Learning to Obey Traffic Rules using Constrained Policy Optimization

Xiao Wang*, Christoph Pillmayer*, and Matthias Althoff

Abstract— **When planning motions for autonomous vehicles, traffic rules must be obeyed to ensure safety and reject liability claims. However, present solutions do not scale well with the complexity of traffic rules or even consider them. To solve this problem, we propose a scalable approach based on constrained policy optimization to improve traffic rule compliance of motion planners for autonomous vehicles. Our approach encodes traffic rules as constraints of the optimization problem and does not require an explicit model of the environment. We evaluate our approach using the highway dataset highD and show that agents trained using our method can effectively learn to reach a goal region while following traffic rules.**

I. Introduction

Motion planning for autonomous vehicles is challenging because traffic conditions are very dynamic and unpredictable. Fortunately, traffic rules help traffic participants to better handle the uncertainty and dynamics of traffic. Obviously, not only conventional traffic participants, but also autonomous vehicles must follow traffic rules; this is ideally achieved by formalizing traffic rules in a machine-readable way [1]–[3].

Due to the complexity of complying with traffic rules, we employ constrained reinforcement learning (RL), which explicitly considers constraints originating from traffic rules. Constrained RL is particularly beneficial for learning to obey traffic rules since it does not require an explicit model of the environment.

A. Related Work

Subsequently, we review the literature on traffic-rule-compliant motion planning and constrained RL.

1) Traffic rule compliance: Various temporal logic can be employed to formalize traffic rules in an unambiguous and machine-interpretable way. For instance, linear temporal logic (LTL) is used in [2], [4]–[6]. However, time is not explicitly considered in LTL. To add temporal constraints, Maierhofer et al. [1], [3] use metric temporal logic (MTL) to formalize interstate and intersection traffic rules, which uses Boolean values to represent rule satisfaction. The authors of [7], [8] use signal temporal logic (STL), which extends MTL to continuous time. Furthermore, MTL and STL are equipped with quantitative semantics, indicating the degree of satisfaction of a trajectory with respect to a specification, and are equivalent in our setting.

Different methods have been proposed to synthesize plans that satisfy temporal logic specifications. Automata-based approaches convert the temporal logic specifications to finite automata, incorporate the corresponding system dynamics, and generate rule-compliant trajectories using receding horizon approaches [9], [10] or sampling-based approaches [11], [12]. However, the resulting automata's size grows exponentially with the specification's length [13], limiting their application to high-dimensional problems. Instead, mixed-integer programming is employed in [13] to encode the temporal logic specifications as mixed-integer linear constraints. However, mixed-integer programming struggles to scale to complex specifications and long time horizons. On the other hand, deep RL approaches have shown better scalability for high-dimensional systems [14], [15]. However, no direct comparison with the previously mentioned methods has yet been conducted.

2) Constrained RL: Constrained reinforcement learning methods are classified into primal and primal-dual methods in [16]. Primal methods [16]–[18] satisfy the constraints by considering them in the update process of a policy. Chow et al. [17] project the policy parameter to a safe set formed from constraints defined using Lyapunov functions. Liu et al. [18] employ an interior-point approach to augment the policy optimization objective with logarithmic barrier functions. Xu et al. [16] propose to update the reward and cost objective alternately, without the demand for a dual variable. However, primal methods have been much less popular compared to primal-dual methods, since one cannot tune the trade-off between the primal objective and constraints due to the absence of dual variables. Primal-dual methods augment the optimization objective with the given constraints weighted by dual variables. Therefore, the original constrained problem becomes unconstrained. We further divide primal-dual methods into two sub-categories: trust-region and Lagrangian methods. Trust-region approaches [19], [20] approximate constraints linearly within the trust region. Lagrangian methods [21]–[23] use Lagrange multipliers to tune the trade-off between performance and constraint satisfaction.

Among these researches, the closest to ours is [23], which also combines distributional RL and constrained RL. However, the authors of [23] have not considered temporal logic constraints and did not apply their approach to autonomous vehicles. Additionally, they integrate their approach with an off-policy RL algorithm, which we demonstrate in Section IV is less stable for our problem compared to our method. To summarize, a solution to the problem of learning traffic rule

* The first two authors have contributed equally to this research.

This research was supported by the German Federal Ministry for Digital and Transport (BMDV) within the project *Cooperative Autonomous Driving with Safety Guarantees* (KoSi) and the German Research Foundation (DFG) grant AL 1185/7-1.

All authors are with the Department of Informatics, Technical University of Munich, 85748 Garching, Germany.
xiao.wang@tum.de, christoph.pillmayer@tum.de, althoff@in.tum.de

compliance without an explicit model does not yet exist.

B. Contributions

We present the first study on learning traffic rule compliance for autonomous vehicles using a distributionally constrained policy optimization approach. Particularly, our contributions are threefold:

- We propose the first approach to integrating distributionally primal-dual-constrained RL with a policy gradient approach;
- We develop a traffic-rule-compliant motion planner for autonomous vehicles based on constrained reinforcement learning;
- We demonstrate the effectiveness of our method using the highD dataset [24].

The remainder of this paper is organized as follows: Section II introduces the required preliminaries of our method. In Section III, we present our approach to integrating distributionally constrained RL with policy gradient methods, which is then demonstrated in Section IV using the highD dataset. Finally, Section V concludes our research.

II. PRELIMINARIES

In this section, we introduce the basic concepts of constrained Markov decision processes and distributional RL, notations of cost/risk measures, and the syntax of MTL.

