first; (b) Vehicle 1 yields.

leading Vehicle 1 to take the lead. By contrast, in Fig. 6 (b), Vehicle 1 yields to Vehicle 0 because, in this instance, Vehicle 0 approaches at a higher speed and from a closer distance, prompting Vehicle 1 to adopt a yielding behavior. These results indicate that SCORP is capable of generating diverse cooperative behaviors, thereby providing diverse samples to support the online reinforcement learning stage for continual improvement of cooperative behavior quality.

We further compare pre-training and RL post-training in an out-of-distribution interaction case. As shown in Fig. 7 (a), the pre-trained model generates a left-turn trajectory for Vehicle 3 but fails to adequately resolve the interaction conflict between Vehicle 3 and Vehicle 20, resulting in a collision at $t = 6.0$ s. In Fig. 7(b), after 1M RL steps, the model learns a more conservative yielding strategy in the challenging left-turn scenario, avoiding collisions at the cost of reduced traffic efficiency. In Fig. 7 (c), after 10M RL steps, the model preserves safety while recovering more flexible and decisive interaction behaviors, and generates more efficient trajectories for moving vehicles such as Vehicles 0, 1, 2, 6, 10, and 18. These qualitative cases show that reward-driven RL post-training progressively reshapes the interaction strategy of the planner, improving the balance between safety and efficiency in closed-loop multi-agent driving.

C. Ablations

Unless otherwise stated, ablations use the same training pipeline and evaluation protocol as Section VII.

Impact of AdaLN-Zero Module. AdaLN-Zero consistently improves conditional generation quality and closed-loop safety. As shown in Table III, introducing AdaLN-Zero reduces the collision rate from 2.11% to 2.04% and, more notably, lowers

