Conditional Prediction by Simulation for Automated Driving

Fabian Konstantinidis[*,†], Moritz Sackmann[*], Ulrich Hofmann[*], Christoph Stiller[†]

Abstract: Modular automated driving systems commonly handle prediction and planning as sequential, separate tasks, thereby prohibiting cooperative maneuvers. To enable cooperative planning, this work introduces a prediction model that models the conditional dependencies between trajectories. For this, predictions are generated by a microscopic traffic simulation, with the individual traffic participants being controlled by a realistic behavior model trained via Adversarial Inverse Reinforcement Learning. By assuming various candidate trajectories for the automated vehicle, we generate predictions conditioned on each of them. Furthermore, our approach allows the candidate trajectories to adapt dynamically during the prediction rollout. Several example scenarios are available at https://conditionalpredictionbysimulation.github.io/.

Keywords: Conditional Prediction, Multi-Agent Behavior Modeling, Simulation

1 Introduction

Predicting the future trajectories of surrounding traffic participants plays an essential role in automated driving. By anticipating future movements of nearby agents, such as vehicles and vulnerable road users, an automated vehicle (AV) can better plan maneuvers, reduce the risk of collisions, and ensure smoother interactions with other road users.

Although existing approaches, e.g., [1–3], effectively predict the future movements of individual traffic participants, they limit an AV to a reactive planning strategy, assuming that the predictions of surrounding vehicles remain unaffected by the AV's planned actions. In highly interactive situations, this often leads to the *freezing robot problem* [4], where the AV, unable to engage in cooperative planning, simply stops to avoid potential collisions. For example, when it is unable to merge in dense traffic because the predictions of surrounding vehicles do not react to the AV's plan.

One approach to resolving this is to condition the prediction on the AV's plan, often referred to as *conditional inference* [5]. This enables holistic planning: Repeatedly selecting candidate trajectories and predicting their impact on surrounding vehicles until

This work is a result of the joint research project STADT:up (19A22006E). The project is supported by the German Federal Ministry for Economic Affairs and Climate Action (BMWK), based on a decision of the German Bundestag. The author is solely responsible for the content of this publication.

[*]Pre-Development of Automated Driving, CARIAD SE, 38440 Wolfsburg, Germany. (e-mail: first-name.lastname@cariad.technology)

[†]Institute of Measurement and Control Systems, Karlsruhe Institute of Technology (KIT), 76131 Karlsruhe, Germany. (e-mail: fabian.konstantinidis@partner.kit.edu, stiller@kit.edu)

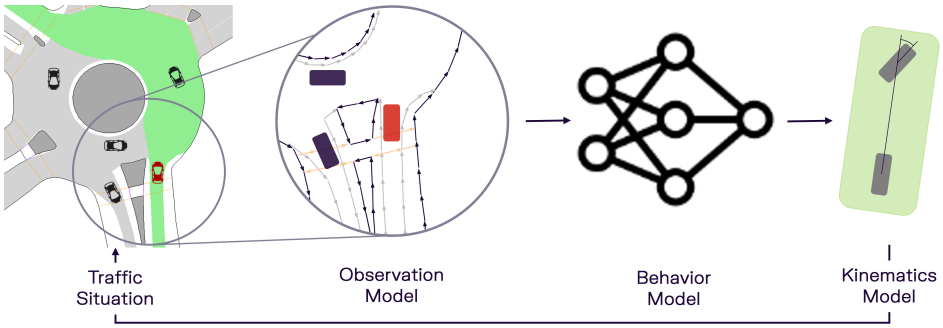

Figure 1: Single simulation step: Each vehicle observes the traffic situation locally, selects an action based on the observation, and executes it using the kinematics model.

a sufficiently good trajectory is found. A straightforward approach to this involves extending the prediction model to incorporate the AV's planned trajectory, improving the prediction by considering the AV's influence. However, this requires the AV's trajectory to be fixed during prediction, preventing truly interactive planning. We argue that an effective prediction model should capture the bidirectional interactions by conditioning the prediction on the plan of the AV while simultaneously allowing the plan to adapt in response to the evolving predictions, e.g., when planning stepwise in a tree structure [6].

