Analyzing Closed-loop Training Techniques for Realistic Traffic Agent Models in Autonomous Highway Driving Simulations

Matthias Bitzer[1] Reinis Cimurs[2] Benjamin Coors[2] Johannes Goth[1]
Sebastian Ziesche[1] Philipp Geiger[1]
Maximilian Naumann[1]
Robert Bosch GmbH, [1]Renningen, Germany; [2]Feuerbach, Germany
{matthias.bitzer3, reinis.cimurs, benjamin.coors, johannes.goth}@de.bosch.com
{sebastian.ziesche, philipp.w.geiger, maximilian.naumann}@de.bosch.com

Abstract

Simulation plays a crucial role in the rapid development and safe deployment of autonomous vehicles. Realistic traffic agent models are indispensable for bridging the gap between simulation and the real world. Many existing approaches for imitating human behavior are based on learning from demonstration. However, these approaches are often constrained by focusing on individual training strategies. Therefore, to foster a broader understanding of realistic traffic agent modeling, in this paper, we provide an extensive comparative analysis of different training principles, with a focus on closed-loop methods for highway driving simulation. We experimentally compare (i) open-loop vs. closed-loop multi-agent training, (ii) generative adversarial vs. deterministic supervised training, (iii) the impact of reinforcement losses, and (iv) the impact of training alongside log-replayed agents to identify suitable training techniques for realistic agent modeling. Furthermore, we identify promising combinations of different closed-loop training methods.

1. Introduction

Modeling the behavior of traffic participants is a crucial component in the development process of autonomous driving systems. Multi-agent driver models are utilized, for example, in simulation [2, 35] to benchmark planners or in planning systems themselves to reason about the behavior of other traffic participants [11]. However, most deployed driver models are rule-based and are not able to capture behavior outside their manually specified rules. Data-driven driver models offer an alternative that is able to learn behavior directly from real-world data.

Many general learning methods have been proposed to learn multi-agent driving policies, including simple one-step supervised learning [2, 12], closed-loop deterministic imitation learning [19, 30, 36], adversarial imitation learning [17, 22, 37] and combinations of imitation with Reinforcement Learning (RL) [23, 24, 30, 31, 43, 45]. Most recent methods have in common that the training is executed closed-loop, meaning that the model directly executes a sequence of actions with a differentiable forward model in the loop (see Figure 1), instead of predicting the next action given a ground-truth state (which we refer to as open-loop). This enables the policy to reason about the future consequences of a given action. While many of the aforementioned works propose new training methods and/or architectures, a systematic, comparative, empirical study of high-level training principles is often missing. But such a study is important to understand the sim-to-real gap induced by different training paradigms, in particular when using them for safety testing. Our goal and contribution in this paper is to obtain a systematic analysis of different training paradigms for multi-agent driver models, with a particular focus on closed-loop methods in highway scenarios to identify most promising methods. The analysis includes the following dimensions:

- *Closed-loop vs. open-loop:* While it is already evident that closed-loop training is beneficial, we reevaluate this claim over a larger set of training methods.
- *Deterministic imitation vs. generative adversarial imitation:* Recent methods used Model-based Generative Adversarial Imitation Learning (MGAIL) [5] to train a probabilistic driver model. However, also training purely deterministically in a closed-loop supervised fashion is possible [19, 30, 36]. We directly compare performance of both methods with the same network architecture.
- *Pure Imitation vs. additional reinforcement learning signal:* There is evidence that training a policy with imitation combined with reinforcement learning is beneficial [24]. We validate and quantify its benefit in closed-loop

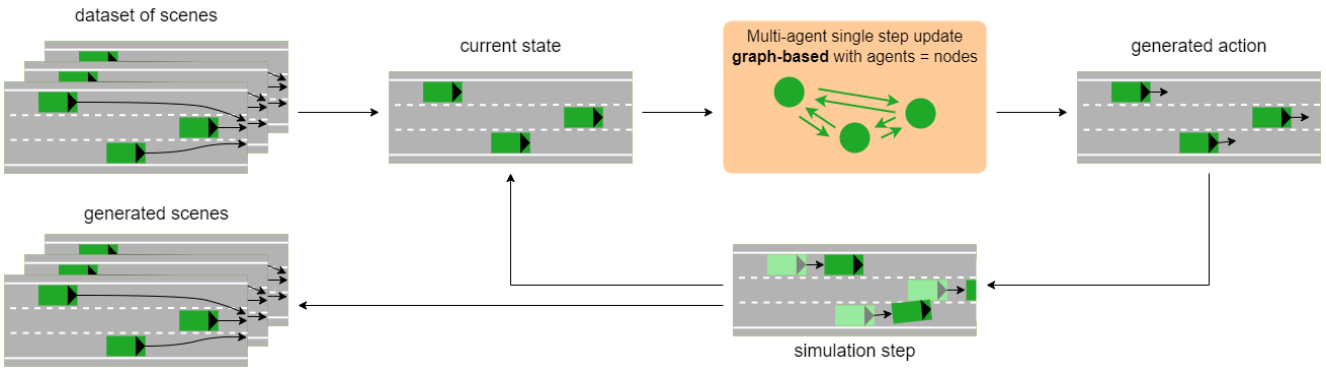

Figure 1. Problem setting - closed-loop multi-agent policy learning for autonomous driving simulation.

training.

- *Log-replay vs. multi-agent training:* Most methods either propose a single-agent training alongside log-replayed agents or a multi-agent learning scheme. The comparison of the two is infrequent. Therefore, we compare performance of both schemes in the same training scenarios.

Each training method has its theoretical benefits and shortcomings. Policies trained via deterministic supervised training might lack diverse behavior, but can be trained in a stable way. Adversarial training might, in theory, be able to match state distributions perfectly but are hard to train and it is unclear if the discriminator matches the distribution on features that are actually relevant for the driving task. Furthermore, one can enforce driving properties that are important for the driving task, such as collision avoidance, via a reinforcement learning signal but it is unclear how much this impacts other realism properties.