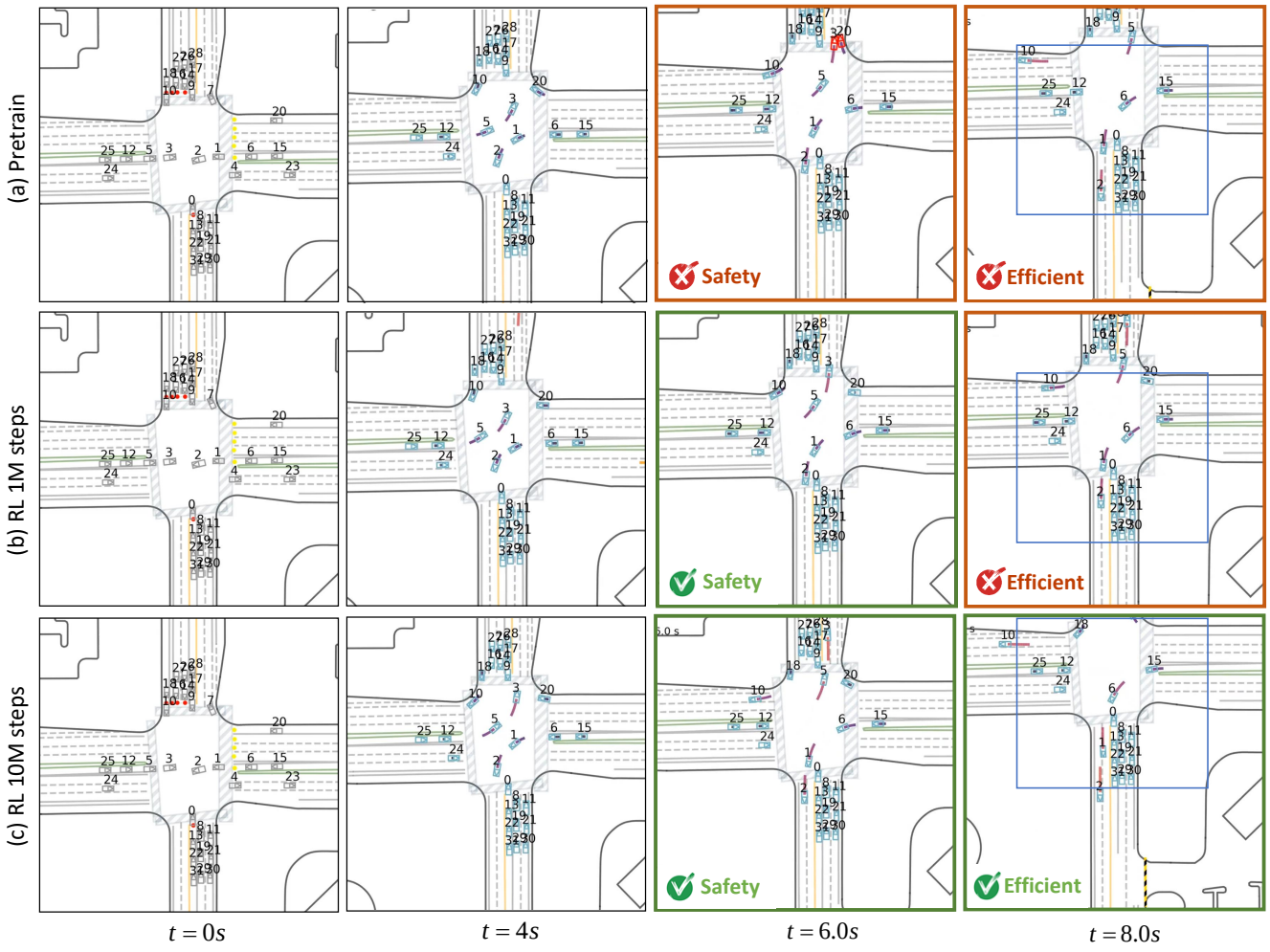
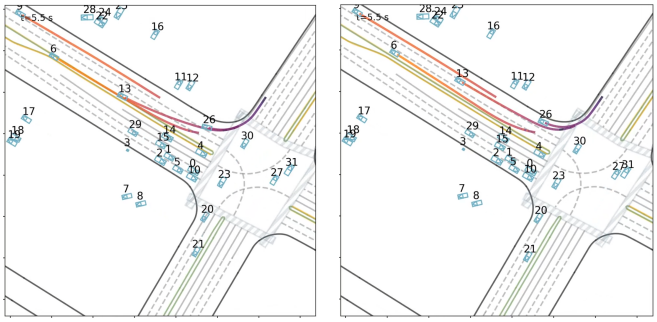


Fig. 7. Qualitative comparison between pre-training and RL post-training over an 8-second closed-loop rollout. (a) Pretrained planner. (b) Planner after 1M RL post-training steps. (c) Planner after 10M RL post-training steps. Safety and efficiency steadily improve as post-training progresses.

TABLE III
ABLATION ON ADALN-ZERO MODULE

Setting	CR (%)↓	OR (%)↓	AS (m/s)↑	ADE (m)↓	Kin (%)↓
w/o AdaLN-Zero	2.11	2.05	**8.40**	1.30	0.26
with AdaLN-Zero	**2.04**	**1.68**	8.36	1.28	0.25



(a) without AdaLN-Zero module (b) with AdaLN-Zero module

Fig. 8. Qualitative comparison with and without the AdaLN-Zero module

the off-road rate from 2.05% to 1.68%, indicating substantially improved scene consistency and boundary adherence. This improvement arises because condition-driven modulation reshapes denoising features and strengthens awareness of road geometry, enabling the model to exploit scene cues more effectively than cross-attention alone and to generate trajectories that better comply with scene constraints. The resulting improvement in boundary compliance further translates into stronger closed-loop stability.

Fig. 8 provides a qualitative comparison on sharp right-turn cases. Without AdaLN-Zero, Vehicles 6 and 13 drift outward and hit the median boundary. With AdaLN-Zero, turn trajectories are more compact and maintain safer boundary margins.

