ReDMan: Reliable Dexterous Manipulation with Safe Reinforcement Learning

Yiran Geng[1,2†], Jiaming Ji[1,2†], Yuanpei Chen[1,2†], Haoran Geng[1,2], Fangwei Zhong[1,2] and Yaodong Yang[1,2*]

[1]Peking University, Beijing, China.
[2]National Key Laboratory of General Artificial Intelligence and Beijing Institute for General Artificial Intelligence, Beijing, China.

*Corresponding author(s). E-mail(s): yaodong.yang@pku.edu.cn;
Contributing authors: gyr@stu.pku.edu.cn; jiamg.ji@gmail.com; yuanpei.chen312@gmail.com; ghr@stu.pku.edu.cn; zfw@pku.edu.cn;
†These authors contributed equally to this work.

Abstract

Dexterous hand manipulation is a crucial ability for robots in various applications. However, ensuring safety and reliability during manipulation poses significant challenges. Safe Reinforcement Learning (Safe RL) algorithms are important to ensure robust performance and prevent damage to the robotic hand, manipulated object, or environment. Realistic and complex simulation platforms are needed to develop and evaluate such algorithms. Unfortunately, existing platforms have limitations in terms of realism, complexity, and customizability. To address these issues, we introduce **ReDMan**, an open-source simulation platform that provides a standardized implementation of safe RL algorithms for **Re**liable **D**exterous **Man**ipulation. ReDMan features challenging tasks based on real-world scenarios that require safety awareness, such as Jenga, as well as multi-modal observations and customizable robotic hardware. This platform facilitates the replication and comparison of experimental results and demonstrates the effectiveness of safe RL methods compared to classical RL algorithms. ReDMan is the first benchmark for safe dexterous manipulation and aims to bridge the gap between safe RL and dexterous manipulation research. The code and demonstration can be found at https://github.com/PKU-MARL/ReDMan.

Keywords: Manipulation, Safety, Reinforcement Learning, Dexterous Hands

1 Introduction

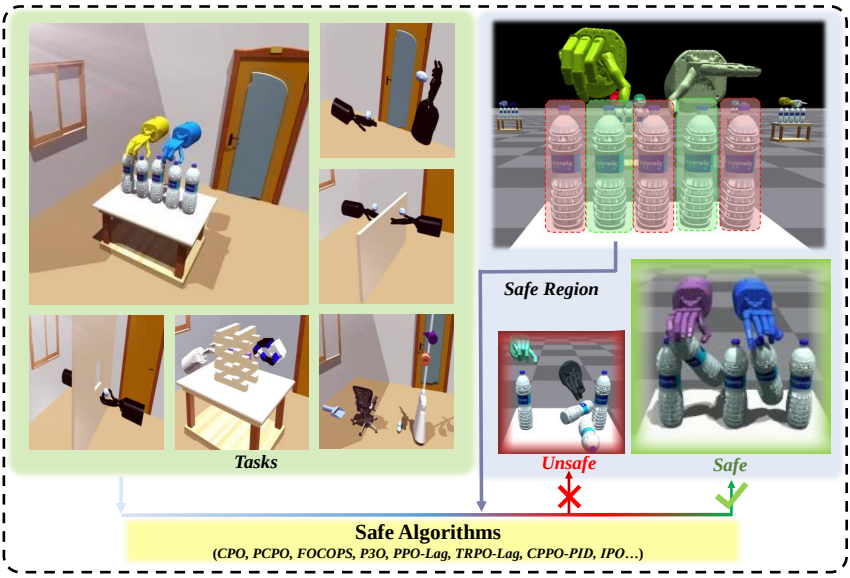

Fig. 1 Overview. In ReDMan, we design a variety of dexterous hand tasks and try to guarantee the safe completion of the tasks using the safe algorithm.

Dexterity is a fundamental aspect of human interaction with the physical world, facilitated by multi-finger manipulators that enable everyday activities such as typing, door opening, Rubik's cube manipulation, and Jenga [1]. In recent years, there has been a growing interest in the development of dexterous manipulation techniques for robots, which is critical for improving robotics and human-robot interaction [2–4]. Exploring dexterous manipulation techniques can also provide insights into the neural and physiological mechanisms that underlie complex motor skills. To create more capable and versatile robotic systems that can assist humans in various domains, including service, entertainment, education, and manufacturing, it is essential to enhance the dexterity and intelligence of robotic hands. Therefore, research focused on improving the dexterity and intelligence of robotic hands is crucial for realizing the full potential of robotics in enhancing human-machine collaboration [5, 6].

Recently, reinforcement learning (RL) has emerged as a highly successful approach for achieving dexterous manipulation in robotics tasks [1, 5, 7–13], surpassing traditional control methods that rely on specific assumptions. The principle of RL is that an agent seeks to maximize cumulative returns through

trial and error, which potentially leads to dangerous or harmful behaviors. Existing RL methods often ignore these safety learning issues, where failure is acceptable and even desirable to learn from bad outcomes. However, in the real world, such exploration can produce undesired consequences, making safety a crucial consideration. With robot manipulation, especially in the challenging task of multi-fingered manipulation, has become a vital application area for RL, safety in RL requires greater attention, as emphasized in recent research [14, 15]. Specifically, dexterous robots are highly complex and require precise control, making them susceptible to errors and malfunctions [5, 12]. Additionally, the sophisticated hardware required for their operation is prone to damage [16, 17], further emphasizing the need for safety. In real-world applications where dexterous robots interact with humans or the environment, safety and trustworthiness become even more critical [18].

Due to its importance, the community has been actively researching safe policy learning (e.g.,[19–25]). However, most of the existing work mainly focuses on algorithm design. Among these works, either the authors did not publish the source code (e.g. P3O [26]), or the algorithms were implemented using different frameworks(e.g. PCPO [27] in Theano [28], CPPO-PID [20] in PyTorch), with divergent approaches (FOCOPS [29] does not parallelize sample collection while others do), and on separate tasks (FOCOPS is tested solely on MuJoCo-Velocity [30] and CPPO-PID solely on Safety-Gym [16]). While there exists safety-starter-agents [16] as a publicly available collection of algorithms, it was implemented using TensorFlow1, required old hardware and system, lack recent updates, and was no longer maintained. As a result, the Safe RL community has experienced serious difficulty in reproducing the experimental results, comparing algorithms fairly, and deriving correct insights. An open-source, standardized algorithm implementation for algorithm verification and empirical study is desperately needed.

To facilitate the consideration we mentioned above and fill the research gap, we developed a simulation platform with a unified re-implementation of Safe RL algorithms for **Re**liable **D**exterous **Man**ipulation, namely **ReDMan**. We highlight three particularly desirable features of ReDMan:

- **For Safe RL researchers.** We present a novel framework for Safe RL algorithms that is unified, highly optimized, and extensible, with re-implementations of commonly used algorithms that support ReDMan and popular environments. Our framework emphasizes abstraction and encapsulation to encourage code reuse and maintain a clean code style. We also provide a suite of intricate and demanding safe dexterous manipulation tasks that are motivated by the requirement for safe robotic manipulation in daily life scenarios, such as cleaning floors without damaging furniture. Exhaustive experiments with our implemented algorithms were conducted in these environments, and we share the results, observations, and analyses to benefit the Safe RL community.
- **For Robotic researchers.** We present the inaugural compilation of tasks aimed at safe dexterous manipulations. Along with safety considerations,

we offer various features, such as multi-modal observation information that includes contact force, RGB image, RGB-D image, point cloud, and more. Additionally, our platform boasts customizable dexterous hands and a robotic arm drive specifically tailored to the dexterous hand. These features collectively provide an all-encompassing platform for robotic research.

2 Related Work

2.1 Environments for Safe RL

Simulator plays a critical role in the training for RL since it is very expensive to collect data in the real world. Safety-gym [31] introduces a robot that has to navigate through a cluttered environment to achieve a task, which is a suite of complex continuous control environments for Safe RL. Safe-control-gym [22] introduces cart-pole, 1D, and 2D quadrotor dynamic systems to achieve control tasks like stabilization or trajectory tracking, which allows us for constraint specification and disturbance injection onto a robot's inputs, states, and inertial properties through a portable configuration system. AI Safety Gridworlds [32] proposes an environment for evaluating various safe properties of intelligent agents, including safe interruptibility, avoiding side effects, safe exploration, distributional shift, etc. MuJoCo-Velocity, originally proposed in [33], consists of a series of safety tasks like constrained velocity based on MuJoCo environment [30]. However, there still lacks a safe environment for safe robot manipulation, which the difficulty lies in requiring safe high-dimensional continuous space control and dealing with the dynamic environment. So we introduce ReDMan, which aims to apply Safe RL to dexterous manipulation, providing a more challenging environment for evaluating Safe RL algorithms.

2.2 Safe RL Algorithms

Safe Reinforcement Learning (Safe RL) seeks to enable robots to perform complex tasks safely and effectively by finding a balance between maximizing expected total rewards and avoiding harmful or negative actions. This is critical in real-world applications such as robotics and autonomous systems, where safety is a primary concern. Safe RL aims to develop algorithms and methods that ensure safety constraints are not violated while achieving desired tasks [34, 35]. With the rise of deep RL, CMDPs are also moving to more high-dimensional continuous control problems. CPO [36] presents a general-purpose policy search algorithm that guarantees near-constraint satisfaction at each iteration. PCPO [37] uses a two-step approach to maximize the return and then projects the policy back into the safety region in terms of the minimum KL divergence. FOCOPS [33] adopts a similar idea, directly solving the constrained policy optimization problem via the primal-dual approach and projecting the solution back into the parametric policy space. Traditional robot

control also considers the safety problem. [38] presents a method via constructing the Lyapunov function to guarantee constraint satisfaction during training. [39] combines PID control with Lagrangian methods which dampen cost oscillations resulting in reduced constraint violations. It is still lacking a unified and efficient framework to cover these algorithms. Therefore, we provide PyTorch-version re-implementations of widely used safe policy optimization algorithms, hoping to facilitate experimental validation in Safe RL research.