To execute our study, we propose an intuitive multi-agent policy parameterization that can be employed in all training methods with minimal adjustments. The architecture is based on a multi-agent Graph Neural Net (GNN) encoder that is invariant to rotation and translation of the scene and only operates on differentiable features. The decoder can be a stochastic head (for MGAIL), a deterministic head (for deterministic supervised learning) or a head that maps to discriminator scores. Our experiments are executed on the exiD dataset [26] consisting of sixteen hours of real-world driving data that focuses on agent-to-agent interaction on highway on-ramp and off-ramp scenarios. We identify the following useful strategies for training realistic traffic agent models on these scenarios:

1. All investigated closed-loop training strategies show superior performance over their open-loop counterparts.
2. A reinforcement signal improves crucial metrics such as collision rate, but easily deteriorates other realism metrics.
3. Deterministic supervised learning can be competitive with probabilistic adversarial learning in highway scenarios.
4. Combining different closed-loop policy learning strategies improves crucial metrics such as collision rate while maintaining realism.

2. Related Work

Rule-based driver models. The driver modeling task is often solved by employing rule-based methods for creating a decision-making policy. Generally, rule-based driver models express the driving task as a set of parameterized functions, such as the Intelligent Driver Model (IDM) [40] and its extensions [32, 48]. Their popularity stems from the simple implementation and parameterization based on ego agent's velocity, distance to other vehicles, and the velocity difference. To obtain the best performing parameters for this method, data driven approaches have been used in [18, 27, 46]. Other works [7, 15, 20, 21, 41] parameterize car following models based on surrounding vehicle features. While rule-based models are highly interpretable and computationally efficient, they lack expressiveness w.r.t. realism of the driver behavior and suffer from poor generalization [9, 38]. To address these issues, different learning-based methods have come to the forefront in autonomous driving research.

Driver modeling with Reinforcement Learning. In RL implementations [8, 33, 39] the driving policy is trained to maximize a pre-specified reward obtained through the interaction with the environment. Typically, the environment dynamics are non-differentiable or even unknown. Designing a reward function that captures realistic motion behavior is extremely difficult due to the subtle intricacies of human decision making. Assuming that the reward is a function purely defined by the state and is dynamics invariant [13], its function can be inferred through Inverse Reinforcement Learning (IRL) [47]. Yet, IRL is expensive to run and difficult to scale [42].

Driver Modeling with Imitation Learning. Imitation Learning (IL) is used to train a policy solely from expert

demonstrations. Classically, Behavioral Cloning (BC) can be used in an open-loop setting to obtain a trained policy directly from ground truth demonstrations [1, 2, 12]. However, these methods suffer from distribution shift in long tail rollouts [29]. This can be alleviated with applying augmentations [2] or training in closed-loop where the training is performed directly in a sequential decision-making manner. Here, differential simulation accumulates the loss over multiple steps [17, 19, 31, 35, 36]. This requires either augmenting the input data if working with rasterized representations of environment [31] or using vector representations [14] to have fully differentiable features. However, pure IL can be insufficient to learn safe and reliable policies due to a rareness of critical scenarios in the ground truth data [24]. Therefore, loss in IL is augmented with RL rewards or common sense losses. In [23, 30, 31, 43, 45] a level of infraction is used to penalize the collisions and road departures in the loss.

Driver Modeling with Generative Adversarial Learning. Adversarial learning is an alternative approach for imitating expert behavior. Here, the loss function is replaced with a learned discriminator, and the driving policy is trained to fool the discriminator. Foundational work that combined adversarial and policy learning was done with Generative Adversarial IL (GAIL) [5, 22, 34, 37]. These methods typically need an interplay between discriminator optimization and solving the RL problem. Importantly, when the environment dynamics is known and differentiable, GAIL can be replaced with model-based generative adversarial IL (MGAIL), which can be used for closed-loop training [34]. In the light of driver modeling, it was used for single-agent planner learning [5] and multi-agent driver-modeling for simulations [17]. While those methods alleviate the necessity to specify a loss, they are difficult to train [10, 25] and can still suffer from a lack of realism in feature distribution if not addressed in the training process [44].

3. Problem Formulation and Base Policy

In this section, we phrase our problem formulation and introduce our base policy parameterization.

Problem Formulation. Our driver modeling goal is to learn a multi-agent policy $\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$, where $\mathbf{a_t} = (a_{1,t}, \dots, a_{M,t})$ denotes the actions of M traffic participants at time step t and $\mathbf{s_t}$ denotes the states of all agents, which contain information like the position, speed etc. , and the local map, at time step t. Given a set of ground truth trajectories $\mathcal{D} = \{\tau_i\}_{i=1}^N$ with $\tau_i = (\mathbf{s}_0^{(i)}, \mathbf{a}_1^{(i)}, \dots, \mathbf{s}_T^{(i)}, \mathbf{a}_T^{(i)})$ the goal is to learn the parameters θ of the policy such that it induces a distribution over trajectories $p(\tau) = p(\mathbf{s}_0)\prod_{t=0}^T p(\mathbf{s}_{t+1}|\mathbf{a}_t, \mathbf{s}_t)\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$ that resembles the distribution of trajectories in $\mathcal{D}$. Here, we assume a known, deterministic and differentiable transition model $p(\mathbf{s}_{t+1}|\mathbf{a}_t, \mathbf{s}_t)$, which is a common assumption in driver modeling [30, 36]. In particular, we use position delta actions for each agent (detailed in differential update step). The main challenge of driver modeling consists of learning the underlying policy π_θ and choosing the method to fit the generated trajectories to the ground truth trajectories. In the following paragraphs we propose our policy parameterization, that induces an intuitive inductive bias for simulated multi-agent driving.

Policy architecture. Our multi-agent policy $\pi_\theta(\mathbf{a}_t|\mathbf{s}_t)$ follows an encoder-decoder architecture. The encoder takes in the multi-agent state $\mathbf{s}_t$ and returns an encoding for each agent:

$$\mathbf{h}_t := [h_{1,t}, \dots, h_{M,t}] = \mathrm{Encoder}(\mathbf{s}_t) \tag{1}$$

For multi-agent scenarios it is crucial that the Encoder can deal with different number of agents in the scene. Moreover, to be able to compare the effects of different training methods in simulation for vehicle-to-vehicle interactions, it is beneficial to explicitly define the agent relations in the network architecture. Therefore, we propose a GNN-based architecture where agents are the nodes in a locally connected graph. For a given multi-agent state $\mathbf{s}_t$ we extract initial node features $\mathbf{n}^{(0)} = [n_1^{(0)}, \dots, n_M^{(0)}]$ and initial edge features $\mathbf{e}^{(0)} = [e_{i\to j}^{(0)}]_{i\neq j}$ and process them via repeating message-passing layers with index $k = 0, \dots, K-1$

$$\mathbf{n}^{(k+1)}, \mathbf{e}^{(k+1)} = \mathrm{MessagePassing}(\mathbf{n}^{(k)}, \mathbf{e}^{(k)}) \tag{2}$$

The output of the encoder is given via $\mathbf{h}_t = \mathbf{n}^{(K)}$.