Impact of the Variance-gated Mechanism. Table IV analyzes the effect of our *variance-gated* mechanism via ablations over several settings, including a non-gated baseline and multiple threshold combinations. Without variance gating, post-training collapses at ∼0.5M steps, evidenced by persistently large gradient-norm fluctuations and a sudden spike in the KL term. The last stable checkpoint also exhibits degraded performance across metrics. We then vary the gating thresholds. With $\text{std}_1/\text{std}_2 = 0.00/0.06$, collapse is delayed to ∼2.0M steps and key closed-loop safety and efficiency metrics

TABLE IV
ABLATION OF THE VARIANCE-GATED MECHANISM (PERFORMANCE IS EVALUATED AT THE LAST CHECKPOINT BEFORE COLLAPSE)

Gating $\text{std}_1/\text{std}_2$	Collapse step	CR[%]↓	OR[%]↓	AS$[m/s]$↑	ADE$[m]$↓	Kin[%]↓
w/o gating	≈ 0.5M	2.15	2.03	8.05	1.74	0.40
0.03/0.06	–	**1.89**	**1.36**	**8.61**	1.36	0.32
0.00/0.06	≈ 2.0M	2.01	1.57	8.42	1.42	0.30
0.03/0.09	–	1.96	1.50	8.49	**1.30**	**0.29**

TABLE V
ABLATION ON POST-TRAINING DATA DISTRIBUTION

Dataset type	CR (%)↓	OR (%)↓	AS (m/s)↑	ADE (m)↓	Kin (%)↓
High-score	1.95	**1.34**	8.53	**1.30**	**0.31**
Low-score	2.18	1.99	8.49	1.40	0.37
Full	**1.89**	1.36	**8.61**	1.36	0.32

improve slightly, indicating that std_2 contributes to training stability. In contrast, $0.03/0.06$ and $0.03/0.09$ prevent collapse completely, suggesting that std_1 is critical for stability by mitigating advantage degeneration when within-group samples are nearly identical. Among stable settings, $0.03/0.06$ achieves the best performance. A larger std_2 appears overly conservative and weakens the effective learning signal, suggesting that moderate thresholds strike a better balance between stability and performance gains.

Impact of Post-training Data Distribution. Table V analyzes how the composition of post-training scenes affects final performance. Training only on the high-score subset yields limited gains, because this subset contains many relatively simple scenes and thus provides weak optimization signals due to insufficient diversity among sampled trajectories. By contrast, training only on the low-score subset substantially increases the collision and off-road rates, while yielding only a slight increase in average speed, indicating clear degradation in overall performance. This suggests that when post-training data are overly concentrated on hard failure cases, policy updates may be dominated by more explicit local efficiency-oriented corrective objectives and fail to establish a stable global optimization direction. The best overall result is obtained on the full dataset, which mixes high-score, low-score, and regular scenes. This observation indicates that a balanced post-training data distribution is important for stable reinforcement post-training: it maintains broader policy coverage and more stable sampling behavior, and thus provides more informative gradient signals for effective optimization and better generalization.

VIII. CONCLUSION

In this paper, we introduced SCORP, a cooperative multi-agent planner that couples condition-enhanced diffusion pre-training with stable online RL post-training in reactive closed-loop environments. For pre-training, SCORP strengthens scene consistency and road adherence by combining inter-agent self-attention with dual-path scene conditioning. For post-training, SCORP achieves stable online learning by combining dense, well-shaped rewards with our proposed VG-GRPO, mitigating advantage collapse and gradient instability while strengthening closed-loop cooperative behaviors. Extensive experiments demonstrate superior closed-loop planning performance on key *safety* and *efficiency* metrics, outperforming state-of-the-art baselines on WOMD and across alternative post-training paradigms.

REFERENCES

[1] J. Luo, T. Zhang, R. Hao, D. Li, C. Chen, Z. Na, and Q. Zhang, "Real-time cooperative vehicle coordination at unsignalized road intersections," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 5, pp. 5390–5405, 2023.

