Model-Based Actor-Critic with Chance Constraint for Stochastic System

Baiyu Peng[1], Yao Mu[1], Yang Guan[1], Shengbo Eben Li[1*], Yuming Yin[1], Jianyu Chen[2]

***Abstract*— Safety is essential for reinforcement learning (RL) applied in real-world situations. Chance constraints are suitable to represent the safety requirements in stochastic systems. Previous chance-constrained RL methods usually have a low convergence rate, or only learn a conservative policy. In this paper, we propose a model-based chance constrained actor-critic (CCAC) algorithm which can efficiently learn a safe and non-conservative policy. Different from existing methods that optimize a conservative lower bound, CCAC directly solves the original chance constrained problems, where the objective function and safe probability is simultaneously optimized with adaptive weights. In order to improve the convergence rate, CCAC utilizes the gradient of dynamic model to accelerate policy optimization. The effectiveness of CCAC is demonstrated by a stochastic car-following task. Experiments indicate that compared with previous RL methods, CCAC improves the performance while guaranteeing safety, with a five times faster convergence rate. It also has 100 times higher online computation efficiency than traditional safety techniques such as stochastic model predictive control.**

I. Introduction

Recent advances in deep reinforcement learning (RL) have demonstrated state-of-the-art performance on a broad set of tasks, including Atari games [1], StarCraft [2] and Go [3]. However, these works do not consider safety since they are usually applied in virtual games. In many real-world tasks such as autonomous driving and unmanned aerial vehicles, the agent should follow some safety rules besides achieving excellent performance. For instance, an autonomous car driving on the highway, while optimizing its velocity, must not take actions that may cause a crash with the surrounding car. Usually, it is nontrivial to learn a driving policy that is both efficient and safe. [4].

The safety consideration has taken different forms in the safe RL community [5], [6]. Tamar (2013) treated safety in a robust view and optimized the worst-case performance of the agent [7]. Chow (2017) used value-at-risk as a metric of safety and a policy was regarded safe if its value-at-risk was high enough [8]. Recently, many researchers also cast safety in the context of Constrained MDPs, where the cumulative cost was constrained below a given threshold [9]–[11]. However, these criteria all focus on reward-related or cost-related measures, and they still lack a direct connection with safety [12]. In other words, given a learned policy with a certain value-at-risk, it is still hard to evaluate how safe the policy is. Indeed, an explicit safety constraint is preferred in real-world applications [13], [14]. In this work, we aim to build a safe policy optimization framework, which can quantitatively constrain the possibility of the control policy violating the state constraint. It should be stressed that plenty of real-world systems are stochastic in nature, and thus the state constraint only holds in a probability form, which is quite different from the hard constraint in deterministic systems. For example, in the case of an unmanned aerial vehicle, the direction and force of wind are uncertain. Thus it can only keep balance at a high probability. Specifically, the state constraint in such a probability form is briefly called chance constraint.

Strategies used to solve the chance constrained reinforcement learning problems can be roughly categorized into two approaches. The first and the most common solution is to add a fixed-weight penalty term to the reward function so as to prevent agents from entering the dangerous states [15], [16]. Although this approach is very straightforward and simple to implement, it requires the penalty weight to strike a balance between safety and performance correctly. Unfortunately, it is usually difficult to select an appropriate fixed-weight. Especially, a large penalty is prone to converge to sub-optimal solutions, while a small penalty is unable to satisfy the constraint [10]. The second approach constrains the lower bound of safe probability to the required threshold, which can be solved through dynamic programming method [17] or model-free primal-dual (MF-PD) method [18], [19]. Nevertheless, the dynamic programming method only works in discrete state and action space, which can not be applied to continuous problems. The model-free primal-dual method is purely data-driven, which leads to high variance and low convergence rate. Moreover, constraining the lower bound of safe probability may produce a policy whose real safe probability is significantly higher than the required threshold, i.e, introduces large conservatism. As shown in our experiments, the learned policy achieves 99% safe probability even when the required threshold is only 90%, and thus influences the performance.

To overcome the aforementioned challenges, this paper proposes a model-based algorithm named chance constrained actor-critic (CCAC). Instead of constraining the lower bound of safe probability like MF-PD, CCAC directly solves the original chance constrained problems through the exterior point methods. In order to improve the convergence rate, the gradient of dynamic model is utilized to guide policy optimization. Finally, CCAC is compared with two RL methods and two traditional safety techniques such as stochastic model predictive control to demonstrate its

This study is supported by Tsinghua-Toyota Joint Research Institute Cross-discipline Program and Xilinx.

[1]State Key Laboratory of Automotive Safety and Energy, School of Vehicle and Mobility, Tsinghua University, Beijing, China * Corresponding author lishbo@tsinghua.edu.cn

[2]Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China jianyuchen2020@163.com

superior performance. The contributions of this paper are as follows,

1) a direct approach to solve the chance constrained problems, rather than indirectly solving by constraining the lower bound of safe probability.
2) a model-based framework of policy optimization for chance constrained problems, where the gradient of the dynamic model is used to accelerate training process.

