Benchmarking Safe Policy Optimization for Constrained Reinforcement Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Safe reinforcement learning (safe RL) tackles decision making problems with safety constraints. Despite the influx of attention in this field, there is a lack of commonly recognized safe RL benchmark. This is partly because the code of many safe RL methods is unavailable and also because new methods often come with new testing tasks. As a result, researchers suffer from incorrect implementations, unfair comparisons, and misleading conclusions. In this study, we offer a solid safe RL benchmark—(`SafePO`)—which benchmarks popular safe policy learning algorithms across a list of common environments. Specifically, we start by standardizing the problem of safe exploration via solving constrained Markov decision processes (CMDP). Then, we provide implementations for CMDP solutions, covering both constrained policy optimization type methods and Lagrangian type methods. Our implementations in SafePO are highly efficient in the sense that learners can collect samples in parallel and synchronize their policy gradients on different physical CPU cores. We test them on four types of safety-aware robot learning tasks. Based on the benchmark results, we derive new insights by disclosing the interplay of different attributes on safety performance and illustrating the difficulty of safety learning on the sparse cost. Furthermore, to consider the safe RL problem in multi-agent settings, we introduce new tasks based on the DexterousHands environment and report comparison results for both single-agent and multi-agent safe RL algorithms. Our project is released at: https://github.com/PKU-MARL/Safe-Policy-Optimization.

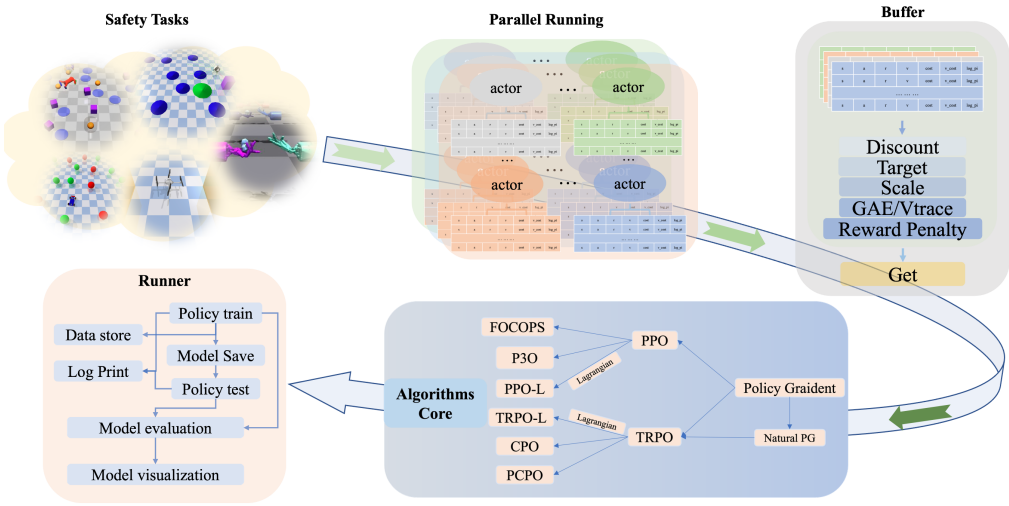
1 Introduction

Safety policy learning is critical in real-world reinforcement learning applications, and dangerous decisions are undesirable. For example, a robot agent should avoid playing actions that irrevocably harm its hardware [43, 21]. Given its importance, recently the community actively consider safe policy learning (e.g.,[2, 47, 28, 59, 27, 54, 37]). However, most existing work only focuses on algorithm design, and it still lacks an open standardized algorithm benchmark for empirical study and a comprehensive comparison for the widely used safety tasks. This results in difficulty reproducing the experimental reports, unfair comparisons, and unclear understanding existing algorithms.

To address the undesirable research status of safe reinforcement learning, we open an efficient safe policy optimization benchmark and provide some empirical insights for safe reinforcement learning.

Standardizing Safety Learning Tasks and Algorithms. We propose the safe policy optimization (`SafePO`), which benchmarks typical and widely used safety tasks and learning algorithms. This benchmark collects significant diversity in the safety tasks, including Robots with Limited Velocity, Robots with Circle, Robots with Goal, and Safety DexterousHands. For the algorithmic part, this

Submitted to the 36th Conference on Neural Information Processing Systems (NeurIPS 2022) Track on Datasets and Benchmarks. Do not distribute.



(a) Overview of safe policy optimization

Figure 1: The proposed safe policy optimization (`SafePO`) provides a rich environment interface, which can be used for different algorithms to verify performance on different tasks. `SafePO` contains 6 baseline algorithms includes PPO-L, TRPO-L, P3O [60], PCPO [57], FOCOPS [61], and CPO [1]. Particularly, the `SafePO` collects samples in parallel and synchronizes the policy gradients on different physical cores of the CPU, which makes `SafePO` efficient to train a tough safety learning task. According to the algorithm configuration, the framework performs customized preprocessing on the collected data to update the actors and critics of different algorithms. During the algorithm training process, we also provide a large amount of visual information to evaluate the algorithms.

benchmarks includes CPO [1], PCPO [57], PPO-L, TRPO-L, FOCOPS [61] and P3O [60]. The proposed `SafePO` standardizes the safe policy optimization via the main formalism with respect to the constrained Markov decision process [43], which provides a training framework from compiling algorithms to the evaluation of algorithms. The `SafePO` collects samples in parallel and synchronizes the policy gradients on different physical cores of the CPU, which makes `SafePO` efficient for training a tough safety learning task. Besides, the proposed `SafePO` is a scalable software package that can be extended to new safety tasks at scale, which makes it an accessible way to promote the reproducibility of the latest safety tasks.

Empirical Explanation for Safety Learning. We consider the key factors that influence the performance of the safety learning, where we provide a comprehensive empirical comparison to benchmark algorithms with 33 different safety robot tasks setting. Our empirical observations include:

1) The first observation is that there is no unambiguous winner: no single algorithm performs best among all the tasks for reward performance and cost limit in all safety tasks. But in general, the approaches implemented by a deep neural network (e.g., P3O) achieve a better performance than the convex approximation method (e.g., CPO); such an empirical observation is sharper when the task is more complex (e.g., Safety DexterousHands, see Figure 6).

2) We observe that the safety task of robots with limited velocity is an *ill defined task*, where a contradictory phenomenon occurs between reward improvement and constrained limited cost for the robots like Ant-v3,Hopper-v3,HalfCheetah-v3, and Walker2d-v3, see Figure 3, and 7.