[2] H. Bai, J. Luo, H. Li, X. Zhao, and Y. Wang, "A robust cooperative vehicle coordination framework for intersection crossing," *IEEE Trans. Veh. Technol.*, vol. 75, no. 4, pp. 5428–5442, 2026.

[3] X. Zhao, C. Wen, X. Zhu, Y. Wang, H. Bai, and W. Dou, "TripleMixer: A triple-domain mixing model for point cloud denoising under adverse weather," *IEEE Transactions on Image Processing*, vol. 34, pp. 7712–7727, 2025.

[4] H. Bai, T. Zhang, C. Guo, Y. Wang, X. Zhao, and H. Zhu, "Robust real-time coordination of CAVs: A distributed optimization framework under uncertainty," *arXiv preprint arXiv:2508.21322*, 2025.

[5] N. Nayakanti, R. Al-Rfou, A. Zhou, K. Goel, K. S. Refaat, and B. Sapp, "Wayformer: Motion forecasting via simple and efficient attention networks," *arXiv preprint arXiv:2207.05844*, 2022.

[6] M. Janner, Y. Du, J. Tenenbaum, and S. Levine, "Planning with diffusion for flexible behavior synthesis," in *Proc. Int. Conf. Mach. Learn. (ICML)*. PMLR, 2022, pp. 9902–9915.

[7] Y. Zheng, R. Liang, K. Zheng, J. Zheng, L. Mao, J. Li, W. Gu, R. Ai, S. E. Li, X. Zhan *et al.*, "Diffusion-based planning for autonomous driving with flexible guidance," *arXiv preprint arXiv:2501.15564*, 2025.

[8] Z. Huang, Z. Zhang, A. Vaidya, Y. Chen, C. Lv, and J. F. Fisac, "Versatile behavior diffusion for generalized traffic agent simulation," *arXiv preprint arXiv:2404.02524*, 2024.

[9] Z. Huang, Z. Zhou, T. Cai, Y. Zhang, and J. Ma, "MDG: Masked denoising generation for multi-agent behavior modeling in traffic environments," *arXiv preprint arXiv:2511.17496*, 2025.

[10] Y. Lu, J. Fu, G. Tucker, X. Pan, E. Bronstein, R. Roelofs, B. Sapp, B. White, A. Faust, S. Whiteson *et al.*, "Imitation is not enough: Robustifying imitation with reinforcement learning for challenging driving scenarios," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*. IEEE, 2023, pp. 7553–7560.

[11] H. Li, T. Li, J. Yang, H. Tian, C. Wang, L. Shi, M. Shang, Z. Lin, G. Wu, Z. Hao *et al.*, "PlannerRFT: Reinforcing diffusion planners through closed-loop and sample-efficient fine-tuning," *arXiv preprint arXiv:2601.12901*, 2026.

[12] H. Gao, S. Chen, B. Jiang, B. Liao, Y. Shi, X. Guo, Y. Pu, H. Yin, X. Li, X. Zhang *et al.*, "RAD: Training an end-to-end driving policy via large-scale 3DGS-based reinforcement learning," *arXiv preprint arXiv:2502.13144*, 2025.

[13] Z. Huang, X. Weng, M. Igl, Y. Chen, Y. Cao, B. Ivanovic, M. Pavone, and C. Lv, "Gen-drive: Enhancing diffusion generative driving policies with reward modeling and reinforcement learning fine-tuning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2025, pp. 3445–3451.

[14] Z. Peng, W. Luo, Y. Lu, T. Shen, C. Gulino, A. Seff, and J. Fu, "Improving agent behaviors with RL fine-tuning for autonomous driving," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Springer, 2024, pp. 165–181.

[15] D. Li, J. Ren, Y. Wang, X. Wen, P. Li, L. Xu, K. Zhan, Z. Xia, P. Jia, X. Lang *et al.*, "Fine-tuning generative trajectory model with reinforcement learning from human feedback," *arXiv e-prints*, 2025.