A. Constrained Markov Decision Process

We model our problem as a constrained Markov decision process (MDP) [25] defined as a tuple $(S, A, p, r, c, c_{\text{limit}}, \gamma)$: a state space S represents a set of continuous states s, an action space A represents a set of actions a, a transition function $p(s'|s,a)$ is the distribution describing the probability of reaching state s' from state s taking action a, r represents the immediate reward, c represents the immediate cost, c_{limit} is a given threshold for the cumulative cost, and $\gamma \in [0,1]$ represents the discount factor. Note that a constrained MDP is a special form of an MDP, where the policy maximizes the cumulative reward subject to constraining the cumulative cost to stay below a user-defined limit c_{limit} [25]. In the rest of this paper, we focus on definitions of costs and omit definitions for rewards, which can be obtained by replacing c with r. Without loss of generality, we use subscripts $*_0$ and $*_t$ to denote a variable at time step 0 and t, respectively. The future discounted cost, called cost *return*, is defined as:

$$C = \sum_{t=0}^{\infty} \gamma^t c_t, \tag{1}$$

and the cost value of the state s following policy π is:

$$V_c^\pi(s) = \mathbb{E}[C|s_0 = s, \pi], \tag{2}$$

where $\mathbb{E}[\cdot|\pi]$ denotes the expected value over all future states following policy π.

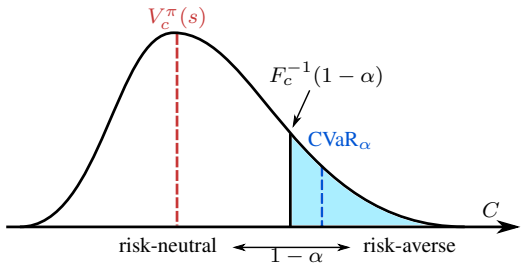

Fig. 1. Conditional value-at-risk (CVaR) of the distribution of C.

B. Distributional RL

Compared with traditional RL, which learns the approximate expectation of returns, distributional RL learns the approximate distribution of returns instead. Bellemare et al. have shown the advantages of learning distributions instead of expectations in [26]. We approximate the distribution of the state-dependent return (1) following policy π with a Gaussian distribution as proposed in [23], [27]:

$$C^\pi(s) \sim \mathcal{N}(V_c^\pi(s), \Upsilon_c^\pi(s)), \tag{3}$$

where the variance $\Upsilon_c^\pi(s)$ is calculated by [28, Eq. (2.35)]:

$$\Upsilon_c^\pi(s) = \mathbb{E}[C^2|s_0 = s, \pi] - V_c^\pi(s)^2. \tag{4}$$

C. Conditional Value-at-Risk

A policy that optimizes the expectation of return (1) might have high variance, therefore frequently generating actions that result in more constraint violations. Risk-sensitive RL tackles this challenge by considering various risk-related metrics of the distributions of return instead. Conditional Value-at-Risk (CVaR) is a popular metric to describe risk representing the percentiles of the distribution of return. For interested readers, Sarykalin et al. demonstrate the relation between chance constraints and percentiles of a distribution in [29, Eq. (14)]. The benefit of CVaR is that we can optimize the policies toward various levels of risk aversion through a parameter $\alpha \in [0,1]$, called risk level, as demonstrated in Fig. 1. Let p_c denote a general distribution of cost return, then CVaR of p_c with risk level α is [23, Definition 1]:

$$\text{CVaR}_\alpha := \mathbb{E}[C|C \geq F_c^{-1}(1-\alpha)], \tag{5}$$

where F_c represents the cumulative distribution function of p_c. CVaR of a Gaussian distribution (3) can be calculated as [30, Proposition 1]:

$$V_{\text{CVaR}}^\pi(s) = V_c^\pi(s) + \alpha^{-1}\phi(\Phi^{-1}(\alpha))\sqrt{\Upsilon_c^\pi(s)}, \tag{6}$$

where $\phi(\cdot)$ and $\Phi(\cdot)$ are the standard normal distribution's the probability density function and cumulative distribution function, respectively. Note that we omit superscript $*^\pi$ in the rest of this paper to enhance readability.

D. Past-Time MTL

The traffic rules that our agent aims to learn are formalized in the past fragment of metric temporal logic (past-MTL) [31], which defines finite traces over Boolean signals using bounded-time operators. Let p denote an atomic proposition that can be either $true$ or $false$; $\neg$ and $\vee$ denote

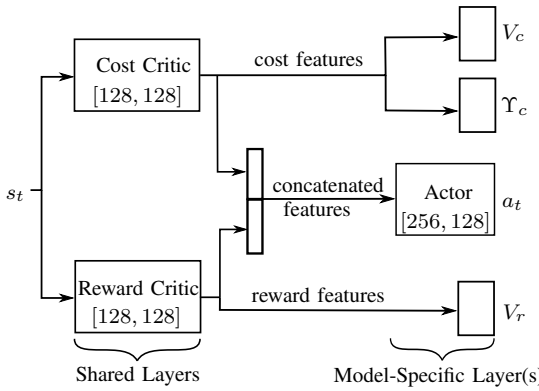

Fig. 2. An overview of our model architecture. The number of neurons in each layer is marked in the respective models. We omit the numbers for the last layer of cost expectation, cost variance, and reward since each of them only has one neuron.

Boolean operators not and or; $\boldsymbol{G}$ denotes the temporal operator $globally$, indicating that formula φ must hold at all times; $\boldsymbol{P}$ and $\boldsymbol{O_I}$ denote the bounded past temporal operator $previously$ and $once$, indicating that φ must hold at the previous time step and at least once during a past time horizon respectively, where the subscript $\boldsymbol{I} \subseteq (0, \infty)$ describes time constraints relative to the current time. With this, the syntax of past-MTL is defined as follows:

$$\varphi := \neg\varphi \vee \varphi \,|\, p \,|\boldsymbol{G}\varphi|\, \boldsymbol{P}\,\varphi|\, \boldsymbol{O_I}\,\varphi. \tag{7}$$

For a formal definition of the semantics of past-MTL, we refer interested readers to [32].