3) We observe that the Lagrange parameter plays an important role in learning a safe policy for the agent, see Figure 4. If the policy violates the cost limit, the Lagrange parameter will be active and help the agent update the policy alongs with a safe direction. Otherwise, the Lagrange parameter keeps calm, which allows the agent improves the reward performance within the safety region.

4) We observe that it is very challenging for the agent to learn a safe policy if the cost is too sparse. The empirical result shows that: the agent learns little to improve the reward performance and always prefers to play an unsafe policy. We suspect that sparse cost implies a small cost limit, which causes a limited safe region. Thus, it is very difficult for the agent to search for a safe policy.

Safety DexterousHands Benchmark. We propose the safety DexterousHands benchmark, a new and challenging robot task for measuring safe policy optimization algorithms. Dexterous manipulation with hands to complete the complex task is one of the most challenging problems in the robotics [38], the proposed Safety DexterousHands tasks provide the way to test the capacity of the algorithm to manipulate the tough tasks. For a deeper comparison to the safe policy optimization algorithms, we present the performance with two settings: single-agent and multi-agent.

2 Related Work

Safety Algorithms. The work [42, 4, 35] provides concrete problems in AI safety, for example, avoiding reward hacking and safe exploration. This benchmark mainly consider the safety learning via the formulation of constrained Markov decision process [3], which is an active community in safe reinforcement learning (e.g., safe exploration [40, 8, 31], risk criteria [11, 12, 44, 45], or primal-dual method [14, 49, 41, 33]). For a comprehensive survey, see recent works [29, 10].

Reinforcement Learning Benchmark. Arcade Learning Environment (ALE) [7] provides the widely used Atari games with score-based reward functions. Open AI Gym [9] includes wide variety of classical control tasks: MuJoCo [50], Atari games high-dimensional continuous control. The work [48, 20, 18] also proposes continuous robotic control tasks. Deepmind Lab [6] provides a first-person 3D game platform designed for the research and development of general artificial intelligence and machine learning systems. CoinRun [15] is a procedurally generated environment, and [15] shows the generalization performance during the training process. Surreal [23] is an open-source reinforcement learning framework and robot manipulation benchmark. Clatech OPE [51] provides eight environments for the empirical study of off-policy policy evaluation. Pybullet [18] provides a framework that combines diverse learning paradigms (such as imitation and reinforcement learning). Assistive gym [22] is a physics simulation framework for assistive robotics. Brax[24] is a differentiable physics engine for large-scale rigid body simulation.

Safety Benchmark. AI Safety Gridworlds [35] provides a framework of safety reinforcement learning tasks with respect to grid world. The work [43] proposes Safety Gym that contains three robots for the empirical study of safe exploration in deep policy optimization. Bullet-safety-gym [27] provide a framework for safety specifications of Bullet [16] in constrained reinforcement learning problems. Safe-control-gym [59] provide an implementation for three dynamics. The work [29] considers safe multi-agent reinforcement learning benchmark.

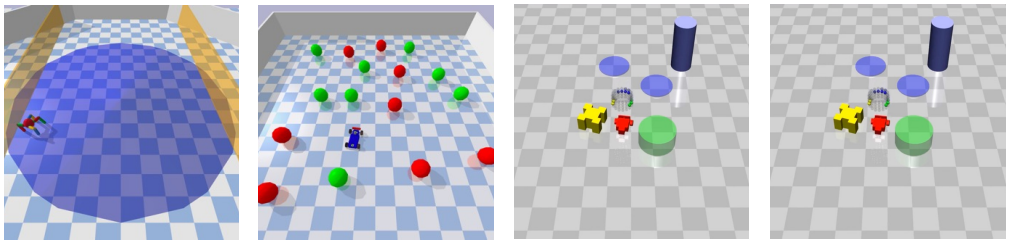
3 Safe Reinforcement Learning

The paper formulates safe reinforcement learning (RL) as a constrained Markov decision process (CMDP) [3], which is defined as $(\mathcal{S}, \mathcal{A}, \mathbb{P}, r, \rho_0, \gamma, \mathcal{C})$. Here $\mathcal{S}$ is state space, $\mathcal{A}$ is action space. $\mathbb{P}(s^{'}|s,a)$ state transition probability from s to $s^{'}$ after playing a. $r(s'|s,a)$ denotes the reward that the agent observes when state transition from s to $s^{'}$ after it plays a. $\rho_0(\cdot) : \mathcal{S} \to [0,1]$ is the initial state distribution and the discounter factor $\gamma \in (0,1)$. The constraint set $\mathcal{C} = \{(c,b)\}$, c is cost functions: $c : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$, and b is cost limit. A stationary policy π is a probability distribution defined on $\mathcal{S} \times \mathcal{A}$, we use Π to denote the set of all stationary policies. Let $\tau = \{s_t, a_t, r_{t+1}, c_{t+1}\}_{t \geq 0} \sim \pi$ be a trajectory generated by π, where $s_0 \sim \rho_0(\cdot)$, $a_t \sim \pi(\cdot|s_t)$, $s_{t+1} \sim \mathbb{P}(\cdot|s_t, a_t)$, $r_{t+1} = r(s_{t+1}|s_t, a_t)$, and $c_{t+1} = c(s_{t+1}|s_t, a_t)$. The *state value function* of π is defined as $V_\pi(s) = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t r_{t+1} | s_0 = s]$. The goal of reinforcement learning is to maximize $J(\pi)$ defined as: $J(\pi) = \mathbb{E}_{s \sim \rho_0(\cdot)}[V_\pi(s)]$. The *cost-return* is: $J^c(\pi) = \mathbb{E}_{s \sim \rho_0(\cdot)}[\sum_{t=0}^{\infty} \gamma^t c_{t+1} | s_0 = s]$, we define the feasible policy set $\Pi_{\mathcal{C}}$: $\Pi_{\mathcal{C}} = \{\pi \in \Pi \text{ and } J^c(\pi) \leq b\}$. The goal of CMDP is to search the optimal policy $\pi_\star$ such that

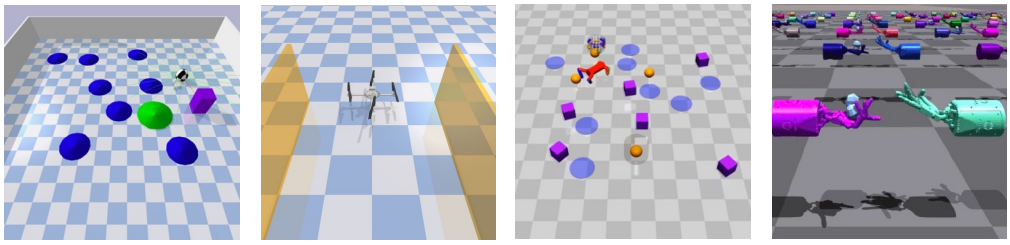
$$\pi_\star = \arg\max_{\pi \in \Pi_{\mathcal{C}}} J(\pi). \tag{1}$$

4 Safety Tasks

In this section, we present four safety robot tasks, including robots with limited velocity, robots with circle, robots with goal and safety dexterousHands. The safety dexterousHands are new safe learning tasks, which are design to make deeper comparisons to the safe algorithms and measure research progress on safe RL. For more details about safety dexterousHands, see Appendix D.