[16] Y. Li, K. Xiong, X. Guo, F. Li, S. Yan, G. Xu, L. Zhou, L. Chen, H. Sun, B. Wang *et al.*, "RecogDrive: A reinforced cognitive framework for end-to-end autonomous driving," *arXiv preprint arXiv:2506.08052*, 2025.

[17] D. Guo, D. Yang, H. Zhang, J. Song, P. Wang, Q. Zhu, R. Xu, R. Zhang, S. Ma, X. Bi *et al.*, "DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning," *arXiv preprint arXiv:2501.12948*, 2025.

[18] Q. Yu, Z. Zhang, R. Zhu, Y. Yuan, X. Zuo, Y. Yue, W. Dai, T. Fan, G. Liu, L. Liu *et al.*, "DAPO: An open-source LLM reinforcement learning system at scale," *arXiv preprint arXiv:2503.14476*, 2025.

[19] K. Zhang, Y. Zuo, B. He, Y. Sun, R. Liu, C. Jiang, Y. Fan, K. Tian, G. Jia, P. Li *et al.*, "A survey of reinforcement learning for large reasoning models," *arXiv preprint arXiv:2509.08827*, 2025.

[20] W. Peebles and S. Xie, "Scalable diffusion models with transformers," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2023, pp. 4195–4205.

[21] S. Ettinger, S. Cheng, B. Caine, C. Liu, H. Zhao, S. Pradhan, Y. Chai, B. Sapp, C. R. Qi, Y. Zhou *et al.*, "Large scale interactive motion forecasting for autonomous driving: The Waymo Open Motion Dataset," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 9710–9719.

[22] S. Chen, B. Jiang, H. Gao, B. Liao, Q. Xu, Q. Zhang, C. Huang, W. Liu, and X. Wang, "VADv2: End-to-end vectorized autonomous driving via probabilistic planning," *arXiv preprint arXiv:2402.13243*, 2024.

[23] S. Shi, L. Jiang, D. Dai, and B. Schiele, "MTR++: Multi-agent motion prediction with symmetric scene modeling and guided intention querying," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 46, no. 5, pp. 3955–3971, 2024.

[24] Z. Zhou, Z. Wen, J. Wang, Y.-H. Li, and Y.-K. Huang, "QCNext: A next-generation framework for joint multi-agent trajectory prediction," *arXiv preprint arXiv:2306.10508*, 2023.

[25] J. Gu, C. Sun, and H. Zhao, "DenseTNT: End-to-end trajectory prediction from dense goal sets," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 15 303–15 312.

[26] B. Varadarajan, A. Hefny, A. Srivastava, K. S. Refaat, N. Nayakanti, A. Cornman, K. Chen, B. Douillard, C. P. Lam, D. Anguelov *et al.*, "Multipath++: Efficient information fusion and trajectory aggregation for behavior prediction," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2022, pp. 7814–7821.

[27] W. Wu, X. Feng, Z. Gao, and Y. Kan, "Smart: Scalable multi-agent real-time motion generation via next-token prediction," *Adv. Neural Inf. Process. Syst.*, vol. 37, pp. 114 048–114 071, 2024.

[28] A. Seff, B. Cera, D. Chen, M. Ng, A. Zhou, N. Nayakanti, K. S. Refaat, R. Al-Rfou, and B. Sapp, "MotionLM: Multi-agent motion forecasting as language modeling," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2023, pp. 8579–8590.

[29] Z. Zhou, H. Haibo, X. Chen, J. Wang, N. Guan, K. Wu, Y.-H. Li, Y.-K. Huang, and C. J. Xue, "BehaviorGPT: Smart agent simulation for autonomous driving with next-patch prediction," *Adv. Neural Inf. Process. Syst.*, vol. 37, pp. 79 597–79 617, 2024.

