Flow-ERD: Agent-type Aware Flow Matching with Entropy-Regularized Distillation for Diverse Traffic Simulation

Seulbin Hwang*, Kiyoung Om*, Daejung Kim, and Jinhan Lee†

***Abstract*— Realistic and diverse traffic simulation is essential to autonomous driving development. Yet prevailing benchmarks predominantly reward realism, and recent methods have optimized accordingly, leaving diversity underexplored. We introduce Flow-ERD, a multi-agent simulator that pursues realism and diversity jointly. Its backbone, Agent-Type Aware Flow Matching (AFM), couples flow matching's multi-modal expressiveness with type-specific kinematic execution. It preserves fine-grained diversity while keeping motions consistent with each agent type. A second stage, Entropy-Regularized Distillation (ERD), fine-tunes the closed-loop rollout distribution with an entropy-regularized reverse-KL objective. This mitigates covariate shift while explicitly preventing collapse onto high-density modes. We evaluate Flow-ERD with a log-free diversity metric alongside standard realism scores. Flow-ERD ranks first on the WOSAC test benchmark and dominates the realism–diversity Pareto front among reproducible baselines. Our project page is available here.**

I. Introduction

Traffic simulation has become core infrastructure for autonomous driving, supporting controlled validation before public-road deployment as well as the development of AV planning policies [1], [2]. For simulation to serve these roles, the surrounding agents, including vehicles, cyclists, and pedestrians, must be *realistic*, imitating real-world traffic behavior and reacting to one another in a closed loop; and *diverse*, spanning the multiple plausible futures of a scene to ensure the ego policy's robustness [3], [4]. These properties must hold jointly, not as alternatives (Fig. 1).

Generative models are well-suited to capturing these properties, learning the distribution of traffic-agent behavior from large-scale data [5] with expressive architectures [6], [7]. Indeed, recent learning-based simulators [8]–[14] have made substantial progress on realism, especially under benchmarks such as the Waymo Open Sim Agents Challenge (WOSAC) [3]. The benchmark's realism score, however, is measured against a single logged future, and thus cannot distinguish a model that merely fits the logged future from one that captures diverse plausible behaviors. As models are increasingly optimized for this benchmark, diversity has been acknowledged but rarely treated as equally important as realism: it is often left to a sampling hyperparameter [9] or assessed only qualitatively [15], [16].

This gap also manifests in how simulators are designed and trained. By design, backbones trade realism against diversity. Next-token-prediction based methods [8]–[10], [17] draw each action from a predefined discrete vocabulary derived from logged data. Because they encode patterns already present in the data, they provide an inductive bias toward realistic, type-compatible motion. However, the fixed vocabulary collapses fine-grained motion onto a coarse set of tokens, which inherently bounds the attainable diversity [10]. Continuous representations [11], [12], [18], especially diffusion models, remove this bottleneck and are effective at modeling complex multi-modal distributions [19], yet this expressiveness does not guarantee realism. Continuous policies must learn valid motion implicitly, so occasional type-incompatible predictions can be fed back into the closed-loop context and amplify rollout errors.

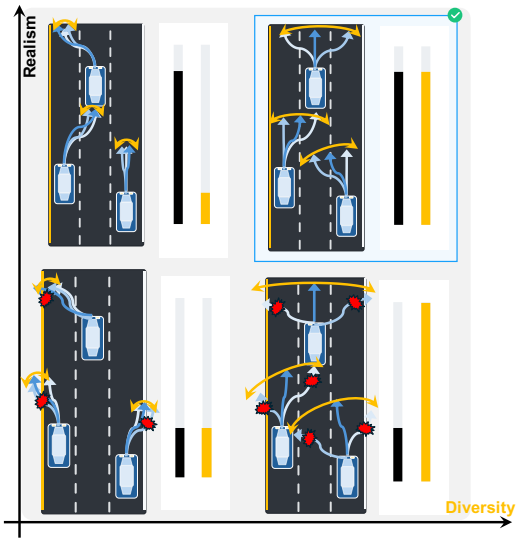

Fig. 1: Low-diversity rollouts concentrate on a dominant behavior, whereas low-realism rollouts deviate from plausible traffic motion. Flow-ERD targets the desired regime of realistic and diverse closed-loop rollouts.

On the training side, backbones trained in open loop are deployed in closed loop, leading to covariate shift [20], in which a model's own actions feed into the next prediction and accumulate error over horizons. Existing methods address this by augmenting the data with log-proximal rollouts [15], [21], [22], by using reinforcement learning to maximize a realism score [16], [23], or by using a generative adversarial reward [24]. While these methods reduce compounding closed-loop error, it remains unclear whether they do so by narrowing the distribution of generated behaviors.

To this end, we introduce **Flow-ERD**, a multi-agent sim-

* Equal contribution;† Corresponding author.
All authors are with NAVER LABS Corp., Republic of Korea (e-mail: {h.sb, se99an, daejung.kim, jinhan.lee}@naverlabs.com).

ulator that jointly addresses realism and diversity. It is built on an **Agent-type Aware Flow-Matching** (AFM) backbone which adopts flow matching [25]—a diffusion-family continuous generator—for its multi-modal expressiveness, and grounds it in kinematics to keep that diversity realizable. During closed-loop rollout, AFM samples continuous actions and executes them through agent-type-specific transitions, avoiding token-codebook limits and type-incompatible unconstrained motion.

The second component, **Entropy-Regularized Distillation** (ERD), preserves the diversity while addressing the covariate shift problem. ERD fine-tunes the closed-loop rollout distribution with an entropy-regularized reverse-KL objective that drives it toward the data distribution while mitigating collapse toward high-density modes.

We evaluate Flow-ERD on the WOSAC using the standard realism meta metric (RMM) together with our proposed **Cross-Pair Diversity** (CPD), a log-free metric that measures rollout spread across multiple samples. This joint evaluation establishes a comprehensive realism–diversity landscape, setting the stage for our core findings.

Our contributions are threefold:

- We propose Agent-Type Aware Flow Matching (AFM), which captures fine-grained multi-modal behaviors through continuous flow matching while preserving realism through agent-type aware transitions.
- We introduce Entropy-Regularized Distillation (ERD), a closed-loop fine-tuning method that mitigates covariate shift while explicitly preserving multi-modality.
- On the WOSAC [3] test benchmark, AFM achieves the state-of-the-art kinematic score, and Flow-ERD ranks first overall in realism; on the validation split, both attain the highest rollout diversity among reproducible baselines, dominating the realism–diversity Pareto front.

II. Related Works

A. Learning-Based Multi-Agent Simulation

Next-token prediction (NTP)- based models have been among the strongest WOSAC submissions [8]–[10], [17], [26]. Their discrete vocabularies provide an inductive bias toward realism by composing motion from data-derived or rule-based primitives that preserve type-specific motion patterns for pedestrians, vehicles, and cyclists. This constrains agents to data-supported, type-compatible motions and prohibits feeding implausible predictions back into their input conditions. However, this also limits diversity. Any motion absent from the vocabulary can only be approximated by its nearest available token, bounding the generated behaviors by the coverage of the token vocabulary [10].

