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 constraints are essential for reinforcement learning (RL) applied in real-world situations. Chance constraints are suitable to represent the safety requirements in stochastic systems. Most existing RL methods with chance constraints have a low convergence rate, and 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 an aggressive car-following task. Experiments indicate that compared with previous methods, CCAC improves the performance by 57.6% while guaranteeing safety, with a five times faster convergence rate.**

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, in many real-world applications 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]–[7]. Tamar (2013) treated safety in a robust view and optimized the worst-case performance of the agent [8]. 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 [9]. Recently, many researchers also cast safety in the context of Constrained MDPs, where the cumulative cost was constrained below a given threshold [10]–[13]. However, these criteria all focus on reward-related or cost-related measures, and they still lack a direct connection with safety [14]. 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 [15], [16]. In this work, we aim to build a safe policy optimization framework, which could 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 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 [17], [18]. 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 [11]. The second approach constrains the lower bound of safe probability to the required threshold, which can be solved through dynamic programming method [19] or model-free primal-dual (MF-PD) method [20], [21] . 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 conversation. 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. Besides, in order to improve the convergence rate, CCAC utilizes the gradient of dynamic model to guide policy optimization. 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.

*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

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 an aggressive car-following task in Section IV. Section V concludes this paper.

II. Preliminary

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

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

where t is the current time step, $x_t \in \mathcal{X} \subset \mathbb{R}^n$ is the state, $u_t \in \mathcal{U} \subset \mathbb{R}^m$ 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 defining a safe state region. We do not make assumptions about the form of $f(\cdot,\cdot,\cdot)$ and $h(\cdot)$, i.e., they can be linear or nonlinear. Note that here the safety constraint takes form of a joint chance constraint with $1-\delta$ as the required threshold. This form is extensively used in stochastic systems control [22]. Intuitively, it can be interpreted as the probability of agent staying within a safe region over the finite horizon N is at least $1-\delta$. For simplicity, we only consider one joint constraint, but it is easy to generalize to the case of multiple constraints.

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

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

where π is the policy. $r(\cdot,\cdot)$ is the reward function. $0 < \gamma < 1$ is the discounting factor and $\mathbb{E}(\cdot)$ is the expectation w.r.t. the initial state and uncertainty. Specifically, the policy is a deterministic mapping from state space $\mathcal{X}$ to action space $\mathcal{U}$:

$$u_t = \pi(x_t) \tag{3}$$

For an agent behaving according to policy π, the values of the state-action pair (x,u) are defined as follows:

$$Q^{\pi}(x,u) = \mathbb{E}\left\{\sum_{t=0}^{\infty} \gamma^t r\left(x_t, u_t\right) \Big| x_0 = x, u_0 = u\right\} \tag{4}$$

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

$$J_r(\pi) = \mathbb{E}\left\{\sum_{t=0}^{N-1} \gamma^t r\left(x_t, u_t\right) + \gamma^N Q^{\pi}(x_N, u_N)\right\} \tag{5}$$

The joint chance constraint in (2) is generally nonconvex and intractable [23]. 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 [19], [20]. More specifically, a cost function $c(x_t, u_t, x_{t+1})$ and the expected cumulative cost J_c are defined as:

$$c(x_t, u_t, x_{t+1}) = \begin{cases} 0 & h(x_{t+1}) < 0 \\ 1 & h(x_{t+1}) \geq 0 \end{cases} \tag{6}$$

$$J_c(\pi) = \mathbb{E}\left\{\sum_{t=0}^{N-1} c(x_t, u_t, x_{t+1})\right\} \tag{7}$$

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

$$\begin{aligned} p_s(\pi) &= \Pr\left\{\bigcap_{t=1}^{N} \left[h\left(x_t\right) < 0\right]\right\} \\ &\geq 1 - \sum_{t=1}^{N} \Pr\left\{h\left(x_t\right) \geq 0\right\} \\ &= 1 - J_c(\pi) \end{aligned} \tag{8}$$

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

$$J_c(\pi) \leq \delta \tag{9}$$

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 conservation, 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 conservation. 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 [24]. The exterior point methods put the

chance constraint into a large and increasing penalty term in the objective function in k-th iteration:

$$\max_{\pi_k} J_{EP}(\pi_k) = J_r(\pi_k) - \frac{1}{2} b_k \max(1 - \delta - p_s(\pi_k), 0)^2 \tag{10}$$

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 prohibitive, 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.