III. Approach

We propose a novel approach to integrate distributional RL, risk-averse RL, and constrained RL, which is then used to train an agent to obey traffic rules.

The architecture of our model is shown in Fig. 2 and the overall algorithm in Alg. 1. The reward value and cost value are estimated using two separate neural networks, namely, the *cost critic* and the *reward critic*. This separate structure decouples the features of cost and reward, therefore simplifying the training. We stack the features outputted by the cost critic and reward critic (i.e., *concatenated features* in Fig. 2) and feed them into the layers of the policy network, also called the *actor*, since the actor requires both features to learn to maximize the reward value while maintaining the cost value below a given threshold c_{limit}.

A. Distributional Cost Critic

As shown in Fig. 2, we used a shared network to estimate the cost return's expected value and variance, i.e., $V_c(s)$ and $\Upsilon_c(s)$, since they might have common features (i.e., *cost features* in Fig. 2). Tang et al. derived the variance of reward return in [27], which depends on the state and action, i.e., their critic approximates the Q-value, whereas our critic approximates the state value. We make this choice because the advantage function [33, Eq. (3)] can be calculated using the state value directly, as introduced in Sec. III-B. The variance of the cost return is

$$\begin{aligned}\Upsilon_c(s) = {} & c(s)^2 - V_c(s)^2 \\ & + 2\gamma c(s) \sum_a \pi(a|s) \sum_{s'} p(s'|s,a) V_c(s') \\ & + \gamma^2 \sum_a \pi(a|s) \sum_{s'} p(s'|s,a) \Upsilon_c(s') \\ & + \gamma^2 \sum_a \pi(a|s) \sum_{s'} p(s'|s,a) V_c(s')^2.\end{aligned} \tag{8}$$

A detailed proof is given in Appendix-A. To obtain the distributional Bellman update of $V_c(s)$ and $\Upsilon_c(s)$, we replace the expected future values with the estimated values for the next state, i.e., we utilize the one-step estimation [34, Chapter 6.1] for an arbitrary variable X:

$$\sum_a \pi(a|s) \sum_{s'} p(s'|s,a) X(s') \approx X(s_{t+1}). \tag{9}$$

We expand (2) as [34, Eq.(3.14)]:

$$V_c(s) = \sum_a \pi(a|s) \sum_{s'} p(s'|s,a)[c_t + \gamma V_c(s')]. \tag{10}$$

Then the Bellman updated expectation and variance are:

$$\begin{aligned}\overline{V}_c(s_t) &\overset{(9)\text{in}(10)}{=} c_t + \gamma V_c(s_{t+1}) \\ \overline{\Upsilon}_c(s_t) &\overset{(9)\text{in}(8)}{=} c_t^2 - V_c(s_t)^2 + 2\gamma c_t V_c(s_{t+1}) \\ &\quad + \gamma^2 \Upsilon_c(s_{t+1}) + \gamma^2 V_c(s_{t+1})^2.\end{aligned} \tag{11}$$

To train our distributional cost critic, we require a loss function to measure the distance between the current cost distribution $\mathcal{N}(V_c(s), \Upsilon_c(s))$ and the Bellman updated distribution $\mathcal{N}(\overline{V}_c(s), \overline{\Upsilon}_c(s))$. We choose the 2-Wasserstein distance, as proposed in [23], [27], since the Bellman update using the 2-Wasserstein distance is a contraction operator [26, Lemma 3]. The loss function is

$$\begin{aligned}L_{V_c} &= \|\overline{V}_c(s_t) - V_c(s_t)\|_2^2 \\ L_{\Upsilon_c} &= \overline{\Upsilon}_c(s_t) + \Upsilon_c(s_t) - 2\sqrt{\overline{\Upsilon}_c(s_t)\,\Upsilon_c(s_t)}.\end{aligned} \tag{12}$$

In contrast to the cost critic, the reward critic is not chosen to be distributional since we aim to constrain the worst-case behavior with respect to the costs while optimizing the average behavior in terms of performance. Thus, we use the temporal difference error as its loss [34, Chapter 6.1]:

$$L_{V_r} = \|r_t + \gamma V_r(s_{t+1}) - V_r(s_t)\|_2^2. \tag{13}$$

Note that we weigh the loss of the reward and cost value equally, but one could perform a hyperparameter search to find a better choice.

B. Worst-case PPO

On-policy approaches tend to be more stable and less sensitive to hyperparameters compared to off-policy approaches [35]. Therefore, we build our approach on top of the state-of-art on-policy approach—proximal policy optimization (PPO) [36]. We denote the actor in PPO as π^θ with a parameter vector θ. The loss of the actor comprises two parts: reward-dependent loss L_r^θ and cost-dependent loss L_c^θ. We first

calculate the distributional cost's advantage function based on CVaR. By replacing the value function with V_{CVaR} in the general advantage estimator (GAE) [33, Eq. (16)], we obtain

$$\hat{A}_{\mathrm{CVaR},t} = \delta_t + (\gamma\beta)\delta_{t+1} + \cdots + \cdots + (\gamma\beta)^{T-t+1}\delta_{T-1},$$
$$\text{with} \quad \delta_t = c_t + \gamma V_{\mathrm{CVaR}}(s_{t+1}) - V_{\mathrm{CVaR}}(s_t), \tag{14}$$

where $\beta \in [0, 1]$ denotes a discount factor of the advantage estimator and makes a compromise between variance and bias. The two loss functions of the actor are [36, Eq. (7)]