A natural remedy is to replace discrete tokens with continuous outputs. UniMM [14] unifies discrete NTP-based models and continuous mixture models under a common mixture-model view, covering both anchor-free and anchor-based variants. However, continuity alone is not sufficient for diversity, as continuous mixture models still capture multi-modality through a finite set of components, and anchor-based variants further tie coverage to predefined anchors, limiting their expressiveness.

Diffusion-based simulators model joint futures in continuous space, naturally supporting multimodality and controllability without committing to a fixed number of modes or anchors [11], [12], [18], [22], [27]–[29]. Although they remove the fixed-vocabulary bottleneck, models must learn valid motion supports implicitly from data. Naive modeling of its continuous state or action representation may therefore generate type-incompatible motions, such as lateral slip for vehicles and cyclists or unnecessarily constrained pedestrian behavior. Fed back as closed-loop history, such states may amplify distribution shift more readily than in token-based systems. Our method samples multi-modal continuous kinematic actions and executes them through type-specific transitions, yielding type-compatible motion.

B. Mitigating Covariate Shift in Traffic Simulation

Learning-based traffic simulators are typically pretrained by behavior cloning (BC), conditioned on logged histories during training but on their own generated states at deployment, causing the standard covariate-shift problem [20], [30]. Recent methods therefore fine-tune pretrained models on closed-loop rollouts to improve realism. One line constructs supervision from log-proximal rollouts: CAT-K [15] selects the top-K action token whose next state is closest to the ground truth (GT) and trains on the resulting DaD-style recovery tokens [31]; RoaD [21] samples many rollouts, filters them by GT distance, and treats the retained rollouts as demonstrations; and LangTraj [22] adapts this principle to diffusion by denoising candidates from lightly noised GT trajectories and learning recovery actions.

Another line directly optimizes realism-oriented objectives. SMART-R1 [16] and RLFTSim [23] apply reinforcement fine-tuning on the WOSAC Realism Meta Metric (RMM) and DecompGAIL [24] stabilizes GAIL [30] in multi-agent settings by decomposing the discriminator into ego–map and ego–neighbor terms.

Although these methods obtain supervision differently, they primarily pull rollouts toward the recorded trajectory, RMM statistics, or the logged-data manifold. This improves closed-loop stability, but leaves scenario-conditioned diversity implicit. In contrast, our method fine-tunes on the model's own rollouts with an entropy-regularized distribution-matching objective that balances covariate-shift reduction with multimodality preservation.

III. Preliminaries

A. Multi-Agent Driving Simulation

A scenario is specified by a road map $\mathcal{M}$ and N traffic participants indexed by $i \in \{1, \dots, N\}$, each with agent type $c_i \in \mathcal{C} = \{\text{VEH}, \text{CYC}, \text{PED}\}$. Agent i at time t has modeled planar state $\mathbf{s}_t^i = (\mathbf{p}_t^i, \psi_t^i) \in \mathcal{S}$, where $\mathbf{p}_t^i \in \mathbb{R}^2$ is the agent's center position, ψ_t^i is the heading and $\mathbf{b}^i = (\ell^i, w^i)$ denotes the 2D box size. The joint scene state is $\mathbf{s}_t = (\mathbf{s}_t^1, \dots, \mathbf{s}_t^N)$. We denote the historical scene context by $\mathcal{H}_t = (\mathbf{s}_{t-L+1:t}, \mathcal{M})$, where L is the history length, and let $\mathcal{H}_0$

be the initial history context. Starting from $\mathcal{H}_0$, a simulator evolves the scene over horizon T and produces a state rollout $\tau = (\mathbf{s}_1, \ldots, \mathbf{s}_T)$. Simulation is closed-loop: each realized scene becomes the context for the next prediction.

B. Holonomic and Bicycle-Style Motion

Traffic-agent motion models commonly distinguish holonomic planar motion for freely moving participants, such as pedestrians, from bicycle-style non-holonomic motion for wheeled road users [32], [33]. The former permits both longitudinal and lateral displacement in the agent frame, whereas the latter does not treat lateral motion as an independent executed input.

For heading ψ, let $R(\psi)$ be the planar rotation matrix, $\mathbf{u}_\parallel(\psi) = (\cos\psi, \sin\psi)^\top$, $\mathbf{u}_\perp(\psi) = (-\sin\psi, \cos\psi)^\top$, and $\mathrm{sinc}(z) = \sin(z)/z$ with $\mathrm{sinc}(0) = 1$. We denote $a_\parallel$, $a_\perp$, and a_ψ for one-step longitudinal displacement, lateral displacement, and heading change in the agent frame.

A holonomic displacement executes both displacement channels:

$$\Delta\mathbf{p}_{\mathrm{hol}} = R(\psi) \begin{bmatrix} a_\parallel \\ a_\perp \end{bmatrix}. \tag{1}$$

For bicycle-style motion, let r be the no-slip offset, the scalar distance from the box center to the no-slip reference point where lateral motion is suppressed. With midpoint heading $\bar{\psi} = \psi + a_\psi/2$, the box-center displacement is

$$\begin{aligned} \Delta\mathbf{p}_{\mathrm{nh}} &= \Delta\mathbf{p}_{\mathrm{prog}} + \Delta\mathbf{p}_{\mathrm{swing}}, \\ \Delta\mathbf{p}_{\mathrm{prog}} &:= a_\parallel \,\mathrm{sinc}\Big(\frac{a_\psi}{2}\Big)\, \mathbf{u}_\parallel(\bar{\psi}), \\ \Delta\mathbf{p}_{\mathrm{swing}} &:= 2r \sin\Big(\frac{a_\psi}{2}\Big)\, \mathbf{u}_\perp(\bar{\psi}). \end{aligned} \tag{2}$$

Here, $\Delta\mathbf{p}_{\mathrm{prog}}$ is the forward progress of the no-slip reference point, and $\Delta\mathbf{p}_{\mathrm{swing}}$ is the signed box-center swing caused by rotating the offset r. Since $\Delta\mathbf{p}_{\mathrm{prog}}$ is parallel to $\mathbf{u}_\parallel(\bar{\psi})$, projecting Eq. (2) onto $\mathbf{u}_\perp(\bar{\psi})$ leaves only the swing term:

$$\Delta\mathbf{p}_{\mathrm{nh}}^\top \mathbf{u}_\perp(\bar{\psi}) = 2r \sin\Big(\frac{a_\psi}{2}\Big). \tag{3}$$

C. Closed-Loop Covariate Shift

Behavior cloning (BC) on a logged dataset $\mathcal{D} = \{(\mathcal{H}_0, \tau)\}$ minimizes the negative log-likelihood for model parameters θ:

$$\mathcal{L}_{\mathrm{BC}}(\theta) = -\,\mathbb{E}_{\tau\sim\mathcal{D}} \sum_{t=0}^{T-1} \log p_\theta(\mathbf{s}_{t+1} \mid \mathcal{H}_t). \tag{4}$$