(a) Circle (b) Goal (c) Goal (d) Goal



(e) Reacher (f) Run (g) Button (h) DexterousHands

Figure 2: Fig.(a) Circle [1]: The robot is restricted to move in a given circle, but it is constrained to stay within a safe region more minor than the radius of the circle. Fig.(b), (c), (d) Goal [43]: Move the robot to a series of goal positions. When a goal is achieved, the goal location is randomly reset to someplace new while keeping the rest of the layout the same. Fig.(e) Reacher [27]: As soon the robot enters the goal zone, the goal is re-spawned such that the agent has to reach a series of goals. Fig.(f) Run [13]: Robots are rewarded for running through an avenue between two safety boundaries. Fig.(g) Button [43]: Press a series of goal buttons. Fig.(h) DexterousHands: Its task is to pass the ball from the right hand to the left hand, but some joints of the hand are freezing.

4.1 Task 1: Robots with Limited Velocity

We present the task of the robots with limited velocity according to [61, 60], where we train the robots to walk but imposed a limited velocity to the robots. We consider the tasks from MuJoCo [9]: Swimmer-v3, Walker2d-v3, Hopper-v3, HalfCheetah-v3, and Ant-v3. For the robots move on a two-dimensional plane, we calculate the cost as follows, $c(s,a) = \sqrt{v_x^2 + v_y^2}$; for the robots move move along a straight line, we calculate the cost as follows, $c(s,a) = |v_x|$, where v_x, v_y are the velocities of the agent in the x and y directions respectively.

4.2 Task 2: Robots with Circle

In the circle tasks, the goal is for an agent to move along the circumference of a circle while remaining within a safety region smaller than the radius of the circle. The reward and cost functions are:

$$R(s) = \frac{-yv_x + xv_y}{1 + |\sqrt{x^2 + y^2} - d|}, \quad C(s) = \mathbb{I}(|x| > x_{\text{lim}}),$$

where x, y are the positions of the agent on the plane, v_x, v_y are the velocities of the agent along the x and y directions, d is the radius of the circle, and x_{lim} specifies the range of the safety region. We set these parameters as follows: for Bullet Circle $d = 7$, $x_{\text{lim}} = 4.5$; for Humanoid circle $d = 10$, $x_{\text{lim}} = 2.5$, and for Ant circle $d = 10$, $x_{\text{lim}} = 3$.

4.3 Task 3: Robots with Goal

In this benchmark, we consider four tasks of Goal: see Figure 2 (b)-(g), where those tasks share a common character that the robots move to a series of goal positions. We have presented some basic details of those tasks in the caption of Figure 2. Now, we present the task of Button according to [43]. Several immobile "buttons" are scattered throughout the environment, and the agent should navigate to and press the currently-highlighted Button. After the agent presses the correct Button, the environment will select and highlight a new goal button, keeping everything else fixed.

4.4 Task 4: Safety DexterousHands

A robot consists of two fixed hands, where the right hand holds a ball, its task is to pass the ball to the left hand. At the beginning, the ball falls randomly in the area around the right side, the right hand grabs the ball, and passes the object to the left hand. The base of the hand is fixed. In this task, the hand that holds the object initially cannot directly touch the target, nor can it directly roll the object to the other hand, so the object must be thrown up and stays in the air in the process. There are 211-dimensional observations and 40-dimensional actions.

For each round t, let $x_{b,t}$ be the position of the ball and $x_{g,t}$ be the position of the goal, we use d_t to denote the distance between the ball and goal $d_t = \|x_{b,t} - x_{g,t}\|_2$. Let $d_{a,t}$ denote the angular position distance between the object and the goal, and the rotational difference $d_{r,t} = 2 \arcsin \min\{|d_{a,t}|, 1.0\}$. The the reward is defined as follows,

$$r_t = \exp\{-0.2(\alpha d_{d_t} + d_{r,t})\}, \tag{2}$$

where α is a constant balances translational and rotational rewards.

Safety Joint. See Figure 11, in this task, we constrain the freedom of the joint ④ of forefinger with a given region. The freedom angle of the joint ④ is among $[-15°, 90°]$, the safety task is to restrict to joint ④ within $[-7.5°, 45°]$. Let the `ang_4` be the angle the joint ④ plays, and the cost is defined as:

$$c_t = \mathbb{I}(\texttt{ang\_4} \notin [-7.5°, 45°]). \tag{3}$$

Safety Finger. In this task, we constrain the freedom of the joint ②, ③ and ④ of forefinger with a given region. The freedom of the joint ③ and ④ is among $[0°, 90°]$, the safety task is to restrict to joint ④ within $[0°, 45°]$. Let the `ang_2`, `ang_3` be the angle the joint ②, ③ plays, and the cost is:

$$c_t = \mathbb{I}(\texttt{ang\_2} \notin [0°, 45°], \text{ or } \texttt{ang\_3} \notin [0°, 45°], \text{ or } \texttt{ang\_4} \notin [-7.5°, 45°]). \tag{4}$$

5 Algorithms

In this benchmark, we consider representative safe policy optimization algorithms. Appendix A contains all the key steps of those algorithms.