2.3 Dexterous Manipulation

Manipulation is one of the essential research topics in robotics, and researchers have long tried to establish a stable theory of manipulation [7]. However, traditional methods mostly rely on various assumptions, such as knowing the environmental dynamics model or having no uncertainty in the process. In recent years, learning-based approaches have been successful in this regard, coping with uncertainty in perception and even generalizing to unseen objects [40]. There are many learning-based benchmarks for robotic manipulation in recent years [41–43], but none of them use dexterous hands or consider safe constraints. Dexterous multi-finger hands provide intrinsic dexterity for better manipulation in unstructured scenes and contact-rich situations, but additionally, bring the challenges of high-dimensional control and complex contact models [10, 11]. Previous research methods have mostly focused on trajectory optimization or model prediction, which highly relied on accurate dynamics models [1, 8, 9]. For example, [44] performs in-hand manipulation of a cube using a trajectory optimization technique known as Model Predictive Path Integral (MPPI). [45] extended the MPPI method to allow objects to be thrown and catch between two hands. [12] solved a Rubik's cube using model-free RL and domain randomization techniques. [13] proposed an in-hand manipulation system to learn how to manipulate a large number of objects of different shapes, and even generalize to unseen objects. [46, 47] studied dexterous manipulation learning from human demonstration. [48] studied the bimanual dexterous manipulation to solve cooperative manipulation and skill generalization problem. While most of them focus on unconstrained dexterous manipulation, how to do dexterous manipulation safely is an unstudied topic. In this paper, we provide a massively parallel benchmark for safe dexterous manipulation, hoping to facilitate research on how to manipulate safely.

3 The Safety Learning Environment

ReDMan is comprised of two fundamental components: safe learning environments and safe policy optimization algorithms. This section primarily focuses on elucidating the high-level design of safe learning environments.

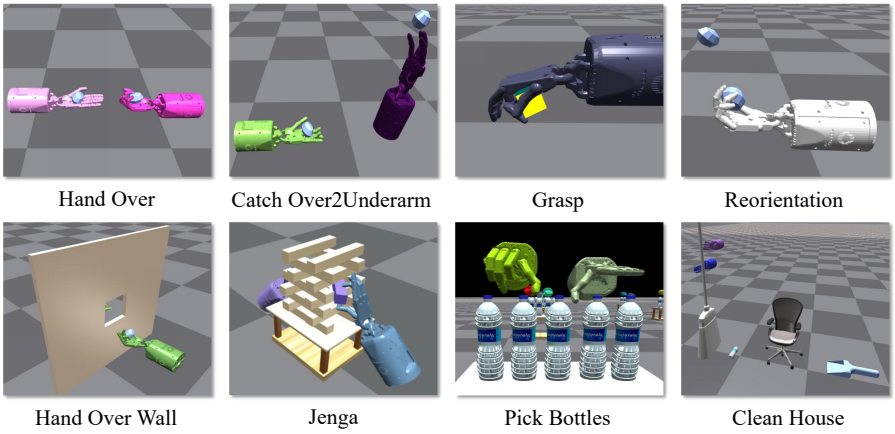

Fig. 2 Eight representative tasks of ReDMan, including Hand Over, Catch Over2Underarm, Reorientation, Hand Over Wall, Jenga, Pick Bottles, Clean House.

3.1 Problem Formulation

In general, Constrained Markov Decision Processes (CMDPs) is defined as $(\mathcal{S}, \mathcal{A}, \mathbb{P}, r, \rho_0, \gamma, \mathcal{C})$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathbb{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition probability function, $r : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the reward function, $\rho_0(\cdot) \in \mathcal{P}(\mathcal{S})$ is the initial state distribution ($\mathcal{P}(X)$ denotes the set of probability distributions over a set X), $\gamma \in [0, 1)$ is the discount factor, and $\mathcal{C} = \{(c_i, b_i)\}_{i=1}^m$ is the constraint set, where $c_i : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the cost function, and b_i is the cost threshold. Without loss of generality, we will restrict our discussion to the case of one constraint with a cost function c and upper bound b.

We use $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$ to denote a stationary policy and 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 the expected total reward, defined as $J(\pi) = \mathbb{E}_{s \sim \rho_0(\cdot)}[V_\pi(s)]$.

We define the cost return function as

$$J^c(\pi) = \mathbb{E}_{s \sim \rho_0(\cdot)} \left[\sum_{t=0}^{\infty} \gamma^t c_{t+1} | s_0 = s \right],$$

and the feasible policy set $\Pi_{\mathcal{C}}$ as $\Pi_{\mathcal{C}} = \{\pi \,|\pi \in \Pi, J^c(\pi) \leq b, \forall (c, b) \in \mathcal{C}\}$. The goal of Safe RL is to learn the optimal policy $\pi^\star$ such that

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

3.2 System Design and Datasets

In ReDMan, there is a collection of challenging dexterous manipulation tasks, underpinned by Isaac Gym [49] and capable of high parallelism on the GPU. All tasks require two dexterous hands to manipulate one or more objects. We design a series of tasks that require policies to perform safe dexterous manipulation, including throwing, grasping, jerking, pulling, etc. At the same time, each task provides the customizability of dexterous hands and objects to support a diverse task.

The construction of the dataset includes the configuration of the robot arm, dexterous hands, and objects. The core goal of our dataset is to **generate a wide variety of scenarios for learning constrained dexterous manipulation**. We collected a variety of robot arms and dexterous multi-finger hands as manipulators, including most of the arms and dexterous hands currently used in robotics. In addition to manipulators, objects also play a crucial role in building datasets. Our manipulation objects are mainly from the YCB [50] and SAPIEN [51] datasets. Both datasets contain many objects used in everyday life.

ReDMan contains 10+ tasks focused on dexterous manipulation. Each task contains one or two dexterous hands and one or more manipulated objects, such as balls, blocks, etc., with the ultimate goal being to manipulate objects placed at the task-specified locations while making the agent satisfied. The default dexterous hand used by our framework is the Shadow Hand [52], more details are provided in Appendix A. The agent performs each task according to its observation, action representation, and reward and cost function definition. We provide more underlying technical details about the tasks in Appendix B.

Observation Space. Here we briefly describe the observation space of the tasks, more details can be seen in Appendix B.1. The observation of all tasks consisted of two or three parts: state information of the single or dual Shadow Hands, and information about the task specification. In each task, the state information of the Shadow Hand is the same, each Shadow Hand contains 24 minimum drive units (which contains four underdriven fingertip units) and its state consists of the following information:

- $\mathcal{D}_p, \mathcal{D}_v, \mathcal{D}_f \in \mathbb{R}^{24}$, corresponds to all joint $\mathcal{D}o\mathcal{F}$ (Degree of Freedom) of angle, velocity, and force with drive units, respectively.
- $\mathcal{P}_w, \mathcal{R}_w \in \mathbb{R}^3$ represents the position and rotation of the base of the hand.
- $\mathcal{FT}_i = [FT_{\text{pose}}, FT_{v_l}, FT_{v_a}, FT_f, FT_t] \in \mathbb{R}^{19}$, corresponds to the pose, linear velocity, angular velocity, force magnitude, and torque of each fingertip, respectively.
- $\mathcal{A} \in \mathbb{R}^{20/26}$, indicates the action executed by the hand in the previous step, which is consistent with the action space.

With the above definitions, the state information of one Shadow Hand can be represented as $\mathcal{H}and = \{\mathcal{D}_p, \mathcal{D}_v, \mathcal{D}, \mathcal{P}_w, \mathcal{R}_w, \{\mathcal{FT}_i\}_{i=1}^5, \mathcal{A}\}$. We character

the observation of each task by the following information:

$$\{\mathcal{H}and_{\text{left}},\ \ \mathcal{H}and_{\text{right}},\ \ \mathcal{G}_{\text{task}}\}, \tag{2}$$

where $\mathcal{G}_{\text{task}}$ represents some observation information specific to different tasks.

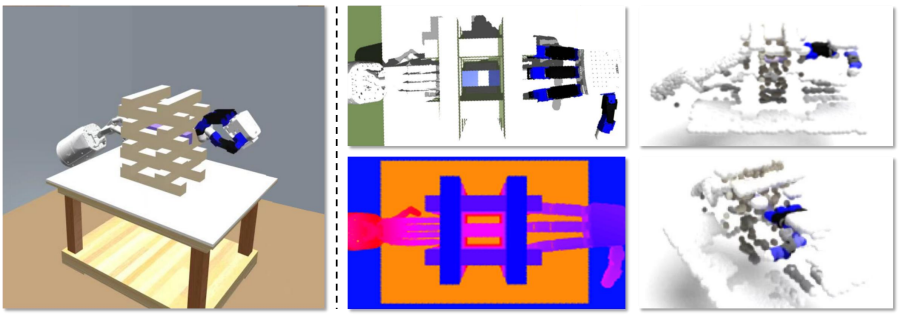

Fig. 3 **Left**: the original scene in the simulation; **Right**: the RGB view and depth view of the scene; the point cloud views of the scene.

Visual Observation. It is very difficult to obtain the state information of the robot in the real world. One way to solve this problem is to use the vision sensor as the input to train the policy. Therefore, we provide multiple modalities of visual information as input, including RGB, RGB-D, and point cloud, see Fig. 3. It is generated using the camera in the Isaac Gym, and the pose and toward of the camera can be customized by the user to obtain the desired visual observation.