This is teacher forcing [34]: the conditioning context $\mathcal{H}_t$ is drawn from logged data. At deployment, however, the simulator conditions on its own previous outputs,

$$\hat{\mathbf{s}}_{t+1} \sim p_\theta\Big(\cdot \,\Big|\, \hat{\mathcal{H}}_t\Big), \qquad t = 0, \ldots, T-1, \tag{5}$$

where $\hat{\mathcal{H}}_t$ is built from the initial history and generated states. The induced closed-loop rollout distribution $p_\theta^{\mathrm{CL}}(\tau \mid \mathcal{H}_0)$ can differ from the data distribution $p_{\mathrm{data}}(\tau \mid \mathcal{H}_0)$ because the model sees off-data contexts created by its own predictions, causing compounding error and closed-loop covariate shift [20]. One remedy is to align closed-loop rollouts with the data distribution, $p_\theta^{\mathrm{CL}}(\tau \mid \mathcal{H}_0) \approx p_{\mathrm{data}}(\tau \mid \mathcal{H}_0)$, for example by minimizing a reverse-KL objective, $D_{\mathrm{KL}}(p_\theta^{\mathrm{CL}} \| p_{\mathrm{data}})$ [35] that penalizes generated rollouts unlikely under the data distribution. However, reverse-KL is mode-seeking [35], [36] and can concentrate mass on a few high-density modes.

D. Training Flow-based Model

Flow Matching [25] defines a probability path $(p_\lambda)_{0\le\lambda\le1}$ from a Gaussian source $p_0 = \mathcal{N}(\mathbf{0}, \mathbf{I})$ to a data target $p_1 = p_{\mathrm{data}}$ over $\mathbf{x} \in \mathbb{R}^d$. It learns a velocity field $\mathbf{v} : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ whose flow map $\varphi : [0,1] \times \mathbb{R}^d \to \mathbb{R}^d$ satisfies

$$\frac{d}{d\lambda}\varphi_\lambda(\mathbf{x}) = \mathbf{v}(\lambda, \varphi_\lambda(\mathbf{x})), \qquad \varphi_0(\mathbf{x}) = \mathbf{x}. \tag{6}$$

Given a clean data sample $\mathbf{x}_1$ and noise $\mathbf{x}_0 \sim p_0$, we use the affine optimal-transport path $\mathbf{x}_\lambda = (1-\lambda)\mathbf{x}_0 + \lambda\mathbf{x}_1$.

Corresponding velocity target $\mathbf{v}^\star = \mathbf{x}_1 - \mathbf{x}_0$, and the encoded context $\mathbf{e}$, derives our flow-matching loss:

$$\mathcal{L}_{\mathrm{FM}}(\theta) = \mathbb{E}_{(\mathbf{x}_1,\mathbf{e})\sim\mathcal{D},\, \mathbf{x}_0\sim p_0,\, \lambda\sim\mathcal{U}[0,1]} \Big[\|\mathbf{v}_\theta(\mathbf{x}_\lambda, \lambda, \mathbf{e}) - \mathbf{v}^\star\|_2^2 \Big], \tag{7}$$

where $\mathbf{v}_\theta$ is a parameterized neural velocity field. After training, samples are generated by solving the learned ODE:

$$\frac{d\mathbf{x}_\lambda}{d\lambda} = \mathbf{v}_\theta(\mathbf{x}_\lambda, \lambda, \mathbf{e}), \qquad \mathbf{x}_0 \sim p_0. \tag{8}$$

IV. Method

In this section, we introduce our two components: the proposed backbone, Agent-Type Aware Flow Matching (AFM), and its diversity-preserving fine-tuning method, Entropy-Regularized Distillation (ERD), illustrated in Fig. 2. AFM captures multimodal diversity while keeping it realistic through agent-type aware modeling in a continuous action space. ERD then fine-tunes this with an entropy-regularized objective that retains minority modes, avoiding the mode collapse of standard reverse KL objectives.

A. Flow-Matching Backbone

1) Motivation and Design Rationale: Token-based simulators execute discrete motion primitives, but finite vocabularies limit continuous variation. Continuous generators remove this codebook bottleneck, but without type-aware execution, they may assign probability to motions invalid for a given agent type; in closed loop, such invalid states become future context. AFM separates generation from execution: the flow model samples multimodal continuous kinematic actions, and the simulator feeds back only poses obtained by executing those actions through type-specific transitions.

2) Kinematic Action-Space Flow Modeling: For agent i at time t, we use the metric kinematic action

$$\mathbf{a}_t^i = (a_{\parallel,t}^i, a_{\perp,t}^i, a_{\psi,t}^i) \in \mathcal{A}_{c_i}, \tag{9}$$

where the entries are local longitudinal, lateral, and heading increments. Given the history context $\mathcal{H}_t$, AFM models the full H-step action sequence $\mathbf{a}_{t:t+H-1}$. In flow-matching pretraining, the clean endpoint $\mathbf{x}_1$ becomes the target action sequence $\mathbf{a}_{t:t+H-1}^\star$; we define $\mathbf{a}_{t:t+H-1}^\star$ below after specifying the transition that executes actions into poses.