CPO (Constrained Policy Optimization). CPO [1] suggests to replace the reward objective $J(\pi)$ and the cost constraint $J^c(\pi)$ with surrogate functions, which evaluates the objective $J(\pi)$ and constraint $J^c(\pi)$ according to the samples collected from the current policy π_k as follows,

$$\pi_{k+1} = \arg\max_{\pi \in \Pi_{\boldsymbol{\theta}}} \quad \mathbb{E}_{s \sim d^{\rho_0}_{\pi_k}(\cdot), a \sim \pi_{\boldsymbol{\theta}}(\cdot|s)} \left[A_{\pi_k}(s,a)\right] \tag{5}$$

$$\text{s.t. } J^c(\pi_k) + \frac{1}{1-\gamma} \mathbb{E}_{s \sim d^{\rho_0}_{\pi_k}(\cdot), a \sim \pi_{\boldsymbol{\theta}}(\cdot|s)} \left[A^c_{\pi_k}(s,a)\right] \leq b, \tag{6}$$

$$\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}, \pi_k) = \mathbb{E}_{s \sim d^{\rho_0}_{\pi_k}(\cdot)}[\mathrm{KL}(\pi_{\boldsymbol{\theta}}, \pi_k)[s]] \leq \delta, \tag{7}$$

where π_k is short for $\pi_{\boldsymbol{\theta}_k}$. For the practical implementation, CPO introduces convex approximations to replace the term $A_{\pi_k}(s,a)$, $A^c_{\pi_k}(s,a)$, and $\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}, \pi_k)$, which is similar to TRPO [46].

PCPO (Projection-based Constrained Policy Optimization). PCPO [57] is an iterative method for optimizing policies in a two-step process: the first step performs a local reward improvement update, while the second step reconciles any constraint violation by projecting the policy back onto the constraint set. For the practical implementation, CPO inherits the key idea from TRPO [46] and CPO [1], where PCPO also use convex approximations to the reward objective and cost constraint.

FOCOPS (First Order Constrained Optimization in Policy Space). FOCOPS [61] finds the optimal update policy by solving a constrained optimization problem in the non-parameterized policy space, then it projects the update policy back into the parametric policy space.

TRPO-L and **PPO-L**. The Lagrangian approach is a standard way to solve CMDP (1), which is also known as primal-dual policy optimization:

$$(\pi_\star, \lambda_\star) = \arg\min_{\lambda \geq 0} \max_{\pi \in \Pi_{\boldsymbol{\theta}}} \left\{J(\pi) - \lambda(J^c(\pi) - b)\right\}. \tag{8}$$

TRPO-Lagrangian and PPO-Lagrangian combine the Lagrangian approach with TRPO and PPO. Concretely, PPO using the following clip term to replace $J(\pi)$ in (8),

$$\mathcal{L}^r_{\text{clip}} = \mathbb{E}_{s\sim d^{\rho_0}_{\pi_k}(\cdot), a\sim\pi_k(\cdot|s)}\left[-\min\left\{\frac{\pi_{\boldsymbol{\theta}}(a|s)}{\pi_k(a|s)}A_{\pi_k}(s,a), \text{clip}\left(\frac{\pi_{\boldsymbol{\theta}}(a|s)}{\pi_{\boldsymbol{\theta}_k}(a|s)}, 1-\epsilon, 1+\epsilon\right)A_{\pi_k}(s,a)\right\}\right].$$

With $A_{\pi_k}(s,a)$ replacing $A^c_{\pi_k}(s,a)$ respectively, and obtain $\mathcal{L}^c_{\text{clip}}$. Let $\mathcal{L}^c_{\text{clip}}$ replace $J(\pi)$, PPO-L updates the policy according to first-order optimizer. TRPO-L shares a similar idea but it is adaptive to TRPO, due to the limitation of space, we provide the details in Appendix A.5.

P3O (Penalized Proximal Policy Optimization). P3O [60] solves the cumbersome constrained policy iteration via a single minimization of an equivalent unconstrained problem:

$$\pi_{k+1} = \arg\min_{\pi\in\Pi_{\boldsymbol{\theta}}}\left\{\mathbb{E}_{s\sim d^{\rho_0}_{\pi_k}(\cdot), a\sim\pi_k(\cdot|s)}\left[\frac{\pi(a|s)}{\pi_k(a|s)}A_{\pi_k}(s,a)\right] + \kappa B(\pi, b)\right\}, \tag{9}$$

where κ is a positive scalar, and the penalty term $B(\pi, b)$ is defined as follows,

$$B(\pi,b) = \max\left\{0, \mathbb{E}_{s\sim d^{\rho_0}_{\pi_k}(\cdot), a\sim\pi_k(\cdot|s)}\left[\frac{\pi(a|s)}{\pi_k(a|s)}A^c_{\pi_k}(s,a)\right] + (1-\gamma)\left(J^c(\pi_k) - b\right)\right\}. \tag{10}$$

P3O utilizes a simple yet effective penalty approach to eliminate cost constraints and removes the trust-region constraint by the clipped surrogate objective.

6 Empirical Observations

In this section, we present the key factors influence the performance of the safety policy learning on a comprehensive empirical comparison over 33 different safety robot tasks setting.

6.1 What Algorithm is Winner?

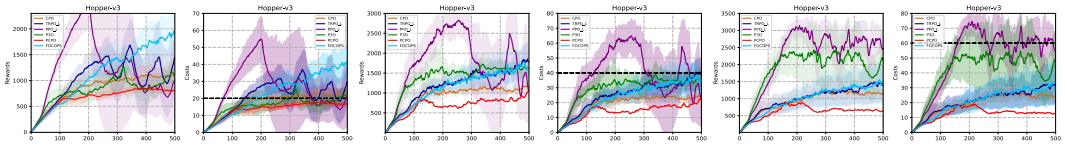
The first observation is that *there is no unambiguous winner*: no single algorithm performs best among all the tasks for both reward performance and cost limit in all of the safety tasks, see Figure 3-6. In general, the approaches implemented by a deep neural network (e.g., P3O) achieve a better performance than the convex approximation method (e.g., CPO); such an empirical observation is sharper when the task is more complex (e.g., Safety DexterousHands, see Figure 6). For more comprehensive empirical comparisons, see Appendix C.

CPO, TRPO-L, and PCPO obtain a relatively worse empirical performance than other algorithms. This is not an accidental phenomenon due to the essential implementation mechanism of CPO, TRPO-L, and PCPO. In practice, CPO and PCPO use convex approximations to replace $A_{\pi_{\boldsymbol{\theta}_k}}(s,a)$ and $\bar{D}_{\text{KL}}(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_k})$ Eq.(5)-(7). Concretely, CPO suggests to use first-order Taylor expansion to replace (5)-(6), uses second-oder approximation to replace (7). Those convex approximations turn a non-convex problem (5)-(7) to be a convex problem, it seems to make a simple solution, but this approach results in many error sources and troubles in practice. It still lacks a theory analysis to show the difference between the non-convex problem (5)-(7) and its convex approximations Policy optimization is a typical non-convex problem [56, 55]. Those convex approximations may introduce some errors for the original issue. TRPO-L and PCPO inherit key ideas concerning convex approximations from CPO, making TRPO-L and PCPO perform relatively worse in some complex safety tasks.

