Two-Stage Safe Reinforcement Learning for High-Speed Autonomous Racing

Jingyu Niu
Research Center for Intelligent Computing Systems
State Key Laboratory of Computer Architecture, ICT, CAS
University of Chinese Academy of Sciences
Beijing, China
niujingyu17b@ict.ac.cn

Yu Hu *
Research Center for Intelligent Computing Systems
State Key Laboratory of Computer Architecture, ICT, CAS
University of Chinese Academy of Sciences
Beijing, China
huyu@ict.ac.cn

Beibei Jin
Research Center for Intelligent Computing Systems
State Key Laboratory of Computer Architecture, ICT, CAS
University of Chinese Academy of Sciences
Beijing, China
jinbeibei@ict.ac.cn

Yinhe Han
Research Center for Intelligent Computing Systems
State Key Laboratory of Computer Architecture, ICT, CAS
University of Chinese Academy of Sciences
Beijing, China
yinhes@ict.ac.cn

Xiaowei Li
Research Center for Intelligent Computing Systems
State Key Laboratory of Computer Architecture, ICT, CAS
University of Chinese Academy of Sciences
Beijing, China
lxw@ict.ac.cn

***Abstract*—Decision making for autonomous driving is a safety-critical control problem. Prior works of safe reinforcement learning either tackle the problem with reward shaping or with modifying the reinforcement learning exploration process. However, the former cannot guarantee the safety during the learning process, while the latter relies heavily on expertise to design exquisite exploration policy. Currently, only short-term decision makings for low-speed driving were achieved in road scenes with basic geometries. In this paper, we propose a two-stage safe reinforcement learning algorithm to automatically learn a long-term policy for high-speed driving that guarantees safety during the entire training. In the first learning stage, model-free reinforcement learning is followed by a rule-based safeguard module to avoid danger at low speed without expert fine-tuning. In the second learning stage, the rule-based module is replaced with a data-driven counterpart to develop a closed-form analytical safety solution for high-speed driving. Moreover, an adaptive reward function is designed to match the different objectives of the two learning stages for faster convergence to an optimal policy. Experiments are conducted on a racing simulator TORCS which has complex racing tracks (e.g. sharp turns, hills). Compared with the state-of-the-art baselines, the results show that our method achieves zero safety violation and quickly converges to a more efficient and stable policy with an average speed of 127 km/h (3.3% higher than the best result of baselines) and an average swing of 3.96 degrees.**

***Index Terms*—safe reinforcement learning, autonomous racing**

I. Introduction

Deep reinforcement learning (DRL) is a promising decision-making paradigm for autonomous driving. It tries to find an optimal policy to maximize the expected cumulative reward by balancing the exploitation of current policy and the exploration of unknown space. The exploration process may lead to some unsafe actions and cause personal injury and economic loss in the real world. As autonomous driving is a safety-critical field, how to ensure the safety during DRL learning and execution is a key research challenge.

TABLE I compares different researches on safe RL from various perspectives. From the perspective of safe RL [1] for autonomous driving, prior works can be broadly divided into two categories: (i) reward shaping [2-5]: designing reward with risk terms; (ii) modifying the RL exploration process by action masking [6-11] or correction with safety checker [12-14]. However, the safety guarantees of those approaches are strongly dependent on expertise to design exquisite exploration policy. And they are only experimented at low speed for a short distance of road without considering the complexity of road geometry.

Meanwhile, there are some data-driven approaches that seek to solve the problem of safety learning with theoretical support instead of manual efforts. These methods are based on constrained Markov Decision Processes (CMDP) [15], which aims to maximize the expected cumulative reward of RL while also satisfying the expected cumulative constraints. One way [16-18] is to map policy parameters into the safety set by using the trust-region policy optimization (TRPO) [19]. Another way takes advantage of modular design and gives a closed-form solution to correct unsafe actions with a single-step dynamics model [20] or the Lyapunov stability theory[21]. Although no expert prior knowledge is needed, these methods require pre-training or the assumption of the feasible baseline as a start to ensure learning safety. And they only demonstrate on robot locomotion tasks with simple 2-dimensional scenes, but not high-speed, long-term and complex autonomous driving tasks.

In this paper, we propose a two-stage learning safe RL approach for end-to-end continuous control. It can solve the problem of learning an efficient and stable policy with safety guarantee during both training and execution for autonomous driving tasks, without the prerequisite of expert prior knowl-

∗ Corresponding author: Yu Hu. This work is supported in part by the National Key RD Program of China under grant No. 2018AAA0102701, in part by the Science and Technology on Space Intelligent Control Laboratory under grant No. HTKJ2019KL502003, and in part by the Innovation Project of Institute of Computing Technology, Chinese Academy of Sciences under grant No. 20186090.

TABLE I
COMPARISON OF VARIOUS APPROACHES IN SAFE RL

Domain	Methods	Safety during Learning	Expert Prior Knowledge		Action Space[b]	Average Speed (km/h)	Road Complexity[c]
			dependence	*timescope*[a]			
autonomous driving	our method	yes	weak	initial	L, C	128	M
	reward shaping [2-5]	no	no	no	L, C or D	<=95	M or S
	action masking mechanism [6-11]	yes	strong	all	H, D	<=63	S
	safety checker[12-14]	no	strong	all	H, D or L, C	<=108	M or S
robot locomotion pursuing safety during learning	trust-region-based safety projection [16-18]	no	no	no	L, C	NA	simple two-dimensional panel
	signle-step safety layer [20]	yes	weak	pretrain	L, C	NA	
	Lyapunov-based a-projection [21]	no	no	no	L, C	NA	

[a]initial: expert prior knowledge is needed during initial learning phase; all: expert prior knowledge is needed during the entire learning and testing phase; pretrain: the agent is pretrained with expert prior knowledge before policy learning.
[b]L: low-level control, such as $\{steer, throttle, brake\}$ in driving task and $\{joint\ torque\}$ in robot task; H: high-level options, such as $\{go, wait, lanechange\}$ in driving tasks; C: continuous action; D: discrete action.
[c]M: complex tracks of multiple structures and physics; S: urban or highway roads with simple geometry.

