# Learning a Diverse and Cooperative Policy for Predicting Roundabout Traffic Situations

Moritz Sackmann[*], Henrik Bey[*], Ulrich Hofmann[†], and Jörn Thielecke[*]

**Abstract:** Predicting other drivers' trajectories is challenging. We address the issue by introducing a method to derive a driving policy based on multi-agent reinforcement learning. For this, we let multiple vehicles interact in a roundabout scenario and reward desirable behavior. While typically, all vehicles follow the same policy, we foster diversity by assigning different preferences, e.g., cautious or sporty driving, to each vehicle during the training stage. These preferences are part of the policy network inputs as well as the reward function. This enables us to learn one single policy that can express different driving styles.

**Keywords:** Multi-Agent Reinforcement Learning, Behavior Modelling, Cooperative Behavior

## 1 Introduction

Predicting other drivers' trajectories by modeling their behavior is an important challenge in automated driving. One prominent application is cooperative behavior planning, i.e., estimating the influence of the own plan on other vehicles [1, 2].

A driver behavior model is a function that maps the current local observation of a driver to their next action. Repeated execution of the model coupled with a kinematic model leads to the prediction of the driver's trajectory. There are many possible ways to obtain driver models: Manual specification [3], learning from observed driver behavior [4–7], and reinforcement learning (RL) [8].

As the manual specification of driver behavior is a Sisyphean task, many recent works focus on learning driver behavior from observations of real-world driving. While the idea of directly learning to imitate driver behavior is appealing due to its conceptual simplicity, it suffers from accumulating errors [4–7], which we investigated in [4]. Moreover, the learned policy is inherently limited by the scope of the data used for training. It is unlikely to learn appropriate reactions to situations that are not represented in the training data, which for example leads to collisions of approximately 3% of all vehicles in the closed-loop simulation [4].

To sidestep these issues, this work builds upon the idea of RL, where the behavior model (policy) is learned by interacting with a simulated environment that can generate

[*]Institute of Information Technology, FAU Erlangen-Nürnberg, (e-mail: firstname.lastname@fau.de).
[†]Pre-Development of Automated Driving, AUDI AG, 85045 Ingolstadt (ulrich.hofmann@audi.de).

Acknowledgment: This work is a result of the research project @CITY – Automated Cars and Intelligent Traffic in the City. The project is supported by the Federal Ministry for Economic Affairs and Energy (BMWi), based on a decision taken by the German Bundestag. The author is solely responsible for the content of this publication.

an arbitrary amount of training data to improve the policy. During the simulation, each state transition is assigned a reward, and the goal of the procedure is to find a policy that maximizes the sum of rewards.

**Outline** When applying RL to learning a driving policy, a range of interesting questions arises: 1.) How can we derive a behavior model from a minimal set of assumptions, e.g., the goal to move forward and to avoid collisions? 2.) The interaction with other drivers is one key feature of behavior models. How can we learn a cooperative behavior model without any model of how other drivers behave and react to us? 3.) How can the behavior model be adapted to represent different types of drivers, e.g., more and less cautious ones?

The foundations of multi-agent reinforcement learning (MARL), questions 1 and 2, are discussed in section 2, while our approach to learning a diverse policy, question 3, is the subject of section 3. Finally, the properties of the learned policy are presented in section 4.

The core contribution of our work is the introduction of a method to train one single adjustable policy that can be used to represent different driving styles. Further contributions are the introduction of a reward function that fosters cooperative behavior as well as the application of the independent proximal policy optimization method [9] for learning the policy.

## 2 Technical Background

**Reinforcement Learning** The field of RL is concerned with solving partially observable Markov decision processes [10], which model sequential decision problems: one agent makes an observation $o \in O$ and performs an action $\alpha \in A$. As a consequence of his action, his state $s \in S$ transitions stochastically to the next state[1] $s^+ \sim T(\cdot|s, \alpha)$, according to the conditional transition density $T(\cdot|s, \alpha)$. Each transition is assigned a reward $\mathcal{R} : S \times A \to \mathbb{R}$. Each new state $s^+ \in S$ stochastically leads to a new observation $o^+ \sim \mathcal{O}(\cdot|s^+)$, forming the basis for the next decision of the agent.

