2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Informed Reinforcement Learning for Situation-Aware Traffic Rule Exceptions

Daniel Bogdoll[1,2,*], Jing Qin[2,*], Moritz Nekolla[1],
Ahmed Abouelazm[1], Tim Joseph[1], and J. Marius Zöllner[1,2]

***Abstract*— Reinforcement Learning is a highly active research field with promising advancements. In the field of autonomous driving, however, often very simple scenarios are being examined. Common approaches use non-interpretable control commands as the action space and unstructured reward designs, which are unsuitable for complex scenarios. In this work, we introduce Informed Reinforcement Learning, where a structured rulebook is integrated as a knowledge source. We learn trajectories and asses them with a situation-aware reward design, leading to a dynamic reward that allows the agent to learn situations that require controlled traffic rule exceptions. Our method is applicable to arbitrary RL models. We successfully demonstrate high completion rates of complex scenarios with recent model-based agents.**

I. Introduction

The rapid development of autonomous driving has led to multiple SAE Level 4 fleets available to the public in small, restricted Operational Design Domains [35]. The high flexibility necessary to progress in complex traffic scenarios is especially challenging [16]. In research, often simple scenarios are being examined, which can hardly be transferred to real-world scenarios [5]. Especially hierarchical traffic rules, which sometimes override others in specific situations, are often neglected but are a typical occurrence in everyday traffic [9]. Especially in the field of Reinforcement Learning, which has shown rapid advancements, often very simple and conflicting reward functions are being used in the domain of autonomous driving, which do not have the potential to solve such challenges [24].

Research Gap. While the development of autonomous vehicles has rapidly progressed, their ability to comprehend hierarchical traffic rules, i.e., rules that override others in certain situations, remains a challenge [9]. Current methods predominantly focus on standard traffic scenarios [2], often neglecting the nuances of exception handling. Additionally, even though Reinforcement Learning (RL) has made strides in behavior planning and control instructions for autonomous driving, the potential of RL in direct trajectory generation is not extensively researched [27].

Contribution. Our work leverages the capabilities of Informed Reinforcement Learning [39] to enhance the decision-making and adaptability of autonomous vehicles, especially in traffic rule exception scenarios. The core contributions are:

[1]Authors are with the FZI Research Center for Information Technology, Germany bogdoll@fzi.de
[2]Authors are with the Karlsruhe Institute of Technology, Germany
*These authors contributed equally

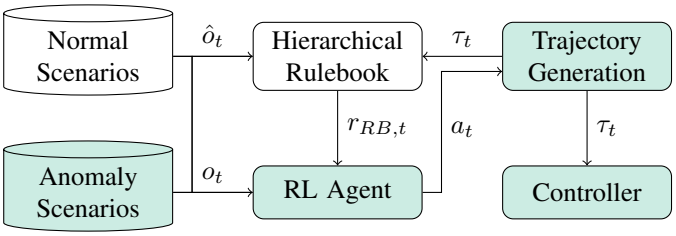


Fig. 1: Architecture of our approach. We use Curriculum Learning, where normal scenarios are used first to learn basic driving behavior. Then, anomalies are provided to learn controlled rule exceptions. Given an observation o_t, the Reinforcement Learning agent chooses an action a_t as the parametric input for generating a trajectory τ_t. The rulebook then evaluates the trajectory in the context of an abstracted environment $\hat{o}_t$ and provides the partial reward $r_{RB,t}$. Finally, a controller follows the trajectory. During evaluation, only the path in green is executed.

- Introduction of a *rulebook* in RL reward design: A structured, machine-comprehensible encoding of hierarchical traffic rules, leading to a situation-aware reward
- Our work is the first one to successfully learn *feasible trajectories* in Frenet space purely based on raw sensory RGB observations of the environment
- We provide a benchmark of 1,000 *anomaly scenarios* in the CARLA simulation environment, where the agent must be able to execute controlled rule exceptions to reach the goal

More details are available in [30], all code is on GitHub.

II. Related Work

In this section, we review related work on the application of Reinforcement Learning (RL) in motion planning for self-driving vehicles, typical traffic scenarios for training, and traffic rule formalization.

A. Reinforcement Learning for Motion Planning

Motion planning in the context of autonomous driving can be split into behavioral planning, trajectory planning, and control instructions [2].

In the context of *behavioral planning*, Fayjie et al. [12] proposed an RL-based autonomous driving strategy for urban traffic scenarios, where the discrete action space consists of *left*, *right*, and *keep going* to symbolize lane changing behaviors. Ye et al. [41] proposed a strategy for automatic

lane changing with RL based on Proximal Policy Optimization. This strategy enables a trained agent to make efficient lane-changing decisions even in dense traffic scenarios. Furthermore, numerous studies have been conducted exploring RL-based behavioral planning for autonomous vehicles, with findings demonstrating reliable performance [21], [7], [8], [20], [17].