Instead of using a convex approximation to the objective function, FOCOPS, PPO, and P3O directly optimize the surrogate objective function via the first-order method, and it does not depend on any convex approximation. Additionally, the first-order method effectively avoids the expensive computation for the inverse Fisher information matrix in CPO, PCPO, and TRPO-L. Those have partially explained why FOCOPS, PPO, and P3O obtain better performance.

6.2 Comments on Robots with Limited Velocity

An important observation is that the safety task of robots with limited velocity is an *ill defined task*, where a contradictory phenomenon occurs between reward improvement and constrained limited cost for the robots like Ant-v3,Hopper-v3, HalfCheetah-v3, and Walker2d-v3.



(a) Cost limit = 20 (b) Cost limit = 40 (c) Cost limit = 60

Figure 3: Learning curves for Hopper-v3 with limited velocity tasks. The shaded region represents the standard deviation of the score over three trials. Curves are smoothed uniformly for visual clarity. The x-axis represents the epoch, and each epoch updates the policy with 80×10^4 iterations.

Concretely, for example, the reward of Hopper-v3 is mainly determined by the following term [1]

$$r_t(s,a) \propto \frac{\|(x_{t'}, y_{t'}) - (x_t, y_t)\|_2}{t' - t} \approx \sqrt{v_x^2 + v_y^2} = c_t(s,a), \tag{11}$$

where (x_t, y_t) is the two-dimensional coordinate of the robot at time t, and $r_t(s,a)$ denotes the reward with respect to the robot transforms from (x_t, y_t) to $(x_{t'}, y_{t'})$ after playing action a. The relationship (11) shows that the reward function is proportional to the cost function. Recall the goal of reinforcement learning is to search a policy improves the cumulative reward while controls the boundedness of the cumulative cost with a given limit b, which is very difficult to be reconciled between the reward improvement and the bounded cost limit. Additionally, from Eq.(11), we also know that the learning curve of reward performance and cost performance can be very similar potentially. The result of Figure 3 is consistent with our analysis, where the reward and cost learning curve shares a very similar shape. Besides, from Figure 3, we know the reward increases while the cost also increases, which is undesirable for safety learning.

Thus, the safety task of robots with limited velocity is ill-defined; for more empirical results, see Figure 7. Such a worse case also occurs in the robots of Ant-v3, HalfCheetah-v3, and Walker2d-v3 since those robots also share a similar formulation (11). For more discussion, see Appendix B.

6.3 Comments on Lagrange Multiplier

We observe that the Lagrange parameter plays an important role for the agent in learning a safe policy. If the policy violates the cost limit, the Lagrange parameter will be active and help the agent update the policy alongs with the safe direction. Otherwise, the Lagrange parameter keeps calm, which allows the agent improves the reward performance within the safety region.

Recall (8), the Lagrange method (e.g., PPO-L or TRPO-L) consider the following update rule of λ:

$$\lambda_{k+1} = \{\lambda_k + \eta(\hat{J}_k^c - b)\}_+, \tag{12}$$

where $\hat{J}_k^c$ is an estimator for the cost function, $\eta > 0$ is step-size. Those empirical result shown in Figure 4 is consistent with the update rule of λ in Eq.(12). If the estimated cost function is under the target threshold b, then λ keeps calm, which implies λ is not activated. Such an empirical phenomenon gives significant expression to the Humanoid environment. While if the estimated cost exceeds the target threshold b, λ will be activated, which requires the agent to play a policy on the safe region. This implies Lagrange parameter plays an important role for the agent to learn a safe policy. Additionally, from the above discussion, we know Figure 4 provides a visualization way to show the difficulty of different tasks, where the task actives much quantification of λ, such a task is more challenging to obtain a safe policy since the agent needs more trials and errors.

6.4 Sparse Cost

We observe that it is very challenging for the agent to learn a safe policy if the cost is too sparse. From Figure 5, in the SafetyDroneCircle, TRPO-L, FOCOPS, CPO, and PCPO achieve a very low reward performance. This is because the total cost limit is tiny, so the robot obtains a small cost for each step, implying the robot knows very little about the environment. Thus the robot pays much effort into exploring the environment to find a safe policy.

[1] We present the details with respect to reward according to the open implementation: https://www.gymlibrary.ml/environments/mujoco/hopper/.

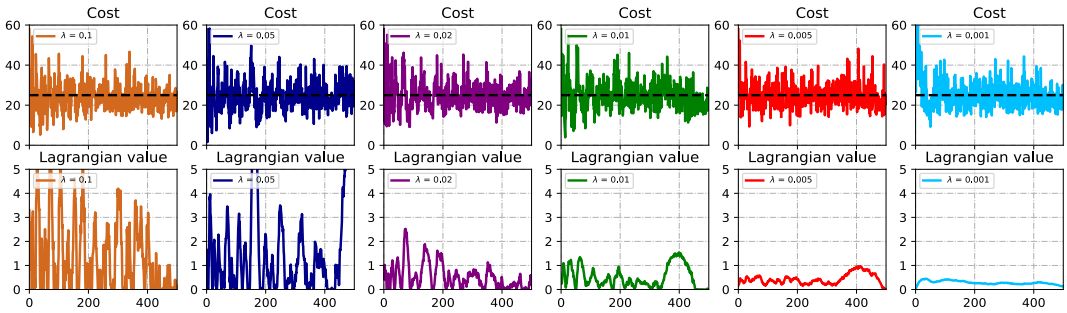
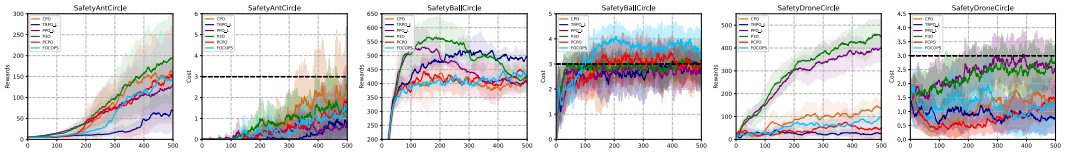


Figure 4: Cost constraint with respect to the Lagrangian hyper-parameter λ.