In this work, we realize such a system by learning a reactive behavior policy and utilizing it for simulating the surrounding vehicles in a closed-loop simulation. As depicted in Figure 1, our framework predicts the evolution of traffic situations by stepwise simulating the movements of all surrounding vehicles using the learned behavior policy until the prediction horizon is reached. At each simulation step, the model processes an observation describing the traffic situation from each agent's perspective and executes appropriate control actions accordingly. This process is conducted stepwise and independently for each target vehicle, allowing agents to respond to each other's movements in the subsequent simulation steps. This approach not only facilitates scene-consistent predictions but also allows the AV to adapt its planned trajectory dynamically during the prediction rollouts.

Realistically simulating driving behavior requires a behavior model, typically derived from data-driven methods like Reinforcement Learning (RL) [7] or Imitation Learning (IL) [8]. While RL enhances robustness, it requires the definition of a reward signal that describes realistic behavior, which is a complex task. In contrast, Behavior Cloning (BC), an IL method, learns from expert demonstrations using supervised learning but suffers from *covariate shift* [9], where compounding errors during inference lead to situations differing from the training data, resulting in undefined behavior.

Similar to our previous work [10], we address these limitations by employing Adversarial Inverse Reinforcement Learning (AIRL) [11] to learn the behavior model. AIRL combines RL and IL by reconstructing the reward function that best explains the behavior of real-world drivers, and simultaneously learning a behavior model that maximizes this reward.

Contributions: The core contribution of this work is the integration of flexible behavior models, learned via AIRL, into a simulation framework to enable conditional motion prediction. Specifically, this study emphasizes the prediction of surrounding vehicles conditioned on a predefined plan for the AV, thereby capturing the bidirectional interactions between the AV and its surrounding agents. As we focus on prediction, we maintain fixed paths for the AV during prediction; however, the proposed stepwise approach enables the planning algorithm to adapt its strategy during the prediction rollout. In addition, we apply the model to various traffic situations with distinct road layouts.

2 Method

Traffic situations are simulated by executing a learned behavior model for every agent in the scene simultaneously. From the perspective of an individual agent, surrounding drivers are treated as part of the environment. When driving, the uncertainty arising from their unknown driving characteristics (e.g., individual driving styles or intentions) adds complexity to the sequential decision-making process, often formulated as a Partially-Observable Markov Decision Process (POMDP). Formally, a POMDP is characterized by the tuple $(S, O, A, T, R, \Omega, \gamma)$. In this framework, the agent cannot directly observe the true state $s \in S$ but instead is limited to a (noisy) observation $o \in O$ of the environment, determined by the observation model $\Omega : S \to O$ mapping from states to observations. Upon executing an action $a \in A$, the state of the environment is updated stochastically according to the transition probability density $T : S \times A \times S \to [0, \infty[$. Additionally, the agent receives a numerical reward defined by the reward function $R : S \times A \to \mathbb{R}$ as well as an observation of the new state of the environment. The discount factor $\gamma \in [0, 1[$ balances the trade-off between immediate and future rewards. The solution of a POMDP is the optimal policy $\pi^* : O \times A \to [0, \infty[$ mapping from observations to distributions over actions that maximize the expected cumulative reward over time $J(\pi) = \mathbb{E}_\pi \left[\sum_{k=0}^{\infty} \gamma^k r_k\right]$, where $r_k = R(s_k, a_k)$.

2.1 Reinforcement Learning

A common approach to maximizing $J(\pi)$ is through RL, where the agent interacts with a (simulated) environment via trial and error to discover effective actions. This involves two alternating steps: 1) The agent collects a new set of experiences $E = \{e_1, \ldots, e_M\}$, where $e_k = (o_k, a_k, r_k)$ represents one experience obtained by executing a single step in the environment. 2) The experiences gathered are used to adjust the policy, promoting actions that lead to higher rewards and vice versa for actions that lead to lower rewards. These steps are repeated until a sufficiently good policy is found.