In the context of *trajectory planning*, Feher et al. [13] learned waypoints an agent should follow. For that, they used the Deep Deterministic Policy Gradient (DDPG) algorithm. A limitation of this methodology lies in its sole focus on lateral planning. Moghadam et al. [27] proposed an RL agent that learns input parameters for a trajectory planner on the Frenet Space for highway scenarios. They use a continuous action space with processed time-series data as observation space instead of raw sensory observations. Coad et al. [10] presented a continuous RL agent in a static occupancy grid. The agent's action space is a sequence of changes in curvilinear coordinates, lateral displacement, and velocity with a fixed longitudinal step. Lu et al. [25] proposed a hierarchical reinforcement learning framework for trajectory planning given a state space composed of BEV images and lidar data. The framework consists of a high-level action responsible for choosing the direction of motion and a medium-level action sampling the vehicle's next waypoint from a fixed-size semi-circle, which can also sample off-road waypoints.

In the aerospace sector, Goddard et al. [15] proposed a framework that adds recorded flight trajectories from and after training to a database of motion primitives, which are then processed and used for subsequent classical motion planning. Williams et al. [37] propose a similar method for space-shuttle trajectories, where their RL agent learns the parameters of motion primitives, i.e., the length and defining node points of splines, which are then combined into a full trajectory. Only during training does the agent learn to avoid infeasible trajectories, as the action space does not guarantee this.

In the context of *control instructions*, where we focus on end-to-end learning, many methods leverage raw sensor inputs as the input space and directly output control commands for autonomous vehicle control, which include steering angle and acceleration [3], [22], [29], [33], [42]. However, these approaches are challenging to interpret, which makes them difficult to apply in real-world scenarios.

B. Traffic Scenarios

Most RL-based autonomous driving studies set up a specific autonomous driving environment for the vehicle. Given the relatively straightforward nature of *highway traffic* conditions, these environments present comparatively less complex scenarios for autonomous driving. Consequently, A substantial number of studies has opted to utilize highway scenarios as the benchmark for evaluating RL-based autonomous driving strategies [40], [3], [4], [20], [29], [38].

However, further approaches have focused on *urban area traffic*, encompassing elementary urban traffic, intersections, as well as dense urban traffic situations [7], [12], [41], [43].

Nonetheless, there is a lack of literature considering traffic rule exception scenarios [2], [5]. Talamini et al. [34] utilize RL to train a driving strategy when controlled traffic rule exceptions become necessary. Their approach considers behavior planning with lateral motion only and does not provide a structured approach regarding the integration of rules into the reward.

C. Formalism of Traffic Rules

Many studies have explored formalizing traffic rules into a machine-readable format. A number of methods has been used, e.g., temporal logic [26], linear temporal logic (LTL) [23], signal temporal logic (STL) [1], Isabelle theorem proving [31], and fuzzy logic [28]. However, these studies primarily concentrate on translating individual rules without considering the prioritization among different rules. Censi et al. [9] introduced a theoretical "rulebook" to structure different rules, establish a hierarchy between rules, and analyze traffic rule exception scenarios, but did not provide a framework or implementation to actually utilize it.

Consequently, there is a noticeable research gap in the development of an RL application for autonomous vehicles that not only addresses the trajectory generation in traffic rule exception scenarios, but also efficiently incorporates a structured set of traffic rules into the reward function.

III. Method

This section presents our methodology as visualized in Fig. 1. We first introduce the problem statement that outlines the challenges discussed in this paper. Next, the generation of vehicle trajectories using the Frenet Space [32] is detailed. Subsequently, the process of structuring traffic rules for computational interpretation is discussed, including using the rulebook and its integration into a reward function. Our methodology can be utilized with arbitrary RL agents.

A. Problem Statement

We focus on handling scenarios requiring controlled traffic rule exceptions, as discussed in Section I. These situations contain apparent conflicts of traffic rules, where one rule can override another. For an agent to solve the tasks, situation-awareness is necessary to apply the correct set of rules at any given time. As the system dynamics are unknown, we model the problem as a Partially Observable Markov Decision Process (POMDP), with the state space comprising high-dimensional sensory RGB data from a BEV camera and the action space consisting input parameters for the generation of a trajectory in Frenet Space.

B. Trajectory Generation

Following the approach presented by Werling et al. [36], we generate trajectories in Frenet Space. For this, terminal manifolds, symbolized as $A\{v, d, t\}$, are altered according to the current state of the vehicle:

- v: Desired velocity at the termination of the trajectory
- d: Lateral offset relative to the reference trajectory

- t: Time needed to reach the desired target state

The goal state of a trajectory in Frenet space is fully defined by v, d, t, which means that each set of parameters corresponds to a trajectory. As shown in Fig. 1, a controller is then utilized to follow the generated trajectory.