(a) Cost limit = 3 (b) Cost limit = 3 (c) Cost limit = 3

Figure 5: Learning curves for SafetyAntCircle, SafetyBallCircle, and SafetyDroneCircle.The shaded region represents the standard deviation of the score over three trials. Curves are smoothed uniformly for visual clarity.

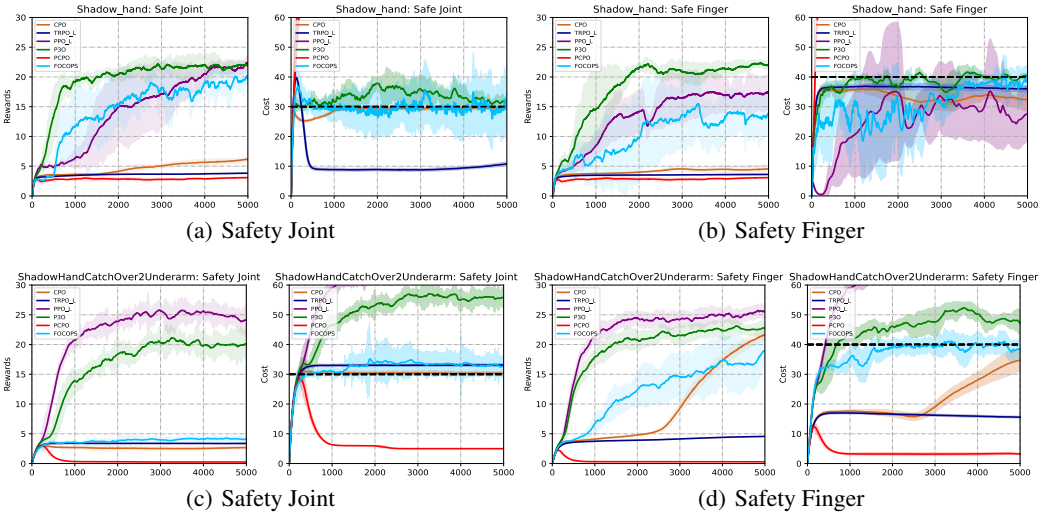
A more general observation is that the cost helps the robot search for a safe policy. From Figure 5, we know the algorithm near the cost limit usually obtains better performance than that far away from the cost limit. This is because the higher cost implies the robot learns more about the environment, which helps the robot understand a safer policy. This observation further illustrates that cost plays a quantization measurement of exploration. Additionally, we know that if the cost limit is too large, the task could be reduced to an unconstrained policy learning since the robot explores the environment with a huge cost. Thus, we should consider the trade-off between a small and large cost limit for real-world applications to design a safety policy learning algorithm.

7 Safe Robot Learning in Dexterous Hands

Dexterous manipulation with hands to complete complex task is one of the most challenging problem in the robotics [38]. In this section, we propose the Safety DexterousHands tasks, which is a new and challenging robot task for measuring research progress on safe reinforcement learning. For a deeper comparison to the safe algorithms, in this section we present the performance with two settings: single-agent and multi-agent.

In contrast to single agent RL , in Multi-Agent Reinforcement Learning (MARL) [28], it's more difficult to optimise policy because of the unstability of multi-agent systems. In MARL settings, when an agent optimises its policy, it needs to consider other agents' policy simultaneously. When it comes to safe MARL settings, it's more challenging to improve policy performance than purely reward optimisation, since it not only needs to improve agents' reward, but also needs to guarantee agents' safety. We provide a safe MARL baseline benchmark for safe MARL research on challenging tasks of safety DexterousHands, in which the MACPO [28], MAPPO-L [28], MAPPO [58], HAPPO [32], IPPO [17] are all implemented to investigate the safety and reward performance. MACPO is developed based CPO and HATRPO [32] by leveraging contrained trust region optimisation and multi-agent sequential decision theory, MACPO can achieve hard constrained optimisation to ensure safety for multi-agent systems; MAPPO-L is developed based on primal-dual methods, which can achieve soft constrained policy optimisation to improve the safety of multi-agent systems.

Single-Agent Safety Learning. Figure 6 and Table 4 show the results of single-agent safe RL algorithm over the Safety Joint and Safety Finger tasks. In general, although the tasks are all two-hand cooperation, Safety Joint outperforms Safety Finger in most cases, which implies finger manipulation is more difficult than joint manipulation. This observation seems to opposite the human behavior: a person makes a better finger manipulation than a single joint, which implies that a task



(a) Safety Joint (b) Safety Finger

(c) Safety Joint (d) Safety Finger

Figure 6: Learning curves for Single-agent Safety DexterousHands. The shaded region represents the standard deviation of the score over three trials. Curves are smoothed uniformly for visual clarity.

is difficult for human but it is simple for a machine. Besides, we observe that FOCOPS and P3O outperforms CPO and PCPO significantly, this is because the previous task is relatively easy. The advantage of our VRMPO becomes more powerful when the task is more difficult. CPO and PCPO obtain a relatively worse empirical performance than other algorithms. This is not an accidental phenomenon due to the essential implementation mechanism of CPO and PCPO. Instead of using a convex approximation for the objective function, FOCOPS and P3O directly optimizes the surrogate objective function via the first-order method, and it does not depend on any convex approximation.

Multi-Agent Safety Learning. The empirical study reveals the findings that MACPO and MAPPO-lagrangian algorithms can achieve comparable reward performance compared with the state-of-art MARL algorithms while guaranteeing safety of agents. Figure 12 shows performance comparisons on tasks of Safe ShadowHandOver 2x6 task two in terms of cost and reward, the results indicate that MACPO and MAPPO-lagrangian algorithms are the only ones that learn the safety constraints compared with MAPPO, HAPPO and IPPO, while achieving satisfying performance in terms of the reward. This is consistent with what has been found in [28] work.

8 Conclusion and Future Work

In this study, we standardize the safe policy optimization method for solving constrained Markov decision processes and introduced the first benchmark named SafePO. We implemented constrained policy optimization (CPO) type algorithms and Lagrangian based algorithms and test them on four kinds of safety-aware robotic tasks. Moreover, we design Safety DexterousHands tasks for safe multi-agent RL tasks. Importantly, we design SafePO in a way that it can collects samples in parallel and synchronizes the policy gradients on different physical CPU cores, this makes SafePO highly efficient for training a safe policy. On top of the benchmark comparison, we also highlight the interplay of different attributes on the safety performance, and illustrate the difficulty of safety learning on the sparse cost.

