Feasible Actor-Critic: Constrained Reinforcement Learning for Ensuring Statewise Safety

Haitong Ma[1] **Yang Guan**[1] **Shengbo Eben Li**[1]* **Xiangteng Zhang**[2]

Sifa Zheng[1] **Jianyu Chen**[3]

[1]School of Vehicle and Mobility [2]School of Aerospace Engineering
[3]Institute for Interdisciplinary Information Sciences, Tsinghua University
{maht19, guany17, zhangxt18}@mails.tsinghua.edu.cn
{lishbo, zsf, jianyuchen}@tsinghua.edu.cn

Abstract

The safety constraints commonly used by existing safe reinforcement learning (RL) methods are defined only on *expectation* of initial states, but allow each certain state to be unsafe, which is unsatisfying for real-world safety-critical tasks. In this paper, we introduce the feasible actor-critic (FAC) algorithm, which is the first model-free constrained RL method that considers *statewise safety*, e.g, safety for each initial state. We claim that some states are inherently unsafe *no matter* what policy we choose, while for other states there exist policies ensuring safety, where we say such states and policies are *feasible*. By constructing a statewise Lagrange function available on RL sampling and adopting an additional neural network to approximate the statewise Lagrange multiplier, we manage to obtain the optimal feasible policy which ensures safety for each feasible state, and the safest possible policy for infeasible states. Furthermore, the trained multiplier net can indicate whether a given state is feasible or not through the statewise complementary slackness condition. We provide theoretical guarantees that FAC outperforms previous expectation-based constrained RL methods in terms of both constraint satisfaction and reward optimization. Experimental results on both robot locomotive tasks and safe exploration tasks verify the safety enhancement and feasibility interpretation of the proposed method.

1 Introduction

Reinforcement learning (RL) has achieved superhuman performance in solving many complicated sequential decision making problems like Go [26, 27], Atari games [18], and Starcraft [32]. These groundbreaking capabilities attract many real-world applications like autonomous driving [12, 15], energy system management [16] and surgical robotics [22]. However, most successful RL applications with these fields are with simulation platforms, for example, CARLA [6], TORCS [33] and SUMO [21] in autonomous driving. The lack of safety guarantee is one of the major limitations preventing these achievements with RL from being transferred to real world, and becomes an urgent problem for widely benefiting from high-level artificial intelligence.

Current safety consideration of constrained or safe RL studies are to optimize the expected return while guaranteeing safety constraints. The constrained objective is computed by expectations on

*Corresponding author.

Preprint. Under review.

trajectories, i.e., $\mathbb{E}_{\tau\sim\pi}\{\sum_{t=0}^{\infty} c_t\} \leq d$, where τ is the trajectories generated by policy π starting from an initial state distribution $d_0(s)$[1, 2, 20]. However, the are several fatal flaws with the expectation safety constraint: (1) Learned policy can only guarantee the trajectories expectation to satisfy the safety constraints. For a specific state, whether the state is safe remains unclear, which is not acceptable for real-world safety-critical tasks since the real agent must be initialized with a state but not a distribution; (2) We do not know which states are safe with the learned policy, and it is all luck that the current initial state is safe. Intuitively, the most optimistic case is that learned policy can guarantee the whole state space to be safe, but it is impossible in many realistic tasks. For example, Figure 1 depicts an emergency braking task. If the vehicle is too close or too fast, the collision is inevitable. Those states where safety constraints must be violated no matter what policy we choose is defined as infeasible region, and states which are possible to be safe are included in the feasible region.

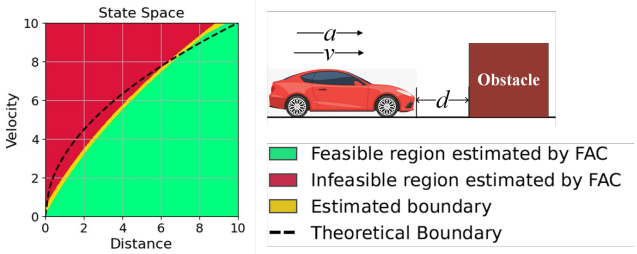


Figure 1: Feasible region in emergency braking. States include distance d and velocities v, the action is vehicle deceleration limited at $a_{max} = -5m/s^2$. The real feasible region is limited by the uniform deceleration curve $d \leq v^2/2|a_{max}|$ plotted by the dash line. Estimated region by FAC is shaded with green.

Therefore, what we can do is to ensure that as many states as possible are safe, while identifying those infeasible states. In this paper, we propose the feasible actor-critic (FAC) algorithm to guarantee safety of feasible states and indicate statewise feasibility. We give the mathematical formulation of feasible partition of state space, including feasible and infeasible region. The safety constraint is defined on cost value function of each possible states in the initial distribution. We use the Lagrangian approach to solve the problems with infinite number of constraints, and the statewise multipliers are approximated by a neural network. The multiplier network stores additional information from learning progress, and we manage to explain that it is exactly the feasibility interpretation by introducing the statewise complementary slackness conditions. By training policy and multiplier networks with the primal-dual gradient ascent, we can find an optimal feasible policy to guarantee safety on each feasible state a feasibility indicator to tell which states are infeasible.

Our main contributions are:
(1) We introduce the complete definitions about statewise safety for RL, and the feasible region to discuss feasible partition of state space.
(2) We introduce FAC, an algorithm to guarantee each possible initial state in the initial distribution is safe. An additional neural network is introduced to indicate whether current state is feasible.
(3) We evaluate the proposed algorithm on environments in Gym and Safety Gym with different safety constraints. FAC shows solid safety guarantee and great robustness compared to baselines. The ability to interpreter feasible region is also depict.

2 Preliminaries

2.1 Constrained Markov Decision Process

The constrained Markov decision process (CMDP), $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, \mathcal{P}, r, c \rangle$, is to maximize the expected return of rewards while satisfying the constraints of expected return of costs:

$$\max_{\pi} J(\pi) = \mathbb{E}_{\tau\sim\pi}\left\{\sum_{t=0}^{\infty}\gamma^t r_t\right\} \quad \text{s.t. } C(\pi) = \mathbb{E}_{\tau\sim\pi}\left\{\sum_{t=0}^{\infty}\gamma_c^t c_t\right\} \leq d \tag{EP}$$

where $\mathcal{S}$ and $\mathcal{A}$ are a state and action space, $\mathcal{P}$ is the environment transition model, $r,\ c : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the reward and cost functions,$\gamma,\ \gamma_c$ is the discounted factor for reward and cost functions, $\tau = \{s_0, a_0, s_1, a_1 \dots\}$ is the trajectory under policy π starting from an initial state distribution $d_0(s)$[2]. The feasible policy set is defined as

$$\Pi_C = \{\pi | C(\pi) \leq d\} \tag{2.1}$$

CMDP does not discuss which states are feasible. Notably, the percentile indicators rather than mean value are used to define worst-case safety constraints [8, 34]. Nevertheless, they still wrap it with the trajectory expectation, and the problems that certain state may violate the constraints still exist in these studies. Our motivation is to solve the failure of safety guarantee caused by expectation on initial state distributions, which is different from these studies.