C. Situation-Aware Reward Design

In the case of traffic rule exception scenarios, some rules can override others. For this, we present a formal rulebook as part of the reward function to represent situation-specific hierarchies between different rules. Assuming no rule conflicts as the default state, the agent first needs to assess the current situation to activate dynamic rewards.

Situation Awareness. In order to assess traffic situations, an agent needs to have an understanding of traffic rules and capabilities to monitor them [6]. As shown in Fig. 1, the abstracted environment $\hat{o}_t$ provides information for this purpose, such as map data. Similar to Censi et al. [9], we introduce *Rule Realizations*:

Definition III.1 (Rule Realization)**.** Let τ_t be a sequence of states, i.e., a trajectory. A rule realization $\psi : \tau_t \to \mathbb{R}$ assigns a real number $\psi(\tau_t)$ to τ_t . This is an expression of the degree of compliance of τ_t with an underlying traffic rule.

Based on this knowledge of existing traffic rules, an agent can then set rule coefficients ρ_j depending on the current situation, e.g., diminishing the relevance of a rule if it is overwritten by another one.

Hierarchical Rulebook. Inspired by the conceptual hierarchical rulebook by Censi et al. [9], we present an implementation of a rulebook within the reward function of an RL agent. We define a rulebook as follows:

Definition III.2 (Rulebook)**.** A rulebook $\mathcal{R}$ is defined as a tuple $(\Psi, \preceq)$, where Ψ represents a finite set of rule realizations and $\preceq$ denotes a pre-order relation. Specifically, $\psi \preceq \psi'$ implies that ψ is of a lower hierarchy than ψ'.

We utilize Linear Temporal Logic (LTL) syntax for rule descriptions and their integration into the reward function. The rulebook is only activated when the situation awareness module detects a situation where a controlled rule exception becomes possible. The hierarchical structure of the rulebook is instrumental in determining which rules have precedence over others. Its hierarchical structure can be visualized as a graph, with each rule realization as a node and edges indicating priority relationships. Nodes of equal priority can be merged.

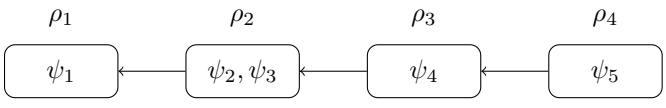


Fig. 2: Graph representation of a hierarchical rulebook Ψ with rule realizations ψ_i and hierarchy coefficients ρ_j, where j indicates the hierarchy index.

For instance, in Fig. 2, ψ_1 holds the highest hierarchy, ψ_2 and ψ_3 share the same, followed by ψ_4, and ψ_5 with the lowest. Such a representation assures the rulebook's scalability.

Linear Temporal Logic. LTL is a powerful logic language utilized in defining sequences of events or states. Its syntax contains various logical operators: negation ($\neg$), conjunction ($\wedge$), disjunction ($\vee$), and implication ($\rightarrow$), along with temporal operators: *Next* ($\mathbf{X}$), *Globally* ($\mathbf{G}$), *Finally* ($\mathbf{F}$), and *Until* ($\mathbf{U}$). To evaluate a trajectory τ_t, which can be defined as a sequence of vehicle states, we apply LTL for each rule realization ψ_i. This reward calculation can be described by a function $f(\psi_i, \tau_t)$.

Reward Design. The rulebook's hierarchical structure is incorporated into the reward function based on hierarchy coefficients $\rho_j \in [0, 1]$ for each hierarchy, as shown in Fig. 2. The coefficient scales the reward or penalty associated with a given level's rules so that higher-hierarchy rules have a higher weight in the reward. This way, the reward is dynamically adapted to the current situation. All coefficients default to 1 in standard scenarios. The realization is shown in Eq. (1).

$$r_{RB,t} = \sum_{\psi_i \in \Psi} \left(\prod_{\psi_j = \psi_i}^{\psi'} \rho_j \right) \cdot f(\psi_i, \tau_t) \cdot c_{\psi i} \qquad (1)$$

Here, Ψ are all the rules involved in the rule exception, ψ' is the rule with the highest priority in the rulebook, and ρ_j is the hierarchy coefficient of ψ_j. $f(\psi_i, \tau_t)$ is the reward value. $c_{\psi i}$ is a scaling factor for each rule that allows fine-tuning.

IV. Experiments

This section presents our experimental setup for the training and evaluation of our Reinforcement Learning agents. We present the traffic scenarios and rules we examined, the state and action spaces, our reward function, as well as the training process and parameters.

Traffic Scenarios. For our experiments, we chose a typical scenario in everyday urban traffic. Based on the German road traffic regulations, it is generally forbidden to cross a solid line. However, in certain situations, e.g., when the lane of the ego vehicle is blocked, there exists a rule exception [14], as shown in Fig. 3.