The rest of this paper is organized as follows. The chance constrained RL problem is formulated in Section II. The CCAC algorithm is proposed in Section III. The effectiveness of CCAC is illustrated by a stochastic car-following task in Section IV. Section V concludes this paper.

II. Preliminary

For a discrete-time stochastic system, the dynamics with the chance constraint is mathematically described as:

$$\begin{aligned} &s_{t+1} = f(s_t, a_t, \xi_t), \quad \xi_t \sim p(\xi_t), \\ &\Pr\left\{\bigcap_{i=1}^{N} \left[h\left(s_{t+i}\right) < 0\right]\right\} \geq 1-\delta \end{aligned} \tag{1}$$

where t is the current time step, $s_t \in \mathcal{S}$ is the state, $a_t \in \mathcal{A}$ is the action, $f(\cdot,\cdot,\cdot)$ is the environmental dynamics, $\xi_t \in \mathbb{R}^n$ is the uncertainty following an independent and identical distribution $p(\xi_t)$, $h(\cdot)$ is the state constraint function and the set $\{s \mid h(s) < 0\}$ defines a safe region in which the agent should remain. We do not make assumptions about the form of $f(\cdot,\cdot,\cdot)$ and $h(\cdot)$, i.e., they can be linear or nonlinear. The safety constraint takes form of a joint chance constraint with $1-\delta$ as the required threshold. Such a form is extensively used in stochastic systems control [20]. Intuitively, it can be interpreted as the probability of agent staying within a safe region $\{s \mid h(s) < 0\}$ over the horizon N is at least $1-\delta$. For simplicity, we only consider one constraint.

The objective of chance constrained RL problems is to maximize the expectation of cumulative reward J_r, while constraining the safe probability p_s:

$$\begin{aligned} &\max_{\pi} J_r\left(\pi\right) = \mathbb{E}_{s_0,\xi}\left\{\sum_{t=0}^{\infty} \gamma^t r\left(s_t, a_t\right)\right\} \\ &\text{s.t. } p_s(\pi) = \Pr\left\{\bigcap_{t=1}^{N} \left[h\left(s_t\right) < 0\right]\right\} \geq 1-\delta \end{aligned} \tag{2}$$

where π is the policy, $r(\cdot,\cdot)$ is the reward function, $0 < \gamma < 1$ is the discounting factor and $\mathbb{E}_{s_0,\xi}(\cdot)$ is the expectation w.r.t. the initial state s_0 and uncertainty ξ. Specifically, the policy is a deterministic mapping from state space $\mathcal{S}$ to action space $\mathcal{A}$ with parameters $\theta : a_t = \pi(s_t;\theta)$.

The joint chance constraint in (2) is generally nonconvex and intractable [21]. Therefore, previous methods like model-free primal-dual (MF-PD) usually solve the chance constraint indirectly, i.e, derive a lower bound of the joint probability p_s through the Boole's inequality and turn to constrain this lower bound [17], [18]. More specifically, a cost function $c(s_t, a_t, s_{t+1})$ and the expected cumulative cost J_c are defined as:

$$c(s_t, a_t, s_{t+1}) = \begin{cases} 0 & h(s_{t+1}) < 0 \\ 1 & h(s_{t+1}) \geq 0 \end{cases} \tag{3}$$

$$J_c(\pi) = \mathbb{E}_{s_0,\xi}\left\{\sum_{t=0}^{N-1} c(s_t, a_t, s_{t+1})\right\} \tag{4}$$

Given the Boole's inequality, a lower bound of p_s is derived as:

$$\begin{aligned} p_s(\pi) &= \Pr\left\{\bigcap_{t=1}^{N} \left[h\left(s_t\right) < 0\right]\right\} \\ &\geq 1 - \sum_{t=1}^{N} \Pr\left\{h\left(s_t\right) \geq 0\right\} \\ &= 1 - J_c(\pi) \end{aligned} \tag{5}$$

The original chance constraint in (2) is indirectly imposed by constraining its lower bound:

$$J_c(\pi) \leq \delta \tag{6}$$

Many previous methods (e.g. MF-PD) adopt above reformation because J_c has similar additive structure as the objective function J_r, making it easier to impose the constraints in the RL framework. However, constraining the lower bound may lead to a policy whose real safe probability is significantly higher than the required threshold, i.e., introduces conservatism, as our experiments show in section IV. This problem is also a main challenge of a class of existing methods.

III. Chance Constrained Actor-Critic Algorithm

Different from previous methods which constrain the lower bound of safe probability, we propose a model-based approach to directly solve the original chance constrained problem (2) with less conservatism. Besides, our method also takes use of the gradient of dynamic model to accelerate the training process.