$$L_c^\theta = \hat{\mathbb{E}}_t[\max(k_t(\theta)\hat{A}_{\mathrm{CVaR},t}, k_{\mathrm{clip},t}(\theta)\hat{A}_{\mathrm{CVaR},t})]$$
$$L_r^\theta = -\hat{\mathbb{E}}_t[\min(k_t(\theta)\hat{A}_t, k_{\mathrm{clip},t}(\theta)\hat{A}_t)], \tag{15}$$

where the probability ratio is defined by $k_t(\theta) = \pi_\theta(a_t|s_t)/\pi_{\theta_{\mathrm{old}}}(a_t|s_t)$, $\pi_{\theta_{\mathrm{old}}}$ denotes the old policy before the update, $k_{\mathrm{clip},t}(\theta) = \mathrm{clip}(k_t(\theta), 1-\epsilon, 1+\epsilon)$, ϵ is a hyperparameter, defining the trust region's size, clip is an operator for limiting the operand in a given range, and $\hat{\mathbb{E}}_t[\cdots]$ denotes the empirical average over a finite batch of samples.

C. PID Lagrangian PPO

To combine the two loss functions of the actor (15), we choose Lagrangian optimization since it introduces a Lagrange multiplier that enables us to avoid tuning the trade-off between performance and constraint satisfaction manually. The combined loss is defined as:

$$\min_\theta L^\theta = \frac{1}{1+\lambda}(L_r^\theta + \lambda L_c^\theta), \tag{16}$$

where $\lambda \in [0, +\infty)$ represents the Lagrange multiplier. Note that we use an extra weight $1/(1+\lambda)$ to rescale the original Lagrangian objective as proposed in [22] to prevent large updates of θ when λ is large.

Naïve Lagrangian methods calculate λ as a learnable parameter along with θ. However, as shown in [22, Figure 1], dual updates of λ and θ could cause oscillations in the cost and thus unstable training. To overcome this problem, we update λ using the approach proposed in [22, Sec. 5.3], which adopts a PID controller to tune the value of λ based on L_c^θ.

Algorithm 1 *Distributional PID Lagrangian PPO*

Require: c_{limit}, initial parameters of the actor θ_0, reward critic $\upsilon_{r,0}$, cost critic for expectation $\upsilon_{c,0}$ and variance $\upsilon_{\Upsilon,0}$, risk-level α, and number of episodes N

1: **for** i = 1,2,..., N **do**
2: $\mathcal{T} \leftarrow$ execute policy π^{θ_i} in the environment and collect samples of (s, a, r, c)
3: $V_{\mathrm{CVaR}} \leftarrow$ (6)
4: $\hat{A}_{\mathrm{CVaR}} \leftarrow$ (14), $L_c^\theta \leftarrow$ (15)
5: $\lambda \leftarrow$ update with [22, Sec.5.3]
6: $\theta_{i+1} \leftarrow$ update with L^θ computed in (16)
7: $\upsilon_{r,i+1}, \upsilon_{c,i+1}, \upsilon_{\Upsilon,i+1} \leftarrow$ update with $L_{V_c}, L_{\Upsilon_c}, L_{V_r}$ computed in (12) and (13)
8: **end for**

TABLE I

TRAFFIC RULES CONSIDERED FOR OUR PROBLEM [38].

Rule	past-MTL formula
R_G1	in-same-lane$(x_{\mathrm{ego}}, x_{\mathrm{obs}})$ $\wedge$ in-front-of$(x_{\mathrm{ego}}, x_{\mathrm{obs}})$ $\wedge\neg\mathbf{O}_{[0,t_c]}$(cut-in$(x_{\mathrm{obs}}, x_{\mathrm{ego}})$ $\wedge$ $\mathbf{P}(\neg$cut-in$(x_{\mathrm{obs}}, x_{\mathrm{ego}}))$) $\Rightarrow$ keeps-safe-distance-prec$(x_{\mathrm{ego}}, x_{\mathrm{obs}})$
R_G2	brakes-abruptly(x_{ego}) $\Rightarrow$ necessary-to-brake$(x_{\mathrm{ego}}, x_{\mathrm{obs}})$*
R_G3	keeps-lane-speed-limit(x_{ego}) $\wedge$ keeps-type-speed-limit$(x_{\mathrm{ego}})\wedge$ keeps-brake-speed-limit(x_{ego}) $\wedge$ keeps-fov-speed-limit(x_{ego})
R_G0	R_G1 $\wedge$ R_G2 $\wedge$ R_G3

* necessary-to-brake$(x_{\mathrm{ego}}, x_{\mathrm{obs}})$:= $\exists$obs $\in \mathcal{I}_{\neg\mathrm{ego}}$: precedes$(x_{\mathrm{ego}}, x_{\mathrm{obs}})$ $\wedge$ ($\neg$keeps-safe-distance-prec$(x_{\mathrm{ego}}, x_{\mathrm{obs}})$ $\vee\neg$brakes-abruptly-relative$(x_{\mathrm{ego}}, x_{\mathrm{obs}})$)

TABLE II

HYPERPARAMETERS USED IN TRAINING AND TRAFFIC RULES

Parameters as in [22]	Value	Constants as in [38]	Value
PID-K_P	0.5	a_{abrupt}	-2.0
PID-K_I	0.001	v_{fov}	50.0
PID-K_D	0.0	v_{type}(truck)	22.22
Samples / Epoch	8192	v_{brake}	43.0
PPO Batch Size	2048	c_{limit}	7.5
PPO Epochs	8		

IV. NUMERICAL EXPERIMENTS

We evaluate the proposed method on the highD dataset and demonstrate its effectiveness by comparing it to other state-of-the-art methods.