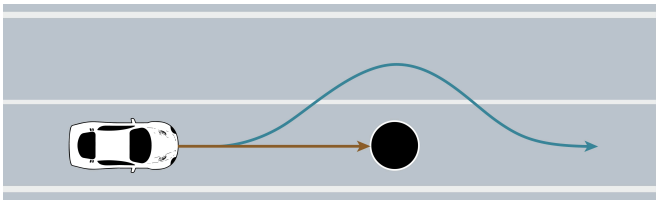


Fig. 3: Traffic scenario that shows an atypical scenario with an anomaly. In the illustrated scenario, the ego vehicle's lane is blocked, enabling it to perform a controlled rule exception. Adapted from [30].

We provide a benchmark with 1,000 such scenarios in the CARLA simulation environment [11] and also the codebase

(a) Action space for $d = 0$ (b) Action space for $d \neq 0$

Fig. 4: Dynamic action space in Frenet space. The left side shows a scenarios where the ego vehicle is in its intended lane while it is in the opposite lane on the right side.

to generate more if needed. Each scenario is defined by a reference trajectory with a length of 80 meters and an arbitrary object that blocks the lane at some point along the trajectory, as shown exemplary in Fig 6.

Agent Selection. For our experiments, we selected two established Reinforcement Learning Models. First, we decided to utilize the current state-of-the-art model-based algorithm DreamerV3, which demonstrated superior performance in a wide variety of domains [18]. Second, we utilized the model-free Rainbow algorithm [19], an improved version of the well-known Deep Q-Networks (DQN). In both cases, CNNs were used to encode the observations.

State Space. The state space comprises BEV RGB images with a resolution of $128x128$, as shown in Fig. 6. This state space inherently captures essential aspects like road geometry, the ego vehicle's position, obstacles, and the planned path.

Action Space. In order to generate trajectories, we utilize the terminal manifold $A\{v, d, t\}$ introduced earlier. Here, d helps to avoid obstacles, while v and t affect trajectory curvature and length. These translate into full trajectories in Frenet space. As we are most interested in the ability of the agent to avoid obstacles, we simplify the discrete action space. We set v and t to constants and d to specific values in dependence on the vehicle's position, as illustrated in Fig. 4. As shown in Fig. 1, a PID controller is implemented to follow the generated trajectory.

Situation-Aware Reward. For the total reward r_t, we combine two aspects, as shown in Eq. (2). The first component utilizes the current state of the ego vehicle and the second one is based on our situation-aware rulebook.

$$r_t = r_{ego,t} + r_{RB,t} \tag{2}$$

The first component r_{ego} is shown in Eq. (3). It consists of $r_{finish} = 10$ if the vehicle reaches s_{target} but not d_{target}, 60 if both are reached and 0 otherwise. Additionally, $r_{speed} = -1$ if the speed is not within $10-50km/h$, otherwise, it is 0. The trajectory length traveled in the past step is denoted by l. All values were determined by a small set of experiments.

$$r_{ego} = r_{finish} + r_{speed} * l \tag{3}$$

For the second reward component $r_{RB,t}$, we utilize a set of simplified traffic rules necessary for the designed scenarios. As shown in Table I, our agents monitor three rules, focussing on collision avoidance and adherence to road layout. Per default, the agent should adhere to all rules.

Rule	Realization	LTL Formula	j	ρ_j
Avoid collisions	ψ_1	G(no_collision)	1	1
Stay in lane	ψ_2	G(in_lane)	2	0.1
Stay on road	ψ_3	G(no_out_road)	2	0.1

TABLE I: Rule overview including rule realization IDs, LTL formulas, hierarchy levels j and coefficients ρ_j.

All rules can be monitored based on the temporal operator $\mathbf{G}$ from LTL, as introduced in Section III. As shown in Eq. (4), states breaking the rule receive a penalty of -1 per rule realization, otherwise 0. The expression $\tau_t \not\models \mathbf{G}\psi$ refers to whether the states in the generated trajectory satisfy a rule. The trajectory is considered to violate a rule if any state does not satisfy it.

$$f(\mathrm{G}\psi, \tau_t) = \begin{cases} -1, & \text{if } \tau_t \not\models \mathrm{G}\psi \\ 0, & \text{Otherwise} \end{cases} \tag{4}$$

Given the concrete set of rules and rule realizations from Table I and the rulebook reward as defined in Eq. (1), we express $r_{RB,t}$ as follows:

$$\begin{aligned} r_{RB,t} = {} & \rho_1 * r_{collision} * c_{col} \\ & + \rho_1 * \rho_2 * r_{in\_lane} * l * c_{lane} \\ & + \rho_1 * \rho_2 * r_{no\_out\_road} * l \end{aligned} \tag{5}$$

When the scenarios demand it, controlled rule exceptions become necessary to proceed. Based on ground truth, the situation awareness module of the agent activates the rulebook when it approaches an obstacle. This becomes evident in Eq (5), where all coeffients ρ_j are then set to their values as defined in Table I instead of their default value 1. Thus, when necessary, the agent can leave the road with only minor negative influence on the reward in order to perform a controlled rule exception.