2.2 Related Works

Constrained RL with expectation constraint. We focus on constrained RL algorithms applicable for continuous high-dimensional tasks. One branch is to compute a constraint-satisfying policy gradient, for example, using Rosen gradient projection [31] or approximate solver for natural policy gradient (PG) [1, 35, 36]. These constrained gradient methods are difficult to implement, and their approximations harm the performance in complex tasks. Another branch is the Lagrangian-based constrained RL, which is to reach the saddle point of the objective function with a scalar Lagrange multiplier. There are studies applying expected Lagrangian modifications on vanilla PG, actor-critic (AC), trust-region policy optimization (TRPO), and proximal policy optimization (PPO) [8, 20, 23, 24]. Lagrangian approach is simple to implement and outperforms the constrained gradient methods in complex tasks. However, the oscillation issue seriously affects the performance of Lagrangian approach. Peng et al. [19], Stooke et al. [28] use a proportional-integral (PI) control approach to handle the oscillation problem in primal-dual ascent.
Shielding RL or RL with controllers. Some studies in the control theory have similar motivation like safety barrier certificate or reachability theory [3, 15, 17]. With these control-based approaches to compute a a safe set $\mathcal{U}$ at a specific state, some RL studies projects the policy output into $\mathcal{U}$ achieve safety of current state [7, 9]. The major limitation of shields or controllers is that they require a prior model and can only handle a single state, while FAC is a model-free constrained RL algorithm with feasibility indicator of the whole state space. We do not consider these methods as the baselines since we do not have prior models for the empirical environments.

3 Statewise Safety and Feasibility Analysis

3.1 Definitions of Statewise Safety

Definition 1 (Feasible state)**.** We define that a state is feasible if its expected return of cost c_t satisfy the inequality:

$$v_C^\pi(s) = \mathbb{E}_{\tau\sim\pi}\left\{\sum_{t=0}^{\infty}\gamma_c^t c_t \middle| s_0 = s\right\} \leq d \tag{3.1}$$

where $\mathbb{E}_{\tau\sim\pi}[\cdot|s_0 = s]$ denotes that the expected value of trajectory generated by policy π starting from a given initial state s. $v_C^\pi(s)$ is named as safety critic, or cost value function.

Before defining safety constraints, we first define the feasible partition of state space:

Definition 2 (Infeasible region)**.** The infeasible region $\mathcal{S}_I$ is defined as the set of states can not be safe no matter what policy we choose

$$\mathcal{S}_I = \{s | v_C^\pi(s) > d,\ \forall\pi \in \Pi\} \tag{3.2}$$

Definition 3 (Feasible region)**.** Given a policy π, the set of all feasible state is defined as the *feasible region* $\mathcal{F}^\pi \subseteq \mathcal{S}$:

$$\mathcal{S}_F^\pi = \{s \mid v_C^\pi(s) \leq d\} \tag{3.3}$$

Note that the feasible region is relevant with policy π, and the infeasible region is not. The policy may be not good enough to guarantee that all possibly feasible states to be feasible, so the relation of feasible region and infeasible region is

$$\mathcal{S}_F^\pi \subseteq \complement_{\mathcal{S}}\mathcal{S}_I \tag{3.4}$$

The states inside infeasible region are those cannot guaranteed to be safe, so we assume that the initial distribution does not start from the infeasible region. The initial state distribution $d_0(s)$ is designed as follow, and $\mathcal{I}$ represents possible initial states set:

$$d_0(s) > 0 \text{ if } s \in \mathcal{I}, d_0(s) = 0 \text{ if } s \notin \mathcal{I},\ \mathcal{I} \subseteq \complement_{\mathcal{S}}\mathcal{S}_I \tag{3.5}$$

Definition 4 (Statewise safety constraint)**.** The statewise safety constraint is defined to guarantee every state in the possible initial states set to be safe:

$$v_C^\pi(s) \leq d, \forall s \in \mathcal{I} \tag{3.6}$$

We define a policy to be feasible if it satisfies the statewise safety constraint. The feasible policy set under statewise safety constraints is $\Pi_F = \left\{\pi \middle| v_C^\pi(s) \leq d, \forall s \in \mathcal{I}\right\}$. It is equivalent to the possible initial states set are included in the feasible region, i.e.,

$$\mathcal{I} \subseteq \mathcal{S}_F^\pi \tag{3.7}$$

Finally, without changing the RL maximization objective, we formulate the optimization problem with statewise constraints:

$$\max_\pi\ J(\pi) = \mathbb{E}_{\tau\sim\pi}\left\{\sum_{t=0}^{\infty} r_t\right\} \quad \text{s.t. } v_C^\pi(s) \leq d, \forall s \in \mathcal{I} \tag{SP}$$

3.2 Multipliers: Statewise Feasibility Indicators

We use the Lagrangian approach to solve problem (SP). As each state in $\mathcal{I}$ has a constraint, there exists the corresponding Lagrange multiplier at each state, and we denote the statewise multiplier as $\lambda(s)$. The multipliers have the physical meaning of solution feasibility, which comes from the statewise complementary slackness conditions. If $v_C^\pi(s) = d$, we say that the constraint at state s is active. An active constraint represents that the constraint is preventing the objective function to be further optimized. We assume the strong duality holds here for simplicity, and the following experimental results depict that the feasibility analysis still works in complex environments where the strong duality assumption may fail.

Proposition 1 (Statewise complementary slackness condition)**.** For the problem (SP), at state s the optimal multiplier and optimal safety critic are $\lambda^*(s), v_C^*(s)$, the following conditions hold:

$$\lambda^*(s) = 0\ , v_C^{\pi*}(s) \leq d,\ \text{or } \lambda^*(s) > 0,\ v_C^{\pi*}(s) = d \tag{3.8}$$

The proposition comes from the KKT necessary condition for the problem (SP). Whether the multiplier is zero or positive represents whether the constraints is active. A state whose safety constraint is not active must lie inside the feasible region.