A. Environment

1) Dataset: We built the training environment on top of CommonRoad-RL [37]. All training and testing scenarios are generated from the highway drone (highD) dataset [24], which contains $16.5\,\mathrm{h}$ of vehicle trajectories with a time step of $\Delta t = 0.04\,\mathrm{s}$. Note that we choose recorded data for its diverse driving behavior compared to existing driver models in a simulator. Additionally, the scenario becomes more critical when other vehicles do not react to the ego vehicle, since the existing driver models are more conservative than human drivers. We convert the dataset into 2000 scenarios with a duration of $40\,\mathrm{s}$. For each scenario, we randomly choose a vehicle as the ego vehicle, create a planning problem using its initial and final states, and remove this vehicle from the scenario. The scenarios are split into 70% training set and 30% test set. An exemplary scenario in the highD dataset is shown in Fig. 5.

2) Traffic rules: We use three general highway traffic rules formalized in [38] in our experiments, as listed in Table I. The variables x_{ego} and x_{obs} represent the ego vehicle and obstacles' states, respectively. Note that the traffic rules are formalized in STL in [38]. However, we reformalize them in past-MTL, since they are equivalent for discrete time [39]. Additionally, we add rule R_G0, the conjunction of all rules, to track the overall traffic rule compliance. For detailed definitions of predicates used in Table I, we refer interested readers to [38, Sec. III-C]. User-defined parameters for the predicates are given in Table II.

3) Training settings: The cost is defined by the indicator function representing whether rule R_G0 is violated:

$$c_t = \mathbf{1}_{\text{R\_G0\_violated}} \tag{17}$$

We make sure that the agent always begins in a safe state, i.e., $c_0 = 0$, as done in the benchmark environment for constrained RL safety-gym [40]. The reward is chosen as:

$$\begin{aligned} r_t = & 50 \cdot \mathbf{1}_{\text{reach\_goal}} - 20 \cdot \mathbf{1}_{\text{collision}} - 20 \cdot \mathbf{1}_{\text{off\_road}} \\ & - 10 \cdot \mathbf{1}_{\text{time\_out}} + 0.025\,[s_{\text{goal}}(t-1) - s_{\text{goal}}(t)] \\ & + 0.025\,[d_{\text{goal}}(t-1) - d_{\text{goal}}(t)], \end{aligned} \tag{18}$$

where $s_{\text{goal}}(t)$ and $d_{\text{goal}}(t)$ represent the longitudinal and the lateral distances between the ego vehicle and the goal region at time t, respectively. $\mathbf{1}_*$ is an indicator function that equals one if an event $*$ happens and zero otherwise.

We adopted the observations from [37, Table II] and captured the surrounding vehicles in a lane-based fashion as demonstrated in [37, Fig. 2(a)]. For each vehicle, the longitudinal distance, relative velocity, and relative acceleration are observed. Note that we include relative acceleration since this information is crucial for learning rule R_G2. Additionally, we include the speed limits used in the rule R_G3 in the observations. The ego vehicle's actions are defined as the control inputs, i.e., the accelerations of a point-mass model defined in [41, Sec. III-A]. Note that since the absolute acceleration is bounded by the friction circle in the used vehicle model, the RL agent's actions are always feasible.

B. Results and Discussions

1) Training results: We train five groups of agents: unconstrained as a baseline, constrained ones using our method with $\alpha = 0.5$ and $\alpha = 0.9$, respectively, and constrained ones using a worst-case soft actor-critic (WCSAC) proposed in [23], each trained with two random seeds. The hyperparameters are given in Table II, which are found empirically using grid search. Note that strict constraints that should never be violated can be enforced by setting $c_{\text{limit}} = 0$, which however did not converge in our experiments. Fig. 3 shows the learning progress of all constrained agents, where the curves and shaded area indicate the mean and the standard deviation of two runs, respectively. Both agents trained using our method converged to a high goal-reaching rate fast and steadily learned to reduce the cost. Instead, WCSAC agents showed high variance regarding the goal-reaching rate between two random seeds, therefore they are less stable and less robust compared to our approach. Furthermore, the converged costs of all agents are lower than the threshold because of the impact of α. The smaller α is, the lower the converged costs. Additionally, although the WCSAC agents converged to slightly lower total costs than our agents, the curves of episode length suggested that their episodes were terminated earlier. Therefore, the average cost for each step was actually higher.

2) Test results: Next, we analyze the performance and the traffic rule compliance of the trained agents in test scenarios. We measure the agents' performance based on collision rate, off-road rate, and goal-reaching rate since they are the primary aspects covered by the reward function (18). The performance of all constrained agents is given in Table III, whereas the rates of traffic rule compliance of all agents are shown in Fig. 4. In addition to the agents introduced above, we include the evaluation of the recorded human trajectories as a baseline.

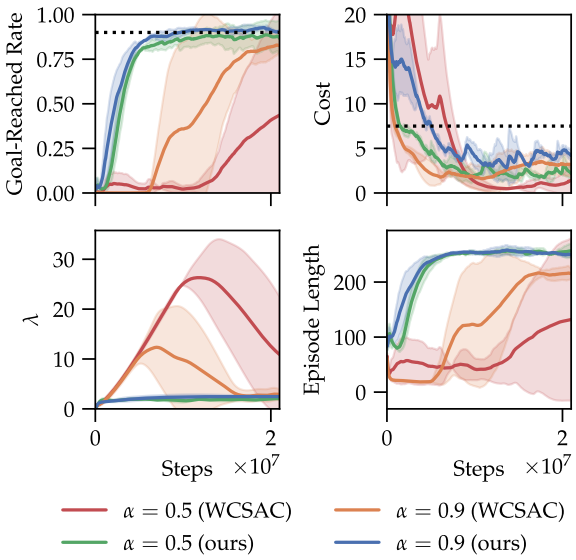

Fig. 3. Learning curves of our method and WCSAC.