Curriculum Learning. We divided our training strategy into two steps. First, the agent shall learn regular driving behavior. After that, we introduce situations that require controlled traffic rule exceptions, as shown in Fig. 1. Thus, for the first 3,000 steps, our agent is trained in a simple urban environment. Subsequently, we continue the training with scenarios that require the previously introduced traffic rule exceptions. The key training parameters are detailed in Table II.

Parameters	Value	Parameters	Value
steps	$4e4$	device	A100
action repeat	1	obs_size	[128,128]
train_scenarios	800	batch_size	32
actor_dist	one hot	batch_length	64
value_lr	3e-5	discount	0.997
model_lr	1e-4	actor_lr	3e-5

TABLE II: Training parameters

V. Evaluation

In this section, we compare and analyze the results of our experiments. We compare two Reinforcement Learning Agents and perform a variety of ablation studies in order to attribute the performance of our approach to the individual components. We show both quantitative results for the whole training process as well as qualitative demonstrations of how our most successful agent performs in scenarios that require controlled traffic rule exceptions.

A. Quantitative Evaluation

For the evaluation, we opted for two key metrics based on the vehicle's performance in avoiding obstacles, returning to the original lane, and adhering to traffic rules:

- **Arrived Distance:** The distance the vehicle was able to travel along the s-axis in the Frenet coordinate at the end of each episode, reflecting the distance traveled along the lane.
- **Finished Score:** Value ranging from 0 to 1 that quantifies the success in completing the anomaly scenario navigation task. A value of 1 denotes full success, 0.5 indicates returning to the correct longitudinal but not lateral position, and 0 is assigned otherwise.

These metrics collectively assess the agent's ability to manage scenarios which require controlled traffic rule exception.

As we extended existing RL models with our trajectory generation component and the situation-aware reward design, we performed four types of *ablation studies*:

- Baseline: Here we implemented the baseline RL agents in an end-to-end setting with a discrete control based action space, consisting of three acceleration values $(-1, 0, 1)$ and three angular velocity values $(-1, 0, 1)$.
- Trajectory: Here, we implemented only the trajectory generation without the situation-aware reward. This means that all coefficients ρ_j are set to 1 constantly.
- Rulebook: Here, we implemented only the situation-aware reward function but did not utilize our trajectory generation. In this case, Eq. (4) will only check the state of the vehicle at each timestep
- Combination: Here, we implemented both the trajectory generation and the situation-aware reward function.

We trained a total of six different models in accordance with our ablation study design, as shown in Fig. 5. Independant of the underlying RL model, the scenarios in combination with our reward were too challenging for both baseline models. Including only the rulebook had no clear influence. At this point we decided to focus on the generally more capable DreamerV3 agent. In combination with our trajectory planning module, DreamerV3 consistently outperformed other methods on both metrics **Arrived Distance** and **Finished Score**. This approach exhibited a steeper learning curve, suggesting rapid adaptation to guide the vehicle efficiently. When we additionally activated our situation-aware reward function, the performance of the DreamerV3 model improved further. This indicates that the reward function

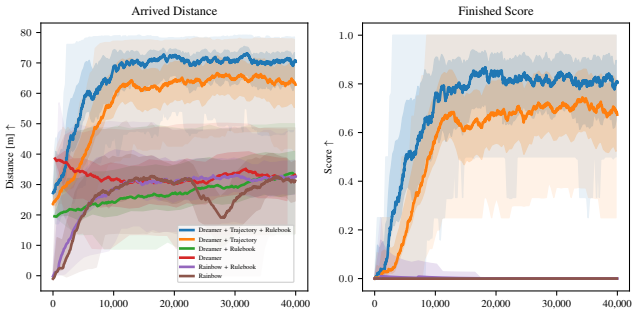


Fig. 5: Evaluation of the *arrived distance* and *finished score* during training, showing the running average, standard deviation, and 5th and 95th percentiles. We compared agents that worked with direct controls as their output or a *trajectory* and utilized either a conservative reward or our *rulebook*.

is beneficial for the agent's learning process, as the total performance stas consistently above the approach without the situation-awareness, while not achieving a 100% success rate in either of the metrics. In order to better understand failure cases, we have also performed a qualitative evaluation, which will be presented in the following section.

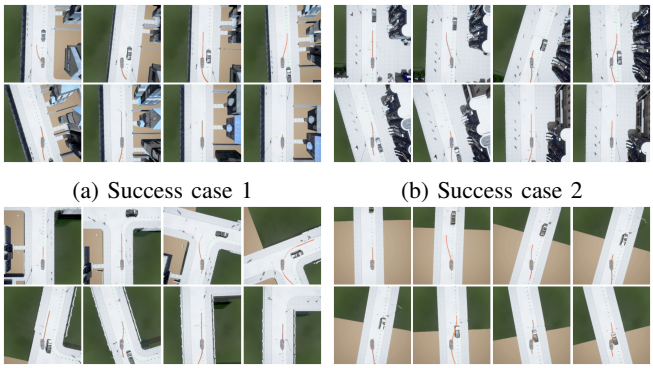
B. Qualitative Evaluation

For a better understanding of individual scenarios, we visualize both the agent's driving performance and its adherence to the defined traffic rules as shown in Fig. 6 and Fig. 7.