edge or feasible baselines. The specific autonomous driving scenario we investigate is a high-speed racing environment with complex road geometry (e.g. sharp turns, hills). For the sake of clarity, we define safety as no collision happened or not out of track. The policy efficiency is evaluated by the average speed of completing a task, the higher the better. And the policy stability is evaluated by the amplitude of the car's swing from side to side, the lower the better. As shown in Fig. 1, our two-stage safe RL approach adopts a modular design that combines an independent safeguard mechanism with model-free DRL and an adaptive reward function. The safeguard mechanism contains a rule-based module and a data-driven module. In the first learning stage, the rule-based module only helps RL agent avoid dangers, but does not require expert fine-tuning for an optimal action. This coarse correction does not care whether the performance is optimal or not and it is the difference between our rule-based module and prior rule-based approaches. The adaptive reward encourages policy to update in the direction of improving safety before efficiency. At the same time, the data-driven module is trained by data collected at this stage. Once the data-driven module has acquired sufficient correction capability, our algorithm enters the second learning stage by replacing the rule-based safeguard module with the data-driven safeguard module. Inspired by recent works [20, 21] but not limited by pre-training and the assumption above, we develop an analytical feasible solution to ensure safety and accelerate convergence. The adaptive reward now changes to guide policy convergence to high efficiency and better stability.

Our contributions are three-fold. First, we present a two-stage safe RL algorithm for end-to-end continuous control. It combines a model-free RL algorithm with an independent safeguard mechanism and an adaptive reward function. Second, we develop a closed-form analytical solution to correct unsafe actions for ensuring safety during learning and execution. Finally, we apply our approach to a challenging task of high-speed racing on tracks with complex geometry. And we evaluate it in the open racing simulator TORCS [22]. The experiments show that safety is guaranteed during the learning and execution phase. Meanwhile, driving policy can converge faster to higher efficiency and better stability than baselines.

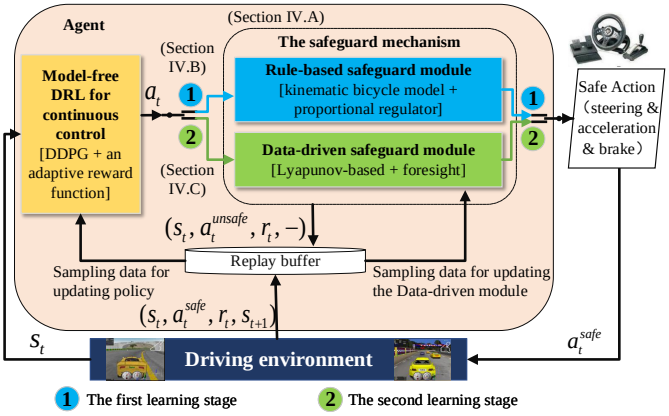


Fig. 1. Overview of our proposed safe RL algorithm

The rest of this paper is organized as follows. In Section II, we introduce the related work in safe reinforcement learning. In Section III, we briefly review the concept of DDPG and CMDP. In Section IV, we describe the architecture and learning procedure of our two-stage safe RL method in detail. In Section V, we evaluate our approach in 3D racing simulator TORCS. Finally, we conclude the paper and propose the future direction in Section VI.

II. RELATED WORK

The comparison of various safe RL approaches are shown in TABLE I.

Safe RL for autonomous driving. There are two categories of safe autonomous driving with DRL. (i) Reward shaping approaches [2-5] directly modify the RL objective function by integrating risk into reward to reduce safety violations. But this category cannot guarantee the absolute safety of the training process. (ii) Another category is to modify the RL exploration process [1] with expert prior knowledge, which can be further divided into action masking mechanism and safe checker. The action masking mechanism forces the discrete-action DRL to only choose the best action from a list of safe actions. This masking mechanism can be implemented by a rule-based [6-8], a linear temporal logic [9, 10] or a prediction [11] approach. Although the masking mechanism can achieve safety during learning, these approaches rely heavily on manual design for short distance, low-speed urban or highway scenarios with

simple geometry, e.g. a straight road. The safe checker is more flexible than action masking mechanism to deploy on discrete or continuous action DRL. The safe checker assesses danger level of current actions generated by DRL and corrects unsafe actions to ensure safe driving [12-14]. But these methods still fail to achieve 100% safety because the safety checker heavily relies on expert experience and lacks theoretical safety guarantees that make it difficult to apply to unseen scenes.

Safe RL with theoretical safety guarantees during learning. There are some approaches for robotics that seek to analytically solve the problem of safety during DRL learning based on continuous constrained Markov Decision Processes (CMDP) [15]. [16-18] extend the trust-region policy optimization [19] with CMDP to update policy within safety bounds in expectation. These trust-region methods achieve near-safety satisfaction, but cannot guarantee zero safety violation especially at the beginning of training, due to approximation errors of the Fisher information matrix. And they are only suitable for RL based on TRPO. Gal Dalal et al. [20] propose the safety layer concept which models single-step dynamics and corrects unsafe actions to accomplish zero safety violation. The safety layer can be directly added to the output of a continuous RL algorithm to obtain a closed-form analytical solution of action correction. The a-projection approach in [21] leverages the idea in [20], but further uses the notion of Lyapunov function, which is an important tool for stability certification of dynamical systems [23, 24], to represent constraints from an energy perspective for better stability and robustness. Although [20,21] can guarantee safety during learning in an elegant way, they only work reliably with adequate pre-training or assume a feasible policy as the starting baseline. In addition, they are not applied to the complex, long-horizon and high-speed autonomous driving tasks.

Inspired by [20, 21], we propose a two-stage learning safe RL algorithm, which includes a safeguard mechanism and an adaptive reward function to assist model-free RL in a high-speed racing environment with complex road geometry. Different from [20, 21], (i) our method can keep safe learning from scratch by adding the simplified dynamics knowledge in the initial learning phase without additional assumptions and meticulous expert efforts; (ii) we extend to a foresight safeguard solution with multi-step prediction for timely response and faster convergence to an optimal policy.

III. PRELIMINARIES

In this section, we briefly introduce Deep Deterministic Policy Gradient (DDPG) [25] and CMDP, and explain why DDPG is chosen as a part of our safe RL algorithm.