[30] Y. Yuan, X. Weng, Y. Ou, and K. M. Kitani, "Agentformer: Agent-aware transformers for socio-temporal multi-agent forecasting," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2021, pp. 9813–9823.

[31] M. Jiang, Y. Bai, A. Cornman, C. Davis, X. Huang, H. Jeon, S. Kulshrestha, J. Lambert, S. Li, X. Zhou *et al.*, "SceneDiffuser: Efficient and controllable driving simulation initialization and rollout," *Adv. Neural Inf. Process. Syst.*, vol. 37, pp. 55 729–55 760, 2024.

[32] C. Jiang, A. Cornman, C. Park, B. Sapp, Y. Zhou, D. Anguelov *et al.*, "MotionDiffuser: Controllable multi-agent motion prediction using diffusion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2023, pp. 9644–9653.

[33] 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 *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*. IEEE, 2023, pp. 3560–3566.

[34] 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 *Proc. Conf. Robot Learn. (CoRL)*. PMLR, 2023, pp. 144–177.

[35] D. Zhang, J. Liang, K. Guo, S. Lu, Q. Wang, R. Xiong, Z. Miao, and Y. Wang, "Carplanner: Consistent auto-regressive trajectory planning for large-scale reinforcement learning in autonomous driving," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2025, pp. 17 239–17 248.

[36] B. Jiang, S. Chen, Q. Zhang, W. Liu, and X. Wang, "AlphaDrive: Unleashing the power of VLMs in autonomous driving via reinforcement learning and reasoning," *arXiv preprint arXiv:2503.07608*, 2025.

[37] Q. Li, X. Jia, S. Wang, and J. Yan, "Think2Drive: Efficient reinforcement learning by thinking with latent world model for autonomous driving (in CARLA-v2)," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*. Springer, 2024, pp. 142–158.

[38] X. Tang, M. Kan, S. Shan, and X. Chen, "Plan-R1: Safe and feasible trajectory planning as language modeling," *arXiv preprint arXiv:2505.17659*, 2025.

[39] A. Q. Nichol and P. Dhariwal, "Improved denoising diffusion probabilistic models," in *Proc. Int. Conf. Mach. Learn. (ICML)*. PMLR, 2021, pp. 8162–8171.

[40] K. Black, M. Janner, Y. Du, I. Kostrikov, and S. Levine, "Training diffusion models with reinforcement learning," in *Proc. ICML 2023 Workshop on Structured Probabilistic Inference and Generative Modeling*, 2023.

[41] A. Z. Ren, J. Lidard, L. L. Ankile, A. Simeonov, P. Agrawal, A. Majumdar, B. Burchfiel, H. Dai, and M. Simchowitz, "Diffusion policy policy optimization," in *Proc. CoRL 2024 Workshop on Mastering Robot Manipulation in a World of Abundant Data*, 2024.

[42] J. Schulman, "Approximating KL divergence," *http://joschu.net/blog/kl-approx.html*, 2020.

[43] J. Roberts and R. Tedrake, "Signal-to-noise ratio analysis of policy gradient algorithms," *Advances in neural information processing systems*, vol. 21, 2008.

[44] H. Zhong, J. Zhai, L. Song, J. Bian, Q. Liu, and T. Tan, "Rc-grpo: Reward-conditioned group relative policy optimization for multi-turn tool calling agents," *arXiv preprint arXiv:2602.03025*, 2026.

[45] Z. Zhang, C. Sakaridis, and L. Van Gool, "TrafficBots V1.5: Traffic simulation via conditional VAEs and transformers with relative pose encoding," *arXiv preprint arXiv:2406.10898*, 2024.

[46] Z. Zhang, P. Karkus, M. Igl, W. Ding, Y. Chen, B. Ivanovic, and M. Pavone, "Closed-loop supervised fine-tuning of tokenized traffic models," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2025, pp. 5422–5432.