In our case, each agent can select an action $\alpha = (a, \delta)$, composed of acceleration $a$ and steering angle $\delta$. The state transitions are performed through the kinematic bicycle model [11]. The observations are generated by a simulation model and describe the local environment of that agent. This procedure is visualized in Fig. 1. We assign the rewards through a manually defined function that rewards forward movement and penalizes collisions and leaving the track. The reward function is described in section 3. This function might also be learned from real data, for example proposed by [12], but this is beyond the scope of this work.

The goal of RL is to find a possibly stochastic policy $\pi(\cdot|o)$, i.e., which action to choose at which observation, that maximizes the sum of discounted rewards $g_k = \sum_{i=k}^{N} \gamma^{i-k} r_{i-k}$, starting from the initial state $k = 0$. The reward in the $i$-th step $r_i \in \mathbb{R}$ is determined by $\mathcal{R}$. We denote $g_k$ as the return. The discount factor $\gamma \in [0, 1]$, typically close to 1, ensures a preference for policies that gather high rewards fast. Moreover, it supports the convergence of $g_k$ for large numbers of steps $N$.

[1] We follow the convention of [10] to use $\sim$ as a sampling operator: $s^+ \sim T(\cdot|s, \alpha)$ indicates that $s^+$ is a sample from a random variable that is distributed according to the conditional density $T(X = x|s, \alpha)$.

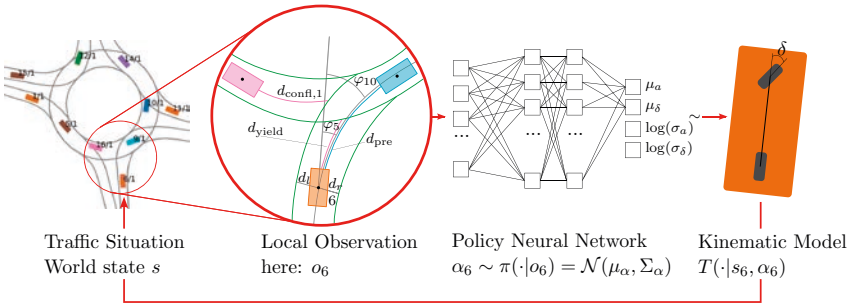



Figure 1: One step of the policy roll-out: The local observations of each vehicle are generated from the traffic situation. Observations include information on relevant vehicles, e.g., conflicting and preceding vehicles. The observation vector is the input to the policy network. An action is sampled from the action distribution. The kinematic model determines the next vehicle state. While this diagram visualizes the steps for vehicle #6, the same procedure is simultaneously executed for each vehicle in the situation.

**Proximal Policy Optimization (PPO)** PPO [13] is a policy gradient algorithm to solve RL problems. One central advantage of policy gradient methods compared to value-based RL approaches is the ability to handle continuous high-dimensional observation and action spaces [10]. For the sake of brevity, we introduce a simplified algorithm here and refer to the literature [13, 14] for a comprehensive description. At its core, two neural networks are responsible for learning the policy: The *policy network* $\pi_\theta$ with parameters $\theta$ predicts the mean and covariance $(\mu_\alpha, \Sigma_\alpha)$ of an action distribution, in our case a 2D Gaussian distribution with diagonal covariance matrix. The *value network* $v_\phi : O \to \mathbb{R}$ with parameters $\phi$ predicts the expected value of an observation, i.e., the return $g_k$, when executing the current policy starting in state $s_k$.