Commonly, a parameterized policy π_θ is used, where θ denotes the trainable parameters of the policy. During the policy update, these parameters are updated according to

$$\theta \leftarrow \theta + \alpha \frac{1}{M} \sum_{e_k \in E} A(o_k, a_k) \nabla_\theta \log \pi_\theta(a_k \mid o_k), \tag{1}$$

where α is the learning rate. Here, the gradients of the policy $\nabla_\theta \log \pi_\theta(a_k \mid o_k)$ are weighted by the advantage $A(o_k, a_k)$ denoting how much the action a_k is better ($A(o_k, a_k) > 0$) or worse ($A(o_k, a_k) < 0$) when making an observation o_k compared to acting according to π_θ.

The difficulty in our setting is that we are learning a policy for every vehicle in the scene simultaneously, making the environment non-stationary from an agent's point of view. To mitigate this issue, similar to our previous work [7], every agent uses a copy of the same shared policy, allowing the use of single-agent methods to solve this multi-agent task. For our experiments, we use Generalized Advantage Estimation (GAE) [12] for obtaining an estimate of the advantage $\hat{A}(o_k, a_k)$ and the Proximal Policy Optimization (PPO) [13] algorithm for learning the driver model.

2.2 Adversarial Inverse Reinforcement Learning

Our goal is to predict human driving behavior by learning and executing a policy that accurately models human driving behavior. Utilizing RL for learning the policy requires the definition of a reward function that accurately captures the incentive structure of real-world drivers. For human driving, finding such a reward function is a time-consuming and tedious task. However, it is easy to demonstrate the desired behavior in the form of a set of demonstrations recorded in real traffic.

One way to automate the process of defining a reward function is AIRL, where a surrogate reward signal is reconstructed, which explains the demonstrated behavior. Utilizing the reconstructed reward for learning the policy model yields policies that mimic expert demonstrations, given as $\mathcal{D} = \{(o_1, a_1), (o_2, a_2), \dots\}$. Specifically, AIRL combines RL with the ideas of Generative Adversarial Network (GAN) and applies them to the task of IL. The policy π_θ is still learned via RL maximizing a reward signal, but the reward signal is now approximated by a discriminator model. The discriminator model D_ϕ tries to distinguish generated samples from demonstrated ones by assigning higher scores to more realistic samples, whereas the generator model (the trainable policy π_θ) tries to fool the discriminator by generating samples matching the distribution of the demonstrated data. These adversarial objectives can be modeled with the following two-player minimax game:

$$\min_{\phi} \max_{\theta} \Bigg[- \mathop{\mathbb{E}}_{(o,a)\sim\mathcal{D}} \left[\log\left(D_\phi(o,a)\right)\right] - \mathop{\mathbb{E}}_{(o,a)\sim\pi_\theta} \left[\log\left(1 - D_\phi(o,a)\right)\right] \Bigg], \tag{2}$$

where the discriminator assigns the probability $D_\phi(o,a) \in [0,1]$ to the observation-action pair being real. By imposing a special structure on the discriminator

$$D_\phi(o,a) = \frac{\exp\left(f_\phi(o,a)\right)}{\exp\left(f_\phi(o,a)\right) + \pi\left(a \mid o\right)}, \tag{3}$$

a surrogate reward signal is reconstructed, where samples with high rewards are exponentially more likely than samples with low rewards. This structure corresponds to the odds ratio between the policy $\pi\left(a \mid o\right)$ and the exponentiated reward distribution $\exp\left(f_\phi(o,a)\right)$. The discriminator is trained by minimizing the binary cross-entropy loss in (2) and the generator via RL with the surrogate reward function

$$\begin{aligned} \tilde{r}(o,a) &= \log\left(D_\phi(o,a)\right) - \log\left(1 - D_\phi(o,a)\right) + c \\ &= f_\phi(o,a) - \log \pi\left(a \mid o\right) + c, \end{aligned} \tag{4}$$

where c is a constant reward and $-\log \pi\left(a \mid o\right)$ rewards policies for higher entropy, thus promoting exploration during training and robustness to action noise.