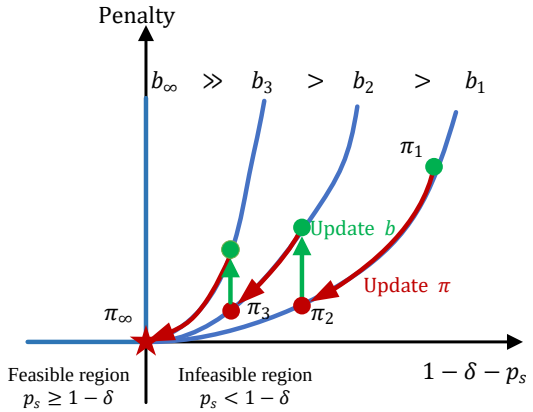

Fig. 1. Exterior point methods.

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

$$\theta^{k+1} = \theta^k + \alpha_\theta \nabla_\theta J_{EP} \tag{11}$$

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

$$\nabla_\theta J_{EP} = \begin{cases} \nabla_\theta J_r & p_s \geq 1-\delta \\ \nabla_\theta J_r + b_k(1-\delta-p_s)\nabla_\theta p_s & p_s < 1-\delta \end{cases} \tag{12}$$

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 is 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 policy evaluation and policy improvement. 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 (8), 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:

$$\nabla_\theta J_{PR} = \begin{cases} \nabla_\theta J_r & p_s \geq 1-\delta \\ \nabla_\theta J_r - b_k(1-\delta-p_s)\nabla_\theta J_c & p_s < 1-\delta \end{cases} \tag{13}$$

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.

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

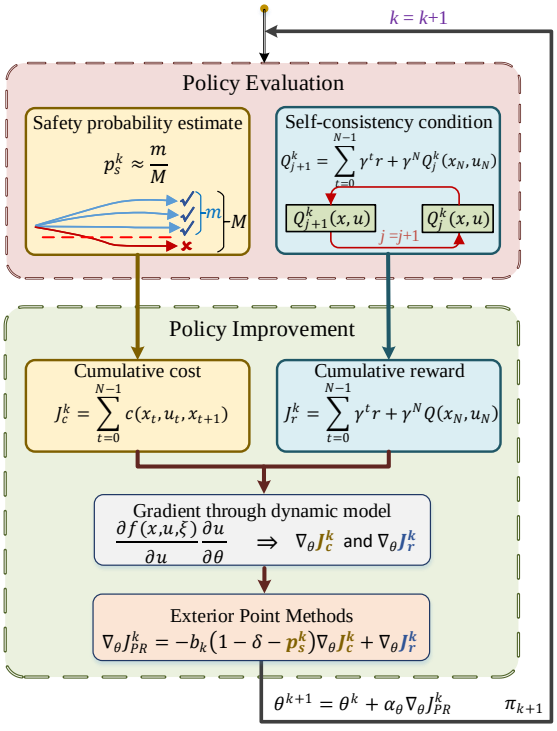

Fig. 2. The framework of CCAC algorithm.