We also propose a point cloud parallel acceleration function to adapt Isaac Gym and provide an example of using it to train the Hand Over task. We replace the object state information with point clouds in the case of 128 parallel environments. The point cloud is captured by the depth camera and down-sampled to 2048 points. The features are extracted using PointNet [53] to a 128-dimensional vector and concated with other observations. It can be seen that under the same episode and the same number of environments, the performance of point cloud input is not as good as full-state input, but it can also achieve some performance. But also using an RTX 3090 GPU, the point cloud RL has only 200+ fps, and the full state can reach 30000+. In fact, we can only open up to 128 environments when using point clouds. This was a problem with Isaac Gym's poor parallel support for cameras. We further refined the method to enhance the parallelization of the point cloud extraction in order to close this gap. When compared to Isaac Gym's original code, the speedup is 1.46 times, going from 232 fps to 339 fps.

Action Space. The single or dual Shadow Hands have 26 or 52 dimensions of action space, where each Shadow Hand has five fingers with a total of 24 degrees of freedom, the thumb has 5 joints and 5 degrees of freedom, and all

other fingers have 3 degrees of freedom and 4 joints (where the joint at the end of each finger is uncontrollable). Therefore, the action space of each hand is 20 dimensions. If the base of the hand is not fixed, there are six dimensions to represent the translation and rotation of the hand base. For the lower and upper limits of the joint angle, see Tab. A1. In each step, we use the absolute value of each joint angle as the target and use the PD controller to make it move.

Reward. We designed some auxiliary rewards to help RL agents learn more consistently, and each task contains a task-specific bonus. In general, our reward design is goal-based and follows the same set of logic. For object-catching tasks, our reward is simply related to the difference between the pose of the object and the target. For other tasks that require the hand to hold the object, our reward generally consists of two or three parts: the distance from each hand to a grip point on the object, and the distance from the object to the object's target.

Cost. Each task contains different constraints (e.g., the ball needs to be thrown to a specified height or a specified angle to prevent damage to other items; the robots need to clean the floor without hitting other furniture). The specific constraint design depends on the safety requirements of each task.

3.3 Safety Constraints

Safety constraints possess varying implications across diverse tasks. This study classifies safety constraints into two distinct categories based on their reliance on the robot's state:

- **Constraints that are contingent on the robot's state.** Constraints are contingent on the robot's state that addresses its safe operation, including limitations on certain joints and motor strength restrictions. The results of this experiment indicate that conventional algorithms, such as MPC [54], perform more effectively and with greater stability when dealing with constraints related to the robot's state stability.
- **Constraints that depend on the environment's state.** Constraints are reliant on the environment's state that consider potential dangers arising from the robot's interaction with its surroundings. For example, when completing the Jenga task, the robot arm must ensure the stability of the entire wooden block structure, or when cleaning a house, it must avoid colliding with fragile objects. In Human-Robotic collaborative control scenarios [55, 56], the robot must avoid harmful behaviors that could harm people or objects while performing delicate tasks. The fulfillment of these constraints is critical for the successful deployment of robots. The study findings indicate that reinforcement learning methods for security based on deep learning, such as PPO-Lag [31], perform well for tasks with large state spaces.

3.4 Diverse Tasks

ReDMan has a collection of 10+ different safe tasks. These tasks form an evaluation suite for benchmarking the performance of Safe RL algorithms. Fig. 2 shows the environment for some of the tasks. For more details, please refer to Appendix B.

Hand Over. In this task, one hand needs to throw the ball to the other hand. The task places emphasis on the limitations encountered within a specific behavioral framework, such as the restrictions imposed on joint degrees of freedom, and necessitates the consideration of the collective functioning of the entire hand-driven mechanism.

Catch Over2Underarm. In this task, akin to Hand Over, the objective entails the propulsion of an object from a hand held in a vertical position to one that is oriented in a palm-upward direction. The task at hand needs a heightened level of safety to facilitate the attainment of greater accuracy in the throwing direction.

Grasp. The task at hand requires the use of a singular dexterous hand to aptly seize an object situated in a terrestrial location and raise it aloft. Safety considerations pertaining to the act of grasping, such as constraints in joint mobility and range of motion, are emphasized in this task.

Reorientation. The objective of the task is to align the orientation of the object held in the hand with the intended target orientation, while concurrently avoiding the application of excessive force by the dexterous hand, which may lead to the undesirable crushing of the object, exemplified by the egg.

Hand Over Wall. In this task, the dexterous hand is tasked with executing a similar throwing maneuver to that of Hand Over. However, the inclusion of a wall with minimal clearance between the hands necessitates that the object being thrown traverse through the barrier seamlessly to achieve the desired goal.

Jenga. The accomplishment of the designated task necessitates the cooperative involvement of both dexterous hands in the extraction of specific blocks from an unsteady stack formation while concurrently mitigating any inadvertent disruption to the integrity of the remaining blocks.

Pick Bottles. The task involves the retrieval of two bottles from a closely spaced row of five bottles, with the dual hands executing the maneuver smoothly and without any inadvertent contact with the remaining bottles.

Clean House. Within the given environment, the task necessitates the coordinated usage of both hands to manipulate the broom for the purpose of sweeping debris into the designated dustpan, while concurrently overcoming obstacles such as the presence of a chair.

In the present work, we propose a classification of constraints in two distinct categories as described in Section 3.3. By incorporating these constraints, we formulate different safe tasks that can be performed with a dexterous hand:

- The first category is defined as **tasks with robot's state constraints**, which encompasses the control of the joints and fingers of the hand. To

this end, we define four safety tasks as follows: (a) Safety Joint task of Hand Over, (b) Safety Finger task of Hand Over, (c) Safety Joint task of CatchOver2Underarm, and (d) Safety Finger task of CatchOver2Underarm.

- The second category is described as **tasks with environment's state constraints**. These tasks focus on environmental constraints, such as avoiding damage to external objects while performing the task. We define six tasks that belong to this category: (a) Grasp, (b) Reorientation, (c) Hand Over Wall, (d) Jenga, (e) Pick Bottles, and (f) Clean House.

3.5 Customizable Dexterous Hands

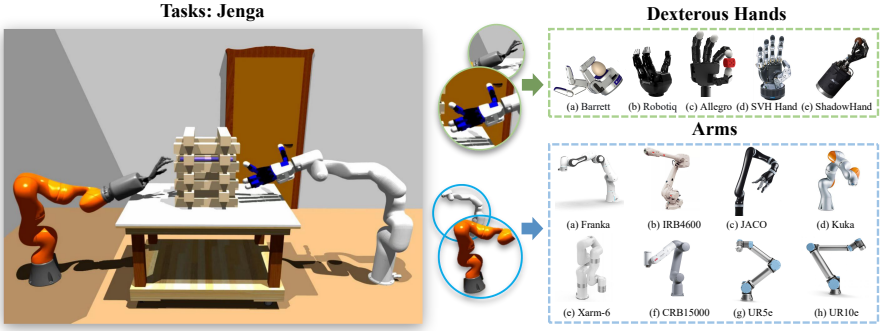

Fig. 4 Using different dexterous hands and robot arms in ReDMan provides diversity. The left image is the dexterous hands with a robot arm driver doing the Jenga task, the demo can be found in https://github.com/PKU-MARL/ReDMan. where the left hand is Kuka connected with Shadow Hand, and the right hand is X-arm6 connected with Allegro Hand. Among them, we support five kinds of dexterous hands shown in the upper right corner, and eight kinds of robot arms shown in the lower right corner, which can be customized by the user.

The availability of multiple types of dexterous hands, including but not limited to the Shadow Hand, such as the Allegro Hand and Tri-Finger, is critical for promoting research and community development in this field. To further facilitate progress, ReDMan offers not only the Shadow Hand but also an additional selection of five dexterous multi-finger hands.

Incorporating a robotic arm drive at the base of the dexterous hand not only mimics real-world conditions but also represents an essential step in achieving successful sim-to-real transfer. Due to the inherent difficulty in reproducing the true dynamics of a flying hand, the ReDMan platform simplifies the deployment process from simulation to real-world applications by permitting adjustments to the dynamics and physics parameters of the arm to minimize the reality gap.

Furthermore, the provision of a variety of arms and dexterous hand combinations yields several benefits. For instance, researchers can select the most appropriate hand for their specific needs, thereby enhancing the versatility of our benchmark. Moreover, the use of diverse arms and hands enables the

study of policy adaptability and generalization, which poses a challenge for future multi-task learning and meta-learning research. A schematic diagram illustrating this feature is depicted in Figure 4.

4 Experiments

4.1 Safe RL Algorithms Implementation

Based on their original papers or public code base, we re-implement eight algorithms (CPO [57], PCPO [27], FOCOPS [29], P3O [26], PPO-Lag [16], TRPO-Lag [16], CPPO-PID [20], and IPO [58]), covering major safe policy optimization algorithms. A brief introduction to each algorithm is given in Appendix D.

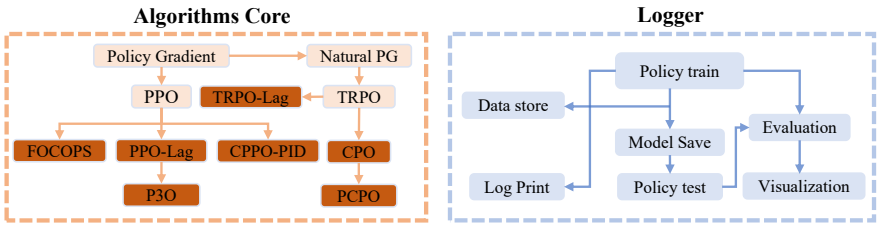

Fig. 5 An overview of algorithms core design and logger.