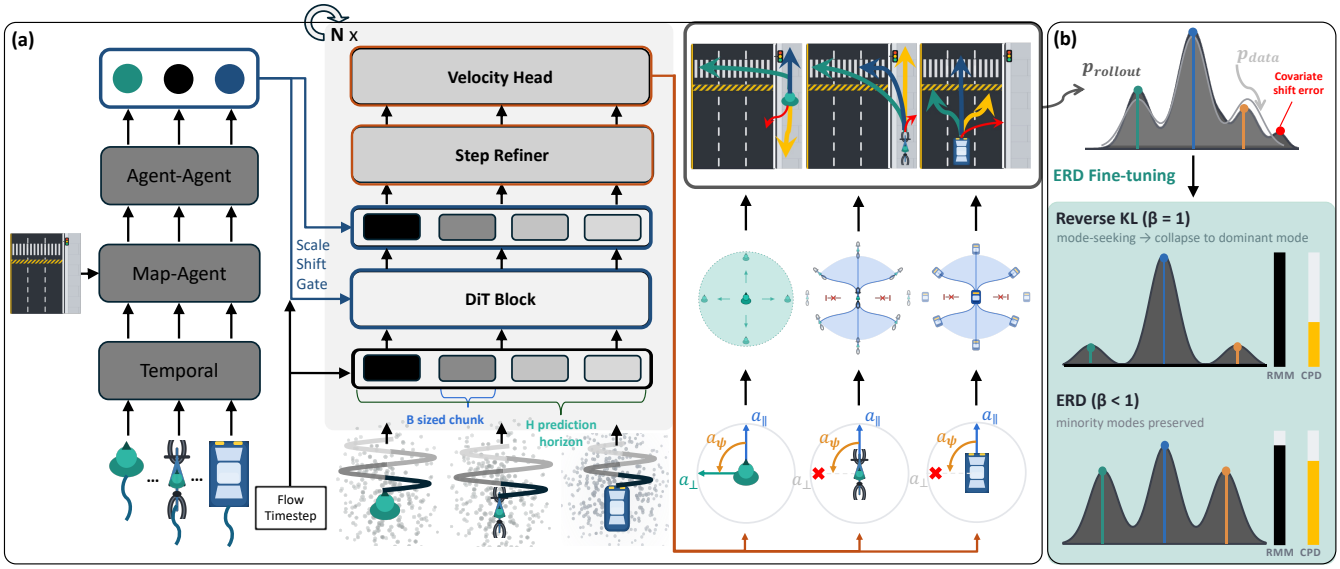

Fig. 2: Overview of Flow-ERD. (a) The Agent-Type Aware Flow-Matching (AFM) backbone generates a shared continuous action representation, executed through type-specific kinematics (non-holonomic for vehicles/cyclists, holonomic for pedestrians). (b) Entropy-Regularized Distillation (ERD) then fine-tunes the closed-loop distribution: the vanilla reverse-KL objective ($\beta = 1$) is mode-seeking, easier to collapse onto the dominant (straight) mode, whereas ERD ($\beta < 1$) targets an entropy-tempered distribution that preserves minority modes while still mitigating closed-loop covariate shift.

3) Type-Specific State Transition: To map an action to the next pose, AFM applies the agent-type-specific transition $\mathcal{F}_{c_i}$ to the planar state $\mathbf{s}_t^i = (\mathbf{p}_t^i, \psi_t^i)$:

$$\begin{aligned} \mathbf{s}_{t+1}^i &= \mathcal{F}_{c_i}(\mathbf{s}_t^i, \mathbf{a}_t^i), \\ \text{with} \quad \psi_{t+1}^i &= \mathrm{wrap}(\psi_t^i + a_{\psi,t}^i), \\ \mathbf{p}_{t+1}^i &= \mathbf{p}_t^i + \Delta\mathbf{p}_t^i, \end{aligned} \tag{10}$$

where wrap maps angles to $[-\pi, \pi]$. Thus, $\mathcal{F}_{c_i}$ is fully specified by the heading update and the type-dependent displacement below. The map-frame displacement uses the motion models in Section III-B:

$$\Delta\mathbf{p}_t^i = \begin{cases} \Delta\mathbf{p}_{\mathrm{hol}}^i, & c_i = \mathrm{PED}, \\ \Delta\mathbf{p}_{\mathrm{nh}}^i, & c_i \in \{\mathrm{VEH}, \mathrm{CYC}\}. \end{cases} \tag{11}$$

For vehicles and cyclists, the non-holonomic branch uses forward displacement, heading change, and the no-slip offset r_i; the lateral channel $a_{\perp,t}^i$ remains in the shared action vector but is not used in their pose update. This preserves a shared action space while enforcing type-compatible execution.

4) Training: Transition-Consistent Action Targets: To supervise AFM in action space, logged pose pairs must be converted into actions. For vehicles and cyclists, this requires inverting Eq. (2), which depends on the no-slip offset r. The dataset logs label only box-center poses, not the no-slip point, so this offset must be estimated from the logged trajectories. We set $r_i = \rho_{c_i}\ell_i$, where c_i is agent i's type and ℓ_i its box length, and estimate one ρ_c per non-holonomic type.

The ratio must be estimated from logged turning intervals. In straight intervals, $a_\psi = 0$, so Eq. (3) gives zero lateral swing for any r; such intervals contain no information about ρ_c. For each logged turning interval, substituting $r_i = \rho_{c_i}\ell_i$ into Eq. (3) gives

$$\hat{\rho}_i = \frac{\Delta\mathbf{p}_i^\top \mathbf{u}_\perp(\bar{\psi}_i)}{2\ell_i \sin(a_{\psi,i}/2)}. \tag{12}$$

We aggregate these interval-level estimates by type with a robust statistic, yielding one ρ_c for each non-holonomic type.

With the no-slip ratio fixed, we convert logged poses into an executable action sequence. During inference, actions are executed sequentially, and each executed pose becomes the state for the next action. To reduce the train-inference gap, we construct the targets in the same way: rather than starting from ground truth position $\mathbf{s}_t^{i,\star}$, we initialize previously executed state $\tilde{\mathbf{s}}_t^i$ for $k = 0, \ldots, H-1$, then recover the action toward the next logged pose:

$$\mathbf{a}_{t+k}^{i,\star} = \mathcal{F}_{c_i}^{-1}(\tilde{\mathbf{s}}_{t+k}^i, \mathbf{s}_{t+k+1}^{i,\star}), \tag{13}$$

where $\mathcal{F}_{c_i}^{-1}$ inverts $\mathcal{F}_{c_i}$. We then execute this action and use the resulting pose as the state for the next step sequentially:

$$\tilde{\mathbf{s}}_{t+k+1}^i = \mathcal{F}_{c_i}(\tilde{\mathbf{s}}_{t+k}^i, \mathbf{a}_{t+k}^{i,\star}). \tag{14}$$

This makes $\mathbf{a}_{t,0:H-1}^\star$ an executable flow-matching target whose state sequence follows the same transition used at inference. We filter out the targets with large re-execution error $\left|\tilde{\mathbf{s}}_{t+k+1}^i - \mathbf{s}_{t+k+1}^{i,\star}\right|$ to improve training stability.

5) Closed-Loop Inference: At inference, AFM generates a full H-step action sequence. The simulator executes it with $\mathcal{F}_{c_i}$, commits only the first $B < H$ steps in a receding horizon manner, and appends their poses to the next context as in Eq. (5). We denote this receding-horizon closed-loop rollout distribution as $p_\theta^{\mathrm{CL}}(\tau|\mathcal{H}_0)$ and the H-step direct prediction distribution as $p_\theta^{\mathrm{OL}}(\tau|\mathcal{H}_0)$. Under $p_\theta^{\mathrm{CL}}(\tau|\mathcal{H}_0)$, since feedback uses type-specific executed poses rather than unconstrained pose predictions, AFM preserves continuous multimodality while reducing type-incompatible closed-loop states.