Algorithm 1 CCAC algorithm

Initialize $x_0 \in \mathcal{X}$, b_0, $k = 0$
repeat
  Rollout M trajectories by N steps via dynamic model
  Estimate safe probability through trajectories
    $p_s \approx \frac{m}{M}$
  Policy evaluation according to (16):
    $\omega^{k+1} = \omega^k + \alpha_\omega \nabla_\omega J_Q$
  Policy improvement according to (17):
    $\nabla_\theta J_{PR} = \nabla_\theta J_r - \max(1 - \delta - p_s, 0) b_k \nabla_\theta J_c$
    $\theta^{k+1} = \theta^k + \alpha_\theta \nabla_\theta J_{PR}$
  Update penalty factor b_k
  $k = k + 1$
until $|Q^{k+1} - Q^k| \le \zeta$ and $|\pi^{k+1} - \pi^k| \le \zeta$

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. Especially, the gradient of dynamic model will be utilized to attain an accurate ascent direction and thus improve the convergence rate [25], [26].

For large and continuous state spaces, both value function and policy are parameterized, as shown in (14). 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" [25].

$$Q(x, u) \cong Q(x, u; w) \quad u \cong \pi(x; \theta) \tag{14}$$

The parameterized critic is trained by minimizing the average square error (15) in policy evaluation, i.e.,

$$J_Q = \mathbb{E}_\xi \left\{ \frac{1}{2} \left(Q_{\text{target}} - Q^k(x_0, u_0; w) \right)^2 \right\} \tag{15}$$

where $Q_{\text{target}} = \sum_{t=0}^{N-1} \gamma^t r(x_t, u_t) + \gamma^N Q^k(x_N, u_N)$ 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

$$\nabla_\omega J_Q = \mathbb{E}_\xi \left\{ \left(Q^k(x_0, u_0; w) - Q_{\text{target}} \right) \frac{\partial Q^k(x_0, u_0; w)}{\partial w} \right\} \tag{16}$$

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 $\frac{\partial x_t}{\partial \theta}$ as ϕ_t, $\frac{\partial u_t}{\partial \theta}$ as ψ_t, then $\nabla_\theta J_r$ is derived as:

$$\begin{aligned} \nabla_\theta J_r = & \mathbb{E}_\xi \left\{ \sum_{t=0}^{N-1} \gamma^t \left[\frac{\partial r(x_t, u_t)}{\partial x_t} \phi_t + \frac{\partial r(x_t, u_t)}{\partial u_t} \psi_t \right] \right. \\ & \left. + \gamma^N \left[\frac{\partial Q(x_N, u_N)}{\partial x_N} \phi_N + \frac{\partial Q(x_N, u_N)}{\partial u_N} \psi_N \right] \right\} \end{aligned} \tag{17}$$

where

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

with $\phi_0 = 0$, and

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

The gradient $\nabla_\theta J_c$ can be derived similar to (17). Considering that $c(x, u, x')$ is an indicator function with zero gradient, it is replaced by the sigmoid function:

$$c(x, u, x') = \text{sigmoid}(\eta h(x')) \tag{18}$$

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 [26].

IV. Numerical Experiment

In this section the proposed CCAC is applied to an aggressive 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.

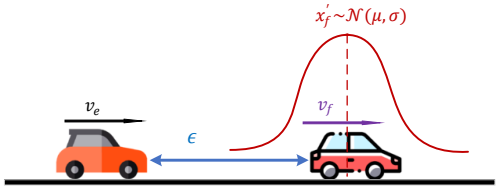

Fig. 3. Car-following scenario.

The discrete-time stochastic system is described by $x_{t+1} = Ax_t + Bu_t + D\xi_t$

$$\begin{aligned} A &= \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -T & T & 1 \end{bmatrix} \\ B &= [T, 0, 0]^\top, \quad D = [0, 0, T]^\top \end{aligned} \tag{19}$$