We abstract a similar structure of the safe policy optimization algorithms and modularize the code into interaction with environments, parallel sample collection, buffer storage, and computation, algorithm core update, and auxiliary functionalities such as visualization and logger. Maximum abstraction and encapsulation take place at the implementation of the core of the algorithms, where each algorithm inherits directly from its base algorithm, thus only unique features have to be implemented and all other code can be reused. An overview of the core of algorithms and logger is shown in Fig. 5.

For algorithms implementation, it is critical to ensure its correctness and reliability. To achieve this goal, we examine the implementation of our algorithms carefully. To test the performance of our implementation, we run the eight algorithms on 30 tasks (for a complete list of the tasks, please refer to Appendix E.1) contained in the four environment suites and present our experimental results for the reference of the community in Appendix E.2.

4.2 Evaluation Protocol

$$\max_{\pi_\theta} \mathop{\mathbb{E}}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T} r_t\right], \text{ s.t. } \mathop{\mathbb{E}}_{\tau\sim\pi_\theta}\left[\sum_{t=0}^{T} c_t\right] \leq b \tag{3}$$

Metrics. We define the following metrics to depict the safety performance of an agent in different tasks. (1) the average return of trajectories, $J^r(\theta)$; (2) the average cumulative cost of trajectories, $J^c(\theta)$. In the Safe RL domain, for any two agents, the superiority of the agents is determined by the following priority comparisons. On the one hand, the agent that satisfies the constraint will definitely outperform the unconstrained one. On the other hand, two agents that satisfy the constraint are determined by comparing the magnitude of their cumulative returns.

Algorithms. For the UnSafe RL algorithm, we uniquely use PPO [59], where the reward function contains no information about the auxiliary costs. For Safe RL algorithms, we evaluate the performance of PPO-Lag [31], FOCOPS, P3O, and PCPO algorithms on ReDMan, and the remaining Safe RL algorithms we implemented are in our anonymous GitHub repository.

4.3 Results

To elucidate the distinctive traits of various safe RL methodologies, subject to two distinct forms of constraint, we conducted empirical validation and conducted separate analyses of the outcomes. Additionally, we endeavored to optimize the training process utilizing point clouds as observations, demonstrating the practicability of utilizing safe RL algorithms for visually-oriented or high-dimensional input tasks. Specifically, our experiments and analysis are divided into the following categories:

(1) The performance of Safe RL algorithms on four tasks with robot's state constraints.
(2) The performance of Safe RL algorithms on the tasks with the environment's state constraints.
(3) The performance of point cloud RL on the Hand Over Wall task.

For (1), we tested the entire eight algorithms we implemented on tasks with the robot's state constraints, which is shown in Tab. 1 and Fig. 6. We the specific details can be found in Appendix C.

Different algorithms exhibit varying performances in distinct environments. The CPO method ensures that the final policy converges within the feasible domain for all four tasks. However, second-order methods like CPO demonstrate inferior reward acquisition compared to PPO-based methods. This disparity can be attributed to CPO's reliance on multiple approximations during the implementation process, including the use of conjugate gradients for Hessian matrix calculations, and more conservative updates to satisfy safety constraints. Such an approach restrains the need for higher reward performance to a certain extent.

Lagrangian-based techniques, such as TRPO-Lag, PPO-Lag, and FOCOPS, generate dissimilar outcomes across tasks while also satisfying constraint requirements. For instance, PPO-Lag achieves high rewards while satisfying constraints in the ShadowHand Finger task, but exceeds safety constraints by a factor of two in the ShadowHand Joint task. FOCOPS achieves

high rewards in ShadowHand Joint but experiences significant cost oscillations due to the instability of Lagrangian multiplier updates in the primal-dual framework, which heavily relies on initial values and update steps. Suboptimal initial values or smaller update steps can impede policy convergence within the feasible domain during the entire training process, while larger update steps may result in unstable algorithm updates.

The CPPO-PID, a PID-Lagrangian-based algorithm, partially mitigates the issue of traditional Lagrangian's poor stability by using PID-Control to update the Lagrangian multiplier. On the other hand, IPO, a penalty-based method, exhibits varying performance in different environments due to the distinct nature of the interior-point method of the penalty function, which necessitates the initialized policy to lie within the feasible domain.

Table 1 Performance on robot's state constraints: The color red is indicative of a constraint violation, while bold text signifies that the optimal reward has been achieved under the given constraint.

	HandOver_Finger		HandOver_Joint		HandOver Underarm_Finger		HandOver Underarm_Joint	
Performance	Reward ↑	Cost(≤ 40)	Reward ↑	Cost (≤ 30)	Reward ↑	Cost (≤ 40)	Reward ↑	Cost (≤ 30)
CPO	4.57 ± 0.02	32.41 ± 0.01	**6.18 ± 0.01**	29.54 ± 0.02	**21.63 ± 0.01**	34.72 ± 0.01	2.68 ± 0.01	29.3 ± 0.02
TRPO-Lag	3.62 ± 0.01	36.02 ± 0.01	3.83 ± 0.01	10.77 ± 0.02	4.55 ± 0.03	15.6 ± 0.01	3.37 ± 0.02	33.13 ± 0.01
PPO-Lag	**17.4 ± 0.04**	27.69 ± 0.01	22.44 ± 0.04	71.15 ± 0.02	25.5 ± 0.01	65.56 ± 0.02	24.11 ± 0.03	64.39 ± 0.05
P3O	21.93 ± 0.07	40.54 ± 0.02	21.9 ± 0.03	31.34 ± 0.02	22.72 ± 0.0	47.3 ± 0.02	20.09 ± 0.05	55.84 ± 0.09
PCPO	3.08 ± 0.02	70.15 ± 0.01	3.08 ± 0.12	70.15 ± 0.01	0.3 ± 0.01	3.22 ± 0.02	0.26 ± 0.01	5.0 ± 0.04
FOCOPS	13.89 ± 0.04	39.68 ± 0.01	19.79 ± 0.37	33.45 ± 0.15	18.95 ± 0.02	38.57 ± 0.01	4.1 ± 0.01	32.54 ± 0.02
CPPO-PID	3.63 ± 0.01	29.41 ± 0.02	5.21 ± 0.03	28.54 ± 0.04	0.31 ± 0.1	3.28 ± 0.2	3.81 ± 0.07	32.6 ± 0.023
IPO	3.12 ± 0.03	69.23 ± 0.02	3.01 ± 0.21	69.21 ± 0.03	0.27 ± 0.02	4.18 ± 0.01	**0.32 ± 0.02**	5.24 ± 0.05

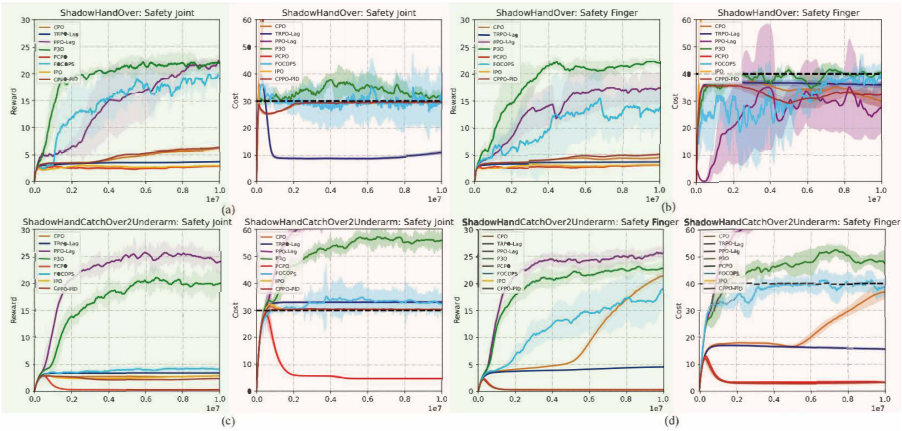

Fig. 6 Performance on robot's state constraints: (a) Safety Joint task of ShadowHandOver, (b) Safety Finger task of ShadowHandOver, (c) Safety Joint task of ShadowHandCatchOver2Underarm, (d) Safety Finger task of ShadowHandCatchOver2Underarm.

For (2), On the other hand, we evaluate the performance of PPO, PPO-Lag, FOCOPS, CPPO-PID, and P3O algorithms on six tasks, and the performance of each algorithm is shown in Fig. 7. It can be observed that PPO-Lag can

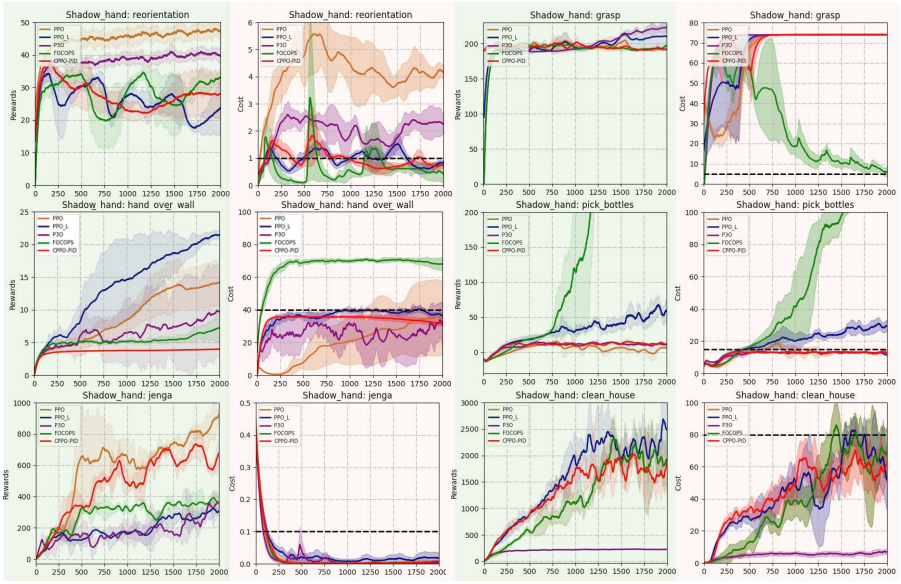

Fig. 7 Performance on environment's state constraints: Learning curves for all six tasks. The shaded region represents the standard deviation of the score over 3 trials. Curves are smoothed uniformly for visual clarity. All algorithms interact with environments in 100M steps and the number of parallel simulations is 2048.

achieve high performance within the range allowed by the cost and is the best-performing algorithm here. Comparing the performance of PPO and PPO-Lag, it can be found that PPO-Lag can perform similarly to PPO in Jenga, and Safe Finger tasks, but the cost is constrained to a lower range, which indicates that the model has learned how to safely manipulate. A remarkable result is that in the Janga and Pick Bottle task, the performance of PPO-Lag is far superior to PPO. This is because in these environments, learning safe manipulation is beneficial. For example, in Jenga, when the policy learns to not mess up other blocks, the target objects are also easier to be removed, resulting in a higher reward. It fully illustrates the advantage of Safe RL in learning better policies on manipulation tasks. However, on most of the tasks, FOCOPS and P3O are basically unable to achieve the performance of PPO-Lag, or even can not complete the task. Therefore, the performance of the current Safe RL algorithm on manipulation still has a lot of room for exploration.