A. Constrained Policy Optimization via Exterior Point Methods

The adopted approach follows the idea of exterior point methods, which are extensively used in constrained optimization area [22]. The exterior point methods put the chance constraint into a large and increasing penalty term in the objective function in k-th iteration:

$$\max_{\pi_k} J_{EP}\left(\pi_k\right) = J_r\left(\pi_k\right) - \frac{1}{2} b_k \max(1 - \delta - p_s\left(\pi_k\right), 0)^2 \tag{7}$$

where $b_k \gg 0$ is the penalty factor and $\{b_k\}$ is a given monotone increasing sequence. Intuitively, the exterior point methods penalize the violation of constraint as shown in Fig. 1. As b_k increases, the penalty will become tremendous, pushing π_k to the feasible region. Although the intermediate policy may be infeasible, the convergent policy will be feasible. This is also the reason why it is called exterior point method.

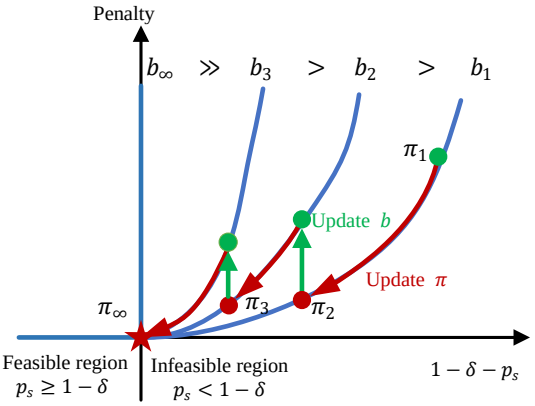


Fig. 1. Exterior point methods.

The chance constrained RL problem (2) is solved by iteratively updating π_k and b_k as shown in Fig. 1. However, in practice, the cost of solving π_k in every iteration until convergence is computationally prohibitive, and an alternative is to replace the maximization by a gradient ascent step

$$\theta^{k+1} = \theta^k + \alpha_\theta \nabla_\theta J_{EP} \tag{8}$$

where $\alpha_\theta > 0$ is the learning rate of policy and the policy gradient $\nabla_\theta J_{EP}$ is derived as

$$\nabla_\theta J_{EP} = \begin{cases} \nabla_\theta J_r & p_s \geq 1-\delta \\ \nabla_\theta J_r + b_k(1-\delta-p_s)\nabla_\theta p_s & p_s < 1-\delta \end{cases} \tag{9}$$

In order to compute the above gradient, we have to obtain the current safe probability p_s and its gradient $\nabla_\theta p_s$. Thanks to the available model, the safe probability p_s can be easily estimated by sampling large numbers of trajectories. Specifically, we rollout M trajectories with policy π. Suppose there are m safe trajectories, then the safety probability is estimated by $p_s \approx \frac{m}{M}$. Note that this rollout procedure will not impose extra computation burden since these trajectories are also necessary for the update of actor-critic as we will discuss in III-B. Unfortunately, the gradient $\nabla_\theta p_s$ is still hard to obtain, which is also a key difficulty in solving the chance constrained problems. One possible solution is to find a substitute ascent direction to replace $\nabla_\theta p_s$. Inspired by the inequality $p_s \geq 1 - J_c$ as shown in (5), a decreasing J_c will push p_s to the ascent direction, so we replace $\nabla_\theta p_s$ with $-\nabla_\theta J_c$. Consequently, the new proxy policy gradient becomes:

$$\nabla_\theta J_{PR} = \begin{cases} \nabla_\theta J_r & p_s \geq 1-\delta \\ \nabla_\theta J_r - b_k(1-\delta-p_s)\nabla_\theta J_c & p_s < 1-\delta \end{cases} \tag{10}$$

This policy gradient can be intuitively interpreted as follows. In order to solve the chance constrained problem (2), we simultaneously optimize the cumulative reward and the safe probability by gradient ascent. To balance the two objectives, the weight of $\nabla_\theta J_c$ is adaptively adjusted according to the current safe probability. We stress that CCAC is essentially different from previous methods which only constrain the lower bound. In CCAC, as long as the chance constraint $p_s \geq 1-\delta$ is satisfied, the weight of $\nabla_\theta J_c$ becomes zero and the safe probability will not be optimized. While previous methods keep optimizing p_s until $J_c \leq \delta$, even when $p_s \geq 1-\delta$ is already satisfied. That is the underlying reason why CCAC is not conservative as previous methods. Finally, in practice, since the weight $b_k(1-\delta-p_s)$ in (10) may become excessively large, we instead use the relative weights between $\nabla_\theta J_r$ and $\nabla_\theta J_c$.

The proposed algorithm CCAC is summarized in Algorithm 1 and Fig. 2.