Depending on the training method (see Section 4) the decoder of the policy either maps the agent-wise encoding directly to the 2D action space via a weight-shared MLP,

$$\mathbf{a}_t = [\mathrm{MLP}(h_{1,t}), \dots, \mathrm{MLP}(h_{M,t})] = \mathrm{Decoder}(\mathbf{h}_t). \tag{3}$$

or each MLP maps to the parameters $\theta_{i,t}$ of a 2D distribution like a Gaussian or Gaussian Mixture (GMM) which defines an agent-specific distribution over our action space.

Node features. The initial node features of the GNN contain information about the agents' poses and kinematics, the agents' local map, and further information about the agent, like the agent type and its dimensions. First, we embed poses, kinematics, agent type and dimensions into a single embedding vector $h_{i,t}$ via an MLP. The map is represented as a sequence of line strings in the local coordinate system of the agent. Those line strings are associated with the boundary lines of each lane segment. The line strings are embedded with Multi-Head Attention (MHA) resulting in a sequence of line embeddings $[l_{i,j}]_{j=1}^J$. The initial embedding to the GNN is computed by fusing map information via cross attention (with residual) from $h_{i,t}$ to $[l_{i,j}]_{j=1}^J$. All features are computed in the local reference frame of each agent and are thus rotation and translation invariant.

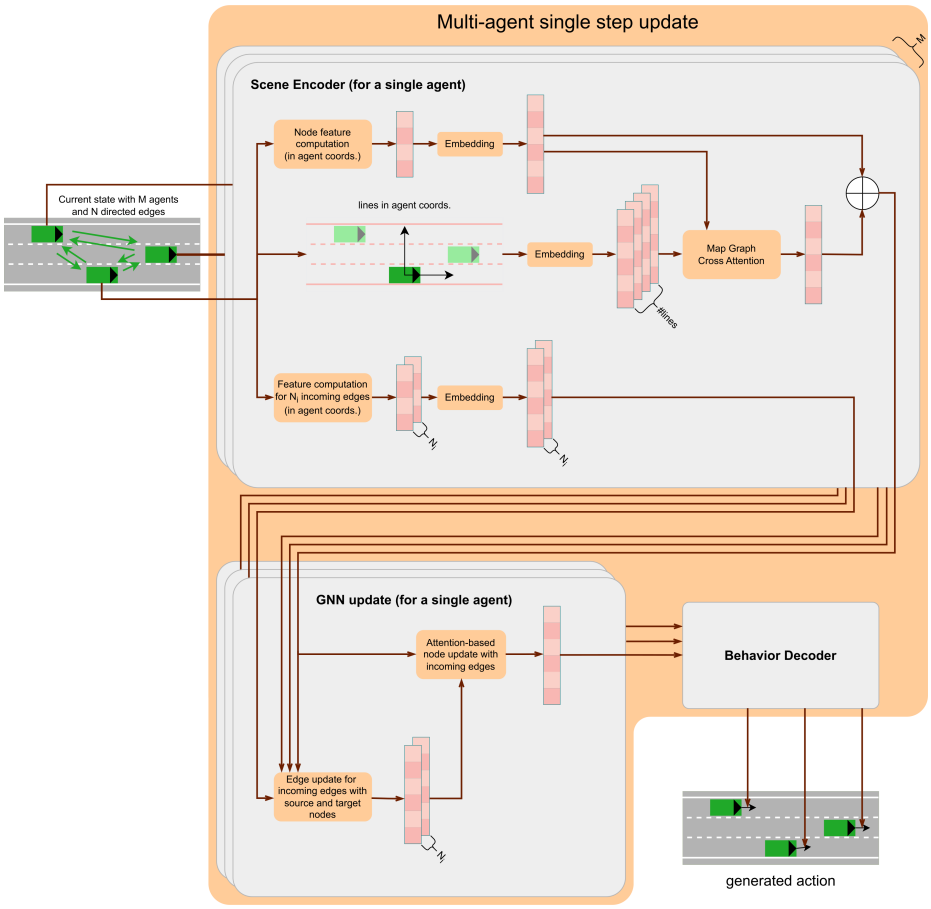

Figure 2. Architecture Overview

Edge features. The edge features capture relevant properties for interaction between source and target agents. The edge features therefore consist of the distance and velocity difference between source and target agents, relative position history between source and target agent for $t-2:t$ in target agents' coordinate frame, heading difference[1], and the time to collision between the source and target agent, clipped at a maximum value of 10s. All these features are invariant to rotations and translations of the agent pair.

Message-Passing. Intuitively, our GNN has the inductive bias that the agents are the nodes in the graph, their interaction is modeled via edges, and the reasoning over others and the resulting behavior is computed with the message-passing steps, as principally proposed by [6]. Concretely, our message passing module employs an edge model that updates the edge features via $e_{i\rightarrow j}^{(k+1)} = \text{MLP}([h_i^{(k)}, h_j^{(k)}, e_{i\rightarrow j}^{(k)}])$ and uses cross-attention (with residual) between $h_j^{(k)}$ and $[e_{i\rightarrow j}^{(k+1)}]_{i=1}^{M}$ to get $h_j^{(k+1)}$. In this way, each target agent can focus on the relevant agents that might interact with it.

[1]Using cos and sin of the difference, to ensure continuity of the feature.

Differentiable update step. A crucial component to enabling closed-loop training is a differentiable forward step, that enables propagating gradients through the steps of the trajectory. As actions $a_{i,t}$, we use the position deltas in the local reference frame of each agent, i.e., for i's local 2D position at time $t+1$, we have $(x_{i,t+1}, y_{i,t+1}) = (x_{i,t}, y_{i,t}) + a_{i,t}$. After transforming this to the global reference frame, the position and the heading of each agent are updated (see Figure 1). The new multi-agent state $\mathbf{s}_{t+1}$ is calculated via differentiable transformations of all features given the new locations and headings. Crucially, this enables MGAIL and differentiable simulation training to propagate future error through time to earlier actions. We investigate the impact of different variants of closed-loop training in our experiments. Full architecture overview is depicted in Fig. 2.