5 Potential Research Topics

ReDMan provides ample opportunities to study trustworthy manipulation of dexterous hands based on Safe RL. We found that the primal-dual-based approach [60] results in great volatility in the update of Lagrange multipliers. A potential research direction is to consider combining feedback control methods in control systems, such as PID [61, 62], ADRC [63], etc., to mitigate the

instability and volatility of Lagrange multipliers in the learning process. Therefore, it would be interesting to combine control methods of complex systems with Safe RL methods to solve complex manipulation problems of dexterous hands.

Sim to real is an important research direction about transferring the simulation result to the real robot. Around this theme, our benchmark includes many of the robot arms and dexterous hands, which were accepted by many research labs. It is convenient for different researchers to choose their own arms and hands for training in the simulation. Meanwhile, the tasks in our benchmark, such as picking bottles[1], Jenga[2], etc., are meaningful in the real world but also needed to ensure safety if transferring the trained policy from simulation to the real world. So our benchmark can also be used to study how to perform sim to real more safely from the perspective of Safe RL.

Training policy with a state-based observation space is difficult for sim to real transfer because such inputs are not available in the real world. So it also makes sense to study the more readily available policy inputs in the real world, such as point clouds. Our environment supports a multimodal input such as visual and forces information, which can support research in this direction. We hope that our benchmark can serve as a tool to study the sim to the real transfer of dexterous hands.

Finally, generalization is an important direction to explore, which is a potential strength of RL. ReDMan supports self-customization, enabling switching and linking different hands and arms to evaluate the generality of different algorithms. Users can use ReDMan as a platform for modification or secondary development to design richer and more challenging target tasks, and we hope that this work will contribute to the flourishing of the RL community.

6 Conclusion and Future Work

In this work, we presented ReDMan, which is the first benchmark focused on safe dexterous manipulation. We standardize the safe policy optimization methods for solving CMDPs and introduce a unified, highly-optimized, extensible, and comprehensive algorithms re-implementation. We checked the correctness of our algorithms on the existing Safe RL benchmark and tested it on ReDMan. The results show that the Safe RL algorithm can better solve the safety problem in dexterous manipulation. For example, using Safe RL can grab the target bottle without touching other bottles, and avoid collision with obstacles when sweeping the floor. However, it is difficult for unconstrained RL algorithms to have this guarantee. These situations are very important for robots in real-world environments because RL-based methods tend to lead to unpredictable behaviors that are prone to danger and damage to robots.

Additionally, we support two features regarding visual policy input and various arms and dexterous hands. Some sort of visual input is becoming

[1] https://www.youtube.com/watch?v=hvibZrLxYyQ
[2] More details in https://en.wikipedia.org/wiki/Jenga

increasingly common in real-world RL-trained robots, so benchmarks for this setting are important. Diverse arms and hands increase the applicability of our benchmarks and allow us to study policy generalization between different robots.

We posit that ReDMan is a valuable resource for future research on safe manipulation, reinforcing the integration of reinforcement learning with robotic control, and making a significant contribution to the reinforcement learning community. Its provision of a comprehensive platform for safe dexterous manipulations and a range of challenging tasks, along with a set of unified and optimized Safe RL algorithms, offers an opportunity to achieve breakthroughs in safe robot manipulation. As such, we expect ReDMan to expedite the pace of Safe RL research and encourage further collaboration between Safe RL and robotic manipulation research.

Declarations

Some journals require declarations to be submitted in a standardised format. Please check the Instructions for Authors of the journal to which you are submitting to see if you need to complete this section. If yes, your manuscript must contain the following sections under the heading 'Declarations':

- Funding
- Conflict of interest/Competing interests (check journal-specific guidelines for which heading to use)
- Ethics approval
- Consent to participate
- Consent for publication
- Availability of data and materials
- Code availability
- Authors' contributions

If any of the sections are not relevant to your manuscript, please include the heading and write 'Not applicable' for that section.

Editorial Policies for:

Springer journals and proceedings:
https://www.springer.com/gp/editorial-policies

Nature Portfolio journals:
https://www.nature.com/nature-research/editorial-policies

Scientific Reports:
https://www.nature.com/srep/journal-policies/editorial-policies

BMC journals:
https://www.biomedcentral.com/getpublished/editorial-policies

Appendix A More details about Shadow Hand

The Shadow Dexterous Hand [52] is an example of a robotic hand designed for human-level dexterity; it has five fingers with a total of 24 degrees of freedom. The hand has been commercially available since 2005; however it still has not seen widespread adoption, which can be attributed to the daunting difficulty of controlling systems of such complexity [5].

The limits of each joint in Shadow hand are as Tab. A1. The thumb has 5 joints and 5 degrees of freedom, while all other fingers have 3 degrees of freedom and 4 joints. It should be noted that the joints at the end of each finger are not controllable.

The distal joints of the fingers are coupled like that of human fingers, making the angle of the middle joint always bigger or equal to the angle of the distal joint. This allows the middle phalange is curved, while the distal phalange is straight. There is an extra joint (LF5) at the end of the little finger to allow the little finger to rotate in the direction of the thumb. There are two joints at the wrist, which guarantees that the entire hand can rotate 360 degrees.

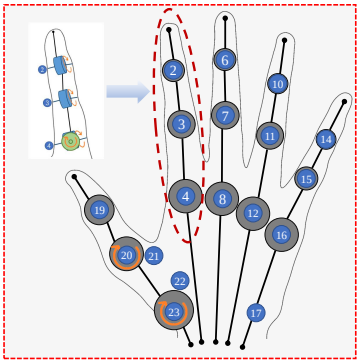

Fig. A1 Illustration of the joints on a dexterous robotic hand.

Stiffness, damping, friction, and armature are also important physical parameters in robotics. For each Shadow hand's joint, we show our DoF properties in Tab. A2. This part can be adjusted in the Isaac Gym simulator.

Table A1 Finger range of motion.

Joints	Corresponds to the number of Fig. A1	Min	Max
Finger Distal (FF1,MF1,RF1,LF1)	15, 11, 7, 3	0°	90°
Finger Middle (FF2,MF2,RF2,LF2)	16, 12, 8, 4	0°	90°
Finger Base Abduction (FF3,MF3,RF3,LF3)	17, 13, 9, 5	-15°	90°
Finger Base Lateral (FF4,MF4,RF4,LF4)	18, 14, 10, 6	-20°	20°
Little Finger Rotation(LF5)	19	0°	45°
Thumb Distal (TH1)	20	-15°	90°
Thumb Middle (TH2)	21	-30°	30°
Thumb Base Abduction (TH3)	22	-12°	12°
Thumb Base Lateral (TH4)	23	0°	70°
Thumb Base Rotation (TH5)	24	-60°	60°
Hand Wrist Abduction (WR1)	1	-40°	28°
Hand Wrist Lateral (WR2)	2	-28°	8°

Appendix B Task Specifications

B.1 Basic State Space and Action Space

The state space dimension of each environment is up to 400 dimensions in total, and the action space dimension is up to 40 dimensions. All environments are goal-based, and each epoch will randomly reset the object's starting pose and target pose to improve generalization. We only use the shadow hand and object state information as observation at present. The observation of all tasks is composed of three parts: the state information of the left and right hands, and the information of objects and target. The state information of the left and right hands were the same for each task, including hand joint and finger positions, velocity, and force information. The state information of the object

Table A2 DoF properties of Shadow Hand.

Joints	Stiffness	Damping	Friction	Armature
WR1	100	4.78	0	0
WR2	100	2.17	0	0
FF2	100	3.4e+38	0	0
FF3	100	0.9	0	0
FF4	100	0.725	0	0
MF2	100	3.4e+38	0	0
MF3	100	0.9	0	0
MF4	100	0.725	0	0
RF2	100	3.4e+38	0	0
RF3	100	0.9	0	0
RF4	100	0.725	0	0
LF2	100	3.4e+38	0	0
LF3	100	0.9	0	0
LF4	100	0.725	0	0
TH2	100	3.4e+38	0	0
TH3	100	0.99	0	0
TH4	100	0.99	0	0
TH5	100	0.81	0	0

and goal are different for each task, which we will describe in the following. Tab. B3 shows the specific information of the left-hand and right-hand state.

Table B3 Observation space of bimanual shadow hands.