The system state vector is $x = [v_e \quad v_f \quad \epsilon]^\top$, 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 control input u is the acceleration of ego car, and the disturbance $\xi_t \sim \mathcal{N}(0, 1)$. $T = 0.1$ is the simulation time step. With a chance constraint on the minimum gap, the chance constrained RL problem is

defined as

$$\begin{aligned} &\max_{\pi} \sum_{t=0}^{\infty} \gamma^t (0.2 v_{e,t} - 0.05 \epsilon_t) \\ &\text{s.t. } \Pr\left\{ \bigcap_{t=1}^{N} [\epsilon_t > 2] \right\} \geq 1 - \delta \end{aligned} \tag{20}$$

where $v_{e,t}$ denotes the ego car velocity at time step t. We implement CCAC algorithm on the problem above. The main hyper-parameters are listed in Table I. The penalty factor in (13) is set as $b_k = \min(1000 * 1.01^k, 10000)$. In practice, the weight $b_k(1-\delta-p_s)$ in (13) may become excessively large, so we instead use the relative weights of the two gradients.

TABLE I

HYPER-PARAMETERS

Parameters	Symbol	Value
trajectories number	M	8192
constraint horizon	N	80
discounting factor	γ	0.98
learning rate of policy network	α_θ	3.6e-4
learning rate of value network	α_ω	2e-4
scale factor	η	10

To demonstrate the advantages of CCAC, we compare the performance of CCAC with model-based actor-critic with the fixed-weight penalty (FWP) and model-free primal-dual methods (MF-PD) proposed by [21]. Especially, FWP is tested with different penalty weights 10 and 20.

The cumulative reward and safe probability of the three methods are compared under two chance constraint thresholds 90.0% and 99.9%, i.e., $\delta = 0.1$ and $\delta = 0.001$. The comparison of asymptotic performance is summarized in Table II. The cumulative reward and safe probability under 90.0% chance constraint are plotted in Fig. 4, Fig. 5. The results under 99.9% chance constraint are plotted in Fig. 6 and 7. Each curve is averaged over five independent experiments. Besides, the curve of FWP-10 is omitted since it wins unreasonable reward by totally ignoring the constraint.

TABLE II

COMPARISON OF ASYMPTOTIC PERFORMANCE.

	Chance constraint	Safe probability	Cumulative reward
CCAC	99.90%	99.88%	44.04
CCAC	90.00%	90.28%	50.17
MF-PD	99.90%	99.43%	29.89
MF-PD	90.00%	99.72%	31.84
FWP-20	None	100.00%	43.07
FWP-10	None	0.18%	196.10

In all experiments, the model-based method CCAC converges faster and more stably than the model-free algorithm MF-PD, which confirms the advantage of the utilization of the analytical gradient given by the dynamic model. Besides, CCAC also achieves the highest cumulative reward in both safety levels, while MF-PD exhibits conservation and attains less reward. Especially, MF-PD learns a policy with 99.72% safe probability even

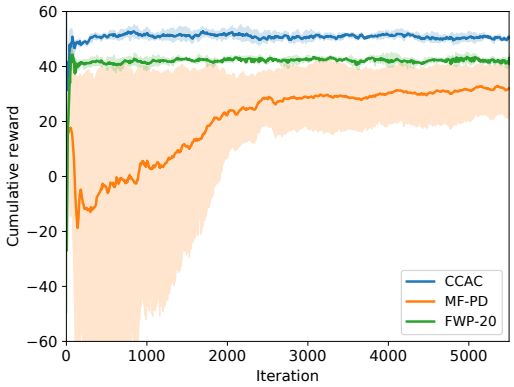

Fig. 4. Comparison of cumulative reward under 90.0% chance constraint among CCAC (chance constrained actor-critic), FWP-20 (penalty with fixed weight 20) and MF-PD (model-free primal-dual).

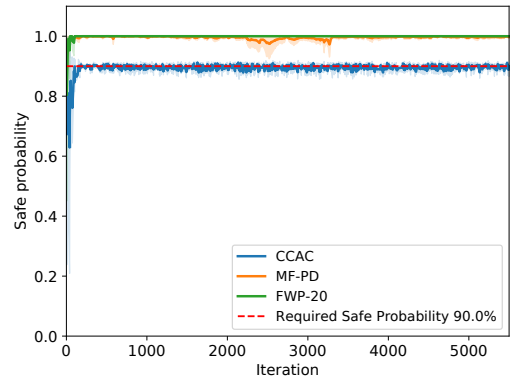

Fig. 5. Comparison of safe probability under 90.0% chance constraint among CCAC (chance constrained actor-critic), FWP-20 (penalty with fixed weight 20) and MF-PD (model-free primal-dual).