4. Compared Training Approaches

We introduce the different training paradigms we analyse for learning realistic driver models trained in simulation.

Behavioral Cloning. The first intuitive training method

is supervised one-step imitation learning, also called behavioral cloning. Here, we minimize an imitation loss over the one-step state-action distribution. For example, one might minimize the expected negative log likelihood of the policy under the one-step data distribution, $\mathcal{L}_{\text{BC}}(\theta) = \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}[-\log \pi_\theta(\mathbf{a}|\mathbf{s})]$. While it appears to be an intuitive principle to train a policy, it has been shown repeatedly [16, 29] that it can easily lead to compounding errors and unrealistic distributions over (multi-step) trajectories $p(\tau)$. However, we use this method for comparison purposes, as pretraining (see Section 5) and as regularizer (see MGAIL).

Differentiable Simulation. Since our forward model is differentiable, we can alleviate the compounding error problem via training π_θ through differentiable simulation, aka propagating gradients through time. Here, we consider the policy to be a deterministic mapping from states to actions $\mathbf{a}_t = \pi_\theta(\mathbf{s}_t)$ rather than a probability distribution. Given initial state $\mathbf{s}_0$ and a generated trajectory $\tilde{\tau} = (\mathbf{s}_0, \tilde{\mathbf{a}}_0, \tilde{\mathbf{s}}_1, \ldots, \tilde{\mathbf{s}}_1)$, we use as loss $\mathcal{L}_{\text{DS}}(\theta) = \mathbb{E}_{\mathbf{s}_0\sim\mathcal{D}}[\sum_{t=1}^{T} d(\mathbf{s}_t, \tilde{\mathbf{s}}_t)|\theta]$, where $d(\mathbf{s}_t, \tilde{\mathbf{s}}_t)$ is a weighted mean squared error (MSE) between the (x, y) positions in the states averaged over all agents. When training via differentiable simulation we pretrain the weights with BC, where we also replace the log likelihood loss with the weighted MSE loss.

MGAIL. Recent methods [5, 22, 37] utilized generative adversarial networks to learn driver models. Here, we train a discriminator D_ψ in addition to the policy π_θ. The discriminator is trained to classify states into the ones that come from ground truth and the ones that are generated via the π_θ. It maps from states to probability scores for each agent, thus $D_\psi(\mathbf{s}) \in [0, 1]^M$, and is trained via the cross entropy loss $\mathcal{L}(\psi) = \mathbb{E}_{\mathbf{s}\sim\mathcal{D}}[-\log D_\psi(\mathbf{s})] + \mathbb{E}_{\mathbf{s}\sim\pi_\theta}[-\log(1 - D_\psi(\mathbf{s}))]$ (here the expectation includes an additional averaging along the time dimension, over all states $\mathbf{s}$ of the individual trajectories). We parameterize our discriminator in the same way as the policy, except that the MLP in the decoder maps to $[0, 1]$ instead of the 2D action space.

The policy/generator is trained to fool the discriminator and thus minimizes the loss $\mathcal{L}_{\text{MGAIL}}(\theta) = \mathbb{E}_{\mathbf{s}\sim\pi_\theta}[\log(1 - D_\psi(\mathbf{s}))]$. It is important to note that gradients in this loss, can also propagate to previous time points, because of the differentiable forward model. Here, the decoder of the policy maps to the parameters of a proper probability distribution and is either parameterized via a Gaussian or Gaussian mixture distribution. The loss can be approximated via sampling with the reparameterization trick.

Differentiable Collision Loss. Combining data-based losses with reinforcement learning (RL) losses has been shown to be beneficial for learning planner policies [24]. Also, for driver modeling, enforcing certain aspects, like preventing collisions, is crucial. However, directly applying additional RL-like losses can induce unrealistic trajectories. To investigate the impact of RL-losses we consider a differentiable collision loss $L_{\text{Collision}}(\theta)$, as proposed in [35], that we use as an auxiliary loss in addition to the data-based losses.

Combination of Methods. In our experiments, we combine different methods, and denote this by "+" signs. For example, we can combine differentiable simulation and MGAIL via minimizing the loss $\mathcal{L}(\theta) = \mathcal{L}_{\text{DS}}(\theta) + \mathcal{L}_{\text{MGAIL}}(\theta)$ for the generator and train the discriminator as in MGAIL. Similarly, we can combine differentiable simulation with the collision loss, which can be seen as a closed-loop version of the method in [24]. Mixing different methods can combine benefits of both methods and alleviate problems that one training principle has when used on its own. Importantly, one needs to consider proper weighting of the loss functions.

5. Experiments & Results

In the following, we present our ablation study on the different training methods for modeling highway traffic agents. First, we introduce our experimental setup and show the results. In Section 6, we summarize the high-level findings of our experiments.

Dataset. We evaluate on the exiD dataset [26], a real-world trajectory dataset that contains drone-recorded driving data from highway entries and exits in Germany. We extract training and evaluation data by cutting the exiD recordings into snippets of 10 second length, which we downsample to a frequency of 2 Hz. In total, our dataset consists of 5750 recording snippets, which we refer to as rollouts, where all rollouts of one recording are assigned to one split. The dataset is organized into a train, validation and test split with 4461, 737 and 552 rollouts, respectively. Furthermore, we ensure that each split contains rollouts from each of the seven exiD recording locations.

Simulation setup. We simulate for the full 10 seconds of our rollouts at 2 Hz. In order to enable the computation of the node features, which include the agent's speed and acceleration, we only start controlling an agent after 3 steps of it being present in the rollout. This means that an agent effectively performs three initial log-replay steps before it is controlled by the model. **Evaluated Methods.** We compare the methods presented in Section 4 along with their combinations. Concretely, for open-loop training, we evaluate BC training with maximum likelihood (`BC-LL`) using a Gaussian head (`BC Gaussian-LL`) as well as a Gaussian mixture head (`BC GMM-LL`). We consider training BC with weighted MSE combined with orientation loss (`BC wMSE + Orientation`). Here, `BC wMSE` refers to an MSE loss where different weights are applied for x and y axis deviations, since lateral motion is predominantly smaller in

Table 1. Loss ablation study for multi-step closed-loop trainings when trained on controlling **all agents** and evaluated when controlling **all agents**.