A. RL & Deep deterministic policy gradient (DDPG)

RL is a decision-making method based on Markov Decision Progress (MDP), defined by the tuple (S, A, P, r, γ), where state $s_t \in S$, action $a_t = \pi(s_t) \in A$, reward $r_t = r(s_t, a_t)$ at time step t, $P : S * A \rightarrow Distr(S)$ is the probabilistic transition function, and $\gamma \in [0, 1]$ is the discount factor to control the influence of future rewards. The goal of RL is to learn an optimal policy π to maximize the excepted discounted sum of future rewards J_R^π.

Since we focus on continuous control for autonomous driving task, the off-policy actor-critic (AC) algorithm is a kind of RL that can be well suited for, because it has high data-efficiency by allowing the use of data from different policies. DDPG is currently the most representative work among AC algorithms. Without loss of generality, we use DDPG in this work to introduce our two-stage safe RL algorithm. In the future, we will try other AC algorithms to show the effectiveness of the proposed algorithm.

DDPG contains an actor network $\pi(s_t|\theta^\pi)$, a critic network $Q(s_t, a_t|\theta^Q)$, a replay buffer and two target networks $\hat{\pi}(s_t|\theta^{\hat{\pi}})$, $\hat{Q}(s_t, a_t|\theta^{\hat{Q}})$ corresponding to the actor and critic network respectively. The actor network is responsible for generating an action and is updated by policy gradient and the critic estimation, as shown in (1). The critic network is the action-value function to measure the quality of a state-action pair. Based on the Bellman equation [25], the critic value can be approximated by a neural network as (2). The critic learning depends on minimizing the error between the target value y_i and the current value estimation (TD error, as seen in (3)).

$$\nabla_{\theta^\pi} \mathrm{J} \approx \frac{1}{\mathrm{N}} \sum_{i=1}^{N} \nabla_a Q(s, a|\theta^Q)|_{s=s_t, a=\pi(s_t)} \nabla_{\theta^\pi} \pi(s|\theta^\pi)|_{s=s_t} \tag{1}$$

$$Q(s_t, a_t) = E[r_t + \gamma Q(s_{t+1}, \pi(s_{t+1}))] \tag{2}$$

$$\begin{cases} L(\theta^Q) = \frac{1}{\mathrm{N}} \sum\limits_{i=1}^{N} \left(y_i - Q(s_i, a_i|\theta^Q)\right)^2 \\ y_i = r_i + \gamma \cdot \hat{Q}(s_{i+1}, \hat{\pi}(s_{i+1}|\theta^{\hat{\pi}})|\theta^{\hat{Q}}) \end{cases} \tag{3}$$

where N in equation (2) and (4) is the batch size of sampling. Two target networks are updated by copying the corresponding networks with a delay factor τ to stabilize learning.

$$\begin{aligned} \theta^{\hat{Q}} &= \tau \cdot \theta^Q + (1-\tau)\theta^{\hat{Q}} \\ \theta^{\hat{\pi}} &= \tau \cdot \theta^\pi + (1-\tau)\theta^{\hat{\pi}} \end{aligned} \tag{4}$$

B. Constrained Markov Decision Process (CMDP)

CMDP extends the MDP by introducing an immediate constraint cost $d(s_t, a_t)$ just like reward and an upper bound d_0 on the expected cumulative constraint cost J_d^π. The RL problem in CMDP can be expressed as:

$$\max_{\theta^\pi} J_R^\pi = E[\sum_{t=0}^{\infty} \gamma^t r_t], \ s.t. \ J_{\mathrm{d}}^\pi = E[\sum_{t=0}^{\infty} \gamma^t d_t] \leq d_0 \tag{5}$$

IV. APPROACH

In this section, we first introduce the architecture and learning procedure of our two-stage safe RL method in Section IV.A. Afterwards, we give detailed explanations for the two learning stages in Section IV.B and IV.C, respectively.

A. Architecture

Fig. 1 shows the architecture of our approach. We design a two-stage safe RL method, which consists of a safeguard mechanism and an adaptive reward function to assist model-free RL (DDPG) in ensuring safety during learning. The

safeguard mechanism contains two modules: a rule-based safeguard module, a data-driven safeguard module. These modules work at different learning stages to jointly guarantee the agent's safety.

The pseudo-code of the two-stage safe RL algorithm is summarized in Algorithm 1. In the first learning stage (the blue part in Fig.1), DDPG and data-driven safeguard module have not been trained well because of lacking training data, we activate the rule-based safeguard module after DDPG with the reward objective of learning a safe policy at low driving speed. We store unsafe transitions monitored by the safeguard module $(s_t, a_t, -, r_t, d_t)$ and safe transitions applied to environment $(s_t, a_t^{safe}, s_{t+1}, r_t, d_t)$ in replay buffer. Then, data in the replay buffer can be used to update DDPG and improve the accuracy of the data-driven safeguard module. Once the data-driven safeguard module can reliably detect and correct the unsafe action from DDPG, the learning process enters the second stage of learning (the green part in Fig.1), where the rule-based safeguard module is replaced with the data-driven safeguard module and the reward function changes to pursue a safe, efficient and stable racing policy. To sum up, the proposed two-stage algorithm is safe to learn from scratch and alleviates the requirement of expert prior knowledge in terms of dependence degree and time scope. Therefore, the proposed algorithm is suitable for the safety-critical autonomous driving task. For the high-speed racing task in this paper, safety is expressed as a single constraint about the minimum distance between the agent and two road boundaries. We denote the threshold of safety constraint as T_s. In order to further accelerate the optimization of policy on the basis of safety satisfaction, we set a lower bound of driving speed constraint as T_v. The detailed design of each learning stage is described below.