Fig. 6, shows raw observations o_t from the BEV camera including the planned trajectory during an episode. This visualization illustrates the vehicle's ability to change lanes to avoid obstacles and then successfully return to the original lane immediately. The corresponding trajectory and traffic rule compliance graphs, plotted in Frenet coordinates, are provided in Fig. 7. As we have seen from the quantitative results, the agent performs controlled rule exceptions successfully most of the time. However, in most failure cases, we observer too early returns to the original lane of the ego vehicle, as shown in the last scenario.

VI. Conclusion

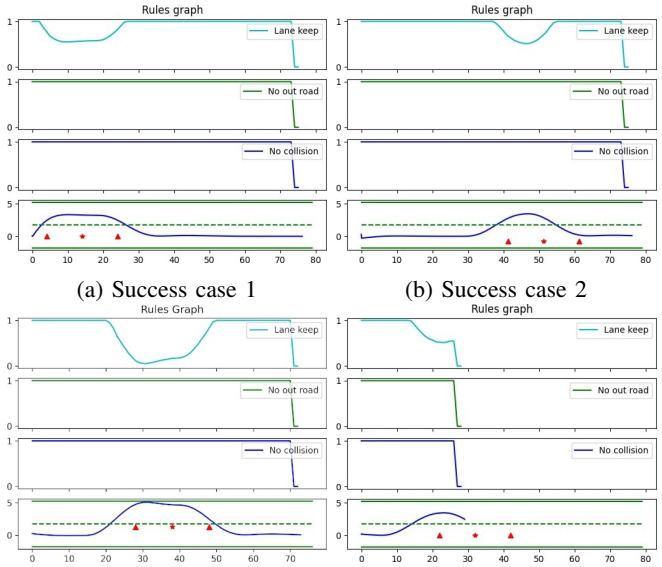
In this work, we proposed Informed Reinforcement Learning to perform controlled traffic rule exceptions in autonomous driving. For this, we extend any given RL algorithm by learning trajectories in the Frenet frame, formulating a hierarchical rulebook, and implementing a situation-aware reward design. Our proposed method achieved faster learning convergence than the baseline, displayed robust performance in traffic rule exception scenarios, and successfully incorporated real-world traffic laws into RL reward design via a rulebook. However, there are some limitations to our work. In order to focus on the performance given scenarios that require controlled traffic rule exceptions, we implemented a discrete action space. As already shown in the literature, a continuous action space for the generation



(a) Success case 1 (b) Success case 2

(c) Success case 3 (d) Failure case 1

Fig. 6: Our Reinforcement Learning agent has learned to detect situations in which allowed and controlled rule exceptions are necessary in order to progress. Trajectories that avoid obstacles on the road ahead are learned, changing to the oncoming lane temporarily, and returning to the default state as soon as possible. Reprinted from [30].



(a) Success case 1 (b) Success case 2

(c) Success case 3 (d) Failure case 1

Fig. 7: Overview of rule adherence for the scenarios shown in Fig. 6. The bottom row depicts the scenario, where ⋆ shows the position of the obstacle and ▲ visualize the area in which the rulebook was active. The top three rows show the rule adherence of the analyzed traffic rules, where 1 means full compliance and 0 violation. Reprinted from [30].

of trajectories could be implemented into our work [27]. We utilize ground-truth information for our situation-awareness, which could be replaced by an independent module. With extensive engineering effort, the number of rules and scenarios can be extended in order to examine the scalability properties of our approach.

However, our results demonstrate that extending an RL agent by both a learned trajectory and a situation-aware reward design accelerates both the training progress and the final performance in challenging scenarios.

VII. Acknowledgment

This work results partly from the project KI WISSEN (19A20020L), funded by the German Federal Ministry for Economic Affairs and Climate Action (BMWK).

References

[1] E. A. Aguilar, L. Berducci, A. Brunnbauer, R. Grosu, and D. Ničković. From STL Rulebooks to Rewards, 2021.

[2] S. Aradi. Survey of Deep Reinforcement Learning for Motion Planning of Autonomous Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[3] S. Aradi, T. Becsi, and P. Gaspar. Policy Gradient Based Reinforcement Learning Approach for Autonomous Highway Driving. In *IEEE Conference on Control Technology and Applications (CCTA)*, 2018.

[4] Z. Bai, W. Shangguan, B. Cai, and L. Chai. Deep reinforcement learning based high-level driving behavior decision-making model in heterogeneous traffic. In *Chinese Control Conference (CCC)*, 2019.