Modifications: In general, AIRL is a domain-agnostic method. However, when applied to learning a driver model, as proposed in [8, 10], two modifications are required: 1) In the original implementation [11], $c = 0$ is used in (4), leading to a negative expected value for the surrogate reward $\tilde{r}(o,a)$, as the discriminator is typically able to classify correctly. In our setting, this resulted in suicidal vehicles leaving the track immediately to avoid the pain of continuously receiving negative rewards. To alleviate this issue, we set $c = 5$, thus promoting survival without changing the optimal behavior with respect to the discriminator model. 2) While during RL training, a large variance in the actions drawn from the policy is crucial to gather a diverse set of experiences, it allows the

discriminator to easily detect generated samples. Therefore, to smoothen the decision boundary, during discriminator training, we add random noise to the actions executed by the experts matching the standard deviation of the learned policy.

2.3 Model Details

To ensure accurate predictions, the model must understand both its environment and the interactions between traffic participants. To this end, we employ a flexible graph-based observation and model architecture, similar to [10], which is particularly effective for handling complex road topologies. Specifically, we use agent-centric observations that capture both agents and road elements within a 30 m radius. Agents are described by their size (width and length), position, heading, velocity, and current speed limit: $\mathbf{x} = [\mathbf{x}_{\text{size}}, \mathbf{x}_{\text{pos}}, \mathbf{x}_{\text{head}}, x_{\text{vel}}, x_{\text{limit}}]^\intercal$. Road elements, such as road markings and boundaries, are represented as sets of vectors, with each vector being defined by its start and end points, a one-hot type encoding, and a binary flag indicating whether it is part of the assigned route: $\mathbf{v} = [\mathbf{v}_{\text{start}}, \mathbf{v}_{\text{end}}, \mathbf{v}_{\text{type}}, v_{\text{route}}]^\intercal$. Vectors corresponding to the same road element are grouped into polylines, which are then interconnected with the agent nodes, forming a higher-level interaction graph.

We use a similar model for both the policy and the discriminator, with the primary distinction being in the decoder. In our model, polylines of varying length are encoded through multiple layers of Message Passing (MP), as expressed by:

$$\mathbf{v} \leftarrow f_{\text{rel}}\left(g_{\text{enc}}\left(\mathbf{v}\right), f_{\text{agg}}\left(\{g_{\text{enc}}\left(\mathbf{v}_l\right)\}_{l=1}^{L}\right)\right), \tag{5}$$

with $g_{\text{enc}}(\cdot)$ being an multi-layer perceptron (MLP), $f_{\text{agg}}(\cdot)$ an element-wise max-pooling operation and f_{rel} a simple concatenation. Here, $v_l \in \{v_1, v_2, \cdots, v_L\}$ denotes one of the L vectors forming the polyline. Lastly, the polyline embeddings $\mathbf{q}$ are obtained by applying an element-wise max-pooling operation over all polyline vectors. Agent feature vectors are encoded into the same embedding space using a simple MLP returning the agent embedding $\mathbf{z} = \text{MLP}(\mathbf{x})$. To make the target agent aware of its surroundings, we use a cross-attention operation $\mathbf{z}_{\text{target}} \leftarrow \text{CA}\left(\text{Q}\colon \mathbf{z}_{\text{target}},\ \text{KV}\colon (\mathbf{z}, \mathbf{q})\right)$ [14] between the target agent embedding $\mathbf{z}_{\text{target}}$ and the combined embeddings $(\mathbf{z}, \mathbf{q})$. Finally, the decoder, implemented as another MLP, maps the interaction- and map-aware embedding $\mathbf{z}_{\text{target}}$ to the mean and standard deviation of the next acceleration and steering angle for the generator, and the expert probability $D_\phi(o, a)$ for the discriminator. As the discriminator classifies observation-action pairs, the executed action is concatenated to its decoder input.

2.4 Conditional Prediction by Simulation

The primary motivation of this work is not to demonstrate how AIRL can be used to learn behavior models that imitate human drivers, as this has already been shown in existing works, e.g., [8, 10]. Instead, we focus on demonstrating how a behavior model learned via AIRL can be used to predict the future evolution of traffic situations conditioned on the planned movement of an AV. This is achieved by embedding the behavior model within a closed-loop simulation framework. In our simulation framework, as illustrated in Figure 1, each vehicle is assigned a predefined route. When predicting trajectories online, this information must be inferred beforehand by estimating the likelihood of possible route hypotheses, as done in [15].