Algorithm 1 Entropy-Regularized Distillation (ERD)

Require: pretrained θ_0; dataset $\mathcal{D}$; temperature β; critic steps n_{critic}; phase length L_{phase}; fake-score step size η; stop-gradient operator sg.

initialize generator θ and fake-score parameters ϕ from θ_0; freeze real score $\mathbf{g}^{\text{OL}}_{\theta_0}$
repeat
  for $q = 1$ **to** L_{phase} ▷ FAKE ONLY ϕ **do**
    $(\hat{\mathbf{a}}_{t:t+H-1}, \hat{\tau}) \sim p^{\text{CL}}_\theta(\tau|\mathcal{H}_0)$
    $\phi \leftarrow \phi - \eta \nabla_\phi \mathcal{L}_{\text{FM}}(\phi; \mathbf{x}_1 = \text{sg}[\hat{\mathbf{a}}_{t:t+H-1}])$
  end for
  for $q = 1$ **to** L_{phase} ▷ BOTH θ, ϕ **do**
    $(\hat{\mathbf{a}}_{t:t+H-1}, \hat{\tau}) \sim p^{\text{CL}}_\theta(\tau|\mathcal{H}_0)$
    $\phi \leftarrow \phi - \eta \nabla_\phi \mathcal{L}_{\text{FM}}(\phi; \mathbf{x}_1 = \text{sg}[\hat{\mathbf{a}}_{t:t+H-1}])$
    if $q \bmod n_{\text{critic}} = 0$**:** update generator θ via Eq. (19), backpropagation through t, λ.
  end for
until converged
return θ

6) Architecture: Fig. 2 summarizes the AFM architecture. Given $\mathcal{H}_t$, a SMART-style scene encoder builds temporal, map–agent, and agent–agent context features, summarized as $\mathbf{e}_t = \text{Enc}(\mathcal{H}_t)$. The temporal history is encoded with continuously executed motion features rather than discrete motion tokens.

The flow decoder takes $\mathbf{e_t}$, a noisy H-step action sequence $\mathbf{x}_\lambda$, with flow noise timestep λ. It embeds the sequence into B-step chunks and applies chunk-level self-attention with DiT-style scale–shift–gate conditioning [37]. A step refiner applies within-chunk self-attention, and an MLP velocity head outputs the action-space velocity field $\mathbf{v}_\theta(\mathbf{x}_\lambda, \lambda, \mathbf{e}_t)$ used in Eq. (8). The resulting action sequence is executed by the type-specific transition in Eq. (11).

B. Entropy-Regularized Distillation

We fine-tune the AFM backbone under closed-loop rollouts to mitigate covariate shift, while explicitly preserving the diversity it already represents. Following the matching goal of Section III-C, $p^{\text{CL}}_\theta(\tau \mid \mathcal{H}_0) \approx p_{\text{data}}(\tau \mid \mathcal{H}_0)$, we instantiate it with a reverse-KL divergence,

$$J(\theta) \;=\; \mathbb{E}_{\mathcal{H}_0 \sim \mathcal{D}}\big[\, D_{\text{KL}}\big(p^{\text{CL}}_\theta(\tau \mid \mathcal{H}_0) \,\big\|\, p_{\text{data}}(\tau \mid \mathcal{H}_0)\big)\big]\,. \tag{15}$$

Reverse-KL is mode-seeking in practice: it penalizes model rollouts in low-density regions of p_{data}, but because the expectation is over p^{CL}_θ, valid data modes rarely sampled by the model contribute little to the loss. Under imperfect capacity or optimization, probability can therefore concentrate on dominant modes and underrepresent minority ones. To counteract this, we add an entropy regularizer on the closed-loop distribution,

$$J_{\text{ERD}}(\theta) = \mathbb{E}_{\mathcal{H}_0}\Big[D_{\text{KL}}\big(p^{\text{CL}}_\theta \,\big\|\, p_{\text{data}}\big) - \gamma\, \text{Ent}\big(p^{\text{CL}}_\theta(\tau \mid \mathcal{H}_0)\big)\Big], \tag{16}$$

where $\text{Ent}(\cdot)$ denotes Shannon entropy and $\gamma \geq 0$ is its regularization weight. For brevity, we hereafter use the temperature $\beta := 1/(1+\gamma) \in (0, 1]$.

Expanding the entropy and dividing by $1+\gamma$ shows that, up to a positive scale and an additive constant, Eq. (16) is an ordinary reverse-KL divergence against a tempered target,

$$J_{\text{ERD}}(\theta) \;=\; (1{+}\gamma)\, \mathbb{E}_{\mathcal{H}_0}\Big[\, D_{\text{KL}}\big(p^{\text{CL}}_\theta \,\big\|\, p^{\beta}_{\text{data}}\big)\Big] {+}\text{const}, \tag{17}$$

$$p^{\beta}_{\text{data}} \propto p_{\text{data}}(\tau \mid \mathcal{H}_0)^\beta. \tag{18}$$

ERD is therefore plain distribution matching toward p^{β}_{data}, whose minimizer is the tempered fixed point $p^{\text{CL}\star}_\theta \propto p^{\beta}_{\text{data}}$. Tempering reduces the density contrast between modes and relaxes the mode-dropping bias of reverse-KL: smaller β (larger γ) flattens p^{β}_{data} toward a uniform distribution over the data support and up-weights minority modes, while $\beta{=}1$ ($\gamma{=}0$) recovers plain distribution matching toward p_{data}.

The likelihood $\log p^{\text{CL}}_\theta$ is unavailable in closed form for autoregressive flow rollouts, so we cannot optimize Eq. (17) directly. Thus, we realize it with Distribution-Matching Distillation (DMD) [38] on the model's generated action sequences, following Self-Forcing [39].

Let $\hat{\mathbf{x}}_1 \equiv \hat{\mathbf{a}}_{t:t+H-1}$ denote the generated clean H-step kinematic-action sequence, and $\hat{\tau}$ be the state rollout induced by executing $\hat{\mathbf{a}}_{t:t+H-1}$ through the type-specific transitions $\mathcal{F} = \{\mathcal{F}_c\}_{c\in\mathcal{C}}$. DMD sidesteps the intractable likelihood by expressing the KL gradient through scores $\mathbf{g}_p(\mathbf{x}) = \nabla_{\mathbf{x}} \log p(\mathbf{x})$ of the two action-sequence distributions, evaluated at each flow noise timestep λ; Eq. (17) is then the standard DMD gradient with the real score replaced by that of the tempered target, $\mathbf{g}_{p^{\beta}_{\text{data}}} = \beta \mathbf{g}_{\text{p}_{\text{data}}}$, where $\mathbf{g}_{p_{\text{data}}}$ is the data score. We make this practical with two substitutions: we use the frozen pretrained backbone $\mathbf{g}^{\text{OL}}_{\theta_0}$ as the data score, where OL denotes the H-step action distribution learned under logged-state conditioning during open-loop pretraining. Since $p^{\text{OL}}_{\theta_0} \approx p_{\text{data}}$ on the data support [39], the real score is evaluated from this frozen OL model, while the CL fake score is trained on action sequences generated by rolling the current policy forward with B-step commitment. We realize the tempered score by scaling it directly, $\mathbf{g}_{p^{\beta}_{\text{data}}} \approx \beta\, \mathbf{g}^{\text{OL}}_{\theta_0}$, which is exact at the clean-data end $\lambda \to 1$, as in classifier-free guidance [40]. This yields the estimator we optimize,

$$\nabla_\theta J_{\text{ERD}}(\theta) \propto \mathbb{E}_{\hat{\mathbf{x}}_\lambda, \lambda}\big[\big(\mathbf{g}^{\text{CL}}_\phi(\hat{\mathbf{x}}_\lambda, \lambda) - \beta\, \mathbf{g}^{\text{OL}}_{\theta_0}(\hat{\mathbf{x}}_\lambda, \lambda)\big)\, \partial_\theta \hat{\mathbf{x}}_1\big]\,, \tag{19}$$

where $\hat{\mathbf{x}}_\lambda$ is $\hat{\mathbf{x}}_1$ noised to flow time λ as in the affine path above and the fake score $\mathbf{g}^{\text{CL}}_\phi$ is kept on-policy. Executing $\hat{\mathbf{a}}_{t:t+H-1}$ with the deterministic transition produces $\hat{\tau}$, so this action-space update changes the induced closed-loop state rollout. Detailed algorithm can be found in Algorithm 1.