After randomly initializing the policy and the value network, the following steps are executed repeatedly during one training epoch:

1. Collect multiple trajectories, i.e., policy roll-outs, by randomly sampling actions from the current policy $\alpha \sim \pi_\theta(\cdot|o)$ and interacting with the environment until termination, e.g., a collision. Calculate the relative likelihood $p = \pi_\theta(\alpha|o)$ of the sampled actions. Calculate the return $g_k$, starting from each state. Store each *experience* $e_k = (o_k, \alpha_k, p_k, g_k)$ without particular order in the current set of experiences $E = \{e_1, e_2, \dots\}$.

2. Estimate the advantage $\mathcal{A}_k = g_k - v_\phi(o_k)$ of each experience in $E$ and store it along with the experience. The advantage is a positive real number if the return $g_k$ is higher than estimated by $v_\phi$; otherwise, it is negative.

3. Train the value network $v_\phi$ to predict returns $g$ based on the current observation by minimizing $\sum_{(o,g)\subset e\in E}(v_\phi(o) - g)^2/|E|$ using gradient descent with respect to the value network parameters $\phi$.

4. Train the policy network to increase the probability of selecting actions with positive

advantages $\mathcal{A}$ and decrease the probability of actions with negative advantages. To do so, maximize $\sum_{(o,\alpha,p,\mathcal{A})\subset e\in E}(\pi_\theta(\alpha|o)/p)\mathcal{A}$ using gradient ascent with respect to $\theta$.

The main principle of each training epoch can be explained as follows: Due to the stochastic sampling in step 1, some trajectories receive higher rewards than expected. Consequently, in step 2, the advantages of observations along these trajectories are positive, and the corresponding actions are reinforced in step 4, i.e., their relative likelihood $\pi_\theta(\alpha|o)$ is increased by shifting the predicted mean action $\mu_\alpha$ towards the good actions $\alpha$. Usually, $\Sigma_\alpha$ also decreases, which again increases the likelihood of $\alpha$. Conversely, $\pi_\theta(\alpha|o)$ is pushed away from actions with negative advantages. Simultaneously, step 3 creates an updated baseline estimate of the expected returns of observations required to determine the truly advantageous actions in the next epoch. As the expected returns of an observation change when the policy changes, the value network needs to be updated along with the policy.

In practice, numerous improvements increase the stability of PPO, mainly by restricting the step length during the gradient ascent of the policy parameters, similar to a trust region optimization [13], and by reducing the variance of the advantage estimate [14].

**Homogenous Multi-Agent Reinforcement Learning** Until now, we have been focusing on a single agent learning to drive through a roundabout. Clearly, the interaction with other vehicles plays a central role in driver behavior models. This leads to three requirements: First, other vehicles must be part of the simulated world, i.e., being represented in the state vector $s$. Secondly, the observation model must give hints on relevant other vehicles. And thirdly, we need a model of how other vehicles behave to simulate the interaction with them.

Back to square one: After all, the goal was to learn a behavior model, but the learning already requires a behavior model. This paradoxical situation is resolved by multi-agent reinforcement learning (MARL): In a traffic situation, MARL treats each vehicle as an agent that interacts with the environment, which includes all other agents, to maximize its own reward. Recent work [9] introduces the independent PPO (IPPO) method, which decomposes the $n$-agent MARL problem into $n$ single-agent RL problems that can be solved with PPO.

As we consider a situation consisting exclusively of cars, we previously trained a *single* policy used by all agents [8]. Thus, each agent executes the same policy, and the policy and value network are updated based on the collective experiences of all agents. This leads to homogenous behavior among agents, e.g., similar velocities in the roundabout and similar bumper-to-bumper distances in the queue before the entrance to the roundabout. In the following sections, we propose and evaluate a concept to foster heterogeneous behavior among agents to better reflect the diverse nature of human behavior in traffic.