TABLE III
PERFORMANCE ON THE TEST SCENARIOS

Agent	Collision rate	Off-road rate	Goal-reaching rate
$\alpha = 0.5$ (WCSAC)	4.0%	45.9%	47.3%
$\alpha = 0.9$ (WCSAC)	2.4%	11.3%	81.0%
$\alpha = 0.5$ (ours)	3.0%	2.4%	91.2%
$\alpha = 0.9$ (ours)	**1.8%**	**1.5%**	**91.3%**

Table III shows that both our agents had a lower collision rate, a lower off-road rate, and a higher goal-reaching rate compared to the WCSAC agents for the same α, thus they performed better. As for the rule compliance behavior, as shown in Fig. 4 all constrained agents outperform the human benchmark for the overall compliance for three rules, i.e., for R_G0. The slight difference between each rule is because R_G1 and R_G2 depend on other vehicles and therefore are harder to learn. Furthermore, our agents obeyed traffic rules very similar to the WCSAC agents. For our method, reducing α from 0.9 to 0.5 increases compliance for R_G0 by 2%, whereas WCSAC shows no difference in overall compliance for different α. Using our method, future studies could include α as an input of the policy, thus making it possible to learn to behave on different levels of risk during deployment.

To further validate the robustness of our method, we introduce noise to the observations. For an error bound of 25%, the compliance for R_G0 of our agent only decreased by 3%.

Combining all findings, we can conclude that our approach outperformed WCSAC since our agents reached the goal

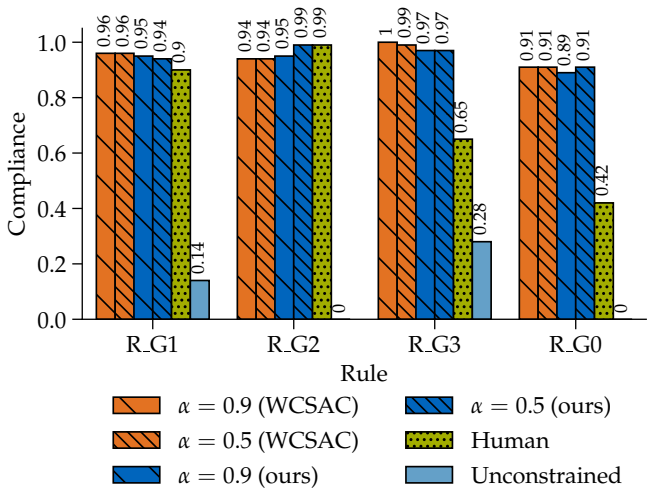

Fig. 4. Rate of traffic rule compliance of trained agents and recorded human trajectories evaluated on the test set.

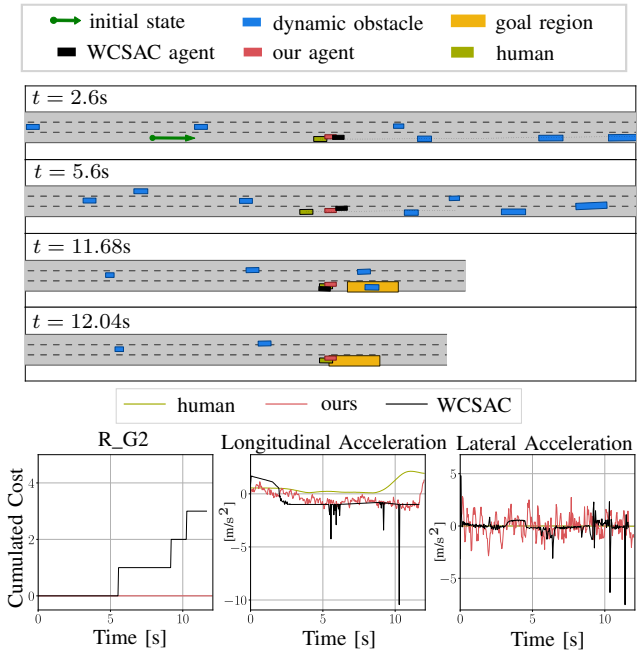

Fig. 5. An exemplary scenario where the WCSAC agent violates R_G2 and collides with the road boundary, whereas our agent reaches the goal at the end. Both agents were trained using $\alpha = 0.5$.

more frequently while experiencing fewer accidents and adhering to traffic rules similarly compared to the WCSAC agents. We show an exemplary scenario in Fig. 5 for which the WCSAC agent violates rule R_G2 and collides with the road boundary, whereas our agent successfully reaches the goal. Note that the noise in the accelerations can be reduced by adding a punishment to the reward function or using the jerk as a control input.

V. Conclusions

We propose a method to integrate PID-controlled Lagrangian constrained PPO with distributional reinforcement learning. Our method trains agents to comply with traffic rules formalized in past-MTL. By encoding the traffic rules in the constraints, our approach can effectively learn complex traffic rules. We evaluated our method using the highD dataset. Our numerical experiments show that our method is more stable and agents trained using our approach reach a goal area more frequently while following the traffic rules similar to state-of-the-art approaches.

References

[1] S. Maierhofer, A.-K. Rettinger, E. C. Mayer, and M. Althoff, "Formalization of interstate traffic rules in temporal logic," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2020, pp. 752–759.

[2] K. Esterle, L. Gressenbuch, and A. Knoll, "Formalizing traffic rules for machine interpretability," in *Proc. of the IEEE Connected and Automated Vehicles Symposium*, 2020, pp. 1–7.

[3] S. Maierhofer, P. Moosbrugger, and M. Althoff, "Formalization of intersection traffic rules in temporal logic," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2022, pp. 1135–1144.