V. EXPERIMENTS

In this section, we evaluate our framework, Flow-ERD, around three questions. (1) Does Agent-Type Aware Flow Matching (AFM) achieve higher realism while maintaining high diversity? (2) Does Entropy-Regularized Distillation (ERD) preserve this diversity while improving realism? (3)

TABLE I: Results on the WOSAC 2025 test split. Best in **bold**, second best underlined, †: fine-tuned from SMART.

Type	Method	RMM ↑	Kin. ↑	Int. ↑	Map ↑	minADE ↓
Backbone	SMART [9]	0.7814	0.4854	0.8089	0.9153	1.3931
	TrajTok [10]	**0.7852**	0.4887	**0.8116**	**0.9207**	1.3179
	UniMM [14]	0.7829	0.4914	0.8089	0.9161	**1.2949**
	MDG [13]	0.7844	0.4928	0.8099	0.9183	1.3123
	AFM backbone (ours)	0.7845	**0.4955**	0.8094	0.9178	1.3358
Fine-tuned	CAT-K† [15]	0.7846	0.4931	0.8106	0.9177	1.3065
	RoaD† [21]	0.7847	0.4932	0.8106	0.9178	1.3042
	RLFTSim† [23]	0.7857	0.4927	0.8129	0.9183	1.3252
	SMART-R1† [16]	0.7858	0.4944	0.8110	**0.9201**	1.2885
	DecompGAIL† [24]	0.7864	0.4919	**0.8152**	0.9176	1.4209
	Flow-ERD (AFM + ERD)	**0.7878**	**0.5062**	0.8110	0.9190	**1.2721**

TABLE II: Results on the WOSAC 2025 4% validation split. Best in **bold**, second best underlined, †: fine-tuned from SMART.

Type	Method	RMM ↑	Kin. ↑	Int. ↑	Map ↑	minADE ↓	CPD ↑
Backbone	SMART [9]	0.7807	0.4873	0.8071	0.9144	1.323	0.1655
	TrajTok [10]	**0.7837**	0.4851	**0.8110**	**0.9193**	1.3258	0.1587
	UniMM [14]	0.7812	0.4863	0.8070	0.9165	**1.2725**	0.1514
	AFM backbone (ours)	0.7836	**0.4914**	0.8096	0.9172	1.3024	**0.1858**
	All holonomic FM ablation	0.7823	0.4905	0.8063	0.9182	1.2993	0.2046
	All non-holonomic FM ablation	0.7824	0.4910	0.8091	0.9144	1.2960	0.1843
Fine-tuned	CAT-K [15]†	0.7842	0.4904	0.8110	0.9175	1.3001	0.1491
	RoaD [21]†	0.7847	0.4904	0.8116	0.9182	**1.2603**	0.1585
	RLFTSim [23]†	0.7848	0.4895	0.8116	**0.9190**	1.2988	0.1510
	DecompGAIL [24]†	0.7854	0.4912	**0.8123**	**0.9190**	1.2868	0.1420
	Flow-ERD ($\beta = 1.0$)	**0.7876**	**0.5063**	0.8108	0.9184	1.2620	0.1684
	Flow-ERD ($\beta = 0.99$)	0.7869	0.5053	0.8099	0.9182	1.2723	**0.1828**

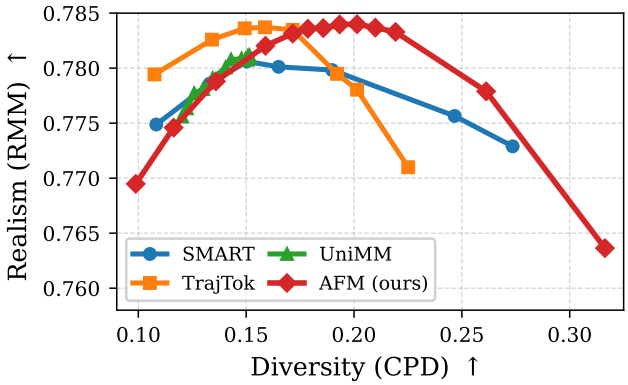

Fig. 3: Realism–diversity trade-off on the validation split. UniMM, SMART, and TrajTok sweep k of top-k decoding during validation rollouts, whereas AFM (ours) sweeps the Gaussian noise scale. AFM traces the upper-right Pareto frontier, reaching an RMM of 0.7840 at noise scale 1.05.

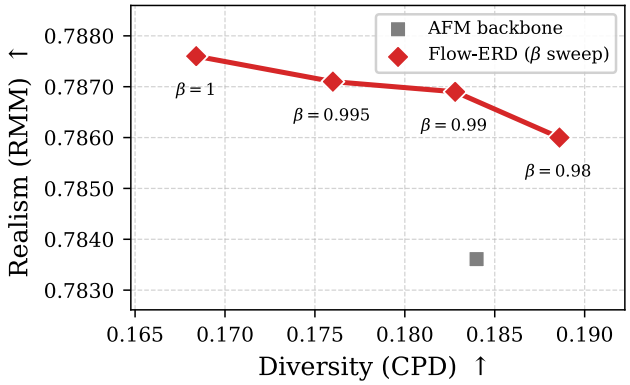

Fig. 4: ERD entropy temperature β sweep on the WOSAC 2025 validation split. Sweeping $\beta \in (0, 1]$ traces Flow-ERD's realism (RMM) versus diversity (CPD, Eq. (21)) trade-off: lowering β flattens the target distribution and raises diversity at a small cost in realism.

Does the preserved diversity translate into semantic multi-modality with minority modes retained? To address these questions, we first introduce our evaluation metrics and then present our analysis.

A. Evaluation Metrics

a) Realism: We follow the WOSAC 2025 evaluation [3], whose realism meta-metric (RMM) aggregates distributional likelihoods over kinematic, interactive, and map-based statistics against the single logged future. We report RMM and its three components, and refer to [3] for exact definitions. We additionally report minADE (per-object minimum ADE over rollouts), which does not enter RMM but is standard for behavior-prediction comparison.

b) Diversity: RMM is a likelihood-based score against a single logged future, so it credits spread around the log but misses plausible modes the log omits. This cannot distinguish dominant-mode tracking from realistic yet diverse rollouts. Therefore, we define the log-independent diversity metric.