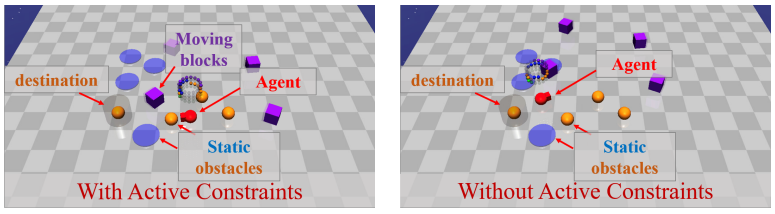


Figure 2: Two example frames in a Safety Gym simulation. The task is to touch the shaded button as fast as possible while avoiding obstacles. In (a), the obstacles block the way towards the destination. This is the case that safety constraint is active, and the state lies on the boundary of feasible region. In (b), the agent can go straight to touch the button without considering safety constraint. The constraint is inactive and the state lie inside the feasible region.

The active constraint can also be represented by the safety critic, or learned cost value net $v_C^\pi(s)$. However, $v_C^\pi(s)$ can only represent if state is feasible under current policy π. It is not able to distinguish two cases if current state is not feasible, i.e., $v_C^\pi(s) \geq d$: (1) The state are not in infeasible region, and only the current policy is not good enough to make it not feasible; (2) The state lies inside the infeasible region, which means it can not be feasible with any policy. The multiplier gives the ability to distinguish them in the primal-dual gradient ascent process by:

Corollary 1. *If s lies in infeasible region, then $\lambda(s) \to \infty$ with the primal-dual descent.*

The main idea is that if the feasible solution do not exists, the subgradient of λ will always be positive. Detailed proof is provided in Appendix A.2. We have the relation between multipliers and feasibility of states listed in Table 1.

Table 1: Relation between multipliers and feasibility.

Multiplier scale $\lambda(s)$	Feasibility situation of s
Zero	Inside feasible region
Finite	Not in infeasible region, possibly on boundary of feasible region
Infinite	Infeasible region

4 Feasible Actor-Critic

4.1 Alternative Objective Function for Practical Implementation

For practical implementation, we use the Lagrangian approach to solve problem (SP), the corresponding Lagrange function is:

$$\mathcal{L}_{\text{stw}}\big(\pi, \lambda\big) = -\mathbb{E}_{s\sim d_0(s)} v^{\pi}(s) + \sum_{s\in\mathcal{I}} \lambda(s)\Big(v_C^{\pi}\big(s\big) - d\Big) \tag{SL}$$

The sum term $\sum_s \lambda(s)(v_C^{\pi}(s) - d)$ is intractable in RL since the we are sampling data from the state distribution, and all computations must be in the formulation of $\mathbb{E}_{s\sim d(s)}\{\cdot\}$. This becomes the major challenge when we want to guarantee statewise feasibility. We propose a theorem to transfer the real Lagrangian to an available alternative objective function:

Theorem 1 (Alternative objective function for statewise Lagrangian)**.** *Construct a statewise Lagrange function as*

$$L(\pi, \lambda, s) = -v^{\pi}(s) + \lambda(s)\big(v_C^{\pi}(s) - d\big) \tag{4.1}$$

The alternative objective function in practical implementation is:

$$J_{\mathcal{L}}(\pi, \lambda) = \mathbb{E}_{s\sim d_0(s)}\big\{L(\pi, \lambda, s)\big\} = \mathbb{E}_{s\sim d_0(s)}\big\{ - v^{\pi}(s) + \lambda(s)\big(v_C^{\pi}(s) - d\big)\big\} \tag{A-SL}$$

If the optimal policy and Lagrange multiplier mapping π^* and $\lambda^*(s)$ exists for problem $\max_{\lambda(s)} \inf_{\pi} J_{\mathcal{L}}(\pi, \lambda)$, then π^* is the optimal policy of problem (SP), and also a feasible policy in Definition 4.

The main idea of this proof is to scale the safety constraints by the probabilistic density $d_0(s)$:

$$d_0(s)\big(v_C^{\pi}(s) - d\big) \leq 0 \tag{4.2}$$

Since $d_0(s) \geq 0$ and the case $d_0(s) = 0$ for $s \notin \mathcal{I}$, it provides the equivalence between the statewise safety constraints and the scaled constraints. The detailed proof is provided in Appendix A.1.

4.2 Performance Comparison with Expected Lagrangian Methods

In this section we explain the relationship between statewise safety and expectation-based safety. Two theorems is proposed to compare the constraint satisfaction and performance and between two approaches.

Theorem 2. *A feasible policy π_f under statewise safety constraints must be feasible under expectation-based safety constraints, i.e., for the same constraint threshold d, $\Pi_F \subseteq \Pi_C$.*

Proof. Expected constraints can be reformulated as

$$C\big(\pi\big) = \mathbb{E}_{\tau\sim\pi}\Big\{\sum_{t=0}^{\infty}\gamma_c^t c_t\Big\} = \sum_s d_0(s)\mathbb{E}_{\tau\sim\pi}\Big\{\sum_{t=0}^{\infty}\gamma_c^t c_t\big|s_0 = s\Big\} = \mathbb{E}_{s\sim d_0(s)}\big\{v_C^{\pi}(s)\big\} \tag{4.3}$$

According to the Definition 4, for a policy $\pi_f \in \Pi_F$ and $\forall s \in \mathcal{I} \subseteq \mathcal{F}^\pi$, $v_C^{\pi_f}(s) \leq d$. Therefore, the expectation on the initial state distribution

$$C(\pi_f) = \mathbb{E}_{s\sim d_0(s)}\{v_C^\pi(s)\} = \sum_s d_0(s) v_C^\pi(s) \leq d \tag{4.4}$$

The performance comparison is under the concave-convex assumption both on policy and state space. The Lagrange function of optimization problem with expectation constraint is:

$$\mathcal{L}_{\text{exp}}(\pi, \lambda) \doteq \lambda\Big(C(\pi) - d\Big) - J(\pi) = \lambda\bigg(\mathbb{E}_{\tau\sim\pi}\Big\{\sum_{t=0}^{\infty}\gamma_c^t c_t\Big\} - d\bigg) - \mathbb{E}_{\tau\sim\pi}\Big\{\sum_{t=0}^{\infty}\gamma^t r_t\Big\} \tag{EL}$$

We name the Lagrange function in (EL) as the *expected Lagrangian*. The optimal policy obtained by the saddle point of problem (EL) and (A-SL) are denoted by π^*_{stw} and π^*_{exp}.