Method	Col. (%)	Off. (%)	ADE (m)	Speed JSD $\times 10^{-2}$	Acc. JSD $\times 10^{-2}$	N_{LC} JSD $\times 10^{-2}$
IDM	0.8 ± 0.0	**0.00 ± 0.0**	5.31 ± 0.0	4.0 ± 0.0	17.64 ± 0.0	0.23 ± 0.0
BC Gaussian-LL	2.67 ± 0.11	6.42 ± 0.42	2.22 ± 0.17	0.63 ± 0.41	8.02 ± 4.6	2.28 ± 0.16
BC GMM-LL	3.1 ± 0.62	7.05 ± 0.78	3.42 ± 0.41	2.41 ± 0.66	23.16 ± 6.52	2.23 ± 0.33
BC wMSE + Orientation	3.1 ± 0.65	4.36 ± 0.73	4.15 ± 0.34	3.51 ± 0.8	12.54 ± 4.33	1.5 ± 0.22
DiffSim MSE	1.1 ± 0.74	3.05 ± 2.11	1.62 ± 0.77	0.28 ± 0.3	1.82 ± 1.57	0.92 ± 1.03
DiffSim wMSE	0.51 ± 0.11	0.68 ± 0.31	1.25 ± 0.13	0.2 ± 0.09	1.05 ± 0.38	0.48 ± 0.09
DiffSim wMSE + Collision	0.39 ± 0.42	2.52 ± 1.15	3.57 ± 3.31	3.24 ± 5.78	2.5 ± 1.7	0.66 ± 0.4
MGAIL + BC-LL (Gauss.)	1.7 ± 0.58	2.95 ± 1.2	1.95 ± 0.24	0.42 ± 0.18	1.82 ± 0.52	0.6 ± 0.56
MGAIL + BC-LL (GMM)	2.87 ± 2.42	2.77 ± 0.52	7.8 ± 8.31	9.34 ± 10.7	18.55 ± 10.69	**0.19 ± 0.13**
MGAIL + DiffSim wMSE (Gauss.)	0.39 ± 0.06	0.59 ± 0.35	**0.93 ± 0.03**	**0.08 ± 0.03**	**0.86 ± 0.15**	0.32 ± 0.15
MGAIL + DiffSim wMSE (GMM)	0.35 ± 0.19	0.36 ± 0.32	1.0 ± 0.18	0.11 ± 0.1	1.03 ± 0.71	0.31 ± 0.15
MGAIL + Diff. wMSE + Col. (Gauss.)	**0.17 ± 0.09**	2.28 ± 0.84	1.47 ± 0.09	0.23 ± 0.05	2.35 ± 0.48	0.26 ± 0.21
MGAIL + Diff. wMSE + Col. (GMM)	0.17 ± 0.11	1.49 ± 0.42	1.48 ± 0.16	0.26 ± 0.06	1.86 ± 0.85	0.35 ± 0.42

highway scenarios. `Orientation` is a loss expressed as:

$$d_{\text{Orientation}}(\hat{\mathbf{a}}, \mathbf{a}) := \frac{1}{M} \sum_{i=1}^{M} d_{\text{MSE}}((\hat{\delta}_{x,i}, \hat{\delta}_{y,i}), (\delta_{x,i}, \delta_{y,i})), \tag{4}$$

where $(\delta_{x,i}, \delta_{y,i})$ is the ground-truth normalized heading of the next time step for agent i and $(\hat{\delta}_{x,i}, \hat{\delta}_{y,i})$ is the heading that would result from the chosen action. M is the number of vehicles. For closed-loop training, we consider deterministic training through differentiable simulation with a weighted MSE loss (`DiffSim wMSE`) and an unweighted MSE loss (`DiffSim MSE`). Furthermore, MGAIL is trained in combination with a BC loss (`MGAIL + BC-LL`) and in combination with differential simulation (`MGAIL + DiffSim wMSE`), using a Gaussian or GMM head. To investigate the impact of an additional reinforcement loss, we consider adding the differentiable collision loss to differentiable simulation (`DiffSim wMSE + Collision`) and to the combination of MGAIL and differentiable simulation (`MGAIL + DiffSim wMSE + Collision`). For all closed-loop methods, we execute a combined multi-agent training and training along log-replay agents.

Training setup. We train each method for 100 epochs. Hyperparameters for each training method are determined with hyperparameter sweeps based on results from evaluation on the validation set. Training of BC methods is performed at every step of the rollout, using the method's respective losses. Differential simulation methods are initialized from a pre-trained `BC wMSE + Orientation` model. Execution is simulated for the full length of the rollout. MGAIL methods are initialized with weights from a pre-trained `BC-LL` model with respective log-likelihood loss. In single-agent log-replay trainings, we randomly select one controlled agent in the rollout, with a probability proportional to the agent's number of time steps in the scene for each batch element.

Evaluation setup. All models are evaluated in closed-loop simulation on the test split over all the time steps in the rollout. In our evaluation, we compare two control settings of the agents. First, the setting of controlling all agents at once. For the test split of the exiD dataset, this means that we control, on average, about 26 agents per scene. Since model training is not deterministic, we evaluate their performance variance by training the base BC models with 5 different seed values, then use these model weights to initialize the respective differential simulation and MGAIL methods. The results are shown in Table 1 and Table 2 which report the mean and standard deviation of the respective 5 evaluations. Second, the setting of only controlling one agent, with all other agents being log-replayed. Here, the controlled agent in the scene is selected deterministically as the agent with the most timesteps in the rollout.

Metrics. We employ a variety of metrics to judge a model's quality and realism. Metrics are computed on the test split for agents and frames, where an agent is controlled in a respective frame by the model.

A first set of metrics evaluates the model's infractions. Here, a **collision rate** is defined as the percentage of agents with at least one collision in a given rollout. It is implemented as a polygon intersection check between the bounding boxes of the agents. An **off-road rate** computes the percentage of frames at which a controlled agent drives off the road, i.e., outside the highway lanes.