Index	Description
0 - 23	right shadow hand dof position
24 - 47	right shadow hand dof velocity
48 - 71	right shadow hand dof force
72 - 136	right shadow hand fingertip pose, linear velocity, angle velocity (5 x 13)
137 - 166	right shadow hand fingertip force, torque (5 x 6)
167 - 169	right shadow hand base position
170 - 172	right shadow hand base rotation
173 - 198	right shadow hand actions
199 - 222	left shadow hand dof position
223 - 246	left shadow hand dof velocity
247 - 270	left shadow hand dof force
271 - 335	left shadow hand fingertip pose, linear velocity, angle velocity (5 x 13)
336 - 365	left shadow hand fingertip force, torque (5 x 6)
366 - 368	left shadow hand base position
369 - 371	left shadow hand base rotation
372 - 397	left shadow hand actions

B.2 Safe Finger

This environment contains two dexterous hands. At the beginning of each episode, a ball falls randomly around the right hand, and the two hands have to collaborate to place the ball in a given position. Since the target is out of

the reach of the right hand, and the right hand cannot pass the ball to the left hand directly, a possible solution is that the right hand grabs the ball, and throws it to the left hand; the left hand catches the ball and puts it to the target. Note that the base of the hand is fixed.

Observations

The 398-dimensional observational space for Hand Over task is shown in Tab. B4. It should be noted that since the base of the dual hands in this task is fixed, the observation of the dual hands is compared to the Tab. B3 of reduced 24 dimensions.

Table B4 Observation space of Safe Finger.

Index	Description
0 - 373	dual hands observation shown in Tab. B3
374 - 380	object pose
381 - 383	object linear velocity
384 - 386	object angle velocity
387 - 393	goal pose
394 - 397	goal rot - object rot

Actions

The 40-dimensional action space for one hand in Safe Finger task is shown in Tab. B5.

Table B5 Action space of Safe Finger.

Index	Description
0 - 19	right shadow hand actuated joint
20 - 39	left shadow hand actuated joint

Reward

For timestep t, let $x_{b,t}$ be the position of the ball and $x_{g,t}$ be the position of the goal. We use $d_{p,t}$ to denote the positional distance between the ball and the goal $d_{p,t} = \|x_{b,t} - x_{g,t}\|_2$. Let $d_{a,t}$ denote the angular distance between the object and the goal, and the rotational difference is $d_{r,t} = 2 \arcsin \min\{\mathsf{d\_a,t''}, 1.0\}$. The reward is defined as follows,

$$r_t = \exp\{-0.2(\alpha d_{p,t} + d_{r,t})\}, \tag{B1}$$

where α is a constant balances positional and rotational rewards.

Cost

In these tasks, we constrain the freedom of joints ②, ③ and ④ of forefinger (please refer to fig. A1 (b)). Without the constraint, joints ② and ③ have freedom of $[0°, 90°]$ and joint ④ of $[-20°, 20°]$. The safety tasks restrict joints ②, ③, and ④ within $[22.5°, 67.5°]$, $[22.5°, 67.5°]$, and $[-10°, 10°]$ respectively. Let $\texttt{ang\_2}, \texttt{ang\_3}, \texttt{ang\_4}$ be the angles of joints ②, ③, ④, and the cost is defined as:

$$\begin{aligned} c_t = &\mathbb{I}(\texttt{ang\_2} \notin [22.5°, 67.5°], && \text{(B2)} \\ &\text{or } \texttt{ang\_3} \notin [22.5°, 67.5°], && \text{(B3)} \\ &\text{or } \texttt{ang\_4} \notin [-10°, 10°]). && \text{(B4)} \end{aligned}$$

B.3 Grasp

The environment of this task is inherited from [49]. The success of the grasp task in robotic hand dexterous manipulation has significant implications for various fields, including manufacturing, healthcare, etc. The grasp task in robotic hand dexterous manipulation involves the ability of a robotic hand to grasp and manipulate objects of different shapes, sizes, and weights. The goal of this task is to develop algorithms that enable robotic hands to perform tasks that are similar to those of human hands, such as grasping, lifting, and manipulating objects in various ways. Among them, the cost we introduced is that the joints of the fingers should not exert excessive force.

B.4 Reorientation

The environment of this task is inherited from [49]. Reorientation is an essential component of robotic manipulation, as many real-world tasks require the manipulation of objects in various orientations. For instance, in a manufacturing setting, robotic hands may need to rotate a part to a specific angle to attach it to another component.

The reorientation task in robotic hand dexterous manipulation involves the ability of a robotic hand to reorient an object to a desired orientation or pose. The goal of this task is to develop algorithms that enable robotic hands to manipulate objects in various ways, such as rotating them to a specific angle, flipping them over, or aligning them with other objects. Among them, the cost we introduced is that the joints of the fingers should not exert excessive force on the dexterously manipulated objects.

B.5 Hand Over Wall

This environment is similar to Safe Finger, except that it has a wall between each hand and a hole in the middle of the wall. We need to learn policy to keep the ball from hitting the wall during the toss.

Observations

The 398-dimensional observational space for the Hand Over Wall task is shown in Tab. B6. It should be noted that since the base of the dual hands in this task is fixed, the observation of the dual hands is compared to Tab. B3 of reduced 24 dimensions.

Table B6 Observation space of Hand Over Wall.

Index	Description
0 - 373	dual hands observation shown in Tab. B3
374 - 380	object pose
381 - 383	object linear velocity
384 - 386	object angle velocity
387 - 393	goal pose
394 - 397	goal rot - object rot

Actions

The 40-dimensional action space for one hand in Hand Over Wall task is shown in Tab. B7.

Table B7 Action space of Hand Over Wall.

Index	Description
0 - 19	right shadow hand actuated joint
20 - 39	left shadow hand actuated joint

Reward

For timestep t, let $x_{b,t}$ be the position of the ball and $x_{g,t}$ be the position of the goal. We use $d_{p,t}$ to denote the positional distance between the ball and the goal $d_{p,t} = \|x_{b,t} - x_{g,t}\|_2$. Let $d_{a,t}$ denote the angular distance between the object and the goal, and the rotational difference is $d_{r,t} = 2 \arcsin \min\{\mathsf{d\_a,t''}, 1.0\}$. The reward is defined as follows,

$$r_t = \exp\{-0.2(\alpha d_{p,t} + d_{r,t})\}, \tag{B5}$$

where α is a constant balances positional and rotational rewards.

Cost

This constraint is more demanding than Wall Down, where we require the ball thrown to fit through a specified narrow hole. If the ball hits the wall, the cost is 1, otherwise it is 0. The size of the wall and the hole can be customized by the user.

B.6 Pick Bottles

This environment contains two hands, a table and five bottles. The five bottles were placed in a row on the table horizontally with very little space between them. We need to pick up two bottles with two dexterous hands, and not touch the bottle around it to cause possible damage.

Observations

The 400-dimensional observational space for Pick Bottles task is shown in Tab. B8. It should be noted that since the base of the dual hands in this task is fixed, the observation of the dual hands is compared to the Tab. B3 of reduced 24 dimensions.

Table B8 Observation space of Pick Bottles.

Index	Description
0 - 397	dual hands observation shown in Tab. B3
398 - 404	left bottle pose
405 - 407	left bottle linear velocity
408 - 410	left bottle angle velocity
411 - 417	right bottle pose
418 - 420	right bottle linear velocity
421 - 423	right bottle angle velocity

Actions

The 52-dimensional action space for one hand in Pick Bottles task is shown in Tab. B9.

Table B9 Action space of Pick Bottles.

Index	Description
0 - 19	right Shadow Hand actuated joint
20 - 22	right Shadow Hand base translation
23 - 25	right Shadow Hand base rotation
26 - 45	left Shadow Hand actuated joint
46 - 48	left Shadow Hand base translation
49 - 51	left Shadow Hand base rotation

Reward

The reward consists of three parts: the distance from the left hand to the left bottle cap, the distance from the right hand to the right bottle cap, and the height of the two bottles that need to be picked. The height of the two bottles that need to be picked is given by d_{height}. The position difference between the left hand to the left bottle cap d_{left} is given by $d_{left} = \|x_{lhand} - x_{lbcap}\|_2$.The

position difference between the right hand to the right bottle cap d_{right} is given by $d_{right} = \|x_{rhand} - x_{rbcap}\|_2$. The reward is given by this specific formula:

$$r = d_{height} * 20 - d_{left} - d_{right} \tag{B6}$$

Cost

The constraint of this environment is that we can't touch other bottles when we pick the bottle. When our hand, or the bottle we picked, touches other bottles, the cost is set to 1, otherwise it is 0.

B.7 Jenga

Jenga is a fitness game that is very suitable for Safe RL algorithm evaluation. Players take turns removing one block at a time from a tower made up of many blocks. In this environment, we need to remove the one we want from the 16 blocks without knocking over the others.

Jenga

The 411-dimensional observational space for Jenga task is shown in Tab. B10. It should be noted that since the base of the dual hands in this task is fixed, the observation of the dual hands is compared to the Tab. B3 of reduced 24 dimensions.

Table B10 Observation space of Jenga.

Index	Description
0 - 397	dual hands observation shown in Tab. B3
398 - 404	object pose
405 - 407	object linear velocity
408 - 410	object angle velocity

Actions

The 52-dimensional action space for one hand in Jenga task is shown in Tab. B11.

Table B11 Action space of Jenga.

Index	Description
0 - 19	right Shadow Hand actuated joint
20 - 22	right Shadow Hand base translation
23 - 25	right Shadow Hand base rotation
26 - 45	left Shadow Hand actuated joint
46 - 48	left Shadow Hand base translation
49 - 51	left Shadow Hand base rotation

Reward

For timestep t, let $x_{b,t}$ as the position of the left middle finger, $x_{g,t}$ as the position of the left end of the object, and $d_{p,t} = \|x_{b,t} - x_{g,t}\|_2$. Define $d_{y,t}$ as the y-axis direction of the position of the object center, the reward is defined as follows:

$$r_t = 30 * (d_{y,t} + 0.6) - d_{p,t} \tag{B7}$$

Cost

The constraint of this environment is that we can not touch other blocks in the Jenda. The cost is 1 if all blocks move more than 0.01 cm, and 0 otherwise.