# 3 Learning a Diverse Policy

We aim to find a safe and cooperative behavior model that can be applied independently by all agents in a roundabout traffic situation. Compared to previous works, we foster diverse behavior among agents by assigning different preferences $\rho_i = (\Delta t_{\min,i}, d_{\min,i}, \omega_{\text{acc,lat},i})$ to each agent. The preferences are reflected in the reward function: At the start of the

Table 1: Components of the observation vector $o$

| **Feature** | **Symbol** | **Unit** |
|---|---|---|
| Velocity | $v$ | m/s |
| Distance to left and right boundary | $d_l, d_r$ | m |
| Heading relative to lane in $\{0, 5, 10, 20\}$m | $\varphi_{0...20}$ | rad. |
| Road curvature in $\{0, 5, 10, 20\}$m | $c_{0...20}$ | $\mathrm{m}^{-1}$ |
| Preceding vehicle's velocity | $v_{\mathrm{pre}}$ | m/s |
| Distance to preceding vehicle | $d_{\mathrm{pre}}$ | m |
| Distance to next yield line | $d_{\mathrm{yield}}$ | m |
| Conflicting vehicle's velocity | $v_{\mathrm{confl},1}$ | m/s |
| Distance of conflicting vehicle to conflict zone | $d_{\mathrm{confl},1}$ | m |
| $2^{\mathrm{nd}}$ conflicting vehicle's velocity | $v_{\mathrm{confl},2}$ | m/s |
| Distance of $2^{\mathrm{nd}}$ conflicting vehicle to conflict zone | $d_{\mathrm{confl},2}$ | m |
| Distance to next priority merge zone | $d_{\mathrm{merge}}$ | m |
| Non-priority vehicle's velocity | $v_{\mathrm{nonpr}}$ | m/s |
| Distance of non-priority vehicle to merge zone | $d_{\mathrm{nonpr}}$ | m |
| Preference: Minimum time gap | $\Delta t_{\mathrm{min}}$ | s |
| Preference: Minimum safety distance | $d_{\mathrm{min}}$ | m |
| Preference: Lateral acceleration weight | $\omega_{\mathrm{acc,lat}}$ | – |

simulation, we assign a random minimum time gap $\Delta t_{\mathrm{min},i} \in (0.5, 1.8)s$, a minimum safety distance to the preceding vehicle $d_{\mathrm{min},i} \in (1, 5)m$ and a lateral acceleration weight $\omega_{\mathrm{acc,lat},i} \in (0.3, 1.5)$ to each agent. These values are also part of the observation vector and consequently can influence the actions selected by the policy.

To learn such a diverse policy, we formulate a MARL problem that we solve using the IPPO approach: The world state $s$ contains the road layout, the kinematic states of all agents, and their planned route, e.g., entering through the first entry and leaving at the third exit.

The observation model $\mathcal{O}(\cdot|s, i)$ determines the observation $o_i$ of the $i$-th agent. Similar to [3, 4, 7, 8], an observation $o_i$ is a vector of manually selected descriptive features of the current local environment of the $i$-th agent. In our case, it includes 24 features that describe the agent's state and relation to the road, relevant other vehicles, and the preferences $\rho_i$. An extensive list of features is given in Tab. 1. Notably, the observation vector includes information on relevant other vehicles, also indicated in the observation step of Fig. 1: The preceding vehicle, the closest two vehicles with priority (Conflicting vehicles) when approaching the roundabout, and the closest non-priority vehicle driving towards the roundabout, when being in the roundabout. The relation to these vehicles is described by their velocities and distances. If a vehicle is not available, these values assume reasonable defaults, i.e., large distances and average velocities. Apart from this representation in the feature vector, communication between agents or policies is neither possible nor needed.

The policy selects an action by mapping the observation to a distribution over the action space. We restrict the feasible actions to physically plausible limits, i.e., steering angle $\delta \in (-\pi/8, \pi/8)\,\mathrm{rad}$ and acceleration $a \in (-7, 3)\,\mathrm{m/s^2}$. The next vehicle state is determined according to the kinematic bicycle model [11].