Algorithm 1 A two-stage safe RL algorithm

Initialize parameters of DDPG θ^{π}, θ^{Q}, $\theta^{\hat{\pi}}$, $\theta^{\hat{Q}}$, the data-driven safeguard module ξ (Lyapunov function), ω (dynamics model), replay buffer Buf
for $i = 0$ to $maximum\ episode$ **do**
  Train w using data from Buf by (13)
  for $j = 0$ to $maximum\ step$ **do**
    Select a_t from DDPG with exploration noise.
    if the data-driven safeguard module is not ready **then**
      Enter the first learning stage
      Get safe action a_t^{safe} through the rule-based safeguard module by (6), apply it to environment
      Store transitions before and after safeguard in Buf
      Update DDPG with an adaptive reward by (1-4, 7)
      Train ξ, ω using data from Buf by (2-4)
    else
      Enter the second learning stage
      Get safe action a_t^{safe} through the data-driven safeguard module by (11,12), apply it to environment
      Store transitions before and after safeguard in Buf
      Update DDPG with an adaptive reward by (1-4, 8)
      Train ξ, ω using data from Buf by (2-4)
    end if
  end for
end for

B. The first learning stage

During the first learning stage, we use a rule-based safeguard module with an adaptive reward to guide DDPG convergence and use the data generated by driving policy to train the data-driven safeguard module. As the current goal of policy learning is to solve safety first at low driving speed, the rule-based module consists of a kinematic bicycle model and a proportional regulator.

The bicycle model is commonly used for describing the vehicle dynamics and has been proved to have the advantages of its simplicity and accurate approximation at low accelerations [26-31]. Therefore, we choose it to check the short-term future trajectory generated by DDPG. The condition of checking safety is to determine whether the next state predicted by the dynamics model is safe and the agent can survive one more step beyond the next state under the maximum steering correction. If this condition is satisfied, the current action from DDPG does not need to be corrected because this module has enough ability to avoid danger by correcting at the next step, or an alternate safe action will be provided by a proportional regulator at the current step. Different from other rule-based methods of pursuing an optimal solution by expert efforts, we prefer to use coarse corrections to only focus on getting out of danger, and let the DRL module search the optimal solution with the adaptive reward. The regulator is formulated as:

$$steer_t^{safe} = \begin{cases} k_1\psi_t + k_2 \cdot \Delta dis_t \text{ if } \hat{dis}_{t+1}|_{s_t,\, a_t} < T_s \text{ or} \\ \qquad\qquad \hat{dis}_{t+2}|_{\hat{s}_{t+1,\,\max\ steer}} < T_s \\ steer_t\ from\ DDPG \quad \text{otherwise} \end{cases}$$

$$accel_t^{safe} = \begin{cases} give\ a\ brake \ \text{ if}(steer_t^{safe} \text{ not from DDPG}) \\ \qquad\qquad \text{and } (steer \text{ not high enough}) \\ accel_t\ from\ DDPG \quad \text{otherwise} \end{cases} \tag{6}$$

where $\hat{dis}_{t+1}$ and $\hat{v}_{t+1}$ are the next minimum distance from two road boundaries and velocity computed by the dynamics model, Δdis_t is the distance between the car location and the central road track, k_1 and k_2 are their coefficients. The steering angle $steer_t$ and acceleration $accel_t$ (as a general term of throttle and brake) are the actions from DDPG.

The reward function in this stage encourages policy to stay away from road boundaries and avoid large deviations from the road axis at low driving speed, as shown in (7). In (7), $\psi\_dis_t = (\cos\psi - |\sin\psi| - |\Delta dis_t|)$, V_{low} is the maximum speed in this phase, ψ is the angle between road and vehicle heading, v is the car speed and Δt is the time interval between two adjacent steps.

$$r_t = \begin{cases} v \cdot \Delta t \cdot \psi\_dis_t & v \le V_{low} \\ (V_{low} - v) \cdot \Delta t \cdot \psi\_dis_t & v > V_{low} \text{ and } \psi\_dis_t \ge 0 \\ v \cdot \Delta t \cdot \psi\_dis_t & v > V_{low} \text{ and } \psi\_dis_t < 0 \end{cases} \tag{7}$$

C. The second learning stage

As the training process goes on, the data-driven safeguard module gradually becomes more reliable. We replace the

rule-based safeguard module with the data-driven safeguard module to find a more elegant solution with CMDP theoretical support. The objective of the second learning stage is to get a safe, efficient and stable racing policy. The reward function also changes adaptively to encourage policy to maximize the effective distance along the road axis at each step for completing tasks quickly and stably, as shown in (8).

$$r_t = \begin{cases} \Delta l \times (\cos\psi - |\sin\psi| - |\Delta dis_t|) & \text{for safe action} \\ -100 & \text{for potential unsafe action} \end{cases} \tag{8}$$

where Δdis_t is the distance between the car location and the central road track, and Δl is the distance between the current and next location.

Our data-driven safeguard module has two submodules, which are based on Lyapunov stability theory and multi-step prediction respectively. Our data-driven safeguard module combines the advantages of them.

(i) Lyapunov-based safeguard submodule

Our Lyapunov-based safeguard submodule is inspired by the Lyapunov-based a-projection proposed by [21]. Let $\pi_{\theta-1}$ denotes the previous policy and π_θ is the current policy. We build the Lyapunov function $Q_{L_{\pi_{\theta-1}}}(s_t,\ a_t;\ \xi)$ which works as the critic network of DDPG to assess safety of the learned policy by J_d^π in (1). The immediate constraint cost is $d(s_t,\ a_t) = \mathbb{1}[|\Delta \mathrm{dis}_t| > 1 - T_s]$. This definition of Lyapunov function satisfies the positive-definite and one-step decease condition for safe action region in Lyapunov stability theory. Then we use the l_2-projection to guide the policy optimization in the direction of reducing Lyapunov value function, as shown in (9):

$$\begin{aligned} a_t^{Ly*} &= \arg\min_{a_t^{Ly}} \tfrac{1}{2}\left\| a_t^{Ly} - \pi_\theta(s_t) \right\|^2 \\ s.t.\ & Q_{L_{\pi_{\theta-1}}}(s_t\,, a_t^{Ly}; \xi) - Q_{L_{\pi_{\theta-1}}}(s_t\,, \pi_{\theta-1}(s_t); \xi) \le \tilde{\varepsilon}(s_0) \\ & \tilde{\varepsilon}(s_0) = (1-\gamma)(d_0 - D_{\pi_{\theta-1}}(s_0)) \end{aligned} \tag{9}$$

where a_t^{Ly} is the action after Lyapunov-based correction, $D(s_0) := E[\sum_{t=0}^{\infty}\gamma^t d(s_t)|\pi,\ s_0]$ where d_0 is the upper-bound value and specifically $d_0 = 1 - T_s$ for our task. (9) can be simplified by approximating the Lyapunov constraint with its first-order Taylor series, as shown in (10):