[5] D. Bogdoll, S. Guneshka, and J. M. Zöllner. One Ontology to Rule Them All: Corner Case Scenarios for Autonomous Driving. In *European Conference on Computer Vision (ECCV) Workshop*, 2022.

[6] D. Bogdoll, M. Nekolla, T. Joseph, and J. M. Zöllner. Quantification of Actual Road User Behavior on the Basis of Given Traffic Rules. In *IEEE Intelligent Vehicles Symposium (IV)*, 2022.

[7] M. Bouton, A. Nakhaei, D. Isele, K. Fujimura, and M. J. Kochenderfer. Reinforcement Learning with Iterative Reasoning for Merging in Dense Traffic. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2020.

[8] Z. Cao, D. Yang, S. Xu, H. Peng, B. Li, S. Feng, and D. Zhao. Highway Exiting Planner for Automated Vehicles Using Reinforcement Learning. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[9] A. Censi, K. Slutsky, T. Wongpiromsarn, D. Yershov, S. Pendleton, J. Fu, and E. Frazzoli. Liability, ethics, and culture-aware behavior specification using rulebooks. In *International Conference on Robotics and Automation (ICRA)*, 2019.

[10] J. Coad, Z. Qiao, and J. M. Dolan. Safe trajectory planning using reinforcement learning for self driving. *arXiv:2011.04702*, 2020.

[11] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. CARLA: An open urban driving simulator. In *Conference on Robot Learning (CoRL)*, 2017.

[12] A. R. Fayjie, S. Hossain, D. Oualid, and D.-J. Lee. Driverless Car: Autonomous Driving Using Deep Reinforcement Learning in Urban Environment. In *International Conference on Ubiquitous Robots (UR)*, 2018.

[13] Á. Fehér, S. Aradi, F. Hegedüs, T. Bécsi, and P. Gáspár. Hybrid DDPG approach for vehicle motion planning. In *International Conference on Informatics in Control, Automation and Robotics (ICINCO)*, 2019.

[14] Führerscheine.de. Achtung durchgezogene Linie – Überholen Verboten! `https://www.fuehrerscheine.de/bussgeldkatalog/durchgezogene-linie/`, 2023. Accessed: 2023-09-15.

[15] Z. C. Goddard, K. Wardlaw, K. Williams, J. Parish, and A. Mazumdar. Utilizing Reinforcement Learning to Continuously Improve a Primitive-Based Motion Planner. *Journal of Aerospace Information Systems*, 19, 2022.

[16] C. Gulino, J. Fu, W. Luo, G. Tucker, E. Bronstein, Y. Lu, J. Harb, X. Pan, Y. Wang, X. Chen, J. D. Co-Reyes, R. Agarwal, R. Roelofs, Y. Lu, N. Montali, P. Mougin, Z. Yang, B. White, A. Faust, R. McAllister, D. Anguelov, and B. Sapp. Waymax: An accelerated, data-driven simulator for large-scale autonomous driving research. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, 2023.

[17] J. Guo, S. Cheng, and Y. Liu. Merging and Diverging Impact on Mixed Traffic of Regular and Autonomous Vehicles. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[18] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap. Mastering Diverse Domains through World Models. *arXiv:2301.04104*, 2023.

[19] M. Hessel, J. Modayil, H. van Hasselt, T. Schaul, G. Ostrovski, W. Dabney, D. Horgan, B. Piot, M. Azar, and D. Silver. Rainbow:

Combining Improvements in Deep Reinforcement Learning. *AAAI Conference on Artificial Intelligence*, 2018.

[20] C.-J. Hoel, K. Wolff, and L. Laine. Automated Speed and Lane Change Decision Making using Deep Reinforcement Learning. In *International Conference on Intelligent Transportation Systems (ITSC)*, 2018.

[21] Y. Hu, A. Nakhaei, M. Tomizuka, and K. Fujimura. Interaction-aware decision making with adaptive strategies under merging scenarios. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[22] M. Jaritz, R. de Charette, M. Toromanoff, E. Perot, and F. Nashashibi. End-to-End Race Driving with Deep Reinforcement Learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2018.

[23] M. Kloetzer and C. Belta. A Fully Automated Framework for Control of Linear Systems from Temporal Logic Specifications. *IEEE Transactions on Automatic Control*, 2008.

[24] W. B. Knox, A. Allievi, H. Banzhaf, F. Schmitt, and P. Stone. Reward (mis)design for autonomous driving. *Artificial Intelligence*, 316, 2023.

[25] X. Lu, F. X. Fan, and T. Wang. Action and trajectory planning for urban autonomous driving with hierarchical reinforcement learning. In *International Conference on Machine Learning (ICML) Workshop*, 2023.

[26] S. Maierhofer, A.-K. Rettinger, E. C. Mayer, and M. Althoff. Formalization of Interstate Traffic Rules in Temporal Logic. In *IEEE Intelligent Vehicles Symposium (IV)*, 2020.