**Reward Function** At each step, a reward is assigned to the $i$-th vehicle by the reward function

$$\begin{aligned} r(s, \alpha, i) =& (\omega_{\text{vel}} r_{\text{vel}} + \omega_{\text{crash}} r_{\text{crash}} + \omega_{\text{offtr}} r_{\text{offtr}} + \omega_{\text{acc,lon}} r_{\text{acc,lon}} \\ &+ \omega_{\text{acc,lat,}i} r_{\text{acc,lat}} + \omega_{\text{tg}} r_{\text{tg}} + \omega_{\text{dist}} r_{\text{dist}} + \omega_{\text{coop}} r_{\text{coop}})(s, \alpha, i). \end{aligned}$$

It is a linear combination of different rewards. The importance of each reward can be altered via the corresponding weight $\omega$. We set all weights to 1 for our experiments, except for $\omega_{\text{acc,lat,i}}$, as described above.

The first three rewards,

$$\begin{aligned} r_{\text{vel}}(s, \alpha, i) &= 1 - |v_i - v_{\max}|/v_{\max} && \text{clipped to [0, 1]}, v_{\max} = 9\,\text{m/s} \\ r_{\text{crash}}(s, \alpha, i) &= -100 - 20 v_i/(1\,\text{m/s}) && \text{if } i \text{ crashed}, \\ r_{\text{offtr}}(s, \alpha, i) &= -200 && \text{if } i \text{ is off track}, \end{aligned}$$

are essential for the driving functionality: $r_{\text{vel}}$ incentivizes progress along the track, whereas $r_{\text{crash}}$ and $r_{\text{offtr}}$ penalize collisions and leaving the track. By assigning negative rewards to longitudinal and lateral accelerations,

$$\begin{aligned} r_{\text{acc,lon}}(s, \alpha, i) &= -a_{\text{lon}}^2/a_{\max}^2 && \text{clipped to [-1,0]}, a_{\max} = 5\,\text{m/s}^2 \\ r_{\text{acc,lat}}(s, \alpha, i) &= -a_{\text{lat}}^2/a_{\max}^2 && \text{clipped to [-1,0]}, \end{aligned}$$

the model learns to use moderate accelerations if possible. The penalty on lateral accelerations effectively limits the velocity in the roundabout. The additional terms

$$\begin{aligned} r_{\text{tg}}(s, \alpha, i) &= -1 && \text{if } \Delta t_i < \Delta t_{\text{min},i}, \\ r_{\text{dist}}(s, \alpha, i) &= -10 && \text{if distance to preceding}(i) < d_{\text{min},i} \end{aligned}$$

facilitate learning a safe behavior because they directly penalize undesirable behavior, i.e., low time gaps and low distances. Thus, the RL agent does not need to experience a collision for learning to avoid these situations. The thresholds $d_{\text{min},i}$ and $\Delta t_{\text{min},i}$ differ per agent, fostering different driving styles.

Finally, the cooperation reward

$$r_{\text{coop}}(s, \alpha, i) = r(s, \alpha, j) \qquad \text{if } i \text{ has to give way to } j$$

assigns the reward of the closest vehicle in the roundabout $j$ to vehicle $i$. This reward is only assigned when $i$ is close ($< 2.5\,\text{m}$) to entering the roundabout. Effectively, this encourages cooperative behavior: If vehicle $j$ can drive unhindered, $i$ also benefits from the rewards of $j$. On the other hand, if the entry of $i$ forces $j$ to brake strongly, $i$ will also receive the resulting penalty.

**Improving Robustness** One major motivation for this work is to improve the robustness, i.e., collision and off-track rate, of the policies compared to our previous behavioral cloning approach [4]. While behavioral cloning does not allow for manual tweaking of the policy, e.g., through the penalties $r_{\text{tg}}$ and $r_{\text{dist}}$ in safety-critical situations, our previous RL-based approach [8] is a first step in this direction. Compared to [8], we now use the IPPO method, which is better suited for MARL than the previously used soft actor-critic (SAC),

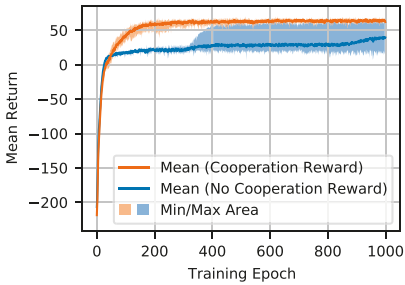



Figure 2: Minimum, mean, and maximum return $g_0$ at each training epoch, six repetitions. Higher values are better.