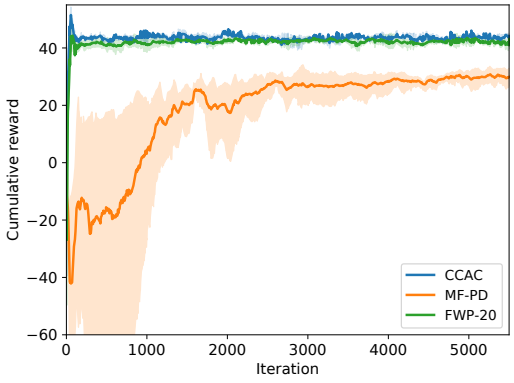

Fig. 6. Comparison of cumulative reward under 99.9% chance constraint among CCAC (chance constrained actor-critic), FWP-20 (penalty with fixed weight 20) and MF-PD (model-free primal-dual).

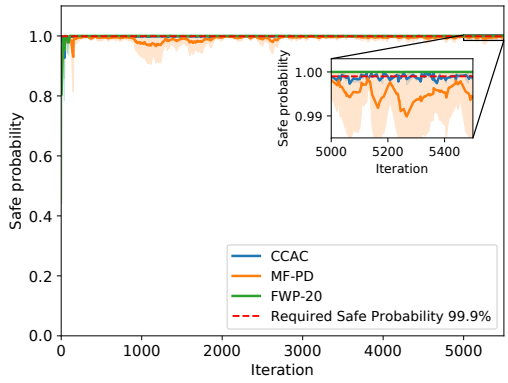

Fig. 7. Comparison of safe probability under 99.9% chance constraint among CCAC (chance constrained actor-critic), FWP-20 (penalty with fixed weight 20) and MF-PD (model-free primal-dual).

when the required threshold is only 90.00%. The root cause of conservation 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. As a result, optimal safe policies can be learned in both safety levels with less conservation.

The FWP has a similar convergence rate as CCAC, but different weights will lead to different safe probabilities and performances. In practice, it is difficult to select an appropriate weight, especially when the required chance is relatively low.

To give an intuitive comparison of three methods, we also test the three policies learned under 90.0% chance constraint on the same car-following scenario, where the initial state is $v_e = 5, v_f = 6$ and $\epsilon = 6$. Fig. 8 shows the varying of car-following gap ϵ averaged on five independent simulations. CCAC keeps the minimum gap while maintaining safety. In contrast, MF-PD is more conservative and keeps a relatively large gap with the front car. FWP method achieves intermediate performance.

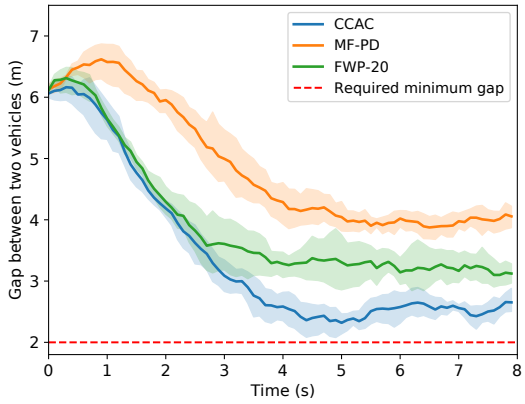

Fig. 8. Car-following gap under 90.0% chance constraint among CCAC (chance constrained actor-critic), FWP-20 (penalty with fixed weight 20) and MF-PD (model-free primal-dual).

In summary, CCAC exhibits the fastest convergence rate during the training process. It also succeeds in learning an optimal policy with required safe probability but not conservative.