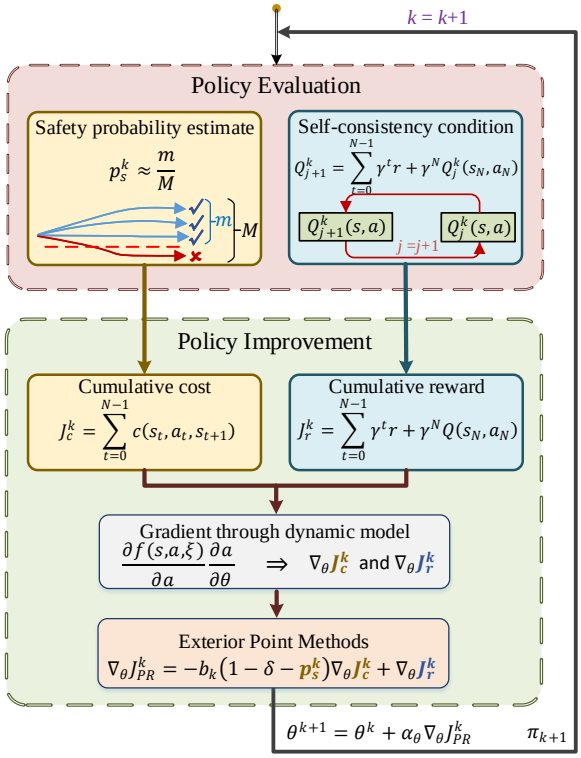


Fig. 2. The framework of CCAC algorithm.

B. Model-based Actor-Critic with Parameterized Functions

In this subsection the main focus is on how to learn the policy and state-action values in the model-based actor-critic framework with parameterized functions. Importantly, the gradient of dynamic model will be utilized to attain an accurate ascent direction and thus improve the convergence rate [23], [24]. For an agent behaving according to policy π, the values of the state-action pair (s,a) are defined as follows:

$$Q^\pi(s,a) = \mathbb{E}_\xi \left\{ \sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \,\middle|\, s_0 = s, a_0 = a \right\} \tag{11}$$

Algorithm 1 CCAC algorithm

Initialize s_0, b_0, $k=0$
repeat
  Rollout M trajectories by N steps via dynamic model
  Estimate safe probability through trajectories
    $p_s \leftarrow \frac{m}{M}$
  Update critic according to (15):
    $\omega^{k+1} \leftarrow \omega^k + \alpha_\omega \nabla_\omega J_Q$
  Update actor according to (16):
    $\theta^{k+1} \leftarrow \theta^k + \alpha_\theta \nabla_\theta J_{PR}$
    $\nabla_\theta J_{PR} = \nabla_\theta J_r - \max(1-\delta-p_s, 0) b_k \nabla_\theta J_c$
  Update penalty factor b_k
  $k \leftarrow k+1$
until $|Q^{k+1} - Q^k| \le \zeta$ and $|\pi^{k+1} - \pi^k| \le \zeta$

The expected cumulative reward J_r can be expressed as a N-step form:

$$J_r(\pi) = \mathbb{E}_{s_0,\xi}\left\{\sum_{t=0}^{N-1} \gamma^t r\left(s_t, a_t\right) + \gamma^N Q^\pi(s_N, a_N)\right\} \quad (12)$$

For large and continuous state spaces, both value function and policy are parameterized, as shown in (13). The parameterized state-action value function with parameter w is usually named the "critic", and the parameterized policy with parameter θ is named the "actor" [23].

$$Q(s,a) \cong Q(s,a;w), \quad a \cong \pi(s;\theta) \quad (13)$$

The parameterized critic is trained by minimizing the average square error:

$$J_Q = \mathbb{E}_{s_0,\xi}\left\{\frac{1}{2}\left(Q_{\text{target}} - Q(s_0,a_0;w^k)\right)^2\right\} \quad (14)$$

where $Q_{\text{target}} = \sum_{t=0}^{N-1} \gamma^t r\left(s_t,a_t\right) + \gamma^N Q\left(s_N, a_N; w^k\right)$ is the N-step target. Note that the rollout length N is equal to the horizon of chance constraint. The semi-gradient of the critic is

$$\nabla_\omega J_Q = \mathbb{E}_{x_0,\xi}\left\{\left(Q(s_0,a_0;w^k) - Q_{\text{target}}\right)\frac{\partial Q(s_0,a_0;w^k)}{\partial w}\right\} \quad (15)$$

As discussed in III-A, the parameterized actor aims to maximize J_{EP} via gradient ascent. The proxy gradient $\nabla_\theta J_{PR}$ is composed of $\nabla_\theta J_r$ and $\nabla_\theta J_c$, which are computed via backpropagation though time. Denoting $\frac{\partial s_t}{\partial \theta}$ as ϕ_t, $\frac{\partial a_t}{\partial \theta}$ as ψ_t, then $\nabla_\theta J_r$ is derived as:

$$\begin{aligned}\nabla_\theta J_r = &\mathbb{E}_{s_0,\xi}\left\{\sum_{t=0}^{N-1}\gamma^t\left[\frac{\partial r(s_t,a_t)}{\partial s_t}\phi_t + \frac{\partial r(s_t,a_t)}{\partial a_t}\psi_t\right]\right.\\ &\left.+\gamma^N\left[\frac{\partial Q(s_N,a_N)}{\partial s_N}\phi_N + \frac{\partial Q(s_N,a_N)}{\partial a_N}\psi_N\right]\right\}\end{aligned} \quad (16)$$

where

$$\phi_{t+1} = \phi_t \frac{\partial f(s_t,a_t,\xi_t)}{\partial s_t} + \psi_t \frac{\partial f(s_t,a_t,\xi_t)}{\partial a_t}$$

with $\phi_0 = 0$, and

$$\psi_t = \phi_t \frac{\partial \pi(s_t;\theta)}{\partial s_t} + \nabla_\theta \pi(s_t;\theta)$$