[27] M. Moghadam, A. Alizadeh, E. Tekin, and G. H. Elkaim. A deep reinforcement learning approach for long-term short-term planning on frenet frame. In *IEEE International Conference on Automation Science and Engineering (CASE)*, 2021.

[28] J. Morse, D. Araiza-Illan, J. Lawry, A. Richards, and K. Eder. A Fuzzy Approach to Qualification in Design Exploration for Autonomous Robots and Systems. In *IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2017.

[29] S. Nageshrao, H. E. Tseng, and D. Filev. Autonomous highway driving using deep reinforcement learning. In *IEEE International Conference on Systems, Man and Cybernetics (SMC)*, 2019.

[30] J. Qin. Reinforcement Learning for Controlled Traffic Rule Exceptions. Master thesis, Karlsruhe Institute of Technology (KIT), 2023.

[31] A. Rizaldi, J. Keinholz, M. Huber, J. Feldle, F. Immler, M. Althoff, E. Hilgendorf, and T. Nipkow. Formalising and Monitoring Traffic Rules for Autonomous Vehicles in Isabelle/HOL. In *Integrated Formal Methods*, 2017.

[32] Robotics Knowledgebase. Trajectory Planning in the Frenet Space. `https://roboticsknowledgebase.com/wiki/planning/frenet-frame-planning/`, 2022. Accessed: 2024-02-06.

[33] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani. End-to-End Deep Reinforcement Learning for Lane Keeping Assist. In *Conference on Neural Information Processing Systems (NeurIPS) Workshop*, 2016.

[34] J. Talamini, A. Bartoli, A. De Lorenzo, and E. Medvet. On the Impact of the Rules on Autonomous Drive Learning. *Applied Sciences*, 2020.

[35] TechCrunch. Cruise and Waymo win robotaxi expansions in San Francisco . `https://techcrunch.com/2023/08/10/cruise-and-waymo-win-robotaxi-expansions-in-san-francisco`, 2023. Accessed: 2023-09-15.

[36] M. Werling, S. Kammel, J. Ziegler, and L. Gröll. Optimal trajectories for time-critical street scenarios using discretized terminal manifolds. *The International Journal of Robotics Research*, 2012.

[37] K. R. Williams, R. Schlossman, D. Whitten, J. Ingram, S. Musuvathy, J. Pagan, K. A. Williams, S. Green, A. Patel, A. Mazumdar, and J. Parish. Trajectory Planning with Deep Reinforcement Learning in High-Level Action Spaces. *IEEE Transactions on Aerospace and Electronic Systems*, 59, 2023.

[38] P. Wolf, K. Kurzer, T. Wingert, F. Kuhnt, and J. M. Zöllner. Adaptive Behavior Generation for Autonomous Driving using Deep Reinforcement Learning with Compact Semantic States. In *IEEE Intelligent Vehicles Symposium (IV)*, 2018.

[39] J. Wörmann, D. Bogdoll, E. Bührle, H. Chen, E. F. Chuo, K. Cvejoski, L. van Elst, P. Gottschall, S. Griesche, C. Hellert, C. Hesels, S. Houben, T. Joseph, N. Keil, J. Kelsch, H. Königshof, E. Kraft, L. Kreuser, K. Krone, T. Latka, D. Mattern, S. Matthes, M. Munir, M. Nekolla, A. Paschke, M. A. Pintz, T. Qiu, F. Qureishi, S. T. R. Rizvi, J. Reichardt, L. von Rueden, S. Rudolph, A. Sagel, T. Scholl, G. Schunk, H. Shen, H. Stapelbroek, V. Stehr, G. Srinivas, A. T. Tran, A. Vivekanandan, Y. Wang, F. Wasserrab, T. Werner, C. Wirth, and S. Zwicklbauer. Knowledge Augmented Machine Learning with Applications in Autonomous Driving: A Survey. *arXiv:2205.04712*, 2023.

[40] X. Xu, L. Zuo, X. Li, L. Qian, J. Ren, and Z. Sun. A Reinforcement Learning Approach to Autonomous Decision Making of Intelligent Vehicles on Highways. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2020.

[41] F. Ye, X. Cheng, P. Wang, C.-Y. Chan, and J. Zhang. Automated lane change strategy using proximal policy optimization-based deep reinforcement learning. In *IEEE Intelligent Vehicles Symposium (IV)*, 2020.

[42] L. Yu, X. Shao, Y. Wei, and K. Zhou. Intelligent Land-Vehicle Model Transfer Trajectory Planning Method Based on Deep Reinforcement Learning. *Sensors*, 2018.

[43] C. Zhang, K. Kacem, G. Hinz, and A. Knoll. Safe and Rule-Aware Deep Reinforcement Learning for Autonomous Driving at Intersections. In *IEEE International Conference on Intelligent Transportation Systems (ITSC)*, 2022.