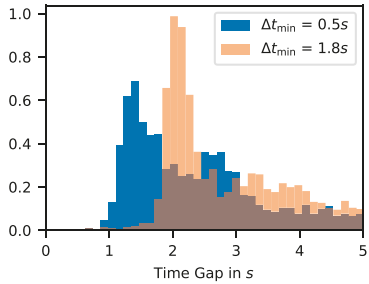



Figure 3: Histogram of time gaps of 2500 vehicles after 100 simulation steps. In these simulations, all vehicles use the same $\Delta t_{\min}$, either $0.5s$ or $1.8s$.

because it exclusively uses experiences from the current policy. SAC and other *off-policy* methods learn from an experience buffer which also contains experiences from policies of earlier training epochs. This increases the *sample efficiency*, i.e., decreases the number of experiences required to converge to a good policy. However, MARL violates the underlying assumption of stationary environment dynamics as these change along with the policy. For example, it might be feasible to enter the roundabout with low speed at early training epochs, as all other agents also drive slowly. The same behavior might be dangerous at later epochs, as the other agents have learned to drive faster; the environment dynamics have changed.

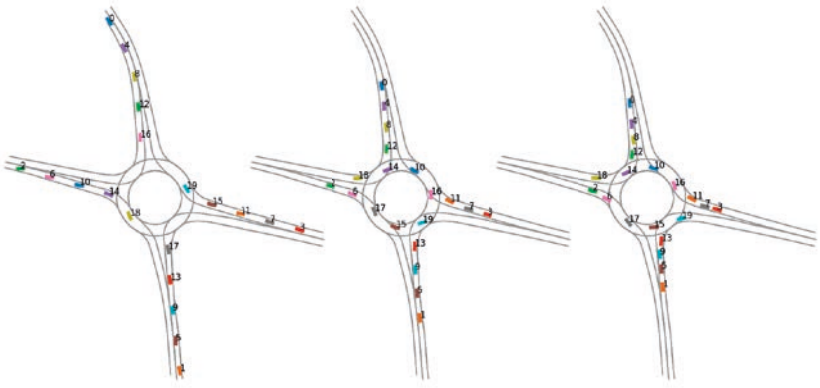
We further improve robustness by enforcing a relatively large minimum $\Sigma_\alpha$ during training. The large variance of the action distribution causes all vehicles to act more randomly, such that the policy needs to be more robust to avoid critical situations, as the randomized actions could otherwise lead to collisions.

# 4 Experiments

The training is performed simultaneously in 50 randomly initialized roundabouts with 1 to 20 vehicles. On average, we collect experiences of 650 vehicles per simulation step. Each simulation terminates after 200 steps with a simulation step size of $dt = 0.2\,\mathrm{s}$. The large simulation horizon of 40 s is required for the agents to experience the long-term effects of their actions, e.g., that waiting at the roundabout entry eventually enables one to drive in.

The training progress is visualized in Fig. 2. Within 10 episodes, most agents learn to stay on track and subsequently further refine their behavior to avoid collisions while slowly increasing their velocity. The training typically converges after approximately 200 epochs, equivalent to 1 hour on an i7-9700 CPU @ 3 GHz. To ensure reproducibility, we repeat the training six times with randomly initialized neural network parameters and random initial simulation states.