Concretely, given K closed-loop rollouts of the same

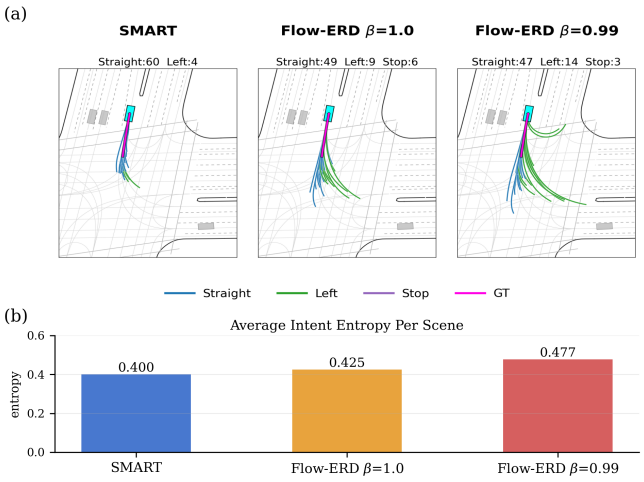

Fig. 5: We run multi-agent closed-loop rollouts over 1,048 validation scenes and label ego-maneuver intents following WOMD [5]. **(a)** Ego-trajectory diversity on a WOSAC scene, shown by overlaying closed-loop rollouts. **(b)** Average per-scene intent entropy of the ego rollouts.

scenario, we define the type-normalized pairwise distance:

$$d(\tau,\tau') \;=\; \sqrt{\sum_{c\in\mathcal{C}} \frac{1}{\sigma_c^2}\cdot\frac{1}{N_c T}\sum_{\substack{i:\,c_i=c\\ 1\le t\le T}} \left\|\mathbf{p}_t^{\,i}-\mathbf{p}_t'^{\,i}\right\|^2}, \qquad (20)$$

where $\mathbf{p}_t^i, \mathbf{p}_t'^i \in \mathbb{R}^2$ are agent i's positions at step t in the two rollouts, σ_c is a per-type scale fixed on the training set, and N_c is the number of agents of type c; types with $N_c = 0$ are omitted. The Cross-Pair Diversity (CPD) averages Eq. (20) over all unordered rollout pairs and scenarios $\omega \in W$:

$$\mathrm{CPD} = \frac{1}{|\mathcal{W}|}\sum_{\omega\in\mathcal{W}} \frac{2}{K(K-1)} \sum_{1\le k<k'\le K} d\big(\tau_\omega^{(k)},\tau_\omega^{(k')}\big). \qquad (21)$$

Higher CPD means more distinct generated futures for the same initial state. Being a pure spread measure, however, it cannot alone separate genuine multimodality from variance due to prediction error or closed-loop drift [4]. As this spurious variance grows with lower realism, we compare CPD only at matched RMM and treat gaps across markedly different realism with caution.

B. Experimental Results

1) AFM breaks the backbone realism–diversity trade-off: On the WOSAC test split (Table I), the AFM backbone attains the highest overall RMM among continuous backbones and is competitive with the strongest tokenized backbones such as TrajTok. Without any fine-tuning, it achieves the best kinematic component among all baselines, including fine-tuned, confirming that agent-type aware kinematic modeling most faithfully reproduces per-type kinematics.

Its advantage is clearest on diversity (Table II): across all other pretrained baselines, AFM attains the highest CPD. To rule out that this merely reflects suboptimal sampling settings for the baselines, we sweep each model's controllable knob in Fig. 3: top-k token/anchor selection for token-based models and UniMM, and the Gaussian noise scale applied to $x_0 \sim p_0$ in Eq. (7) for ours; the baselines cannot reach our frontier under any setting. In most cases, AFM is higher in both RMM and CPD, and against TrajTok, whose RMM matches ours, AFM still attains substantially higher CPD. At matched realism, this gap cannot stem from drift-induced variance, and instead shows AFM placing mass on genuinely more diverse futures, RMM alone would not reveal. The gap to UniMM indicates this diversity stems not only from a continuous representation or higher resolution, but also from the expressiveness of our flow-based modeling.

Table II ablates whether continuous flow matching alone suffices, or whether the transition must be agent-type aware. The all-holonomic backbone yields the largest raw CPD but the lowest kinematic score, indicating that much of its spread comes from type-incompatible motion freedom, lateral slip for vehicles and cyclists. Enforcing a non-holonomic transition for all agents recovers kinematic realism and trims this unrealistic spread, but it over-constrains pedestrians, yielding lower RMM and CPD than AFM. AFM resolves both failure modes by assigning holonomic transitions to pedestrians and non-holonomic ones to vehicles and cyclists, achieving the best RMM and kinematic score among the other variants.

2) ERD improves closed-loop realism without collapsing diversity: Table I shows that Flow-ERD achieves the best RMM among all baselines. We report the official submission result with the vanilla reverse-KL objective ($\beta = 1.0$), since the leaderboard evaluates realism but not diversity. Table II gives the diversity view: even without an entropy term, Flow-ERD preserves higher CPD than all fine-tuned baselines. This indicates that the gain is not merely a generic fine-tuning effect; ERD reshapes probability mass within the broad support learned by the flow-based AFM backbone. Still, $\beta = 1.0$ reduces CPD relative to the pretrained AFM backbone with $\Delta\mathrm{CPD} = 0.0174$, consistent with the mode-seeking behavior described in Section III. Lowering the entropy temperature to $\beta = 0.99$ recovers most of the backbone diversity with $\Delta\mathrm{CPD} = 0.003$ while retaining a large realism gain. While other fine-tuned baselines reduce CPD at least $\Delta\mathrm{CPD} = 0.007$, Flow-ERD remains above the fine-tuned baselines in RMM while preserving higher CPD.

Fig. 4 shows that the entropy temperature β provides a direct realism–diversity control. At $\beta = 1$, ERD reduces to vanilla distribution matching and gives the highest RMM. Lowering β flattens the tempered target $p_{\mathrm{data}}^{\beta}$, trading a small amount of realism for higher CPD. At $\beta = 0.99$, the model recovers nearly all of the backbone's diversity while staying above every fine-tuning baseline in realism, so we report both $\beta = 1.0$ and $\beta = 0.99$ in Table II. Too low β increases diversity at a much steeper realism cost: at $\beta = 0.95$, RMM keeps falling below the backbone, defeating the purpose of fine-tuning; we therefore omit it in Fig. 4.

3) The preserved diversity reflects intent-level multimodality: To verify that the retained CPD corresponds to meaningful behaviors rather than noisy dispersion, we examine diversity at the level of maneuver intents. We sample 64 rollouts across 1,048 validation scenarios and classify the intent

of each ego-trajectory following the WOMD trajectory-type rule [5]. In qualitative Fig. 5(a), SMART assigns almost all of its probability mass to the dominant straight mode. Our $\beta = 1.0$ increases diversity among the same intents, and lowering the temperature to $\beta = 0.99$ further recovers the rare U-turns. These samples do not simply spread away from the map or drift under closed-loop error; they form semantically distinct, physically plausible maneuvers available in the same scene. Fig. 5(b) we further measure the Shannon entropy of intents, which increases monotonically with lower β. The slight RMM drop at lower β is therefore expected, since RMM is tied to benchmark statistics measured against a single logged ground truth and may under-reward rare but plausible alternatives to the logged or dominant mode. This explains why RMM alone is insufficient: a simulator can look realistic while still collapsing to the dominant mode.