$$\begin{aligned} a_t^{Ly*} = \arg\min_{a_t^{Ly}} & \tfrac{1-\eta(s_t)}{2}\left\| a_t^{Ly} - \pi_\theta(s_t) \right\|^2 + \\ & \tfrac{\eta(s_t)}{2}\left\| a_t^{Ly} - \pi_{\theta-1}(s_t) \right\|^2 \\ s.t.\ & \left(a_t^{Ly} - \pi_{\theta-1}(s_t)\right)^T g_{L_{\pi_{\theta-1}}}(s_t;\ \xi) \le \tilde{\varepsilon}(s_0) \\ & g_{L_{\pi_{\theta-1}}}(s_t;\ \xi) := \nabla_{a_t} Q_{L_{\pi_{\theta-1}}}(s_t, a_t;\ \xi)\Big|_{a_t=\pi_{\theta-1}(s_t)} \end{aligned} \tag{10}$$

where $\eta(s) \in [0,\ 1)$ is the mixing parameter that controls the trade-off between maximizing reward and maintaining safety. As safety in our task can be represented as a single constraint, we can obtain the analytical solution to (10) as shown in (11). In (11), λ^* is the optimal Lagrange multiplier associated with the safety constraint.

In this way, we have a theoretically safe and robust solution with fast convergence after we obtained a feasible policy from the first learning stage. Note that when the data-driven safeguard module is activated, the current RL policy may not yet ensure safety. Once this happens, the Lyapunov-based submodule only guarantees that the updated policy is safer than the current policy, but cannot guarantee zero safety violation of the updated policy. To compensate for this problem, we design a forward-looking submodule with multi-step prediction of vehicle dynamics.

$$\begin{aligned} a_t^{Ly*} =& (1-\eta(s_t))\pi_\theta(s_t) + \eta(s_t)\pi_{\theta-1}(s_t) - \\ & \lambda^*(s_t)\cdot g_{L_{\pi_{\theta-1}}}(s_t;\ \xi) \\ \lambda^*(s_t) =& \left[\frac{(1-\eta(s_t))g_{L_{\pi_{\theta-1}}}(s_t;\ \xi)^T(\pi_\theta(s_t) - \pi_{\theta-1}(s_t)) - \tilde{\varepsilon}(s_0)}{g_{L_{\pi_{\theta-1}}}(s_t;\ \xi)^T \cdot g_{L_{\pi_{\theta-1}}}(s_t;\ \xi)}\right]^+ \end{aligned} \tag{11}$$

(ii) foresight safeguard submodule

Inspired by the single-step safety layer in [20], we develop a foresight safeguard solution to handle complex driving problems with high dimension and high speed. For getting a timely response to unsafe actions, this module learns a full dynamics model with the power of deep neural network to endow policy foresight by multi-step prediction. On the one hand, once a danger is predicted in the multi-step future, the action from the Lyapunov-based submodule needs to be further corrected to avoid accidents timely. On the other hand, when safety is guaranteed, the foresight submodule checks and corrects an overly conservative action to avoid slow state changes due to very low speed. Specifically, let t is the current time step, $t+N$ is the max time step of prediction, $t+k, k \in [0,\ N]$ is the time step of a predicted danger or very low speed within the safety field, and $f(s,\ a;\ \omega)$ is the dynamics network with parameter ω. If n-step predictions do not violate safety constraint T_s or low limit of speed T_v, no correction is required.

Otherwise, we get an analytical solution of the positive-definite quadratic objective shown in (12) with the single active constraint by Lagrange multiplier and KKT conditions (the proof is similar to the Proposition 1 of [20]).

$$\begin{aligned} a_t^* &= \arg\min_{a_t} \tfrac{1}{2}\left\| a_t - \pi_\theta^{Ly}(s_t) \right\|^2 \\ s.t.\ & \hat{s}_{t+k} = f(s_{t+k-1},\ \pi_\theta^{Ly}(s_{t+k-1});\ \omega)\ \ 1 \le k \le N \\ & \qquad\qquad \hat{s}_{t+k}^i < 1 - T_s \ or\ \hat{s}_{t+k}^j > T_v \\ & \hat{a}_{t+k}^* = \pi_\theta^{Ly}(\hat{s}_{t+k}) - \lambda_{safe}^* g_i(\hat{s}_{t+k};\ \omega) - \lambda_{speed}^* g_j(\hat{s}_{t+k};\ \omega) \\ & a_t^* = \tfrac{\hat{a}_{t+k}^*}{k} \\ & \lambda_{safe}^* = \left[\frac{\hat{s}_{t+k+1}^i - (1-T_s)}{g_i(\hat{s}_{t+k}^i;\ \omega)^T g_i(\hat{s}_{t+k}^i;\ \omega)}\right]^+ \\ & \lambda_{speed}^* = \left[\frac{T_v - \hat{s}_{t+k+1}^j}{g_j(\hat{s}_{t+k}^j;\ \omega)^T g_j(\hat{s}_{t+k}^j;\ \omega)}\right]^+ \end{aligned} \tag{12}$$

where $\pi_\theta^{Ly}(s_t)$ is the action after Lyapunov-based correction, $g_i(s^i;\ \omega)$ and $g_j(s^j;\ \omega)$ are the action-gradient of the one-step safety and speed constraint prediction which are ith value s^i and jth value s^j in the state vector, respectively. $\hat{s}$ and $\hat{a}$ represent the predicted state and action. Our dynamics network has four fully connected (FC) layers and uses the shortcut connection structure of ResNet [32] as shown in Fig. 2. This network design can well represent the sensitivity of changes

in the action. The loss function is weighted MSE of vehicle's multiple sensors, as shown in (13).

$$L_{dynamics} = \frac{1}{M}\sum_{i=1}^{M}\sum_{n=1}^{N}\frac{k_n}{2}(s_{t+1}^n - f^n(s_t,\ a_t;\ \omega))^2 \quad (13)$$

where M is the batch size, N is the number of sensors, s^n and f^n represent the nth sensor from the sensor vector, k_n is the weighted coefficient.