Theorem 3. *Assume that $\mathcal{I}$ and Π are both nonempty convex set. v^π is concave on $\mathcal{I}$ and Π, and v_C^π is convex on $\mathcal{I}$ and Π. Assume the interior point holds on Π for each s. The optimal objective function under expected Lagrangian in equation (EL) is the lower bound of the objective function on statewise Lagrangian problems (A-SL), i.e.,*

$$J(\pi^*_{stw}) \geq J(\pi^*_{exp}) \tag{4.5}$$

Detailed proof is provided in the Appendix A.3.

4.3 Practical Algorithm

We use an off-policy maximum entropy RL framework to optimize the alternative objective function (A-SL). In practice, the Lagrange multiplier $\lambda(s)$ is approximated with a neural network, i.e., $\lambda(s) \approx \lambda(s;\xi)$. We need to train the value function Q_ϕ, cost value function Q_{ϕ_C}, policy π_θ, and the multipliers networkλ_ξ. Details about the parameterized functions are listed in Appendix B.1. The FAC objective function for both policy and multiplier network is:

$$J_\lambda(\xi) = J_\pi(\theta) = \mathbb{E}_{s_t\sim\mathcal{B}}\Big\{\mathbb{E}_{a_t\sim\pi_\theta}\Big\{\alpha\log\big(\pi_\theta\left(a_t \mid s_t\right)\big) - Q_\phi(s_t,a_t) + \lambda_\xi(s_t)\big(Q_{\phi_C}(s_t,a_t) - d\big)\Big)\Big\} \tag{4.6}$$

where α is the temperature. We use the primal-dual gradient ascent to alternatively update policy and multiplier networks, and detailed gradient computation is provided in Appendix B.2. Additionally, we introduce some tricks used in the practical implementations. Inspired by exiting studies about training with adversarial objective function and delayed policy update tricks [10, 11], we set a separate schedule for policy delay steps m_π and ascent delay steps m_λ to handle the convergence issue. The oscillation problems is tricky for primal-dual ascent, and we start training the multiplier net until the cost value function exceeds the constraint threshold. The pseudocode and detailed gradient computaton is shown in Appendix B.3.

5 Empirical Analysis

We choose two different set of experiments to evaluate the FAC algorithm: (1) MuJuCo robots walking with speed limits; (2) Exploration with limited number of dangerous actions in Safety Gym. The robot speed constraints are rather simple where the concave-convex assumption may hold, and we design them verify the performance improvement in Theorem 3. The environments in the second set is more challenging with realistic observation and constraints as shown in Figure 2, and we want to show that all trajectories satisfy the safety constraint with FAC. All environments are implemented with Gym API and MuJoCo physical simulator [5, 20, 30].

We compare our algorithm against baseline constrained RL algorithms provided by OpenAI, including Constrained Policy Optimization (CPO) , TRPO-Lagrangian (TRPO-L) and PPO-Lagrangian (PPO-L) [1, 20]. TRPO-L and PPO-L are to use the primal-dual decent to compute the saddle point of expected Lagrangian in equation (EL) with a scalar multiplier, and the reward objective function $J_R(\pi)$ is the surrogate formulation in TRPO or PPO.

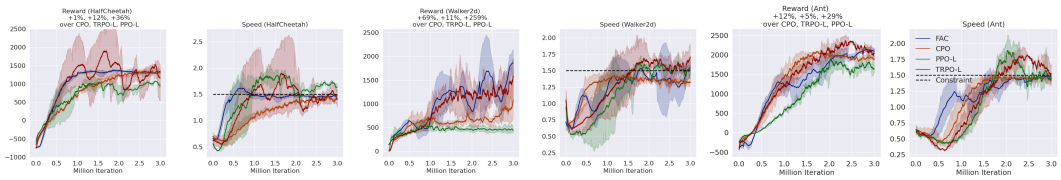


Figure 3: Learning curves for robots with speed limit tasks. The x-axis represent the number of updates used and the y-axis represent the average total reward return and robot speed of the last 100 episodes. The solid line represent the mean of batch samples over 5 random seeds. The shaded regions represent the 95% confidence interval. FAC consistently enforce approximate constraint satisfaction while having a higher performance over other constraint-satisfying algorithms.

5.1 Controlling Robots with Speed Limits

We choose three MuJoCo environments where we attempt to train a robotic agent to walk with speed limits. Results shows that FAC outperforms baselines in all environments. Although CPO has higher average returns in some random seeds, but the average return is lower than FAC. FAC is rather more stable than baseline algorithms in HalfCheetah and Ant environments. For Walker2d environments, all algorithms performs not well since walking with low speed seems more harder than running. FAC still manage to obtain a constraint-satisfying policy with most random seeds. The speed constraints are obeyed by both FAC and CPO. As for TRPO-L and PPO-L, they still suffers from the oscillation issues, the constraints are violated severely during training, and PPO-L fails to learn a policy controlling the agent to walk in Walker2d environment.

Table 2: Average return and speed over 5 random seeds on robot walking with speed limit environments. The bold ones are the maximum average return for each task. $\pm$ corresponds to a single standard deviation over runs.

Environment	Performance (speed limit:1.5)	FAC	CPO	TRPO-L	PPO-L
HalfCheetah-v3	Return	**1318±26.3**	1303±120.4	967±55.9	1180±431.5
	Speed	1.46±0.013	1.41±0.011	1.43±0.125	1.63±0.065
Walker2d-v3	Return	**1651±314**	979±400	1483±384	460±68.8
	Speed	1.43±0.210	1.32±0.051	1.67±0.197	1.50±0.088
Ant-v3	Return	**2121±68.4**	1898±111	2017±119	1646±117
	Speed	1.48±0.052	1.44± 0.053	1.52±0.150	1.43±0.032

5.2 Safe Exploration Tasks

We choose two Safety Gym environments with different agents and tasks. Especially, the cost signal is encoded as the one-hot formulation, which provides $c_t = 1$ once any unsafe action is taken, otherwise $c_t = 0$. The safety constraints are the dangerous action rate $c_{\text{rate}} = \sum c_i/(\text{sampled steps}) \leq 10\%$. It is transformed to the safety critic constraints $v_C^\pi(s) \leq 10$.