[4] A. Rizaldi and M. Althoff, "Formalising traffic rules for accountability of autonomous vehicles," in *Proc. of the 18th IEEE Int. Conf. on Intelligent Transportation Systems*, 2015, pp. 1658–1665.

[5] A. Rizaldi, F. Immler, and M. Althoff, "A formally verified checker of the safe distance traffic rules for autonomous vehicles," in *8th NASA Formal Methods Symposium*, 2016, pp. 175–190.

[6] A. Rizaldi, J. Keinholz, M. Huber, J. Feldle, F. Immler, M. Althoff *et al.*, "Formalising and monitoring traffic rules for autonomous vehicles in Isabelle/HOL," in *Proc. of the 13th Int. Conf. on Integrated Formal Methods*, 2017, pp. 50–66.

[7] N. Aréchiga, "Specifying safety of autonomous vehicles in signal temporal logic," in *Proc. of the IEEE Intelligent Vehicles Symposium*, 2019, pp. 58–63.

[8] M. Hekmatnejad, S. Yaghoubi, A. Dokhanchi, H. B. Amor, A. Shrivastava, L. Karam *et al.*, "Encoding and monitoring responsibility sensitive safety rules for automated vehicles in signal temporal logic," in *Proc. of the 17th ACM-IEEE Int. Conf. on Formal Methods and Models for System Design*, 2019, pp. 1–11.

[9] T. Wongpiromsarn, U. Topcu, and R. M. Murray, "Receding horizon temporal logic planning for dynamical systems," in *Proc. of the IEEE Conf. on Decision and Controlheld jointly with Chinese Control Conf.*, 2009, pp. 5997–6004.

[10] A. Ulusoy, M. Marrazzo, and C. Belta, "Receding horizon control in dynamic environments from temporal logic specifications." in *Robotics: Science and Systems*, 2013.

[11] L. I. Reyes Castro, P. Chaudhari, J. T. S. Karaman, E. Frazzoli, and D. Rus, "Incremental sampling-based algorithm for minimum-violation motion planning," in *Proc. of the IEEE Conf. on Decision and Control*, 2013, pp. 3217–3224.

[12] J. Karlsson, F. S. Barbosa, and J. Tumova, "Sampling-based motion planning with temporal logic missions and spatial preferences," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 15 537–15 543, 2020.

[13] E. M. Wolff, U. Topcu, and R. M. Murray, "Optimization-based trajectory generation with linear temporal logic specifications," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2014, pp. 5319–5325.

[14] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra *et al.*, "Playing Atari with deep reinforcement learning," in *Proc. of the Twenty-seventh Conf. on Neural Information Processing Systems – Workshop on Deep Learning*, 2013.

[15] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez *et al.*, "Mastering the game of Go without human knowledge," *nature*, vol. 550, no. 7676, pp. 354–359, 2017.

[16] T. Xu, Y. Liang, and G. Lan, "CRPO: A new approach for safe reinforcement learning with convergence guarantee," in *Int. Conf. on Machine Learning*, 2021, pp. 11 480–11 491.

[17] Y. Chow, O. Nachum, A. Faust, E. Duenez-Guzman, and M. Ghavamzadeh, "Lyapunov-based safe policy optimization for continuous control," in *Int. Conf. on Machine Learning – Workshop on Reinforcement Learning for Real Life*, 2019.

[18] Y. Liu, J. Ding, and X. Liu, "IPO: Interior-point policy optimization under constraints," in *Proc. of the AAAI Conf.on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 4940–4947.

[19] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *Int. Conf. on Machine Learning*, 2017, pp. 22–31.

[20] L. Wen, J. Duan, S. E. Li, S. Xu, and H. Peng, "Safe reinforcement learning for autonomous vehicles through parallel constrained policy optimization," in *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*, 2020, pp. 1–7.

[21] Y. Chow, M. Ghavamzadeh, L. Janson, and M. Pavone, "Risk-constrained reinforcement learning with percentile risk criteria," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 6070–6120, 2017.

[22] A. Stooke, J. Achiam, and P. Abbeel, "Responsive safety in reinforcement learning by PID Lagrangian methods," in *Proc. of the Int. Conf. on Machine Learning*, 2020, pp. 9133–9143.

[23] Q. Yang, T. D. Simão, S. H. Tindemans, and M. T. Spaan, "WC-SAC: Worst-case soft actor critic for safety-constrained reinforcement learning," in *Proc. of the Thirty-Fifth AAAI Conference on Artificial Intelligence.*, 2021.

[24] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein, "The highD dataset: A drone dataset of naturalistic vehicle trajectories on German highways for validation of highly automated driving systems," in *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*, 2018, pp. 2118–2125.

[25] E. Altman, *Constrained Markov decision processes: stochastic modeling*. Routledge, 1999.

[26] M. G. Bellemare, W. Dabney, and R. Munos, "A distributional perspective on reinforcement learning," in *Int. Conf. on Machine Learning*, 2017, pp. 449–458.

[27] Y. C. Tang, J. Zhang, and R. Salakhutdinov, "Worst cases policy gradients," in *Proc. of the 3rd Annual Conference on Robot Learning*, vol. 100, 2019, pp. 1078–1093.

[28] G. Grimmett and D. Welsh, *Probability: an introduction*. Oxford University Press, 2014.

[29] S. Sarykalin, G. Serraino, and S. Uryasev, "Value-at-risk vs. conditional value-at-risk in risk management and optimization," in *State-of-the-art decision-making tools in the information-intensive age*. Informs, 2008, pp. 270–294.

[30] V. Khokhlov, "Conditional value-at-risk for elliptical distributions," *Evropskỳ časopis ekonomiky a managementu*, vol. 2, no. 6, pp. 70–79, 2016.