VI. Conclusion

We presented Flow-ERD, a multi-agent traffic simulator that jointly improves closed-loop realism and rollout diversity. Its AFM backbone generates continuous action sequences while enforcing type-specific kinematic transitions, and ERD fine-tunes the closed-loop rollout distribution with an entropy-regularized distribution-matching objective. Experiments on WOSAC show that Flow-ERD achieves state-of-the-art realism on the test split and dominates the validation realism–diversity Pareto frontier under CPD. These results suggest that diversity should be evaluated explicitly rather than inferred from realism alone.

References

[1] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conf. Robot Learn. (CoRL)*, 2017.

[2] C. Gulino *et al.*, "Waymax: An accelerated, data-driven simulator for large-scale autonomous driving research," *Adv. Neural Inf. Process. Syst.*, 2023.

[3] N. Montali *et al.*, "The waymo open sim agents challenge," *Adv. Neural Inf. Process. Syst.*, 2023.

[4] S. Suo, S. Regalado, S. Casas, and R. Urtasun, "Trafficsim: Learning to simulate realistic multi-agent behaviors," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021.

[5] S. Ettinger *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.

[6] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, *et al.*, "Improving language understanding by generative pre-training," 2018.

[7] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," *Adv. Neural Inf. Process. Syst.*, 2020.

[8] J. Philion, X. B. Peng, and S. Fidler, "Trajeglish: Traffic modeling as next-token prediction," in *12th Int. Conf. Learn. Represent. (ICLR)*, 2024.

[9] 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.*, 2024.

[10] Z. Zhang *et al.*, "Trajtok: What makes for a good trajectory tokenizer in behavior generation?" in *14th Int. Conf. Learn. Represent. (ICLR)*, 2026.

[11] Z. Huang, Z. Zhang, A. Vaidya, Y. Chen, J. F. Fisac, and C. Lv, "Versatile behavior diffusion for generalized traffic agent simulation," *IEEE Trans. Intell. Transp. Syst.*, 2026.

[12] C. M. Jiang *et al.*, "Scenediffuser: Efficient and controllable driving simulation initialization and rollout," *Adv. Neural Inf. Process. Syst.*, 2024.

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

[14] L. Lin *et al.*, "Revisit mixture models for multi-agent simulation: Experimental study within a unified framework," 2025.

[15] Z. Zhang *et al.*, "Closed-loop supervised fine-tuning of tokenized traffic models," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2025.

[16] M. Pei, S. Shi, and S. Shen, "Advancing multi-agent traffic simulation via r1-style reinforcement fine-tuning," in *14th Int. Conf. Learn. Represent. (ICLR)*, 2026.

[17] J. Zhao *et al.*, "Kigras: Kinematic-driven generative model for realistic agent simulation," *IEEE Robot. Autom. Lett.*, 2024.

[18] Z. Zhong *et al.*, "Guided conditional diffusion for controllable traffic simulation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2023.

[19] B. Liao *et al.*, "Diffusiondrive: Truncated diffusion model for end-to-end autonomous driving," 2025.

[20] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proc. 14th Int. Conf. Artif. Intell. Stat. (AISTATS)*, 2011.

[21] G. Garcia-Cobo *et al.*, "Road: Rollouts as demonstrations for closed-loop supervised fine-tuning of autonomous driving policies," *arXiv:2512.01993*, 2025.

[22] W.-J. Chang, W. Zhan, M. Tomizuka, M. Chandraker, and F. Pittaluga, "Langtraj: Diffusion model and dataset for language-conditioned trajectory simulation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, 2025.

[23] E. Ahmadi *et al.*, "Rlftsim: Realistic and controllable multi-agent traffic simulation via reinforcement learning fine-tuning," *arXiv:2605.19033*, 2026.

[24] K. Guo, H. Liu, X. WU, and C. Lv, "DecompGAIL: Learning realistic traffic behaviors with decomposed multi-agent generative adversarial imitation learning," in *14th Int. Conf. Learn. Represent. (ICLR)*, 2026.

[25] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," in *11th Int. Conf. Learn. Represent. (ICLR)*, 2023.

[26] Z. Zhou *et al.*, "BehaviorGPT: Smart agent simulation for autonomous driving with next-patch prediction," in *Adv. Neural Inf. Process. Syst.*, 2024.

[27] W.-J. Chang, F. Pittaluga, M. Tomizuka, W. Zhan, and M. Chandraker, "SAFE-SIM: Safety-critical closed-loop traffic simulation with diffusion-controllable adversaries," in *Eur. Conf. Comput. Vis. (ECCV)*, 2024.

[28] S. Tan *et al.*, "ProSim: Promptable closed-loop traffic simulation," in *Proc. 8th Conf. Robot Learn. (CoRL)*, 2024.

[29] Y. Zhou *et al.*, "Decoupled diffusion sparks adaptive scene generation," in *ICCV*, 2025.

[30] J. Ho and S. Ermon, "Generative adversarial imitation learning," *Adv. Neural Inf. Process. Syst.*, 2016.

[31] A. Venkatraman, M. Hebert, and J. Bagnell, "Improving multi-step prediction of learned time series models," in *Proc. AAAI Conf. Artif. Intell.*, 2015.

[32] B. Paden, M. Cap, S. Z. Yong, D. Yershov, and E. Frazzoli, "A survey of motion planning and control techniques for self-driving urban vehicles," *IEEE Trans. Intell. Veh.*, 2016.

[33] P. Polack, F. Altché, B. d'Andréa Novel, and A. de La Fortelle, "Guaranteeing consistency in a motion planning and control architecture using a kinematic bicycle model," *IEEE Trans. Intell. Veh.*, 2018.

[34] R. J. Williams and D. Zipser, "A learning algorithm for continually running fully recurrent neural networks," *Neural Comput.*, 1989.

[35] L. Ke, S. Choudhury, M. Barnes, W. Sun, G. Lee, and S. Srinivasa, "Imitation learning as f-divergence minimization," in *Int. Workshop Algorithmic Found. Robot. (WAFR)*, 2020.

[36] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

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

[38] T. Yin *et al.*, "One-step diffusion with distribution matching distillation," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024.

[39] X. Huang, Z. Li, G. He, M. Zhou, and E. Shechtman, "Self Forcing: Bridging the train-test gap in autoregressive video diffusion," in *Adv. Neural Inf. Process. Syst.*, 2025.

[40] J. Ho and T. Salimans, "Classifier-free diffusion guidance," 2022.