We also report the Average Displacement Error (**ADE**) which is the L2 distance between the ground-truth and generated trajectories for a fixed horizon of 5 seconds. This metric measures how closely the generated trajectories follow the ground-truth trajectories.

We evaluate the distributional realism of the models with metrics that compute the Jensen-Shannon Divergence (JSD) between the ground-truth and generated distributions of the agent's **speed**, **acceleration** and **number of lane changes** N_{LC}. The JSD is computed between histograms with 100 equisized bins for speed, acceleration and number of lane changes in a range that covers the minimum and maximum of the generated and ground-truth metric values. **Re-**

Table 2. Loss ablation study for multi-step closed-loop trainings when trained on controlling **one agent** and evaluated when controlling **all agents**.

Method	Col. (%)	Off. (%)	ADE (m)	Speed JSD $\times 10^{-2}$	Acc. JSD $\times 10^{-2}$	N_{LC} JSD $\times 10^{-2}$
DiffSim MSE	1.21 ± 0.64	2.71 ± 1.41	1.69 ± 0.41	0.33 ± 0.27	2.22 ± 0.8	0.45 ± 0.32
DiffSim wMSE	0.63 ± 0.16	1.46 ± 0.38	1.69 ± 0.17	0.27 ± 0.06	1.57 ± 0.45	0.63 ± 0.36
DiffSim wMSE + Collision	**0.39 ± 0.17**	**0.75 ± 0.39**	1.83 ± 0.18	0.45 ± 0.2	4.79 ± 2.81	0.88 ± 0.36
MGAIL + BC-LL (Gauss.)	2.65 ± 0.27	4.57 ± 0.67	2.14 ± 0.12	0.42 ± 0.16	2.54 ± 0.65	1.62 ± 0.43
MGAIL + BC-LL (GMM)	1.88 ± 0.25	3.74 ± 0.16	2.16 ± 0.13	0.8 ± 0.22	4.35 ± 1.23	0.64 ± 0.25
MGAIL + DiffSim wMSE (Gauss.)	0.65 ± 0.21	1.59 ± 0.5	1.16 ± 0.02	0.18 ± 0.05	3.1 ± 1.52	**0.14 ± 0.09**
MGAIL + DiffSim wMSE (GMM)	0.44 ± 0.31	1.12 ± 0.59	**1.14 ± 0.09**	**0.11 ± 0.0**	**1.11 ± 0.25**	0.2 ± 0.14
MGAIL + Diff. wMSE + Col. (Gauss.)	1.28 ± 0.13	2.23 ± 0.47	1.59 ± 0.12	0.41 ± 0.11	2.24 ± 0.23	0.8 ± 0.18
MGAIL + Diff. wMSE + Col. (GMM)	0.54 ± 0.37	1.34 ± 0.5	1.46 ± 0.16	0.18 ± 0.04	1.41 ± 0.5	0.4 ± 0.33

sults. Table 1 presents our results for all mentioned methods when trained in a combined multi-agent way (non log-replay). Here, as baseline we compare the results to IDM [40] with MOBIL [20] to allow for lane changes in highway and on/off-ramping scenarios. In Table 2, we show the results of the closed-loop methods when trained alongside log-replay agents. Videos of ground-truth and generated scenarios can be found in the supplementary material.

6. Analysis of Results

We analyze the experimental results and draw high-level conclusions for driver model training:

Closed-loop can be beneficial over open-loop training: Theoretical and experimental findings have been established in the past regarding the benefit of closed-loop multi-agent training over simpler, open-loop approaches, e.g., the "compounding error" phenomenon of BC training [16, 28, 29]. Nonetheless, open-loop approaches are still often used due to their simplicity [1, 17]. Our experimental findings confirm the case made for closed-loop training. From Table 1, we see that the two closed-loop paradigms - differentiable simulation and MGAIL - significantly outperform open-loop BC methods. This is evident when comparing `DiffSim` to `BC wMSE + Orientation` as well as `MGAIL + BC-LL` to the respective `BC-LL` method with the same head. In all cases, the closed-loop method is better or equal on (almost) every metric. Additionally, single-agent closed-loop training with log replay of surrounding agents (Table 2) can be seen as "less closed-loop" than full multi-agent closed-loop training (Table 1). Also, here, the former outperforms the latter. Furthermore, closed loop methods lead to more realistic scenarios in terms of JSD metrics and and almost always to lower collision rates than an IDM model.

Reinforcement loss can harm realism: High collision rates remain one of the biggest open challenges in learned driver models, and are a key indicator of where realism is still limited [5, 17]. As a remedy, various works [3, 35] have added a "common sense loss", also called "reinforcement loss". Our experiments show that such a reinforcement learning aspect can indeed significantly bring down the collision rate (e.g., compare collision rate metric between `DiffSim wMSE` and `DiffSim wMSE + Collision` or between the respective MGAIL methods). However, this can come at the substantial cost of losing realism in other aspects of the model. For example, all three JSD metrics (speed, acceleration, N_{LC}) are worse (or equal) when comparing `DiffSim wMSE + Collision` **vs.** `DiffSim wMSE` or considering `MGAIL + DiffSim wMSE + Collision` **vs.** `MGAIL + DiffSim wMSE`. Here, the collision loss not only weakens the JSD metrics, but also has a negative effect on the off-road rate.

Deterministic supervised learning can be competitive with MGAIL: Recent work [3, 17] increasingly utilized adversarial learning to train driver models. It has the considerable advantage that this allows for probabilistic policies, and the loss is learned instead of being defined manually. However, adversarial methods come with training challenges and are usually less stable to train compared to supervised or variational inference methods [4]. In our experiments, we observe that methods employing `DiffSim` outperform `MGAIL + BC-LL` on all metrics, except of the acceleration JSD. While both methods are trained closed-loop and improve upon their open-loop BC pretrainings, we observe a stronger improvement in the metrics of `DiffSim MSE`, `DiffSim wMSE`, and `DiffSim wMSE + Collision`. However, we believe that this statement is dependent on the tuning of the MGAIL method as well as the probabilistic characteristics of the dataset, and should therefore be taken with caution. In particular, it might be the case that highway scenarios, including on/off-ramp situations, are not very multi-modal and thus a deterministic method is well suited.