V. SIMULATIONS AND DISCUSSION

A. Experimental setup

We conducted experiments on the open racing simulator (TORCS) [22], which is widely used for learning an intelligent vehicle because of its 3D visualization and a realistic physics engine. The state is a 29-dimension vector from multiple sensors consisting of an angle between agent direction and track axis, 3 speeds along the longitudinal, transverse, Z axis respectively, 19 distances between track edge and agent, a distance between agent and track axis, 4 wheel rotation speeds and an engine rotation speed. The action is a two-element vector [steering; acceleration] $\in [-1,\ 1]^2$ where steering and acceleration have continuous values. We train and test our method and baselines with a racing track named $street1$ which contains many sharp turns, even hairpin bends. Furthermore, we test these methods with five more complex tracks named $Aalborg$, $alpine1$, $alpine2$, $wheel2$, $etrack2$ to check the generalization of the learned policy in an unseen road. The total length of the five tracks is about 5.77 times the length of $street1$. And these tracks have smaller track widths, more varied curves than $street1$. Besides, the five tracks also add uphill and downhill sections. They are shown in Fig. 3.

To compare with the state-of-the-art works, we selected DDPG, DDPG + single-step safety layer from [20] (abbreviated as DDPG+sl) and DDPG + Lyapunov-based a-projection from [21] (abbreviated as DDPG+Ly) as baselines to verify the effectiveness of our work. The reason for this selection is these baselines inspire us (see details in RELATED WORK). Note that since the baselines in [20, 21] are not directly applied to the autonomous driving environment, we replace their original rewards with the race driving reward proposed in [3], as shown in (14). In addition, we use the ablation study to check how much each module of our two-stage safe RL algorithm contributes. We set four configurations: our two-stage method, only second learning stage (abbreviated as only stage2), first learning stage with Lyapunov-based safeguard submodule of the second stage (abbreviated as stage1+ Ly-stage2), and first learning stage with foresight safeguard submodule of the second stage (abbreviated as stage1+ Fs-stage2).

$$R = v_x(\cos\psi - |\sin\psi| - |\Delta dis_t|) \quad (14)$$

The following metrics are used to evaluate safety, efficiency and stability for the racing task. **No. of unsafety**: the total number of safety violations during training and testing, respectively. Notice that zero **No. of unsafety** only ensure there is no danger, but cannot indicate whether the task is completed or not. **No. of success**: the total number of completing the task. **Avg. speed (km/h)**: the average speed indicates how efficient the agent drives with learned policy. **Avg. angle (degree)**: the average angle indicates how stable the learned policy to relieve swing from side to side.

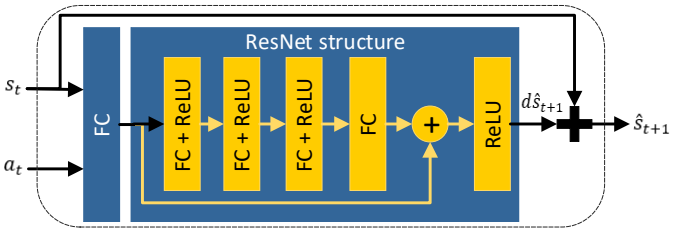


Fig. 2. Our dynamics network architecture

(a) (b)

Fig. 3. (a): The structures and physics of track $street1$ (left) and one of unseen tracks $Aalborg$ (right), (b): Besides $Aalborg$, the structures of the other four tracks ($alpine1, wheel2, alpine2, etrack2$ from left to right and top to bottom respectively)

For all experiments, the actor and critic network of DDPG have two FC hidden layers of (100, 100) and (500, 500) units, respectively. All FC layers use ReLU activations except for their output layers using tanh activations. The optimizers are Adam with learning rate 0.0001 for actor and 0.001 for critic. The size of replay buffer is 10^6. The discount rate is 0.95. The architecture of Lyapunov critic net is the same as the critic network. Its T_s is 0.9 to keep a small gap away from safety limitation. The dynamics model of foresight safeguard submodule uses a FC-ResNet structure shown in Fig. 2, and each FC layer has 350 units. Its optimizer is Adam with learning rate 0.0005. The batch size is 64. We set the max number of episodes and steps per episode to 2000 and 10000 respectively for training on a NVIDIA GTX 1080Ti GPU.

B. Results

(i) Comparison with baselines:

We compare our algorithm with three baselines mentioned above (DDPG, DDPG+Lyap, DDPG+sl) from five aspects: driving safety, efficiency, stability, generalization and policy convergence rate. The results of the first four aspects are shown in TABLE II. In terms of No. of unsafety recorded during training and testing, our algorithm achieves zero safety violation when learning from scratch, while all baselines fail to do this. The closed-form solutions for the safety constraint in DDPG+sl and DDPG+Ly help DRL converge to a safer policy than standard DDPG. As the safety layer of DDPG+sl is pre-trained before RL policy learning, it has much lower safety violations than DDPG+Ly , which learns both Lyapunov function and RL policy from scratch. Failing to guarantee safety in DDPG+sl is partly because short-sighted safety prediction is hard to deal with dangerous situations in time for complex high-speed racing tasks, and partly because the dynamics model of DDPG+sl with simple network structure have limited ability to give an action correction solution

exactly. The reasons for safety violations in DDPG+Ly can be attributed to unreliable Lyapunov function at the beginning of training and dissatisfaction of the assumption of the baseline feasible policy. In addition, one-step TD (as shown in (3)) for updating Lyapunov function is not foresighted enough.