In this benchmark, we only consider the safety policy learning under the framework of constrained Markov decision process, which is one category of safe reinforcement learning [25]. In the future, we will consider the following problems for safety policy learning. 1) **Zero Constraint Violation**: In real-world applications, for example, autonomous vehicle [53, 36] or power systems [26, 52] it is catastrophic if the system plays violations of constraints [5]. Thus, achieving its goal guarantees zero constraint violation, an important problem for safety learning. 2) **State-wise Safety Learning**. This benchmark only considers relaxed trajectory-wise constraints. However, those approaches may fail to satisfy the hard state-wise safety constraints [19], where the state-wise constraint satisfaction is more practical than safe trajectories, which yields a constrained optimization problem at each decision-making step.

References

[1] Joshua Achiam, David Held, Aviv Tamar, and Pieter Abbeel. Constrained policy optimization. In *Proceedings of International Conference on Machine Learning (ICML)*, volume 70, pages 22–31, 2017.

[2] Mohammed Alshiekh, Roderick Bloem, Rüdiger Ehlers, Bettina Könighofer, Scott Niekum, and Ufuk Topcu. Safe reinforcement learning via shielding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

[3] Eitan Altman. *Constrained Markov decision processes*. CRC Press, 1999.

[4] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul Christiano, John Schulman, and Dan Mané. Concrete problems in ai safety. *arXiv preprint arXiv:1606.06565*, 2016.

[5] Qinbo Bai, Amrit Singh Bedi, Mridul Agarwal, Alec Koppel, and Vaneet Aggarwal. Achieving zero constraint violation for constrained reinforcement learning via primal-dual approach. *arXiv preprint arXiv:2109.06332*, 2021.

[6] Charles Beattie, Joel Z Leibo, Denis Teplyashin, Tom Ward, Marcus Wainwright, Heinrich Küttler, Andrew Lefrancq, Simon Green, Víctor Valdés, Amir Sadik, Julian Schrittwieser, Keith Anderson, Sarah York, Max Cant, Adam Cain, Adrian Bolton, Stephen Gaffney, Helen King, Demis Hassabis, Legg Shane, and Stig Petersen. Deepmind lab. *arXiv preprint arXiv:1612.03801*, 2016.

[7] Marc G Bellemare, Yavar Naddaf, Joel Veness, and Michael Bowling. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

[8] Felix Berkenkamp, Matteo Turchetta, Angela Schoellig, and Andreas Krause. Safe model-based reinforcement learning with stability guarantees. *Advances in neural information processing systems*, 30, 2017.

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

[10] Lukas Brunke, Melissa Greeff, Adam W Hall, Zhaocong Yuan, Siqi Zhou, Jacopo Panerati, and Angela P Schoellig. Safe learning in robotics: From learning-based control to safe reinforcement learning. *Annual Review of Control, Robotics, and Autonomous Systems*, 5, 2021.

[11] 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.

[12] Yinlam Chow, Ofir Nachum, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

[13] Yinlam Chow, Ofir Nachum, Aleksandra Faust, Edgar Duenez-Guzman, and Mohammad Ghavamzadeh. Lyapunov-based safe policy optimization for continuous control. *arXiv preprint arXiv:1901.10031*, 2019.

[14] Yinlam Chow, Ofir Nachum, and Mohammad Ghavamzadeh. Path consistency learning in tsallis entropy regularized mdps. In *International Conference on Machine Learning (ICML)*, pages 978–987, 2018.

[15] Karl Cobbe, Oleg Klimov, Chris Hesse, Taehoon Kim, and John Schulman. Quantifying generalization in reinforcement learning. In *International Conference on Machine Learning*, pages 1282–1289. PMLR, 2019.

[16] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning (2016–2019), 2019.

[17] Christian Schroeder de Witt, Tarun Gupta, Denys Makoviichuk, Viktor Makoviychuk, Philip HS Torr, Mingfei Sun, and Shimon Whiteson. Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*, 2020.

[18] Brian Delhaisse, Leonel Rozo, and Darwin G Caldwell. Pyrobolearn: A python framework for robot learning practitioners. In *Conference on Robot Learning*, pages 1348–1358. PMLR, 2020.

[19] Priya L Donti, David Rolnick, and J Zico Kolter. Dc3: A learning method for optimization with hard constraints. *arXiv preprint arXiv:2104.12225*, 2021.

[20] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning (ICML)*, pages 1329–1338, 2016.

[21] Gabriel Dulac-Arnold, Daniel Mankowitz, and Todd Hester. Challenges of real-world reinforcement learning. *arXiv preprint arXiv:1904.12901*, 2019.

[22] Zackory Erickson, Vamsee Gangaram, Ariel Kapusta, C Karen Liu, and Charles C Kemp. Assistive gym: A physics simulation framework for assistive robotics. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10169–10176. IEEE, 2020.

[23] Linxi Fan, Yuke Zhu, Jiren Zhu, Zihua Liu, Orien Zeng, Anchit Gupta, Joan Creus-Costa, Silvio Savarese, and Li Fei-Fei. Surreal: Open-source reinforcement learning framework and robot manipulation benchmark. In *Conference on Robot Learning*, pages 767–782. PMLR, 2018.

[24] C Daniel Freeman, Erik Frey, Anton Raichuk, Sertan Girgin, Igor Mordatch, and Olivier Bachem. Brax–a differentiable physics engine for large scale rigid body simulation. *arXiv preprint arXiv:2106.13281*, 2021.

[25] Javier Garcıa and Fernando Fernández. A comprehensive survey on safe reinforcement learning. *Journal of Machine Learning Research*, 16(1):1437–1480, 2015.

[26] Istemihan Genc, Ruisheng Diao, Vijay Vittal, Sharma Kolluri, and Sujit Mandal. Decision tree-based preventive and corrective control applications for dynamic security enhancement in power systems. *IEEE Transactions on Power Systems*, 25(3):1611–1619, 2010.

[27] Sven Gronauer. Bullet-safety-gym: Aframework for constrained reinforcement learning. 2022.

[28] Shangding Gu, Jakub Grudzien Kuba, Munning Wen, Ruiqing Chen, Ziyan Wang, Zheng Tian, Jun Wang, Alois Knoll, and Yaodong Yang. Multi-agent constrained policy optimisation. *arXiv preprint arXiv:2110.02793*, 2021.