Combinations of methods can be beneficial: One key insight of our experiments is that combinations of methods can counteract individual weaknesses. We already observed that a collision loss can reduce collision rate, at the price of having worse distributional realism. However, adding adversarial learning helps mitigate this effect, as can be seen when comparing `DiffSim wMSE + Collision` with `MGAIL + DiffSim wMSE + Collision`, where distributional

realism is improved over all metrics. Furthermore, `MGAIL + DiffSim wMSE` showed to be the best performing model in all metrics except of collision rate. This holds for combined as well as log-replay training. We think that the supervised learning signal helps to stabilize the adversarial training.

7. Conclusion

To summarize, we conducted a systematic analysis study of closed-loop imitation training principles for realistic traffic agent models for highway scenarios. We utilized the same GNN-based driving policy under different training paradigms, and reported quantitative experimental results as well as high-level insights with qualitative results given in the supplementary material. We find that each method on its own comes with individual weaknesses, and a combination of methods is needed to counteract them. In particular, we find that closed-loop training has significant advantages over open-loop training, that a reinforcement signal can destroy realism and that combinations of different closed-loop learning principles can improve overall performance. While the proposed architecture is intuitive for modeling driver interaction in highway scenarios and the drawn conclusions should carry over to other neural network implementations, additional studies including different interaction modeling paradigms (in particular transformer-based models) as well as extension to non-highway scenarios could add additional insights. This will be the focus of our future study.

References

[1] Mayank Bansal, Alex Krizhevsky, and Abhijit Ogale. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. *arXiv preprint arXiv:1812.03079*, 2018. 3, 7

[2] Luca Bergamini, Yawei Ye, Oliver Scheel, Long Chen, Chih Hu, Luca Del Pero, Błażej Osiński, Hugo Grimmet, and Peter Ondruska. Simnet: Learning reactive self-driving simulations from real-world observations. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages –. IEEE, 2021. 1, 3

[3] Raunak P Bhattacharyya, Derek J Phillips, Changliu Liu, Jayesh K Gupta, Katherine Driggs-Campbell, and Mykel J Kochenderfer. Simulating emergent properties of human driving behavior using multi-agent reward augmented imitation learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 789–795. IEEE, 2019. 7

[4] Sam Bond-Taylor, Adam Leach, Yang Long, and Chris G Willcocks. Deep generative modelling: A comparative review of vaes, gans, normalizing flows, energy-based and autoregressive models. *IEEE transactions on pattern analysis and machine intelligence*, 44(11):7327–7347, 2021. 7

[5] Eli Bronstein, Mark Palatucci, Dominik Notz, Brandyn White, Alex Kuefler, Yiren Lu, Supratik Paul, Payam Nikdel, Paul Mougin, Hongge Chen, et al. Hierarchical model-based imitation learning for planning in autonomous driving. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8652–8659. IEEE, 2022. 1, 3, 5, 7

[6] S. Casas, C. Gulino, R. Liao, and R. Urtasun. Spagnn: Spatially-aware graph neural networks for relational behavior forecasting from sensor data. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9491–9497, 2020. 4

[7] Partha Chakroborty and Shinya Kikuchi. Evaluation of the general motors based car-following models and a proposed fuzzy inference model. *Transportation Research Part C: Emerging Technologies*, 7(4):209–235, 1999. 2

[8] Dian Chen, Vladlen Koltun, and Philipp Krähenbühl. Learning to drive from a world on rails. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15590–15599, 2021. 2

[9] Di Chen, Meixin Zhu, Hao Yang, Xuesong Wang, and Yinhai Wang. Data-driven traffic simulation: A comprehensive review. *IEEE Transactions on Intelligent Vehicles*, 2024. 2

[10] Robert Dadashi, Léonard Hussenot, Matthieu Geist, and Olivier Pietquin. Primal wasserstein imitation learning. *arXiv preprint arXiv:2006.04678*, 2020. 3

[11] Katherine Driggs-Campbell, Vijay Govindarajan, and Ruzena Bajcsy. Integrating intuitive driver models in autonomous planning for interactive maneuvers. *IEEE Transactions on Intelligent Transportation Systems*, 18(12):3461–3472, 2017. 1

[12] Lan Feng, Quanyi Li, Zhenghao Peng, Shuhan Tan, and Bolei Zhou. Trafficgen: Learning to generate diverse and realistic traffic scenarios. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3567–3575. IEEE, 2023. 1, 3

[13] Justin Fu, Katie Luo, and Sergey Levine. Learning robust rewards with adversarial inverse reinforcement learning. *arXiv preprint arXiv:1710.11248*, 2017. 2

[14] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. Vectornet: Encoding hd maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11525–11533, 2020. 3

[15] Peter G Gipps. A behavioural car-following model for computer simulation. *Transportation Research Part B: Methodological*, 15(2):105–111, 1981. 2

[16] Jonathan Ho and Stefano Ermon. Generative adversarial imitation learning. *Advances in neural information processing systems*, 29, 2016. 5, 7

[17] Maximilian Igl, Daewoo Kim, Alex Kuefler, Paul Mougin, Punit Shah, Kyriacos Shiarlis, Dragomir Anguelov, Mark Palatucci, Brandyn White, and Shimon Whiteson. Symphony: Learning realistic and diverse agents for autonomous driving simulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2445–2451. IEEE, 2022. 1, 3, 7

[18] Venkatesan Kanagaraj, Gowri Asaithambi, CH Naveen Kumar, Karthik K Srinivasan, and R Sivanandan. Evaluation of different vehicle following models under mixed traffic con-

ditions. *Procedia-Social and Behavioral Sciences*, 104:390–401, 2013. 2

[19] Peter Karkus, Boris Ivanovic, Shie Mannor, and Marco Pavone. Diffstack: A differentiable and modular control stack for autonomous vehicles. In *Proceedings of The 6th Conference on Robot Learning*, pages 2170–2180. PMLR, 2023. 1, 3

[20] Arne Kesting, Martin Treiber, and Dirk Helbing. General lane-changing model mobil for car-following models. *Transportation Research Record*, 1999(1):86–94, 2007. 2, 7

[21] Stefan Krauß, Peter Wagner, and Christian Gawron. Metastable states in a microscopic model of traffic flow. *Physical Review E*, 55(5):5597, 1997. 2

[22] Alex Kuefler, Jeremy Morton, Tim Wheeler, and Mykel Kochenderfer. Imitating driver behavior with generative adversarial networks. In *2017 IEEE intelligent vehicles symposium (IV)*, pages 204–211. IEEE, 2017. 1, 3, 5