B.8 Clean House

This environment is in a scene we usually clean at home. We need to control the broom with both hands to sweep the trash from the ground into the dustpan without touching other furniture (e.g. chairs).

Observations

The 431-dimensional observational space for Clean House task is shown in Tab. B12. It should be noted that since the base of the dual hands in this task is fixed, the observation of the dual hands is compared to the Tab. B3 of reduced 24 dimensions.

Table B12 Observation space of Clean House.

Index	Description
0 - 397	dual hands observation shown in Tab. B3
398 - 404	object pose
405 - 407	object linear velocity
408 - 410	object angle velocity
411 - 417	goal pose
418 - 421	goal rot - object rot
422 - 424	the bottom of broom position
425 - 427	the left handle position of the broom
428 - 430	the right handle position of the broom

Actions

The 52-dimensional action space for one hand in Clean House task is shown in Tab. B13.

Reward

The reward consists of four parts: the distance from the left hand to the left handle position of the broom, the distance from the right hand to the right handle position of the broom, the object (trash) position to the bottom of

Table B13 Action space of Clean House.

Index	Description
0 - 19	right Shadow Hand actuated joint
20 - 22	right Shadow Hand base translation
23 - 25	right Shadow Hand base rotation
26 - 45	left Shadow Hand actuated joint
46 - 48	left Shadow Hand base translation
49 - 51	left Shadow Hand base rotation

broom position, and the distance from the object to the target (dustpan) point. The distance from the object to the target point is given by d_{target}. The position difference from the left hand to the left handle position of the broom is given by d_{left}. The position difference from the right hand to the right handle position of the broom is given by d_{right}. The object position to the bottom of broom position is given by d_{bottom}. The reward is given by this specific formula:

$$r = 50 - d_{target} * 10 - 5 * d_{left} - 5 * d_{right} \tag{B8}$$

Cost

The constraint of this environment is that we can not damage other furniture when we sweep the floor. So there is a chair in the path of the trash and the dustpan. The cost is 1 when the broom touches the chair and make it move, and 0 otherwise.

Appendix C Environments for Safe Finger.

All environments are comes from Safe Finger. The difference between Safe Finger and Safe Joint is whether it is a joint constrainedor a finger constrained, which is described as follows:

Safety Joint. In these tasks, we constrain the freedom of joint ④ of forefinger (please refer to Fig. A1 (a) and (f)). Without the constraint, joint ④ has freedom of $[-20°, 20°]$. The safety tasks restrict joint ④ within $[-10°, 10°]$. Let `ang_4` be the angle of joint ④, and the cost is defined as:

$$c_t = \mathbb{I}(\texttt{ang\_4} \notin [-10°, 10°]). \tag{C9}$$

Safety Finger. In these tasks, we constrain the freedom of joints ②, ③ and ④ of forefinger (please refer to Fig. A1 (b) and (f)). Without the constraint, joints ② and ③ have freedom of $[0°, 90°]$ and joint ④ of $[-20°, 20°]$. The safety tasks restrict joints ②, ③, and ④ within $[22.5°, 67.5°]$, $[22.5°, 67.5°]$, and $[-10°, 10°]$ respectively. Let $\texttt{ang\_2}, \texttt{ang\_3}, \texttt{ang\_4}$ be the angles of joints ②, ③, ④, and the cost is defined as:

$$c_t = \mathbb{I}(\texttt{ang\_2} \notin [22.5°, 67.5°],$$

$$\text{or } \texttt{ang\_3} \notin [22.5°, 67.5°],$$
$$\text{or } \texttt{ang\_4} \notin [-10°, 10°]).$$

Hand over stands for the situation that two Shadow Hands with palms facing up, opposite each other, and an object that needs to be passed in Safe Finger, and it stands the object that needs to be thrown from the vertical hand to the palm-up hand in Safe Finger. Specific information can refer to Appendix B.2.

Appendix D Details of Benchmark Algorithms

In this section, we review the key steps of typical Safe RL algorithms implemented in this benchmark, which include CPO [57], PCPO [27], FOCOPS [29], P3O [26], PPO-Lag [16], TRPO-Lag [16], CPPO-PID [20], and IPO [58]. We implemented all of these algorithms and check the correctness but only evaluated some of them in ReDMan. Firstly, we will give a brief introduction to these algorithms below and give the hyperparameters of the algorithms we used in our evaluation. Then we have verified our re-implementations in other Safe RL benchmarks.

D.1 CPO

For a given policy $\pi_{\boldsymbol{\theta}_k}$, CPO updates new policy $\pi_{\boldsymbol{\theta}_{k+1}}$ as follows:

$$\pi_{\boldsymbol{\theta}_{k+1}} = \arg\max_{\pi_{\boldsymbol{\theta}} \in \Pi_{\boldsymbol{\theta}}} \mathbb{E}_{s\sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot), a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)} \left[A_{\pi_{\boldsymbol{\theta}_k}}(s,a)\right] \tag{D10}$$

$$\text{s.t.} \quad J^c(\pi_{\boldsymbol{\theta}_k}) + \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot), a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)} \left[A^c_{\pi_{\boldsymbol{\theta}_k}}(s,a)\right] \leq b, \tag{D11}$$

$$\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_k}) = \mathbb{E}_{s\sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot)}[\mathrm{KL}(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_k})[s]] \leq \delta. \tag{D12}$$

It is impractical to solve the problem (D10) directly due to the computational cost. [57] suggest to find some convex approximations to replace the term $A_{\pi_{\boldsymbol{\theta}_k}}(s,a)$ and $\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_k})$ Eq.(D10)-(D12). Concretely, [57] suggest to use first-order Taylor expansion of $J(\pi_{\boldsymbol{\theta}})$ to replace the objective (D10) as follows,

$$\begin{aligned}&\frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot), a\sim\pi_{\boldsymbol{\theta}_k}(\cdot|s)} \left[\frac{\pi_{\boldsymbol{\theta}}(a|s)}{\pi_{\boldsymbol{\theta}_k}(a|s)} A_{\pi_{\boldsymbol{\theta}_k}}(s,a)\right] \\ &\quad = J(\pi_{\boldsymbol{\theta}}) - J(\pi_{\boldsymbol{\theta}_k}) \approx (\boldsymbol{\theta} - \boldsymbol{\theta}_k)^\top \nabla_{\boldsymbol{\theta}} J(\pi_{\boldsymbol{\theta}}).\end{aligned}$$

Similarly, [57] use the following approximations to turn the constrained policy optimization (D10)-(D12) to be a convex problem,

$$\frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot), a\sim\pi_{\boldsymbol{\theta}_k}(\cdot|s)} \left[\frac{\pi_{\boldsymbol{\theta}}(a|s)}{\pi_{\boldsymbol{\theta}_k}(a|s)} A^c_{\pi_{\boldsymbol{\theta}_k}}(s,a)\right] \approx (\boldsymbol{\theta} - \boldsymbol{\theta}_k)^\top \nabla_{\boldsymbol{\theta}} J^c(\pi_{\boldsymbol{\theta}}), \tag{D13}$$

$$\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_k}) \approx (\boldsymbol{\theta} - \boldsymbol{\theta}_k)^\top \mathbf{H} (\boldsymbol{\theta} - \boldsymbol{\theta}_k), \tag{D14}$$

where $\mathbf{H}$ is Hessian matrix of $\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_k})$, i.e.,

$$\mathbf{H}[i,j] =: \frac{\partial^2}{\partial \boldsymbol{\theta}_i \partial \boldsymbol{\theta}_j} \mathbb{E}_{s \sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot)} \left[\mathrm{KL}(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_k})[s] \right],$$

Eq.(D14) is the second-oder approximation of (D12).

Let $\lambda_\star, \nu_\star$ is the dual solution of the following problem

$$\lambda_\star, \nu_\star = \arg \max_{\lambda \geq 0, \nu \geq 0} \left\{ \frac{-1}{2\lambda} \left(\mathbf{g}^\top \mathbf{H}^{-1} \mathbf{g} - 2\nu r + s v^2 \right) + \nu c - \frac{\lambda \delta}{2} \right\};$$

where $\mathbf{g} = \nabla_{\boldsymbol{\theta}} \mathbb{E}_{s \sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot), a \sim \pi_{\boldsymbol{\theta}}(\cdot|s)} \left[A_{\pi_{\boldsymbol{\theta}_k}}(s,a) \right]$, $\mathbf{a} = \nabla_{\boldsymbol{\theta}} \mathbb{E}_{s \sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot), a \sim \pi_{\boldsymbol{\theta}}(\cdot|s)} \left[A^c_{\pi_{\boldsymbol{\theta}_k}}(s,a) \right]$, $r = \mathbf{g}^\top \mathbf{H} \mathbf{a}, s = \mathbf{a}^\top \mathbf{H}^{-1} \mathbf{a}$, and $c = J^c(\pi_{\boldsymbol{\theta}_k}) - b$.

Finally, CPO updates parameters according to conjugate gradient as follows: if approximation to CPO is feasible, then

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \frac{1}{\lambda_\star} \mathbf{H}^{-1} (\mathbf{g} - \nu_\star \mathbf{a}),$$

else,

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \sqrt{\frac{2\delta}{\mathbf{a}^\top \mathbf{H}^{-1} \mathbf{a}}} \mathbf{H}^{-1} \mathbf{a}.$$

D.2 PCPO

Projection-Based Constrained Policy Optimization (PCPO) 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.