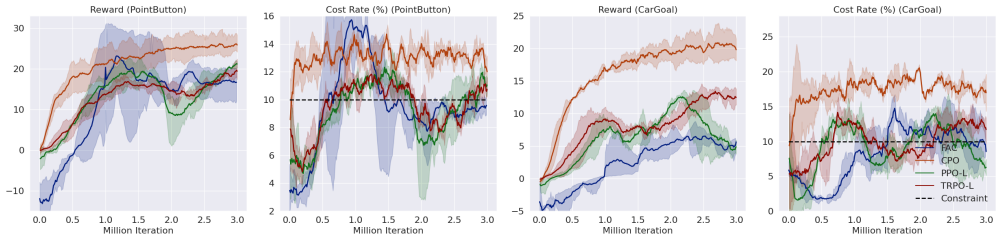


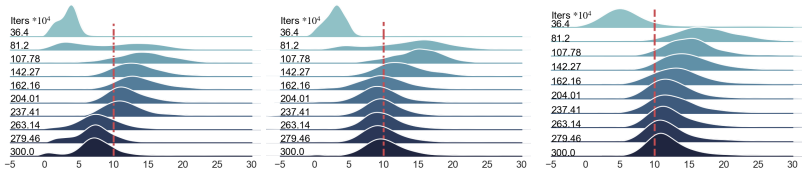
Figure 4: Learning curves for safe exploration tasks. Axis means the same as Figure 3 FAC finally achieves constraints satisfaction in both environments.

Table 3: Batch average return and dangerous action rate over 5 random seeds on safe exploration environments. The bold ones are the maximum average return or the minimum number of dangerous episodes for each task. $\pm$ corresponds to a single standard deviation over runs. $c_{\text{rate}} = \sum c_i/(\text{sampled steps})$ represents the dangerous action rate, and the safety constraint is $c_{\text{rate}} \leq 10\%$. The number of episodes violating the c_{rate} constraints in the 100 test episodes are also listed. FAC maintains low dangerous episodes in both environments with reasonable performance sacrifice.

Agent & Tasks	Performance (Dangerous action rate constraint: 10%)	FAC	CPO	TRPO-L	PPO-L
Point-Button	Return	17.08±3.55	**25.67±1.96**	18.76±1.42	20.48±0.63
	c_{rate} (%)	9.286±0.764	13.192±0.688	10.469±5.357	11.075±0.855
	Dangerous episodes in 100 tests	**3**	73	52	66
Car-Goal	Return	3.80±2.32	**18.62±3.25**	12.49±1.31	5.01±1.46
	c_{rate} (%)	8.786±3.766	16.708±1.874	12.288±1.636	7.047±2.095
	Dangerous episodes in 100 tests	8	77	69	**2**

We count both the episodic dangerous action rate and the number of episodes violating the dangerous action rate constraints. Table 3 shows that FAC confines most episodes to satisfy the safety constraints. As for baseline algorithms, CPO fails to satisfy constraints in all safe exploration tasks. PPO-L achieves a lower average episodic cost and dangerous episodes in Car-Goal environments, but the reason is that it happened to oscillate to the lowest point as shown in Figure 4. For TRPO-L or PPO-L, the mean episodic costs oscillates around the threshold. The number of dangerous test episodes suggest that of the mean value satisfies the safety threshold, there still exist half of the episodes violating the safety constraint. To sum up, expectation-based safety constraints can not guarantee all trajectories to be safe, while FAC can guarantee that most trajectories are safe.
As for the reward performance, baselines are better than FAC in all environments. This contradicts our Theorem 3. The reason is that the safe exploration task is so complex that the concave-convex assumption does not hold. The trade-off relationship between the reward and cost returns in these environments are analyzed by Ray et al. [20], so the performance sacrifice of FAC is reasonable.

5.3 Visualization of Constraint Satisfaction



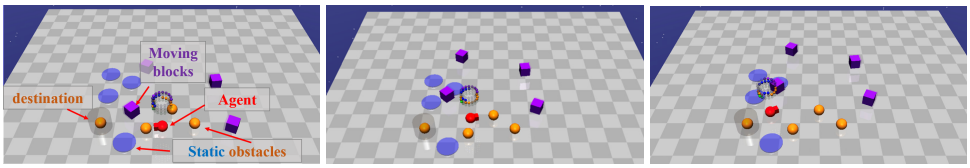
(a) FAC (b) TRPO-Lagrangian (c) CPO

Figure 5: Distribution of batch safety critic values with off-policy algorithms in `Point-Button` safe exploration tasks during training. One Axis represents the cost value distribution of batch states on the specific training iteration labeled at left. The red dash line is the constraint threshold.

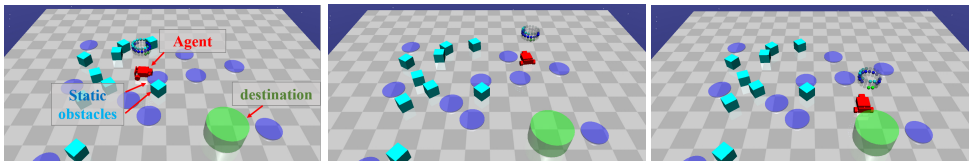
To further understand the constraint satisfaction property with statewise constraints, we visualize the batch safety critic value distribution during the training in Figure 5. Results shows that most states in the distribution with FAC satisfies the safety constraints. With PPO-Lagrangian, the peak of distribution, which corresponds to the expectation, lies exactly on the constraint threshold. Most states violates the constraints with CPO. This visualization depicts that there are still half of the states are dangerous with the expectation-based safety constraint, while statewise safety constraints and FAC render most states in the distribution to be safe.

5.4 Feasibility Indicator by Multiplier Network

We randomly select some cases from two tasks to demonstrate the interpretability of feasibility with multiplier net. Figure 6 shows two example episodes in the safe exploration task with three representative frames in each episode. In Figure 6(a) and (f), no obstacle is blocking the agent to reach the destination (shaded button) the multiplier is zero, which means current state lies inside the feasible region. In Figure 6(b) and (d), When the moving blocks or obstacles blocks the way from the destination, the multiplier is high, which means safety constraints is active here and the agent may face risk.



(a) Step 212, $\lambda = 13.26$ (b) Step 256, $\lambda = 0.005$ (c) Step 275, $\lambda = 0.00$



(d) Step 345, $\lambda = 1.90$ (e) Step 643, $\lambda = 0.024$ (f) Step 664, $\lambda = 0.00$

Figure 6: Representative frames in test episodes of PointButton ((a)-(c)) and CarGoal ((d)-(f)) task with policy trained by FAC. In (a)-(c), the trained agent choose to wait the moving blocks (the purple square) to go away, and bypass the static obstacles from above to reach the destination. In (d)-(f), the trained agent choose to go to the upper side where the obstacles are sparse, then turn around to go straight to the destination.