As illustrated in Figure 2, the predictions unfold step-wise, with each transition corresponding to a single simulation step (see Figure 1). When conditionally predicting the evolution of a traffic situation, the AV first generates a planned trajectory aligned with its driving objective (represented as the orange sequence of states). At each simulation step, the AV's position is updated according to this plan, while the movements of surrounding vehicles are predicted using the learned behavior model. Note that each predicted state depends on the previous state of all vehicles, including itself and the AV. This allows the vehicles to react to each other's predicted movement in subsequent steps, thereby fostering interaction in the prediction. Additionally, by mutually conditioning prediction and planning this way, the AV can not only assess how surrounding vehicles influence its plan but simultaneously how its planned trajectory affects their predictions.

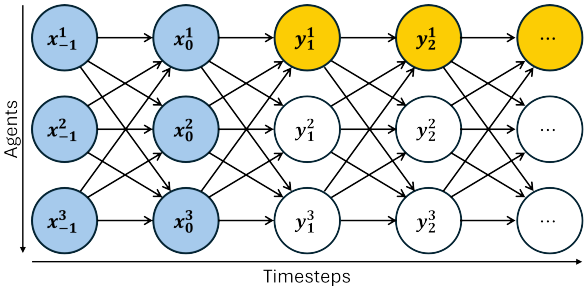

Figure 2: Conditional prediction rollout with past states (blue), AV's plan (orange), and predicted states (white).

In general, the planned movement of the AV is not required to be fixed during the prediction rollout. Instead, it can adapt dynamically in response to the predicted reactions of surrounding agents. Such a reactive planner could be realized by using a behavior model similar to the one used for prediction. However, prediction and planning have distinct requirements. While the prediction model must accurately capture human driving behaviors (e.g., tailgating or reckless driving), the AV must prioritize safe and conservative driving. Defining a safe and cooperative behavior planner is beyond the scope of this work. Therefore, we leave this for future research and use manually defined fixed paths for the AV during prediction.

3 Experiments

Dataset: For our experiments, we use the publicly available INTERACTION dataset [17], which contains recordings from 11 locations, including roundabouts, unsignalized intersections, and merging scenarios. The data consists of 36 279 vehicles, recorded at a frequency of 10 Hz over a total duration of 931 min. The amount of available data varies significantly across the locations due to differences in traffic density and recording times. Hence, to address this imbalance, we downsampled the data to balance sample sizes across locations during training. For validation and final testing, we use 20 % and 30 % of the recordings per location, respectively. The data is divided into 10-second situations, excluding those where vehicles

Parameter	**Value**
Agent feature encoder	$(8, 64, 64)$
Number of MP layers	3
MP MLP (layer 1)	$(11, 64, 32)$
MP MLP (layer 2 & 3)	$(64, 64, 32)$
Interaction module	cross-attention
Policy decoder	$(64, 64, 4)$
Discriminator decoder	$(64{+}2, 64, 1)$
Activation function	ReLU
Batch Size	1024
Optimizer	Adam [16]
Policy learning rate	$2 \cdot 10^{-4}$
Discriminator learning rate	$1 \cdot 10^{-4}$
Discount factor γ	0.95
GAE λ [12]	0.95
PPO clip range	0.2

Table 1: Training Parameters

could not be assigned a route (e.g., due to illegal turns). Modeling vulnerable road users is beyond the scope of this work; thus, the corresponding tracks are omitted.

Training: Table 1 outlines the network shapes and training parameters. For the baseline, a model is trained using BC by minimizing the negative log-likelihood of the expert actions under the predicted action distribution. The AIRL models are trained for 10 000 epochs, requiring 43 h on a single RTX8000 48GB GPU. During each epoch, training situations are randomly initialized, featuring an average of 880 simulated vehicles. The traffic situations used for validation and final testing include 1019 and 2080 vehicles, respectively, evenly distributed across the available locations. Each situation is simulated for 50 timesteps with $\Delta t = 0.2$s between consecutive timesteps. Vehicles that reach the end of their assigned route, leave the track, or collide are removed from the scene and the simulation is continued with the remaining vehicles. For final testing, we use the model with the lowest Root Mean Squared Error (RMSE) after 10 s on the validation data.