We found that the cooperation reward $r_{\mathrm{coop}}$ is essential for learning a policy that respects



(a) Initial situation (b) $d_{\min} = 5\,\mathrm{m}$ (c) $d_{\min} = 1\,\mathrm{m}$

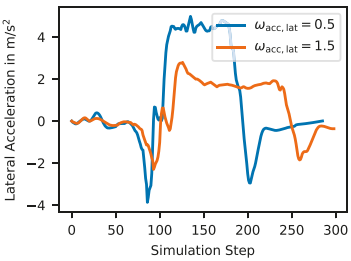
Figure 4: Effect of varying the minimum safety distance $d_{\min}$ of all vehicles, while keeping all other parameters fixed: After 15 s of simulation, the standstill distance between the vehicles at the entry lanes is clearly distinguishable.

the right of way when entering the roundabout. Agents trained without cooperation reward often squeeze into the roundabout, regardless of the impact on other vehicles. Ultimately, this behavior often causes deadlocks, as more and more vehicles enter the roundabout. This effect is reflected in the mean returns during training in Fig. 2: Policies trained without cooperation reward require considerably more training epochs to achieve returns similar to those trained with cooperation reward. Without cooperation, three out of six policies never achieve good performance due to the above-described deadlock phenomenon. Effectively, the cooperation term adds a reflection of the common interest of all agents—flowing traffic—to the otherwise purely egoistic training goal, facilitating the convergence of the IPPO method. All subsequent experiments investigate the cooperative policies.

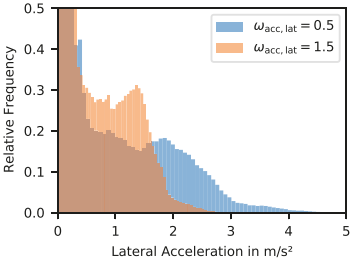
After training, the policy is executed deterministically by selecting the mean value of the action distribution instead of sampling. The preference values are randomly sampled for each vehicle. We evaluate the policy by executing it in 200 randomly generated situations for 200 steps with $dt = 0.1\,\mathrm{s}$, containing a total of 2570 vehicles, which is equivalent to 14 hours of driving. Compared to training, the step size $dt$ is halved to improve the policy performance by enabling faster reaction times. None of the six policies leaves the track in any case. The best two policies produce 0 collisions, while the worst policy produces 5 collisions (0.2%), significantly less than in our previous RL-based works (approx. 1%) [8] and our supervised learning based approach (approx. 3%) [4].

To demonstrate that our method learns an adaptive policy that can be changed at execution time via the preference vector $\rho$, we demonstrate the effect of altering the desired time gap $\Delta t_{\min}$, the minimum safety distance $d_{\min}$, and the lateral acceleration weight $\omega_{\mathrm{acc,lat}}$ in the following. To emphasize the effects, we always assign the same $\rho$ to all vehicles in a situation.

Keeping all other parameters fixed, we evaluate the effect of altering $\Delta t_{\min}$ on the




(a) Lateral accelerations experienced by one vehicle driving through an empty roundabout




(b) Histogram of lateral accelerations of 2500 vehicles during 200 time steps of simulation ($N \approx 500.000$)

Figure 5: Influence of lateral acceleration cost weight $\omega_{\text{acc,lat}}$ on the behavior of vehicles. Higher $\omega_{\text{acc,lat}}$ values penalize lateral accelerations, leading to lower overall accelerations as vehicles reduce their velocity in curves. This is shown for one single vehicle in 5a and for a large number of vehicles in 5b.

time gaps occurring after 100 simulation steps in Fig. 3. Almost no time gaps below the respective $\Delta t_{\text{min}}$ can be observed.

Next, we vary the minimum safety distance $d_{\text{min}}$. The effect is shown in Fig. 4, where the policy is applied to the same initial situation for 150 steps. The effect of switching between $d_{\text{min}} = 5\,\text{m}$ and $d_{\text{min}} = 1\,\text{m}$ is clearly visible when comparing Fig. 4b and 4c.

Finally, the effects of different lateral acceleration costs are visualized in Fig. 5. A lower cost weight $\omega_{\text{acc,lat}}$ permits higher lateral accelerations, enabling the vehicles to drive with higher velocities in the roundabout, and vice versa.