[29] Shangding Gu, Long Yang, Yali Du, Guang Chen, Florian Walter, Jun Wang, Yaodong Yang, and Alois Knoll. A review of safe reinforcement learning: Methods, theory and applications. *arXiv preprint arXiv:2205.10330*, 2022.

[30] Jemin Hwangbo, Joonho Lee, Alexey Dosovitskiy, Dario Bellicoso, Vassilios Tsounis, Vladlen Koltun, and Marco Hutter. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 4(26):eaau5872, 2019.

[31] Torsten Koller, Felix Berkenkamp, Matteo Turchetta, and Andreas Krause. Learning-based model predictive control for safe exploration. In *Conference on Decision and Control (CDC)*, pages 6059–6066. IEEE, 2018.

[32] Jakub Grudzien Kuba, Ruiqing Chen, Munning Wen, Ying Wen, Fanglei Sun, Jun Wang, and Yaodong Yang. Trust region policy optimisation in multi-agent reinforcement learning. *arXiv preprint arXiv:2109.11251*, 2021.

[33] Hoang Le, Cameron Voloshin, and Yisong Yue. Batch policy learning under constraints. In *International Conference on Machine Learning (ICML)*, pages 3703–3712, 2019.

[34] Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science robotics*, 5(47):eabc5986, 2020.

[35] Jan Leike, Miljan Martic, Victoria Krakovna, Pedro A Ortega, Tom Everitt, Andrew Lefrancq, Laurent Orseau, and Shane Legg. Ai safety gridworlds. *arXiv preprint arXiv:1711.09883*, 2017.

[36] Quanyi Li, Zhenghao Peng, Zhenghai Xue, Qihang Zhang, and Bolei Zhou. Metadrive: Composing diverse driving scenarios for generalizable reinforcement learning. *arXiv preprint arXiv:2109.12674*, 2021.

[37] Zuxin Liu, Zijian Guo, Zhepeng Cen, Huan Zhang, Jie Tan, Bo Li, and Ding Zhao. On the robustness of safe reinforcement learning under observational perturbations. *arXiv preprint arXiv:2205.14691*, 2022.

[38] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021.

[39] Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Science Robotics*, 7(62):eabk2822, 2022.

[40] Teodor Mihai Moldovan and Pieter Abbeel. Safe exploration in markov decision processes. *arXiv preprint arXiv:1205.4810*, 2012.

[41] Santiago Paternain, Luiz FO Chamon, Miguel Calvo-Fullana, and Alejandro Ribeiro. Constrained reinforcement learning has zero duality gap. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

[42] Martin Pecka and Tomas Svoboda. Safe exploration techniques for reinforcement learning–an overview. In *International Workshop on Modelling and Simulation for Autonomous Systems*, pages 357–375. Springer, 2014.

[43] Alex Ray, Joshua Achiam, and Dario Amodei. Benchmarking Safe Exploration in Deep Reinforcement Learning. 2019.

[44] Reazul Hasan Russel, Mouhacine Benosman, and Jeroen Van Baar. Robust constrained-mdps: Soft-constrained robust policy optimization under model uncertainty. *arXiv preprint arXiv:2010.04870*, 2020.

[45] Harsh Satija, Philip Amortila, and Joelle Pineau. Constrained markov decision processes via backward value functions. In *International Conference on Machine Learning (ICML)*, pages 8502–8511, 2020.

[46] John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International Conference on Machine Learning (ICML)*, pages 1889–1897, 2015.

[47] 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.

[48] Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy P. Lillicrap, and Martin A. Riedmiller. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

[49] Chen Tessler, Daniel J Mankowitz, and Shie Mannor. Reward constrained policy optimization. *International Conference on Learning Representation (ICLR)*, 2019.

[50] 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.

[51] Cameron Voloshin, Hoang M Le, Nan Jiang, and Yisong Yue. Empirical study of off-policy policy evaluation for reinforcement learning. *arXiv preprint arXiv:1911.06854*, 2019.

[52] Thanh Long Vu, Sayak Mukherjee, Tim Yin, Renke Huang, Jie Tan, and Qiuhua Huang. Safe reinforcement learning for emergency load shedding of power systems. In *2021 IEEE Power & Energy Society General Meeting (PESGM)*, pages 1–5. IEEE, 2021.

[53] Lu Wen, Jingliang Duan, Shengbo Eben Li, Shaobing Xu, and Huei Peng. Safe reinforcement learning for autonomous vehicles through parallel constrained policy optimization. In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–7. IEEE, 2020.

[54] Long Yang, Jiaming Ji, Juntao Dai, Yu Zhang, Pengfei Li, and Gang Pan. Cup: A conservative update policy algorithm for safe reinforcement learning. *arXiv preprint arXiv:2202.07565*, 2022.

[55] Long Yang, Gang Zheng, Haotian Zhang, Yu Zhang, Qian Zheng, and Gang Pan. Policy optimization with stochastic mirror descent. *Association for the Advancement of Artificial Intelligence (AAAI)*, 2022.

[56] Long Yang, Qian Zheng, and Gang Pan. Sample complexity of policy gradient finding second-order stationary points. In *Association for the Advancement of Artificial Intelligence (AAAI)*, 2021.

[57] Tsung-Yen Yang, Justinian Rosca, Karthik Narasimhan, and Peter J Ramadge. Projection-based constrained policy optimization. In *International Conference on Learning Representation (ICLR)*, 2020.

[58] Chao Yu, Akash Velu, Eugene Vinitsky, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative, multi-agent games. *arXiv preprint arXiv:2103.01955*, 2021.

[59] Zhaocong Yuan, Adam W Hall, Siqi Zhou, Lukas Brunke, Melissa Greeff, Jacopo Panerati, and Angela P Schoellig. safe-control-gym: a unified benchmark suite for safe learning-based control and reinforcement learning. *arXiv preprint arXiv:2109.06325*, 2021.

[60] Linrui Zhang, Li Shen, Long Yang, Shixiang Chen, Bo Yuan, Xueqian Wang, Dacheng Tao, et al. Penalized proximal policy optimization for safe reinforcement learning. *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 2022.

[61] Yiming Zhang, Quan Vuong, and Keith Ross. First order constrained optimization in policy space. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, 2020.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A]
   (b) Did you include complete proofs of all theoretical results? [N/A]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [Yes]
   (c) Did you include any new assets either in the supplemental material or as a URL? [Yes]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [Yes]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]