3.1 Prediction Performance

For the final evaluation, we simulate each test situation for a duration of 10 s by employing the learned behavior policies for all agents in the scene. The initial situations are taken from the ground truth test data. Subsequently, the chosen actions as well as the generated trajectories are compared against those of the corresponding ground truth vehicles.

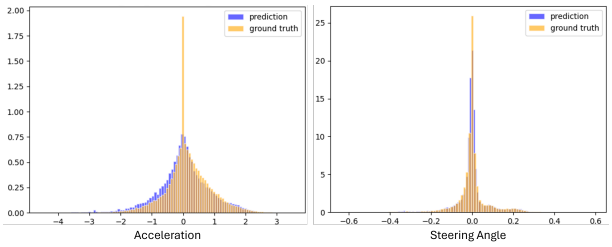

Figure 3: Normalized histograms of executed actions.

Model	RMSE	Collision	Off-Track
BC	$14.35_{\pm 0.42}$m	$10.61_{\pm 1.87}$%	$5.88_{\pm 0.85}$%
AIRL	$13.63_{\pm 0.34}$m	$0.65_{\pm 0.26}$%	$0.23_{\pm 0.07}$%

Table 2: Prediction performance after 10 s.

First, we assess the model's ability to reliably imitate human driving behavior by comparing its chosen actions and generated trajectories with those of the corresponding ground truth vehicles. Figure 3 shows the chosen actions of the learned behavior model (blue) and the corresponding ground truth vehicles (orange) in the test situations. It can be seen that the selected actions largely overlap, indicating that the policy effectively captures the correct distribution of actions. The main discrepancy can be seen in the accelerations: the predicted vehicles exhibit slightly more negative accelerations, whereas the ground truth vehicles show a peak at 0 m/s^2. This can be explained by the preprocessing of the dataset, where for vehicles at a standstill (e.g., waiting at a yield line) both their acceleration and steering angle were set to zero.

Table 2 presents the prediction performance of the models trained using BC and AIRL. Each cell represents the mean and standard deviation of seven models trained with different random seeds. After predicting for 10 s, both models demonstrate a strong performance in terms of RMSE, with the AIRL model outperforming the BC model. However, when looking at the collision and off-track rates, the AIRL models show significantly greater robustness. This improvement can be attributed to the models being trained in simulation, where the model can explore and learn from actions that are not present in the dataset. The result is a realistic behavior model that, when executed in a closed-loop simulation, generates accurate and scene-consistent predictions.

Conditional Prediction: Furthermore, we want to demonstrate how the learned behavior model can be used for making conditional predictions. As outlined in Section 2.4, we modify the planned trajectories for individual vehicles and predict the remaining vehicles. Similarly, an AV could query the prediction model with different planned trajectories for itself and select a plan based on the corresponding predictions. Multiple example scenarios are available at https://conditionalpredictionbysimulation.github.io/.

An example situation is shown in Figure 4. The intersection shown is an all-way stop intersection, where each vehicle is required to stop before entering the intersection. Priority is determined by the order of arrival, resulting in significant interactions between traffic participants. In this scenario, the pink vehicle (ID 6) intends to turn right, while the oncoming red vehicle (ID 3) intends to turn left towards the same exit. Assuming