[31] R. Alur and T. A. Henzinger, "Real-time logics: complexity and expressiveness," in *Proc. of the IEEE Symposium on Logic in Computer Science*, 1990, pp. 390–401.

[32] P. Thati and G. Roşu, "Monitoring algorithms for metric temporal logic specifications," *Electronic Notes in Theoretical Computer Science*, vol. 113, pp. 145–162, 2005.

[33] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in *Int. Conf. on Learning Representations*, 2016.

[34] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[35] R. S. Sutton, A. R. Mahmood, and M. White, "An emphatic approach to the problem of off-policy temporal-difference learning," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 2603–2631, 2016.

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

[37] X. Wang, H. Krasowski, and M. Althoff, "CommonRoad-RL: A configurable reinforcement learning environment for motion planning of autonomous vehicles," in *Proc. of the IEEE Int. Conf. on Intelligent Transportation Systems*, 2021, pp. 466–472.

[38] L. Gressenbuch and M. Althoff, "Predictive monitoring of traffic rules," in *Proc. of the International Intelligent Transportation Systems Conference*, 2021, pp. 915–922.

[39] E. Bartocci, J. Deshmukh, A. Donzé, G. Fainekos, O. Maler, D. Ničković *et al.*, "Specification-based monitoring of cyber-physical systems: A survey on theory, tools and applications," in *Lectures on Runtime Verification*. Springer, 2018, pp. 135–175.

[40] A. Ray, J. Achiam, and D. Amodei, "Benchmarking safe exploration in deep reinforcement learning," in *Proc. NeurIPS – Workshop on Deep Reinforcement Learning*, 2019.

[41] M. Althoff, M. Koschi, and S. Manzinger, "CommonRoad: Composable benchmarks for motion planning on roads," in *IEEE Intelligent Vehicles Symposium*, 2017, pp. 719–726.

APPENDIX

A. *Proof of* (8)

We define $\mathbb{E}_s^\pi[\cdot] := \mathbb{E}[\cdot|s_0 = s, \pi]$ for readability. Note that all variables with a time index are (functions of) random variables in the subsequent proof.

$$\begin{aligned}\mathbb{E}_s^\pi[C^2] &\overset{(1)}{=} \mathbb{E}_s^\pi[(c_0 + \sum_{t=1}^{\infty} \gamma^t c_t)^2] \\ &= c_0^2 + 2c_0\, \mathbb{E}_s^\pi[\sum_{t=1}^{\infty} \gamma^t c_t] + \mathbb{E}_s^\pi[(\sum_{t=1}^{\infty} \gamma^t c_t)^2].\end{aligned} \tag{19}$$

For two events A and B, we get the conditional probability [28, Eq.(1.32)]:

$$p(A, B) = p(B)p(A|B). \tag{20}$$

Combining the conditional expectation [28, Theorem 2.42] and the expected value of a function f of discrete random variables x and y [28, Theorem 3.10], we obtain

$$\mathbb{E}[f(x, y)] = \sum_{y' \in y} p(y')\, \mathbb{E}[f(x, y)|y = y']. \tag{21}$$

By an abuse of notation, we use x and y to represent sequences of random variables. For $i \in \{1, 2\}$, we derive

$$\begin{aligned}&\mathbb{E}_s^\pi[(\sum_{t=1}^{\infty} \gamma^t c_t)^i] = \mathbb{E}^\pi[(\sum_{t=1}^{\infty} \gamma^t c_t)^i|s_0 = s] \\ &\overset{①}{=} \sum_a \sum_{s'} p(s', a|s)\, \mathbb{E}^\pi[(\sum_{t=1}^{\infty} \gamma^t c_t)^i|s_0 = s, a_0 = a, s_1 = s'] \\ &= \gamma^i \sum_a \sum_{s'} p(s', a|s)\, \mathbb{E}^\pi[(\sum_{t=1}^{\infty} \gamma^{t-1} c_t)^i|s_1 = s'] \\ &= \gamma^i \sum_a \sum_{s'} \underbrace{p(s', a|s)}_{\pi(a|s)p(s'|s,a)}\, \mathbb{E}_{s'}^\pi[(\sum_{t=0}^{\infty} \gamma^t c_t)^i],\end{aligned} \tag{22}$$

where ① is based on (21) for $y = (a_0, s_1), y' = (a, s')$, $x = (a_1, s_2, a_2, s_3, \cdots)$, and $(\sum_{t=1}^{\infty} \gamma^t c_t)^i = f(x, y)$, because the cost return depends on all future states and actions.

For $i = 1$, inserting (1),(2) in (22), we obtain

$$\mathbb{E}_s^\pi[\sum_{t=1}^{\infty} \gamma^t c_t] = \gamma \sum_a \pi(a|s) \sum_{s'} p(s'|s, a) V_c^\pi(s'), \tag{23}$$

and for $i = 2$, inserting (1),(2),(4) in (22):

$$\begin{aligned}\mathbb{E}_s^\pi[(\sum_{t=1}^{\infty} \gamma^t c_t)^2] = \gamma^2 \sum_a \pi(a|s) \\ \sum_{s'} p(s'|s, a)(\Upsilon_c^\pi(s') + V_c^\pi(s')^2).\end{aligned} \tag{24}$$

After inserting (23) and (24) into (19), which in turn is inserted in (4), we obtain

$$\begin{aligned}\Upsilon_c^\pi(s) = c_0^2 - V_c^\pi(s)^2 \\ + 2c_0\gamma \sum_a \pi(a|s) \sum_{s'} p(s'|s, a) V_c^\pi(s') \\ + \gamma^2 \sum_a \pi(a|s) \sum_{s'} p(s'|s, a)(\Upsilon_c^\pi(s') + V_c^\pi(s')^2).\end{aligned}$$

□