The gradient $\nabla_\theta J_c$ can be derived similar to (16). Considering that $c\left(s,a,s'\right)$ is an indicator function with zero gradient, it is replaced by the sigmoid function:

$$c(s,a,s') = \text{sigmoid}(\eta h(s')) \quad (17)$$

where $\eta > 0$ is a scale factor. The benefits of calculating $\nabla_\theta J_r$ and $\nabla_\theta J_c$ in the model-based framework is that the gradients of first N steps' reward are computed analytically through the dynamic model. In contrast, model-free methods can not obtain these analytical gradients and thus only relies on the value function, which is usually inaccurate with high variance. In a word, the model-based framework achieves a faster convergence rate due to a more accurate gradient [24]. The convergence and optimality of model-based actor-critic framework have been well-studied in [25].

IV. Numerical Experiment

A. Experiment Setup

In this section the proposed CCAC is applied to a stochastic car-following scenario as shown in Fig. 3, where the ego car expects to drive closely with the front car to reduce wind drag, while keeping a minimum gap between the two cars. Concretely, the ego car and front car follow the kinematics model, where the front car is assumed to drive at a constant speed v_f but its location x_f' is varying with uncertainty (e.g., due to the varying of road grade, wind drag). The minimum gap between the two cars is required to be kept at a high probability.

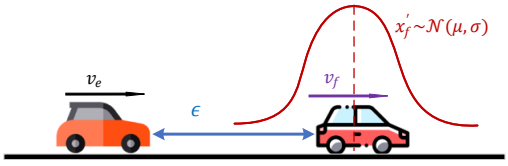
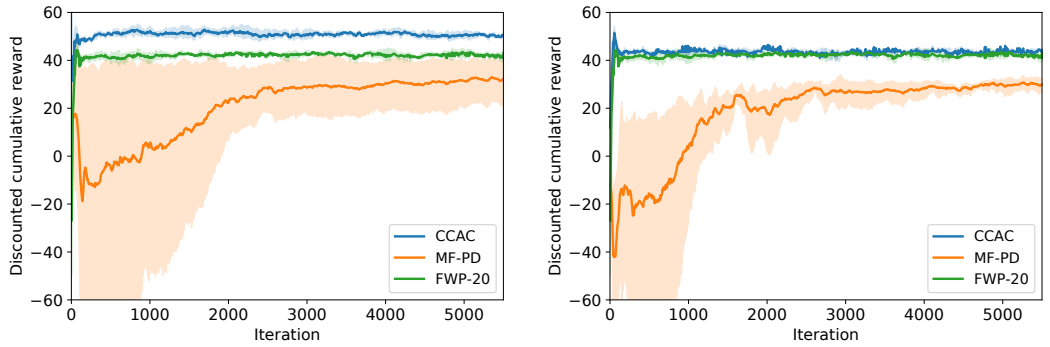


Fig. 3. Car-following scenario.

The discrete-time stochastic system is described by

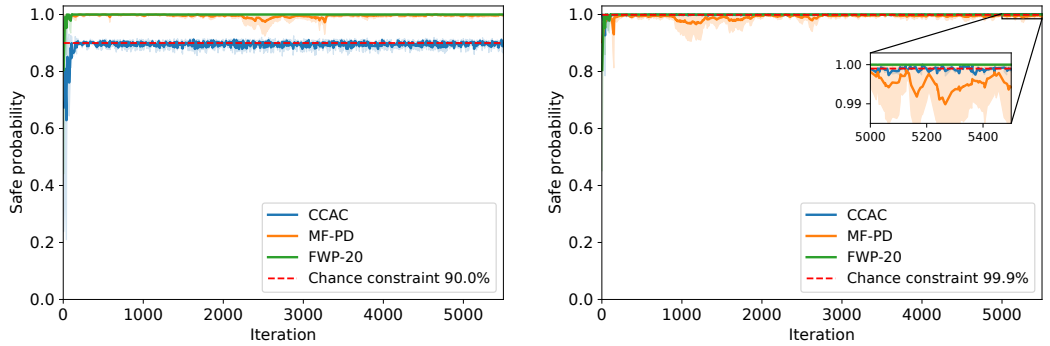
$$\begin{aligned}&s_{t+1} = As_t + Ba_t + D\xi_t\\ &A = \begin{bmatrix}1 & 0 & 0\\ 0 & 1 & 0\\ -T & T & 1\end{bmatrix}\\ &B = [T,0,0]^\top, \quad D = [0,0,T]^\top\end{aligned} \quad (18)$$