6 Discussion

We introduced the statewise safety constraints, which requires starting from arbitrary states the safety constraints should be satisfied. FAC, a Lagrangian-based algorithm is proposed to guarantee statewise safety with a multiplier network. FAC is easy to implement, and shows robustness of guaranteeing statewise safety with interpretation.

Limitations and future work. FAC still suffers from inherent problems of Lagrangian-based algorithms like oscillations and convergence issue, which can be . Some promising avenues come from the feasibility information contained by the multiplier net. For example, the multiplier net can help to design safe exploration rules and safety criterion transformation between different environments.

7 Broader Impact

Safety issues are preventing RL from being applied in more real-world applications. We believe that the reasonable statewise safety take a step forward in the safety of RL, which helps to land this powerful techniques in more safety-critical applications. We advocate for safe RL researchers to consider statewise safety constraints. Moreover, the multiplier net actually learns and stores the information from the environment that is not explored by any other previous RL algorithms other than traditional value and policy. The usage of multiplier net may inspire new breakthrough in RL society.

References

[1] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *International Conference on Machine Learning*, pages 22–31. PMLR, 2017.

[2] Eitan Altman. *Constrained Markov decision processes*, volume 7. CRC Press, 1999.

[3] Aaron D Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *2019 18th European Control Conference (ECC)*, pages 3420–3431. IEEE, 2019.

[4] Dimitri P Bertsekas. Nonlinear programming. *Journal of the Operational Research Society*, 48 (3):334–334, 1997.

[5] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

[6] Jianyu Chen, Shengbo Eben Li, and Masayoshi Tomizuka. Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[7] Richard Cheng, Gábor Orosz, Richard M Murray, and Joel W Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3387–3395, 2019.

[8] Yinlam Chow, Mohammad Ghavamzadeh, Lucas Janson, and Marco Pavone. Risk-constrained reinforcement learning with percentile risk criteria. *The Journal of Machine Learning Research*, 18(1):6070–6120, 2017.

[9] 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.

[10] Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pages 1587–1596. PMLR, 2018.

[11] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014.

[12] Yang Guan, Yangang Ren, Shengbo Eben Li, Qi Sun, Laiquan Luo, and Keqiang Li. Centralized cooperation for connected and automated vehicles at intersections by proximal policy optimization. *IEEE Transactions on Vehicular Technology*, 69(11):12597–12608, 2020.

[13] Yang Guan, Jingliang Duan, Shengbo Eben Li, Jie Li, Jianyu Chen, and Bo Cheng. Mixed policy gradient. *arXiv preprint arXiv:2102.11513*, 2021.

[14] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870. PMLR, 2018.

[15] Haitong Ma, Jianyu Chen, Shengbo Eben Li, Ziyu Lin, and Sifa Zheng. Model-based constrained reinforcement learning using generalized control barrier function. *arXiv preprint arXiv:2103.01556*, 2021.

[16] Karl Mason and Santiago Grijalva. A review of reinforcement learning for autonomous building energy management. *Computers & Electrical Engineering*, 78:300–312, 2019.

[17] Ian M Mitchell, Alexandre M Bayen, and Claire J Tomlin. A time-dependent hamilton-jacobi formulation of reachable sets for continuous dynamic games. *IEEE Transactions on automatic control*, 50(7):947–957, 2005.

[18] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[19] Baiyu Peng, Yao Mu, Jingliang Duan, Yang Guan, Shengbo Eben Li, and Jianyu Chen. Separated proportional-integral lagrangian for chance constrained reinforcement learning. *arXiv preprint arXiv:2102.08539*, 2021.

[20] Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking safe exploration in deep reinforcement learning. *arXiv preprint arXiv:1910.01708*, 2019.

[21] Yangang Ren, Jingliang Duan, Shengbo Eben Li, Yang Guan, and Qi Sun. Improving generalization of reinforcement learning with minimax distributional soft actor-critic. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–6. IEEE, 2020.

[22] Florian Richter, Ryan K Orosco, and Michael C Yip. Open-sourced reinforcement learning environments for surgical robotics. *arXiv preprint arXiv:1903.02090*, 2019.

[23] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

[24] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[25] Alexander Shapiro, Darinka Dentcheva, and Andrzej Ruszczyński. *Lectures on stochastic programming: modeling and theory*. SIAM, 2014.

[26] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[27] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

[28] Adam Stooke, Joshua Achiam, and Pieter Abbeel. Responsive safety in reinforcement learning by pid lagrangian methods. In *International Conference on Machine Learning*, pages 9133–9143. PMLR, 2020.

[29] Sebastian Thrun and Anton Schwartz. Issues in using function approximation for reinforcement learning. In *Proceedings of the Fourth Connectionist Models Summer School*, pages 255–263. Hillsdale, NJ, 1993.

[30] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[31] Eiji Uchibe and Kenji Doya. Constrained reinforcement learning from intrinsic and extrinsic rewards. In *2007 IEEE 6th International Conference on Development and Learning*, pages 163–168. IEEE, 2007.

[32] Oriol Vinyals, Igor Babuschkin, Wojciech M Czarnecki, Michaël Mathieu, Andrew Dudzik, Junyoung Chung, David H Choi, Richard Powell, Timo Ewalds, Petko Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[33] Sen Wang, Daoyuan Jia, and Xinshuo Weng. Deep reinforcement learning for autonomous driving. *arXiv preprint arXiv:1811.11329*, 2018.

[34] Qisong Yang, Thiago D Simão, Simon H Tindemans, and Matthijs TJ Spaan. Wcsac: Worst-case soft actor critic for safety-constrained reinforcement learning. In *Proceedings of the Thirty-Fifth AAAI Conference on Artificial Intelligence. AAAI Press, online*, 2021.

[35] Tsung-Yen Yang, Justinian Rosca, Karthik Narasimhan, and Peter J Ramadge. Projection-based constrained policy optimization. *arXiv preprint arXiv:2010.03152*, 2020.

[36] Yiming Zhang, Quan Vuong, and Keith Ross. First order constrained optimization in policy space. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15338–15349. Curran Associates, Inc., 2020.