[23] Vasileios Lioutas, Adam Scibior, and Frank Wood. Titrated: Learned human driving behavior without infractions via amortized inference. *Transactions on Machine Learning Research*, 2022. 1, 3

[24] Yiren Lu, Justin Fu, George Tucker, Xinlei Pan, Eli Bronstein, Rebecca Roelofs, Benjamin Sapp, Brandyn White, Aleksandra Faust, Shimon Whiteson, et al. Imitation is not enough: Robustifying imitation with reinforcement learning for challenging driving scenarios. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7553–7560. IEEE, 2023. 1, 3, 5

[25] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for GANs do actually converge? In *Proceedings of the 35th International Conference on Machine Learning*, pages 3481–3490. PMLR, 2018. 3

[26] Tobias Moers, Lennart Vater, Robert Krajewski, Julian Bock, Adrian Zlocki, and Lutz Eckstein. The exid dataset: A real-world trajectory dataset of highly interactive highway scenarios in germany. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pages 958–964, 2022. 2, 5

[27] Mitra Pourabdollah, Eric Bjärkvik, Florian Fürer, Björn Lindenberg, and Klaas Burgdorf. Calibration and evaluation of car following models using real-world driving data. In *2017 IEEE 20th International conference on intelligent transportation systems (ITSC)*, pages 1–6. IEEE, 2017. 2

[28] Stéphane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 661–668. JMLR Workshop and Conference Proceedings, 2010. 7

[29] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011. 3, 5, 7

[30] Oliver Scheel, Luca Bergamini, Maciej Wolczyk, Blazej Osinski, and Peter Ondruska. Urban driver: Learning to drive from real-world demonstrations using policy gradients. In *Conference on Robot Learning, 8-11 November 2021, London, UK*, pages 718–728. PMLR, 2021. 1, 3

[31] Adam Ścibior, Vasileios Lioutas, Daniele Reda, Peyman Bateni, and Frank Wood. Imagining the road ahead: Multi-agent trajectory prediction via differentiable simulation. In *24th IEEE International Intelligent Transportation Systems Conference, ITSC 2021, Indianapolis, IN, USA, September 19-22, 2021*, pages 720–725. IEEE, 2021. 1, 3

[32] MN Sharath and Nagendra R Velaga. Enhanced intelligent driver model for two-dimensional motion planning in mixed traffic. *Transportation Research Part C: Emerging Technologies*, 120:102780, 2020. 2

[33] Shinya Shiroshita, Shirou Maruyama, Daisuke Nishiyama, Mario Ynocente Castro, Karim Hamzaoui, Guy Rosman, Jonathan DeCastro, Kuan-Hui Lee, and Adrien Gaidon. Behaviorally diverse traffic simulation via reinforcement learning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2103–2110. IEEE, 2020. 2

[34] Jiaming Song, Hongyu Ren, Dorsa Sadigh, and Stefano Ermon. Multi-agent generative adversarial imitation learning. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2018. 3

[35] Simon Suo, Sebastian Regalado, Sergio Casas, and Raquel Urtasun. Trafficsim: Learning to simulate realistic multi-agent behaviors. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 10400–10409. Computer Vision Foundation / IEEE, 2021. 1, 3, 5, 7

[36] Simon Suo, Kelvin Wong, Justin Xu, James Tu, Alexander Cui, Sergio Casas, and Raquel Urtasun. Mixsim: A hierarchical framework for mixed reality traffic simulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9622–9631, 2023. 1, 3

[37] Mahan Tabatabaie, Suining He, and Kang G Shin. Interaction-aware and hierarchically-explainable heterogeneous graph-based imitation learning for autonomous driving simulation. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3576–3581. IEEE, 2023. 1, 3, 5

[38] Siyu Teng, Xuemin Hu, Peng Deng, Bai Li, Yuchen Li, Yunfeng Ai, Dongsheng Yang, Lingxi Li, Zhe Xuanyuan, Fenghua Zhu, et al. Motion planning for autonomous driving: The state of the art and future perspectives. *IEEE Transactions on Intelligent Vehicles*, 2023. 2

[39] Marin Toromanoff, Emilie Wirbel, and Fabien Moutarde. End-to-end model-free reinforcement learning for urban driving using implicit affordances. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2

[40] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical review E*, 62(2):1805, 2000. 2, 7

[41] Rainer Wiedemann. Simulation des strassenverkehrsflusses. 1974. 2

[42] Zheng Wu, Liting Sun, Wei Zhan, Chenyu Yang, and Masayoshi Tomizuka. Efficient sampling-based maximum

entropy inverse reinforcement learning with application to autonomous driving. *IEEE Robotics and Automation Letters*, 5(4):5355–5362, 2020. 2

[43] Danfei Xu, Yuxiao Chen, Boris Ivanovic, and Marco Pavone. Bits: Bi-level imitation for traffic simulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2929–2936. IEEE, 2023. 1, 3

[44] Xintao Yan, Zhengxia Zou, Shuo Feng, Haojie Zhu, Haowei Sun, and Henry X Liu. Learning naturalistic driving environment with statistical realism. *Nature communications*, 14 (1):2037, 2023. 3

[45] Chris Zhang, James Tu, Lunjun Zhang, Kelvin Wong, Simon Suo, and Raquel Urtasun. Learning realistic traffic agents in closed-loop. In *7th Annual Conference on Robot Learning*, 2023. 1, 3

[46] Duo Zhang, Xiaoyun Chen, Junhua Wang, Yinhai Wang, and Jian Sun. A comprehensive comparison study of four classical car-following models based on the large-scale naturalistic driving experiment. *Simulation Modelling Practice and Theory*, 113:102383, 2021. 2

[47] Guanjie Zheng, Hanyang Liu, Kai Xu, and Zhenhui Li. Objective-aware traffic simulation via inverse reinforcement learning. *arXiv preprint arXiv:2105.09560*, 2021. 2

[48] Mofan Zhou, Xiaobo Qu, and Sheng Jin. On the impact of cooperative autonomous vehicles in improving freeway merging: a modified intelligent driver model-based approach. *IEEE Transactions on Intelligent Transportation Systems*, 18(6):1422–1428, 2016. 2