## 5 Conclusion and Outlook

Our proposed approach learns a diverse policy that can represent different driving styles through its preferences $\rho$: Minimum time gap, minimum safety distance, and lateral acceleration penalty weight. At the same time, we maintain the main advantage of training a single policy: Efficient training by leveraging the experiences of all agents to train the same policy [15]. We demonstrate the emerging properties of the learned policies. Moreover, we introduce a cooperation term that enables the otherwise purely egoistic IPPO method to reliably converge in our roundabout setting.

Future work could address the online estimation of the preference vector, given observations of real-world driver behavior. Similar ideas have been applied in [2, 16] for manually formulated parametric models. Subsequent predictions can then adapt to individual driver behavior. Additionally, uncertainty can be expressed in the low-dimensional preference space. A set of representative trajectory predictions can then be generated by rolling out the policy with different preference values.

## References


[1] C. Hubmann et al. “Automated Driving in Uncertain Environments: Planning With Interaction and Uncertain Maneuver Prediction”. In: *IEEE Transactions on Intelligent Vehicles*. Vol. 3. Mar. 2018, pp. 5–17.

[2] H. Bey et al. “Handling Prediction Model Errors in Planning for Automated Driving Using POMDPs”. In: *IEEE Intelligent Transportation Systems Conference*. Indianapolis, 2021.

[3] J. Schulz et al. “Interaction-Aware Probabilistic Behavior Prediction in Urban Environments”. In: *IEEE International Conference on Intelligent Robots and Systems*. Madrid, 2018.

[4] M. Sackmann et al. “Multi-Step Training for Predicting Roundabout Traffic Situations”. In: *IEEE Intelligent Transportation Systems Conf.* Indianapolis, 2021.

[5] L. Bergamini et al. “SimNet: Learning Reactive Self-driving Simulations from Real-world Observations”. In: *IEEE Int. Conf. on Robotics and Automation*. 2021.

[6] S. Suo et al. “TrafficSim: Learning to Simulate Realistic Multi-Agent Behaviors”. In: *IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021.

[7] J. Schulz et al. “Learning Interaction-Aware Probabilistic Driver Behavior Models from Urban Scenarios”. In: *IEEE Intelligent Vehicles Symposium*. Paris, 2019.

[8] F. Konstantinidis et al. “Parameter Sharing Reinforcement Learning for Modeling Multi-Agent Driving Behavior in Roundabout Scenarios”. In: *IEEE Intelligent Transportation Systems Conference*. Indianapolis, 2021.

[9] C. S. de Witt et al. *Is Independent Learning All You Need in the StarCraft Multi-Agent Challenge?* pre-print, arXiv: 2011.09533v1. Nov. 2020.

[10] R. S. Sutton and A. G. Barto. *Reinforcement learning: an introduction*. Second edition. Cambridge, Massachusetts: The MIT Press, 2018.

[11] J. Kong et al. “Kinematic and dynamic vehicle models for autonomous driving control design”. In: *IEEE Intelligent Vehicles Symposium*. Seoul, 2015.

[12] P. Abbeel and A. Y. Ng. “Apprenticeship Learning via Inverse Reinforcement Learning”. In: *International Conference on Machine Learning*. 2004.

[13] J. Schulman et al. *Proximal Policy Optimization Algorithms*. arXiv: 1707.06347v2. Aug. 2017.

[14] J. Schulman et al. *High-Dimensional Continuous Control Using Generalized Advantage Estimation*. arXiv:1506.02438v6. 2018.

[15] J. K. Gupta, M. Egorov, and M. Kochenderfer. “Cooperative Multi-agent Control Using Deep Reinforcement Learning”. In: *Autonomous Agents and Multiagent Systems*. Ed. by G. Sukthankar and J. A. Rodriguez-Aguilar. Cham: Springer, 2017, pp. 66–83.

[16] S. Hoermann, D. Stumper, and K. Dietmayer. “Probabilistic long-term prediction for autonomous vehicles”. In: *IEEE Intelligent Vehicles Symposium*. Los Angeles, 2017.