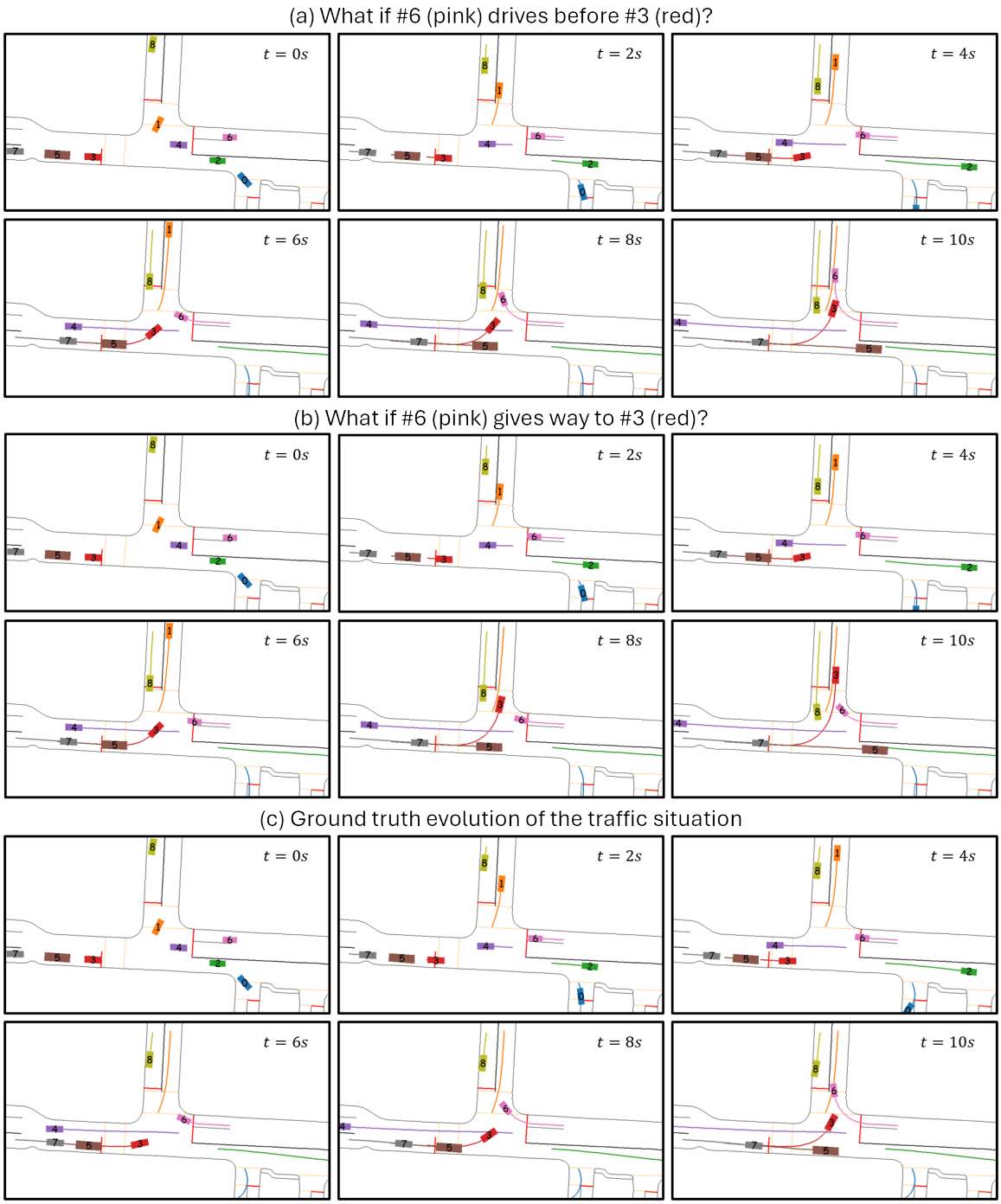

Figure 4: Demonstration of a conditional prediction.

that ID 6 is an AV, it must decide whether to proceed before ID 3 or to yield to it. To evaluate the consequences of both choices, we predict the traffic situation for each of them separately. The original prediction of the model is shown in Figure 4(a). This prediction is derived by applying the learned behavior model to all vehicles in the scene. To simulate the conditioning on the alternative future trajectory of ID 6, its executed accelerations are simply replaced by constant breaking for the first 5 s. The resulting conditional prediction is shown in Figure 4(b). For the first 2 s, both predictions evolve similarly. However, then the predictions start to diverge: Due to ID 6 entering the intersection in 4(a), ID 3 must decelerate and give way to ID 6. Conversely, in 4(b), ID 6 stops, allowing ID 3 to cross the intersection unhindered. As a result, ID 3 advances farther during the prediction, allowing ID 8 to continue driving earlier. The other vehicles' predictions remain largely unaffected, as the model has learned that they are not influenced by the changed trajectory of ID 6.