Focusing on the second and third rows of TABLE II, DDPG without safety constraints has the lowest number of successes, and these unsuccessful episodes are all in danger. Thanks to pre-training and the rule-based safeguard submodule respectively, DDPG+sl and our algorithm are better learned in the same training episodes than DDPG+Ly. However, even in this case DDPG+Ly clearly shows an ability to ensure safety during unsuccessful episodes. From the perspective of policy performance, we observe that three methods of DDPG with safety constraints perform higher efficiency and stability than unconstrained DDPG, since these methods force the policy to explore safe actions that are more likely to get a high reward. Lyapunov-based safety solution in DDPG+Ly has a lower rate of safety violations, smaller swing angle than DDPG+sl. Although DDPG+Ly has a slight decrease in speed than DDPG+sl, it can obtain a more stable and smoother policy. And we further test the generalization of these methods on a mixture of five complex unseen tracks ($Aalborg$, $alpine1$, $alpine2$, $wheel2$, $etrack2$). This test task is considered a success only if the agent can safely drive through all five tracks. DDPG+sl, DDPG+Ly and our two-stage safe RL algorithm also perform better than standard DDPG. And we notice that the three methods of DDPG with safety constraints have an obvious decrease in Avg. speed to improve racing safety in new situations. DDPG+Ly always keeps the agent safe but performs poorly to complete the racing task. DDPG+sl has better generalization capability than DDPG+Ly, but the larger angle swing of single-step safety layer limits its ability to guarantee safety. Our method also works well due to the multi-step safeguard dynamics model.

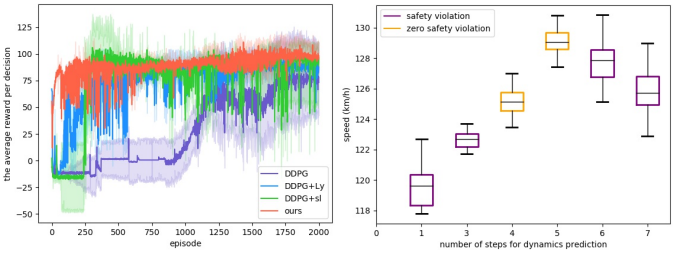
In addition, the comparison of policy convergence is shown in Fig. 4(a). We calculate the average reward per decision by (14). And the results show that DDPG with safety constraints helps policy convergence faster. The reward curve of DDPG+Ly shows that the policy steadily converges to the high-score solution. The curve of DDPG+sl presents a trend of ladder rising style. Our algorithm shows a faster and more stable convergence.

(ii) Ablation study

Our two-stage safe RL algorithm combines the advantages of a Lyapunov-based safety solution and a foresight safety solution with multi-step kinetics prediction to overcome baselines shortcomings and can function well for high-speed racing tasks. In this section, we configured ablation experiments to demonstrate the contributions of each algorithm component. The results are shown in TABLE III. By comparing the first two rows of TABLE III, we can see the first learning stage is proved to be important for achieving safety during learning, but it has no obvious effect on the performance of the final policy. The performance of policy mainly depends on the second learning stage. The last two rows of TABLE III demonstrate the contributions of the two data-driven safeguard submodules in the second learning stage.

TABLE II
COMPARISON WITH BASELINES

Methods		DDPG	DDPG+sl	DDPG+Ly	Ours
Training in street1 (2000 episodes)	***No. of unsafety***	1735	458	816	**0**
Testing in street1 (100 episodes)	***No. of unsafety***	30	5	6	**0**
	No. of success	70	92	84	**100**
	Avg.speed (km/h)	103.89	122.96	120	**127.07**
	Avg.angle (degree)	4.78	4.1	**3.86**	3.96
Testing in a mixture of five unseen tracks (100 episodes)	***No. of unsafety***	96	35	0	**0**
	No. of success	4	57	18	**74**
	Avg.speed (km/h)	99.58	107.84	106.29	**109.13**
	Avg.angle (degree)	4.71	4.9	**4.53**	4.65



(a) (b)

Fig. 4. (a): The average reward per decision during training, (b): The performance of the proposed two-stage safe RL algorithm with multi-step prediction.

The Lyapunov-based safeguard submodule in stage1+Ly-stage2 (defined in Section V.A) starts to work after the Lyapunov function has been trained well, but the RL policy at that moment may not be feasible because it may not satisfy safety. As a result, stage1+Ly-stage2 does not achieve zero safety violation due to the same shortcomings as DDPG+Ly. But Lyapunov-based policy is good at getting a stable policy. The foresight safeguard submodule in stage1+Fs-stage2 is started up after the dynamics model becomes reliable. It shows zero safety violation during testing, but still has few safety violations during training due to angle fluctuations brought by action correction. To further verify the impact of foresight on safety, we compare the performance of our two-stage safe RL algorithm under different foresight steps. As shown in Fig. 4(b), safety and efficiency increase with the number of prediction steps from $n = 1$ to $n = 5$. Especially, our method shows zero safety violation at $n = 4$ and 5. But due to cumulative approximate error of the dynamics model, we can also observe a declining trend from $n = 6$.

VI. CONCLUSIONS & FUTURE WORK

In this paper, we have presented a two-stage safe RL algorithm for autonomous racing tasks. The proposed algorithm combines the model-free RL with a safeguard mechanism and

TABLE III
ABLATION STUDY IN STREET1 TRACK

Methods	Training (2000 episodes)	Testing (100 episodes)			
	No. of unsafety	*No. of unsafety*	*No. of success*	*Avg.speed (km/h)*	*Avg.angle (degree)*
two-stage	0	0	100	127.07	3.96
only stage2	331	0	94	127.18	4.12
stage1 + Ly-stage2	506	0	92	123.73	3.97
stage1 + Fs-stage2	48	7	84	126.03	4.34

an adaptive reward function to guarantee safety during learning and accelerate policy convergence to an optimal solution. It has provided an analytical solution to ensure safety of training without elaborate expert fine design. We have evaluated it on racing simulator TORCS, and the experimental results indicated that our method (i) achieved 100% safety during both learning and execution; (ii) converged faster to a more efficient and stable policy than the most relevant baselines. (iii) showed generalization capability in an unseen and more complex racing task.

In our future work, we will extend this work to other model-free RL algorithms. Further, we plan to test the proposed algorithm in challenging roads with complex terrains, such as mountainous or rural areas.

REFERENCES

[1] Garcíia J, Fernández F, "A comprehensive survey on safe reinforcement learning," Journal of Machine Learning Research, vol.16, no.42, pp. 1437-1480, 2015.

[2] M. Jaritz, R. de Charette, M. Toromanoff, E. Perot and F. Nashashibi, "End-to-end race driving with deep reinforcement learning," in Proc. IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, 2018, pp. 2070-2075.

[3] Lau B, "Using keras and deep deterministic policy gradient to play torcs,"[Online]. Available: https://yanpanlau.github.io/2016/10/11/Torcs-Keras.html, 2016.