The system state vector is $s = [v_e \quad v_f \quad \epsilon]^\top$, where v_e denotes the velocity of ego car, v_f is the velocity of front car, and ϵ is the gap between the two cars. The action $a_t \in (-4,3)$ is the acceleration of ego car, and the disturbance $\xi_t \sim \mathcal{N}(0,1)$ is truncated in the interval $(-5,5)$. $T = 0.1$ is



(a) Discounted cumulative reward under 90.0% threshold

(b) Discounted cumulative reward under 99.9% threshold



(c) Safe probability under 90.0% threshold

(d) Safe probability under 99.9% threshold

Fig. 4. Comparison of training process among CCAC (chance constrained actor-critic), MF-PD (model-free primal-dual method) and FWP-20 (penalty with fixed weight 20).

the simulation time step. The chance constrained RL problem is defined as

$$\begin{aligned} &\max_{\pi} \mathbb{E}_{s_0,\xi}\left\{\sum_{t=0}^{\infty}\gamma^t(0.2v_{e,t}-0.05\epsilon_t)\right\} \\ &\text{s.t. } \Pr\left\{\bigcap_{t=1}^{N}[\epsilon_t>2]\right\}\geq 1-\delta \end{aligned} \tag{19}$$

where $v_{e,t}$ denotes the ego car velocity at time step t. In this setting, the agent is expected to drive fast and close to frontal car while keeping a minimum gap of 2m.

B. Implementation Details

We implement CCAC algorithm on the problem above. Our parameterized actor and critic are both fully-connected neural networks. Each network has two hidden layers using rectified linear unit (ReLU) as activation functions, with 64 units per layer. The main hyper-parameters of the algorithm are listed in Table I. The penalty factor in (10) is set as $b_k = \min(1000 * 1.01^k, 10000)$.

C. Comparison Baselines

To demonstrate the advantages of CCAC, the performance is compared with model-free primal-dual method (MF-PD) [19], model-based fixed-weight penalty method (FWP), and two traditional safety techniques, i.e., stochastic model predictive control (SMPC) and safety shielding. The five methods are evaluated under two chance constraint thresholds 90.0% and 99.9%, i.e., $\delta = 0.1$ and $\delta = 0.001$. For FWP method, we test it with different penalty weights in advance and choose a large weight 20 for 99.9% threshold. Unfortunately, we find it is very hard to select an appropriate weight which produces a 90.0% safe policy. A small decline of the weight will cause a drastic decline of the safe probability. Therefore, we just choose a weight of 10 although it is relatively unsafe. For simplicity, FWP with weight 10 and 20 are shortly labeled as FWP-10 and FWP-20, respectively. The SMPC method adopts the stochastic tube approach to find the feasible actions under uncertainty [26]. The safety shielding method projects the action produced by an unconstrained policy network into the safe action region by solving a constrained optimization problems [27]. To handle the joint chance constraint, Boole's inequality is also used in both SMPC and shielding.

TABLE I

HYPER-PARAMETERS

Parameters	Symbol	Value
trajectories number	M	8192
constraint horizon	N	80
discounting factor	γ	0.98
actor learning rate	α_θ	36e-5→ 2e-5
critic learning rate	α_ω	2e-4
scale factor	η	10

D. Evaluation Results

We will analyse the five methods from the aspects of convergence rate, asymptotic performance and computation time.

TABLE II

ASYMPTOTIC PERFORMANCE UNDER 90.0% CHANCE CONSTRAINT

	Safe probability	Discounted cumulative reward
CCAC	90.28%	50.17
MF-PD	99.72%	31.84
FWP-10	0.18%	196.10
FWP-20	100.00%	43.07
SMPC	99.41%	43.50
shielding	91.84%	42.67

TABLE III

ASYMPTOTIC PERFORMANCE UNDER 99.9% CHANCE CONSTRAINT

	Safe probability	Discounted cumulative reward
CCAC	99.88%	44.04
MF-PD	99.43%	29.89
FWP-10	0.18%	196.10
FWP-20	100.00%	43.07
SMPC	99.95%	42.15
shielding	91.89%	42.65

1) Convergence rate: For three RL methods (i.e., CCAC, MF-PD and FWP), the learning curves of discounted cumulative reward and joint safe probability in horizon N are plotted in Fig. 4. Each curve is averaged over five independent experiments. Besides, the curve of FWP-10 is omitted since it wins unreasonable reward by greatly sacrificing safety. In Fig. 4(a), we notice that model-free algorithm MF-PD converges at about 5000 iterations in terms of reward. Contrarily, the model-based algorithms CCAC and FWP learns at least five times faster and converge within 1000 iterations. Besides, their variances are dramatically lower than MF-PD. These observations confirm the advantage of the utilization of the analytical gradient given by the dynamic model to accelerate and steady the training process.