Appendix

A Proof of Theorems

A.1 Proof of Theorem 1

The alternative Lagrangian in equation (A-SL) with a parameterized policy $\pi(s;\theta)$ is:

$$\begin{aligned} J_{\mathcal{L}}(\theta,\lambda) &= \mathbb{E}_{s\sim d^0(s)}\Big\{v^{\pi}\big(s\big)+\lambda(s)\Big[v_C^{\pi}\big(s\big)-d\Big]\Big\} \\ &= \mathbb{E}_{s\sim d^0(s)}\big\{v^{\pi}\big(s\big)\big\}+\sum_{s\in\mathcal{S}} d^0(s)\lambda(s)\Big[v_C^{\pi}(s)-d\Big] \\ &= \mathbb{E}_{s\sim d^0(s)}\big\{v^{\pi}\big(s\big)\big\}+\sum_{s\in\mathcal{S}} \lambda(s)\Big\{d^0(s)\Big[v_C^{\pi}(s)-d\Big]\Big\} \end{aligned} \tag{A.1}$$

Construct an alternative constrained optimization problem here, whose formulation is exactly corresponding to the optimization problem (SP):

$$\begin{aligned} &\min_{\theta}\ \mathbb{E}_{s\sim d^0(s)} v^{\pi}(s) \\ &\text{s.t. } d^0(s)\Big[v_C^{\pi}(s)-d\Big]\le 0, \forall s\in\mathcal{S} \end{aligned} \tag{A-SP}$$

We focus on the constraint formulation of problem (A-SP). The visiting probability has the property of:

$$d^0(s)\ge 0 \tag{A.2}$$

The initial state set has the property

$$\mathcal{D}=\Big\{s\big|d^0(s)>0\Big\} \tag{A.3}$$

According to the definition of possible initial state set, for those s with $d^0(s)=0$, $s\notin\mathcal{D}$. For those s with $d^0(s)=0$, i.e., $s\in\mathcal{D}$ the constraints in problem (A-SP) is equivalent to $\Big[v_C^{\pi}(s)-d\Big]\le 0$. Therefore, the constraint can be reformulated to

$$v_C^{\pi}(s)-d\le 0, \forall s\in\mathcal{D} \tag{A.4}$$

which is exactly the definition of feasible policy in Definition **??**. □

A.2 Proof of Corollary 1

There always not exists feasible solution means that at state s, the inequality always holds:

$$v_C^{\pi}(s)\ge d \tag{A.5}$$

According to the updating rules of the dual variables, the subgradient at s always satisfies $\hat{\partial}J_{\lambda}\ge 0$. As the subgradient is always positive, the multiplier finally goes to infinity.

A.3 Proof of Theorem 3

We regard the state as a random variable in optimizing problem (SP). Then for each state, we can construct a Lagrange function as

$$L\big(\pi,\lambda,s\big)=-v^{\pi}\big(s\big)+\lambda(s)\Big[v_C^{\pi}\big(s\big)-d\Big] \tag{A.6}$$

The Lagrange dual problem is

$$\max_{\lambda}\inf_{\pi} L(\pi,\lambda,s)=\max_{\lambda}\inf_{\pi}\Big\{-v^{\pi}(s)+\lambda v_C^{\pi}(s)\Big\} \tag{A.7}$$

We denote $G(\lambda,s)$ as the dual problem:

$$G(\mu,s)=\inf_{\pi}\Big\{-v^{\pi}(s)+\mu v_C^{\pi}(s)-d\Big\} \tag{A.8}$$

The expected Lagrangian optimize the expected solution of G.

Lemma 1 (Convex condition for infimum operation [4])**.** *If C is a convex nonempty set, the function f is convex in (x, y), then the infimum on y*

$$g(x) = \inf_{y \in C} f(x, y) \tag{A.9}$$

is concave.

Proposition 2. The dual problem of :

$$G(\mu, s) = \inf_{\pi} \Big\{ -v^{\pi}(s) + \mu v_C^{\pi}(s) - d \Big\} \tag{A.10}$$

is concave on $\mathcal{S}$.

Proof. According to the concave-convex assuption and the linear of convexity, for each μ,

$$-v^{\pi}\big(s\big) + \mu v_C^{\pi}\big(s\big) \tag{A.11}$$

is convex on (s, π). Therfore, according to Lemma 1, for each μ, $G(\mu, s)$ is concave on $\mathcal{S}$. □

Lemma 2 (Lower bound on deterministic equivalent [25])**.** *For a stochastic programming problem with the optimization variable x and random variable ω, if f is convex in ω for each x, then*

$$f(x, \mathbb{E}\omega) \leq \mathbb{E} f(x, \omega) \tag{A.12}$$

We use G_{exp}^* to denote the optimal solution of dual problems of (EL). Assume the π_{stw}^* is the optimal solution of the primal problem, we get

$$\begin{aligned} G_{exp}^* &= \max_{\mu} \Big\{ -\mathbb{E}_s v^{\pi^*}(s) + \mu \mathbb{E}_s v_C^{\pi^*}(s) \Big\} \\ &\geq \max_{\mu} \Big\{ -v^{\pi^*}(\mathbb{E}_s s) + \mu v_C^{\pi^*}(\mathbb{E}_s s) \Big\} = \max_{\mu} G(\mu, \mathbb{E}_s s) \end{aligned} \tag{A.13}$$

The inequality holds because the Jensen inequality under the concave-convex assumption. The statewise Lagrangian optimize the expected solution of L, i.e., assume the θ_{exp}^* is the corresponding infimum, we get

$$G_{stw}^* = \max_{\mu} \mathbb{E}_s \Big\{ -v^{\pi^*}(s) + \mu v_C^{\pi^*}(s) \Big\} = \max_{\mu} \mathbb{E}_s \big[G(\mu, s) \big] \leq \mathbb{E}_s \max_{\mu} \big[G(\mu, s) \big] \tag{A.14}$$

The inequality about expectation and max is analyzed by Thrun and Schwartz [29].

Proposition 3. The optimal solution of dual problem (A.10) $\max_{\mu} G(\mu, s)$ is concave on $\mathcal{S}$.

Proof. According to the property of dual problem, The domain of dual problem (A.10) is convex, and dual problem is concave on it [4]. Use Lemma 1 again, we get the concavity of $g^*(s)$. □

According to **Lemma 2**:

$$G_{exp}^* \geq \max_{\mu} G(\mu, \mathbb{E}_s s) \geq \mathbb{E}_s \max_{\mu} \big[G(\mu, s) \big] \geq G_{stw}^* \tag{A.15}$$

According to the concave-convex and the interior point assumptions, the strong duality condition is satisfied when the objective function is convex on θ. Therefore, the optimal solution of dual problem is exactly the optimal solution of primal problem. i.e., $G_{\diamond}^* = J(\theta_{\diamond}^*)$, where $\diamond$ includes exp or stw. Therefore,

$$J(\pi_{stw}^*) = \mathbb{E}_{s \sim d^{\pi}(s)} \Big\{ v^{\pi_{stw}^*}(s) \Big\} \geq \mathbb{E}_{s \sim d^{\pi}(s)} \Big\{ v^{\pi_{exp}^*}(s) \Big\} = J(\pi_{exp}^*) \tag{A.16}$$

□

B Additional Algorithm Details

B.1 Parameterized Functions

The paramterized functions are listed in Table 4.