[4] Wang S, Jia D, Weng X, "Deep reinforcement learning for autonomous driving," arXiv preprint arXiv:1811.11329, 2018.

[5] P. Wolf, K. Kurzer, T. Wingert, F. Kuhnt and J. M. Zollner, "Adaptive behavior generation for autonomous driving using deep reinforcement learning with compact semantic states," in Proc. IEEE Intelligent Vehicles Symposium (IV), Changshu, 2018, pp. 993-1000.

[6] Shalev-Shwartz S, Shammah S, Shashua A, "Safe, multi-agent, reinforcement learning for autonomous driving," arXiv preprint arXiv:1610.03295, 2016.

[7] Mustafa Mukadam, Akansel Cosgun, Alireza Nakhaei, Kikuo Fujimura, "Tactical decision making for lane changing with deep reinforcement learning," in Proc.NIPS 2017 Workshop on Machine Learning for Intelligent Transportation Systems, 2017.

[8] Hu Y, Nakhaei A, Tomizuka M, Fujimura K, "Interaction-aware decision making with adaptive strategies under merging scenarios," in Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 2019, pp. 151-158.

[9] C. Paxton, V. Raman, G. D. Hager and M. Kobilarov, "Combining neural networks and tree search for task and motion planning in challenging environments," in Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, 2017, pp. 6059-6066.

[10] Alshiekh M, Bloem R, Ehlers R, Knighofer B, Niekum S and Topcu U, "Safe reinforcement learning via shielding," in Proc. AAAI Conference on Artificial Intelligence, New Orleans, 2018.

[11] D. Isele, A. Nakhaei and K. Fujimura, "Safe Reinforcement Learning on Autonomous Vehicles," in Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, 2018, pp. 1-6.

[12] Ye C, Ma H, Zhang X, Zhang K and You S, "Survival-oriented reinforcement learning model: An effcient and robust deep reinforcement learning algorithm for autonomous driving problem," in Proc. International Conference on Image and Graphics, Springer, Cham, 2017, pp. 417-429.

[13] S. Nageshrao, H. E. Tseng and D. Filev, "Autonomous Highway Driving using Deep Reinforcement Learning," in Proc. IEEE International Conference on Systems, Man and Cybernetics (SMC), Bari, Italy, 2019, pp. 2326-2331.

[14] Baheri A, Nageshrao S, Tseng H E, Kolmanovsky I, Girard A, Filev D. "Deep Reinforcement Learning with Enhanced Safety for Autonomous Highway Driving," arXiv preprint arXiv:1910.12905, 2019.

[15] Altman Eitan, "Constrained Markov Decision Processes," CRC Press, Florida, 1999, pp. 260.

[16] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel, "Constrained policy optimization," in Proc. International Conference on Machine Learning, Sydney, 2017, pp. 22-31.

[17] Wen M, Topcu U, "Constrained cross-entropy method for safe reinforcement learning," in Proc. International Conference on Neural Information Processing Systems(NeurIPS), 2018, pp. 7450-7460.

[18] Yang T Y, Rosca J, Narasimhan K, Ramadge P J, "Projection-Based Constrained Policy Optimization," presented at International Conference on Learning Representations (ICLR), Ethiopia, 2020.

[19] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz, "Trust region policy optimization," in Proc. International Conference on Machine Learning (ICML), 2015, pp. 1889-1897.

[20] Gal Dalal, Krishnamurthy Dvijotham, Matej Vecerik, Todd Hester, Cosmin Paduraru, and Yuval Tassa, "Safe exploration in continuous action spaces," arXiv preprint arXiv:1801.08757, 2018.

[21] Chow Y, Nachum O, Faust A, Duez-Guzmn E, Ghavamzadeh M, "Lyapunov-based Safe Policy Optimization for Continuous Control," arXiv preprint arXiv:1901.10031, 2019.

[22] Loiacono D, Cardamone L, Lanzi P L. "Simulated car racing championship: Competition software manual," arXiv preprint arXiv:1304.1672, 2013.

[23] Hassan K Khalil, "Noninear systems," Prentice-Hall, New Jersey, 2(5):5-1, 1996.

[24] Chow Y, Nachum O, Duenez-Guzman E, Ghavamzadeh M, "A lyapunov-based approach to safe reinforcement learning," in Proc. International Conference on Neural Information Processing Systems(NeurIPS), 2018, pp. 8092-8101.

[25] Lillicrap T P, Hunt J J, Pritzel A, Heess N, Erez T, Tassa Y, et al. "Continuous control with deep reinforcement learning," arXiv preprint arXiv:1509.02971, 2015.

[26] J. Kong, M. Pfeiffer, G. Schildbach and F. Borrelli, "Kinematic and dynamic vehicle models for autonomous driving control design," in Proc. IEEE Intelligent Vehicles Symposium (IV), Seoul, 2015, pp. 1094-1099.

[27] Z. Xu, C. Tang and M. Tomizuka, "Zero-shot Deep Reinforcement Learning Driving Policy Transfer for Autonomous Vehicles based on Robust Control," in Proc. International Conference on Intelligent Transportation Systems (ITSC), Maui, HI, 2018, pp. 2865-2871.

[28] De Iaco R, Smith S L, Czarnecki K, "Universally Safe Swerve Manoeuvres for Autonomous Driving," arXiv preprint arXiv:2001.11159, 2020.

[29] M. Toromanoff, E. Wirbel, F. Wilhelm, C. Vejarano, X. Perrotton and F. Moutarde, "End to End Vehicle Lateral Control Using a Single Fisheye Camera," in Proc. IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, 2018, pp. 3613-3619.

[30] Y. Nager, A. Censi and E. Frazzoli, "What lies in the shadows? Safe and computation-aware motion planning for autonomous vehicles using intent-aware dynamic shadow regions," in Proc. IEEE International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 2019, pp. 5800-5806.

[31] P. Polack, F. Altch, B. d'Andra-Novel and A. de La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?," in Proc. IEEE Intelligent Vehicles Symposium (IV), Los Angeles, CA, 2017, pp. 812-818.

[32] K. He, X. Zhang, S. Ren and J. Sun, "Deep Residual Learning for Image Recognition," in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 770-778.