2) Asymptotic performance: The comparisons of asymptotic performance under two chance constraint thresholds are summarized in Table II and Table III. The proposed CCAC achieves the highest discounted cumulative reward in both constraint thresholds. MF-PD exhibits large conservatism and attains fewer rewards. Especially, MF-PD learns a policy with 99.72% safe probability even when the required threshold is only 90.00%. The root cause of conservatism is that MF-PD imposes the chance constraint indirectly by constraining the lower bound of safe probability. On the contrary, CCAC directly solves the original chance constraint and learns an optimal safe policies with less conservatism. The FWP-20 method achieves good performance in terms of both safety and reward in 99.9% threshold, but FWP-10 fails in 90.0% threshold. The traditional safety techniques SMPC and shielding achieve overall good performance, and SMPC is safer than shielding method. But similar to MF-PD, the use of Boole's inequality introduces conservatism, which makes them win fewer rewards than CCAC in 90.0% threshold.

Additionally, one may question that the safe probability of CCAC is slightly lower than the threshold in some iterations. We argue that for chance constrained problems, the chance constraint threshold is just a measurement of the safety level, instead of a physical quantity. Therefore, it will not cause huge difference if the safe probability is slightly below the threshold. In our experiment shown in Fig. 4(d), the fluctuation range around the constraint threshold is only within 0.5%. Actually, since the safe probability is estimated by Monte Carlo simulation with a parameterized policy network, such a fluctuation of safe probability is inevitable.

To give an intuitive comparison of five methods, we implement them under 90.0% threshold on the same initial state, i.e., $v_e = 5, v_f = 6$ and $\epsilon = 6$. Fig. 5 demonstrates the curve of car-following gap ϵ averaged on twenty independent simulations. CCAC keeps the minimum gap while maintaining safety. In contrast, MF-PD is more conservative and retains a large gap from the front car. FWP, SMPC and shielding methods achieve intermediate car-following gaps.

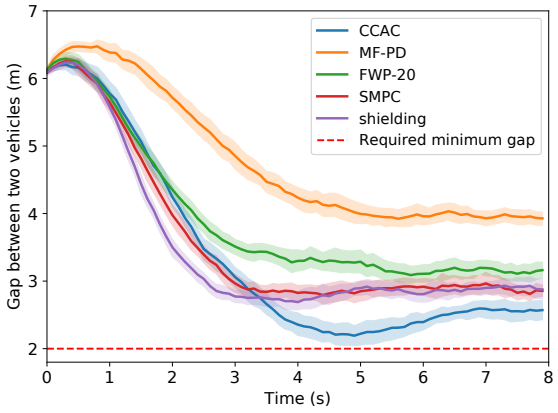


Fig. 5. Simulations of car-following gap among CCAC (chance constrained actor-critic), MF-PD (model-free primal-dual), FWP-20 (penalty with fixed weight 20), SMPC (stochastic model predictive control) and shielding.

3) Online computation efficiency: In some real-world applications like autonomous driving, online computation efficiency is also essential. The computation efficiency is measured by average one-step computation time, i.e., given a certain state, the average time the controller (or neural networks for three RL methods) takes to compute the action. Table IV summarizes the results of five methods, where the optimization package for SMPC and shielding is Ipopt [28]. Benefitting from the parameterized policy networks, CCAC, MF-PD and FWP achieve a dramatically fast computation speed since they only involve the forward propagation of neural networks. However, traditional safety techniques like SMPC and shielding have to solve the constrained optimization problems online. Thus their computation efficiency is nearly 100 times lower. These results indicate the remarkable advantages of CCAC over traditional safety

techniques and confirm the promising potential of CCAC in real-world tasks.

TABLE IV
COMPARISON OF ONE-STEP COMPUTATION TIME

Methods	CCAC	MF-PD	FWP	SMPC	shielding
time (ms)	0.051	0.057	0.051	8.168	4.936

In summary, CCAC succeeds in learning a safe but not conservative policy, with a five times faster convergence rate than existing chance-constrained RL approaches. It also has 100 times higher online computation efficiency than some traditional safety methods.

V. CONCLUSION

This paper proposed a model-based RL algorithm CCAC applied to safety-critical stochastic systems. Instead of constraining the lower bound of safe probability like previous methods, CCAC directly solved the original chance constraint and thus avoided conservatism. Besides, CCAC significantly improved the convergence rate by using the gradient of dynamic model. The benefits of CCAC were demonstrated in simulations of a stochastic car-following task, where it achieved high reward while satisfying the chance constraint. Additionally, CCAC also had five times faster convergence rate than a model-free method and 100 times higher online computation efficiency than traditional safety methods. The application of CCAC to more general environmental dynamics will be investigated in the future.

REFERENCES

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, and M. G. B. et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, 2015.

[2] O. Vinyals, I. Babuschkin, W. Czarnecki, M. Mathieu, A. Dudzik, and J. C. et al., "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, pp. 1–5, 2019.