Table 4: Parameterized Function

Function name	Function type	Notions	Weights
Value function	soft Q-function	$Q_\phi(s_t, a_t)$	ϕ
Cost value function	Q-function	$Q_{\phi_C}(s_t, a_t)$	ϕ_C
Policy	stochastic policy with $tanh$ bijector	$\pi_\theta(s_t)$	θ
Multipliers	multiplier function	$\lambda_\xi(s_t)$	ξ

B.2 Gradients Computation

Therefore, the update of cost Q-function is:

$$J_{Q_C}(\phi_C) = \mathbb{E}_{(s_t,a_t)\sim\mathcal{B}}\left[\frac{1}{2}\left(Q^{\phi_C}\left(s_t,a_t\right) - \left(r\left(s_t,a_t\right) + \gamma\mathbb{E}_{s_{t+1}\sim p, a_{t+1}\sim\pi}\left[Q_C^{\bar{\phi_C}}\left(s_{t+1},a_{t+1}\right)\right]\right)\right)^2\right] \tag{B.1}$$

The stochastic gradients of cost Q-function is

$$\hat{\nabla}_\theta J_{Q_C}(\phi_C) = \nabla_{\phi_C} Q_C^{\phi_C}\left(s_t,a_t\right)\left(Q_C^{\phi_C}\left(s_t,a_t\right) - \left(c(s_t,a_t) + \gamma Q_{\bar{\phi_C}}\left(s_t,a_t\right)\right)\right) \tag{B.2}$$

According to Theorem 1, the loss function for policy update is:

$$J_\pi(\theta) = \mathbb{E}_{s_t\sim\mathcal{B}}\Big[\mathbb{E}_{a_t\sim\pi_\theta}\Big[\alpha\log\left(\pi_\theta\left(a_t \mid s_t\right)\right) - Q_\phi(s_t,a_t) + \lambda_\xi(s_t)\big(Q_{\phi_C}(s_t,a_t) - d\big)\Big]\Big] \tag{B.3}$$

The stochastic policy gradient with the reparameterized policy $a_t = f_\theta(\epsilon_t; s_t)$ can be approximated by [14]:

$$\begin{aligned}\hat{\nabla}_\theta J_\pi(\theta) =& \nabla_\theta\alpha\log\left(\pi_\theta\left(a_t \mid s_t\right)\right) + \\ & \Big(\nabla_{a_t}\alpha\log\left(\pi_\theta\left(a_t \mid s_t\right)\right) - \nabla_{a_t}\big(Q_\phi\left(s_t,a_t\right) - \lambda_\xi(s_t)Q_{\phi_C}\left(s_t,a_t\right)\big)\Big)\nabla_\theta f_\theta\left(\epsilon_t; s_t\right)\end{aligned} \tag{B.4}$$

where the threshold d is neglected since it is irrelevant with θ. The loss function for updating multiplier net is the same as Equation (B.5)

$$J_\lambda(\xi) = \mathbb{E}_{s_t\sim\mathcal{B}}\Big[\mathbb{E}_{a_t\sim\pi_\theta}\Big[\alpha\log\left(\pi_\theta\left(a_t \mid s_t\right)\right) - Q_\phi(s_t,a_t) + \lambda_\xi(s_t)\big(Q_{\phi_C}(s_t,a_t) - d\big)\Big]\Big] \tag{B.5}$$

The stochastic subgradient can be approximated by

$$\hat{\partial} J_\lambda(\xi) = \big(Q_{\phi_C}(s_t,a_t) - d\big)\nabla_\xi\lambda_\xi(s_t) \tag{B.6}$$

where the entropy and reward value function term is neglected since they are irrelevant with the multipliers.

B.3 Pseudocode

The pseudocode of FAC is provided in Algorithm 1.

Algorithm 1 Feasible Actor-Critic

Input: ϕ^1, ϕ^2, ϕ_C^1, ϕ_C^2, θ, ξ. ▷ Initial parameters

$\bar{\diamondsuit} \leftarrow \diamondsuit$ for $\diamondsuit \in \{\phi^1, \phi^2, \phi_C^1, \phi_C^2, \theta\}$ ▷ Initialize target network weights

$\mathcal{B} \leftarrow \emptyset$ ▷ Initialize an empty replay buffer

for each iteration **do**

  for each environment step **do**

    $a_t \sim \pi_\theta(a_t|s_t), s_{t+1} \sim p(s_{t+1}|s_t, a_t)$ ▷ Sample transition from policy and environment

    $\mathcal{B} \leftarrow \mathcal{B} \cup \{(s_t, a_t, r(s_t, a_t), c(s_t, a_t), s_{t+1})\}$ ▷ Store the transition in the replay buffer

  end for

  for each gradient step **do**

    $\phi^i \leftarrow \phi - \beta_Q \hat{\nabla}_{\phi^i} J(\phi^i)$ for $i \in \{1, 2\}$ ▷ Update the Q-function weights

    $\phi_C \leftarrow \phi_C - \beta_Q \hat{\nabla}_{\phi_C} J_C(\phi_C)$ ▷ Update the cost Q-function weights

    if gradient steps $\texttt{mod}\ m_\pi = 0$ **then**

      $\theta \leftarrow \theta - \beta_\pi \hat{\nabla}_\theta J_\pi(\theta)$ ▷ Update policy weights

      $\alpha \leftarrow \alpha - \beta_\alpha \hat{\nabla}_\alpha J(\alpha)$ ▷ Adjust temperature

    end if

    if gradient steps $\texttt{mod}\ m_\lambda = 0$ **then**

      $\xi \leftarrow \xi + \beta_\lambda \hat{\partial}_\xi J_\lambda(\xi)$ ▷ Update multipliers weights[2]

    end if

    $\bar{\diamondsuit} \leftarrow \tau \diamondsuit + (1 - \tau)\bar{\diamondsuit}$ for $\diamondsuit \in \{\phi^1, \phi^2, \phi_C^1, \phi_C^2, \theta\}$ ▷ Update target network weights

  end for

end for

Output: ϕ^1, ϕ^2, ϕ_C^1, ϕ_C^2, θ, ξ.