V. Conclusion

This paper proposes 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 solves the original chance constraint and thus avoid conservation. Besides, CCAC significantly improves the convergence rate by using the gradient of dynamic model. The benefits of CCAC are demonstrated in simulations of an aggressive car-following task. It achieves 57.6% more cumulative reward while satisfying the chance constraint and converges five times faster than a model-free method. 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, M. G. Bellemare, A. Graves, M. A. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "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, J. Chung, D. Choi, R. Powell, T. Ewalds, P. Georgiev, J. Oh, D. Horgan, M. Kroiss, I. Danihelka, A. Huang, L. Sifre, T. Cai, J. P. Agapiou, M. Jaderberg, A. S. Vezhnevets, R. Leblond, T. Pohlen, V. Dalibard, D. Budden, Y. Sulsky, J. Molloy, T. L. Paine, C. Gulcehre, Z. Wang, T. Pfaff, Y. Wu, R. Ring, D. Yogatama, D. Wünsch, K. McKinney, O. Smith, T. Schaul, T. Lillicrap, K. Kavukcuoglu, D. Hassabis, C. Apps, and D. Silver, "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, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, and D. Hassabis, "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. Garcia and F. Fernández, "A comprehensive survey on safe reinforcement learning," *J. Mach. Learn. Res.*, vol. 16, 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] J. Achiam and D. Amodei, "Benchmarking safe exploration in deep reinforcement learning," 2019.

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

[9] Y. Chow, M. Ghavamzadeh, L. Janson, and M. Pavone, "Risk-constrained reinforcement learning with percentile risk criteria," *ArXiv*, vol. abs/1512.01629, 2017.

[10] J. Achiam, D. Held, A. Tamar, and P. Abbeel, "Constrained policy optimization," in *ICML*, 2017.

[11] C. Tessler, D. J. Mankowitz, and S. Mannor, "Reward constrained policy optimization," *ArXiv*, vol. abs/1805.11074, 2019.

[12] Y. Chow, O. Nachum, A. Faust, M. Ghavamzadeh, and E. A. Duéñez-Guzmán, "Lyapunov-based safe policy optimization for continuous control," *ArXiv*, vol. abs/1901.10031, 2019.

[13] K. Narasimhan, "Projection-based constrained policy optimization," *ArXiv*, vol. abs/2010.03152, 2020.

[14] P. Geibel and F. Wysotzki, "Risk-sensitive reinforcement learning applied to control under constraints," *J. Artif. Intell. Res.*, vol. 24, pp. 81–108, 2005.

[15] D. Mayne, J. Rawlings, C. Rao, and P. Scokaert, "Constrained model predictive control: Stability and optimality," *Autom.*, vol. 36, pp. 789–814, 2000.

[16] J. Duan, Z. Liu, S. Li, Q. Sun, Z. Jia, and B. Cheng, "Deep adaptive dynamic programming for nonaffine nonlinear optimal control problem with state constraints," *ArXiv*, vol. abs/1911.11397, 2019.

[17] 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.

[18] Y. Guan, Y. Ren, S. Li, Q. Sun, L. Luo, K. Taguchi, and K. Li, "Centralized conflict-free cooperation for connected and automated vehicles at intersections by proximal policy optimization," *ArXiv*, vol. abs/1912.08410, 2019.

[19] 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.

[20] 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.

[21] 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.

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

[23] D. Bertsimas, D. Brown, and C. Caramanis, "Theory and applications of robust optimization," *SIAM Rev.*, vol. 53, pp. 464–501, 2011.

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

[25] S. E. Li, "Reinforcement learning and control." Tsinghua University: Lecture Notes. http://www.idlab-tsinghua.com/thulab/labweb/publications.html, 2019.

[26] M. Deisenroth and C. Rasmussen, "Pilco: A model-based and data-efficient approach to policy search," in *ICML*, 2011.