The corresponding ground truth evolution of the situation is shown in Figure 4(c). There, the vehicle with ID 6 indeed turns first, and the situation unfolds similarly to prediction 4(a). Notably, the policy exhibits driving dynamics closely resembling real-world drivers, including comparable velocities, accelerations, and also nuanced driving behavior like cutting corners. The main difference in the prediction lies in the duration vehicles stop at a stop line (depicted in red). For instance, the brown vehicle (ID 5) stops for a shorter time in the prediction than its ground truth counterpart. This can be explained by the latency between deciding and continuing to drive, as well as the subjective nature of stopping behavior: some drivers stop for several seconds, while others pause only briefly.

4 Conclusion

This work presents an autoregressive prediction model realized as a simulation framework that executes learned behavior models for all predicted vehicles. The proposed behavior model uses a flexible graph representation as input and is trained using the AIRL method. We evaluated our approach on the INTERACTION dataset, which comprises a variety of traffic scenarios with diverse traffic densities and road layouts. The results show that the model generates realistic driving behaviors, resulting in robust and scene-consistent predictions. Furthermore, we assumed individual vehicles to be AVs and explored the impact of different candidate trajectories for them on the prediction outcome, highlighting the model's ability to evaluate different planning strategies.

While this study focused on prediction, with manually generated plans for the AV, future work could integrate a more advanced planning algorithm to fully leverage the proposed prediction framework.

Acknowledgment

The authors would like to thank Matthias Dingwerth and Matthias Steiner for their support and contributions to the development of the simulation framework.

References

[1] A. Seff, B. Cera, D. Chen, M. Ng, A. Zhou, N. Nayakanti, K. S. Refaat, R. Al-Rfou, and B. Sapp, "Motionlm: Multi-agent motion forecasting as language modeling," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 8579–8590.

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

[3] R. Wagner, Ö. S. Tas, M. Steiner, F. Konstantinidis, H. Königshof, M. Klemp, C. Fernandez, and C. Stiller, "Scenemotion: From agent-centric embeddings to scene-wide forecasts," in *International Conference on Intelligent Transportation Systems (ITSC)*, 2024.

[4] P. Trautman and A. Krause, "Unfreezing the robot: Navigation in dense, interacting crowds," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 797–803.

[5] E. Tolstaya, R. Mahjourian, C. Downey, B. Vadarajan, B. Sapp, and D. Anguelov, "Identifying driver interactions via conditional behavior prediction," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 3473–3479.

[6] Y. Chen, P. Karkus, B. Ivanovic, X. Weng, and M. Pavone, "Tree-structured policy planning with learned behavior models," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 7902–7908.

[7] F. Konstantinidis, M. Sackmann, U. Hofmann, and C. Stiller, "Modeling interaction-aware driving behavior using graph-based representations and multi-agent reinforcement learning," in *2023 IEEE International Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2023.

[8] M. Sackmann, H. Bey, U. Hofmann, and J. Thielecke, "Modeling driver behavior using adversarial inverse reinforcement learning," in *2022 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2022, pp. 1683–1690.

[9] J. Spencer, S. Choudhury, A. Venkatraman, B. Ziebart, and J. A. Bagnell, "Feedback in imitation learning: The three regimes of covariate shift," *arXiv preprint arXiv:2102.02872*, 2021.

[10] F. Konstantinidis, M. Sackmann, U. Hofmann, and C. Stiller, "Graph-based adversarial imitation learning for predicting human driving behavior," in *2024 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 2024, pp. 857–864.

[11] J. Fu, K. Luo, and S. Levine, "Learning robust rewards with adversarial inverse reinforcement learning," *arXiv preprint arXiv:1710.11248*, 2017.

[12] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," *arXiv preprint arXiv:1506.02438*, 2015.

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

[14] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[15] J. Schulz, C. Hubmann, J. Löchner, and D. Burschka, "Multiple model unscented kalman filtering in dynamic bayesian networks for intention estimation and trajectory prediction," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2018, pp. 1467–1474.

[16] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[17] W. Zhan, L. Sun, D. Wang, H. Shi, A. Clausse, M. Naumann, J. Kummerle, H. Konigshof, C. Stiller, A. de La Fortelle *et al.*, "Interaction dataset: An international, adversarial and cooperative motion dataset in interactive driving scenarios with semantic maps," *arXiv preprint arXiv:1910.03088*, 2019.