[3] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, and A. G. et al., "Mastering the game of go without human knowledge," *Nature*, vol. 550, pp. 354–359, 2017.

[4] S. E. Li, X. Hu, K. Li, and C. Ahn, "Mechanism of vehicular periodic operation for optimal fuel economy in free-driving scenarios," *Iet Intelligent Transport Systems*, vol. 9, pp. 306–313, 2015.

[5] J. Garcıa and F. Fernández, "A comprehensive survey on safe reinforcement learning," *Journal of Machine Learning Research*, vol. 16, no. 1, pp. 1437–1480, 2015.

[6] G. Dulac-Arnold, D. J. Mankowitz, and T. Hester, "Challenges of real-world reinforcement learning," *ArXiv*, vol. abs/1904.12901, 2019.

[7] A. Tamar and S. Mannor, "Variance adjusted actor critic algorithms," *ArXiv*, vol. abs/1310.3697, 2013.

[8] 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.

[9] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *International Conference on Machine Learning*, pp. 22–31, PMLR, 2017.

[10] C. Tessler, D. J. Mankowitz, and S. Mannor, "Reward constrained policy optimization," in *International Conference on Learning Representations*, 2018.

[11] T.-Y. Yang, J. Rosca, K. Narasimhan, and P. J. Ramadge, "Projection-based constrained policy optimization," in *International Conference on Learning Representations*, 2019.

[12] P. Geibel and F. Wysotzki, "Risk-sensitive reinforcement learning applied to control under constraints," *Journal of Artificial Intelligence Research*, vol. 24, pp. 81–108, 2005.

[13] D. Q. Mayne, J. B. Rawlings, C. V. Rao, and P. O. Scokaert, "Constrained model predictive control: Stability and optimality," *Automatica*, vol. 36, no. 6, pp. 789–814, 2000.

[14] J. Duan, S. E. Li, Y. Guan, Q. Sun, and B. Cheng, "Hierarchical reinforcement learning for self-driving decision-making without reliance on labelled driving data," *IET Intelligent Transport Systems*, vol. 14, no. 5, pp. 297–305, 2020.

[15] J. Duan, S. E. Li, Y. Guan, Q. Sun, and B. Cheng, "Hierarchical reinforcement learning for self-driving decision-making without reliance on labeled driving data," *ArXiv*, vol. abs/2001.09816, 2020.

[16] Y. Guan, Y. Ren, S. Li, Q. Sun, L. Luo, and K. Li, "Centralized cooperation for connected and automated vehicles at intersections by proximal policy optimization," *IEEE Transactions on Vehicular Technology*, vol. 69, pp. 12597–12608, 2020.

[17] M. Ono, M. Pavone, Y. Kuwata, and J. Balaram, "Chance-constrained dynamic programming with application to risk-aware robotic space exploration," *Autonomous Robots*, vol. 39, pp. 555–571, 2015.

[18] S. Paternain, M. Calvo-Fullana, L. F. O. Chamon, and A. Ribeiro, "Learning safe policies via primal-dual methods," *2019 IEEE 58th Conference on Decision and Control (CDC)*, pp. 6491–6497, 2019.

[19] S. Paternain, M. Calvo-Fullana, L. F. O. Chamon, and A. Ribeiro, "Safe policies for reinforcement learning via primal-dual methods," *ArXiv*, vol. abs/1911.09101, 2019.

[20] A. Mesbah, "Stochastic model predictive control: An overview and perspectives for future research," *IEEE Control Systems*, vol. 36, pp. 30–44, 2016.

[21] D. Bertsimas, D. B. Brown, and C. Caramanis, "Theory and applications of robust optimization," *SIAM review*, vol. 53, no. 3, pp. 464–501, 2011.

[22] S. P. Boyd and L. Vandenberghe, "Convex optimization," *IEEE Transactions on Automatic Control*, vol. 51, pp. 1859–1859, 2006.

[23] S. E. Li, "Reinforcement Learning and Control." Tsinghua University: Lecture Notes. http://www.idlab-tsinghua.com/thulab/labweb/publications.html, 2020.

[24] M. Deisenroth and C. E. Rasmussen, "PILCO: A model-based and data-efficient approach to policy search," in *Proceedings of the 28th International Conference on machine learning (ICML-11)*, pp. 465–472, Citeseer, 2011.

[25] D. Liu, Q. Wei, D. Wang, X. Yang, and H. Li, *Adaptive dynamic programming with applications in optimal control*. Springer, 2017.

[26] T. A. N. Heirung, J. Paulson, J. O'Leary, and A. Mesbah, "Stochastic model predictive control - how does it work?," *Computers & Chemical Engineering*, vol. 114, pp. 158–170, 2018.

[27] G. Dalal, K. Dvijotham, M. Vecerik, T. Hester, C. Paduraru, and Y. Tassa, "Safe exploration in continuous action spaces," *arXiv preprint arXiv:1801.08757*, 2018.

[28] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, pp. 25–57, 2006.