Reward Improvement.

$$\begin{aligned} \pi_{\boldsymbol{\theta}_{k+\frac{1}{2}}} &= \arg \max_{\pi_{\boldsymbol{\theta}} \in \Pi_{\boldsymbol{\theta}}} \mathbb{E}_{s \sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot), a \sim \pi_{\boldsymbol{\theta}}(\cdot|s)} \left[A_{\pi_{\boldsymbol{\theta}_k}}(s,a) \right], \\ \text{s.t.} \bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_k}) &= \mathbb{E}_{s \sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot)} [\mathrm{KL}(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_k})[s]] \leq \delta; \end{aligned}$$

Projection.

$$\begin{aligned} \pi_{\boldsymbol{\theta}_{k+1}} &= \arg \min_{\pi_{\boldsymbol{\theta}} \in \Pi_{\boldsymbol{\theta}}} D \left(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_{k+\frac{1}{2}}} \right), \\ \text{s.t. } J^c(\pi_{\boldsymbol{\theta}_k}) &+ \frac{1}{1-\gamma} \mathbb{E}_{s \sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot), a \sim \pi_{\boldsymbol{\theta}}(\cdot|s)} \left[A^c_{\pi_{\boldsymbol{\theta}_k}}(s,a) \right] \leq b. \end{aligned}$$

Then, [27] follows CPO [57] uses a convex approximation to the original problem, and calculates the update rule as follows,

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \sqrt{\frac{2\delta}{\mathbf{g}^\top \mathbf{H}^{-1}\mathbf{g}}}\mathbf{H}^{-1}\mathbf{g} - \max\left(0, \frac{\sqrt{\frac{2\delta}{\mathbf{g}^\top \mathbf{H}^{-1}\mathbf{g}}}\mathbf{a}^\top \mathbf{H}^{-1}\mathbf{g} + c}{\mathbf{a}^\top \mathbf{L}^{-1}\mathbf{a}}\right)\mathbf{L}^{-1}\mathbf{a},$$

where $\mathbf{L} = \mathbf{I}$ if D is ℓ_2-norm, and $\mathbf{L} = \mathbf{H}$ if D is KL-divergence.

D.3 FOCOPS

[29] propose the First Order Constrained Optimization in Policy Space (FOCOPS) which is a two-step approach. We present it as follows.

Step1: Finding the optimal update policy.

Firstly, for a given policy $\pi_{\boldsymbol{\theta}k}$, FOCOPS finds an optimal update policy $\pi^\star$ by solving the optimization problem (D10)-(D12) in the non-parameterized policy space.

$$\pi^\star = \arg\max_{\pi\in\Pi} \quad \mathbb{E}_{s\sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot), a\sim\pi(\cdot|s)}\left[A_{\pi_{\boldsymbol{\theta}_k}}(s,a)\right] \tag{D15}$$

$$\text{s.t.} \quad J^c(\pi_{\boldsymbol{\theta}_k}) + \frac{1}{1-\gamma}\mathbb{E}_{s\sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot), a\sim\pi(\cdot|s)}\left[A^c_{\pi_{\boldsymbol{\theta}_k}}(s,a)\right] \le b, \tag{D16}$$

$$\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_k}) = \mathbb{E}_{s\sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot)}[\mathrm{KL}(\pi, \pi_{\boldsymbol{\theta}_k})[s]] \le \delta. \tag{D17}$$

If $\pi_{\boldsymbol{\theta}_k}$ is feasible, then the optimal policy for (D15)-(D17) takes the following form:

$$\pi^\star(a|s) = \frac{\pi_{\boldsymbol{\theta}_k}(a|s)}{Z_{\lambda,\nu}(s)}\exp\left(\frac{1}{\lambda}\left(A_{\pi_{\boldsymbol{\theta}_k}}(s,a) - \nu A^c_{\pi_{\boldsymbol{\theta}_k}}(s,a)\right)\right), \tag{D18}$$

where $Z_{\lambda,\nu}(s)$ is the partition function which ensures (D18) is a valid probability distribution, λ and ν are solutions to the optimization problem:

$$\min_{\lambda,\nu\ge 0} \lambda\nu + \nu\tilde{b} + \lambda\mathbb{E}_{s\sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot), a\sim\pi^\star(\cdot|s)}\left[Z_{\lambda,\nu}(s)\right],$$

the term $\tilde{b} = (1-\gamma)(b - J^c(\pi_{\boldsymbol{\theta}_k}))$.

Step 2: Projection.

Then, FOCOPS projects the policy found in the previous step back into the parameterized policy space $\Pi_{\boldsymbol{\theta}}$ by solving for the closest policy $\pi_{\boldsymbol{\theta}} \in \Pi_{\boldsymbol{\theta}}$ to $\pi^\star$ in order to obtain $\pi_{\boldsymbol{\theta}_{k+1}}$:

$$\pi_{\boldsymbol{\theta}_{k+1}} = \arg\min_{\pi_{\boldsymbol{\theta}}\in\Pi_{\boldsymbol{\theta}}} \mathbb{E}_{s\sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot)}[\mathrm{KL}(\pi_{\boldsymbol{\theta}}, \pi^\star)[s]].$$

We usually apply stochastic gradient descent to obtain the solution of above $\boldsymbol{\theta}_{k+1}$.

D.4 PPO-Lag

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

$$(\pi^\star, \lambda_\star) = \arg\min_{\lambda\geq 0}\max_{\pi\in\Pi_{\boldsymbol{\theta}}} \{J(\pi) - \lambda(J^c(\pi) - b)\}. \tag{D19}$$

TRPO-Lag and PPO-Lag combine the Lagrangian approach with TRPO and PPO. Concretely, PPO using the following clip term to replace $J(\pi)$ in (D19),

$$\begin{aligned}\mathcal{L}^r_{\text{clip}}(\pi_{\boldsymbol{\theta}}) =& \mathbb{E}_{s\sim d^{\rho_0}_{\pi_k}(\cdot), a\sim\pi_k(\cdot|s)}\Big[\\ &- \min\Big\{\frac{\pi_{\boldsymbol{\theta}}(a|s)}{\pi_k(a|s)} A_{\pi_k}(s,a), &\text{(D20)}\\ &\text{clip}\left(\frac{\pi_{\boldsymbol{\theta}}(a|s)}{\pi_{\boldsymbol{\theta}_k}(a|s)}, 1-\epsilon, 1+\epsilon\right) A_{\pi_k}(s,a)\Big\}\Big], &\text{(D21)}\end{aligned}$$

where π_k is short for $\pi_{\boldsymbol{\theta}_k}$. With $A_{\pi_k}(s,a)$ replacing $A^c_{\pi_k}(s,a)$ respectively, and obtain $\mathcal{L}^c_{\text{clip}}$ as follows,

$$\begin{aligned}\mathcal{L}^c_{\text{clip}}(\pi_{\boldsymbol{\theta}}) =& \mathbb{E}_{s\sim d^{\rho_0}_{\pi_k}(\cdot), a\sim\pi_k(\cdot|s)}\Big[&\text{(D22)}\\ &- \min\Big\{\frac{\pi_{\boldsymbol{\theta}}(a|s)}{\pi_k(a|s)} A_{\pi_k}(s,a), &\text{(D23)}\\ &\text{clip}\left(\frac{\pi_{\boldsymbol{\theta}}(a|s)}{\pi_{\boldsymbol{\theta}_k}(a|s)}, 1-\epsilon, 1+\epsilon\right) A^c_{\pi_k}(s,a)\Big\}\Big]. &\text{(D24)}\end{aligned}$$

Then, PPO-Lag updates the policy as follows,

$$(\pi_{k+1}, \lambda_{k+1}) = \arg\min_{\lambda\geq 0}\max_{\pi_{\boldsymbol{\theta}}\in\Pi_{\boldsymbol{\theta}}} \left\{\mathcal{L}^r_{\text{clip}}(\pi_{\boldsymbol{\theta}}) - \lambda\left(\mathcal{L}^c_{\text{clip}}(\pi_{\boldsymbol{\theta}}) - b\right)\right\}. \tag{D25}$$

All of the above terms can be estimated according to the policy π_k. Then PPO-Lag updates the policy according to the first-order optimizer as follows,

$$\begin{aligned}\boldsymbol{\theta}_{k+1} &= \boldsymbol{\theta}_k + \eta\frac{\partial}{\partial\boldsymbol{\theta}}\left(\mathcal{L}^r_{\text{clip}}(\pi_{\boldsymbol{\theta}}) - \lambda\left(\mathcal{L}^c_{\text{clip}}(\pi_{\boldsymbol{\theta}}) - b\right)\right)|_{\boldsymbol{\theta}=\boldsymbol{\theta}_k, \lambda=\lambda_k},\\ \lambda_{k+1} &= \lambda_k + \eta\left(\mathcal{L}^c_{\text{clip}}(\pi_{\boldsymbol{\theta}}) - b\right)_+|_{\boldsymbol{\theta}=\boldsymbol{\theta}_k},\end{aligned} \tag{D26}$$

where $\eta > 0$ is step-size.

D.5 TRPO-Lag

TRPO-Lag shares a similar idea but it is adaptive to TRPO, where TRPO-Lag replaces $J(\pi_{\boldsymbol{\theta}})$ as follows,

$$J(\pi_{\boldsymbol{\theta}}) \approx J(\pi_{\boldsymbol{\theta}_k}) + (\boldsymbol{\theta} - \boldsymbol{\theta}_k)^\top\nabla_{\boldsymbol{\theta}} J(\pi_{\boldsymbol{\theta}}) =: \mathcal{L}^r(\pi_{\boldsymbol{\theta}}). \tag{D27}$$

Similarly,

$$J^c(\pi_{\boldsymbol{\theta}}) \approx J^c(\pi_{\boldsymbol{\theta}_k}) + (\boldsymbol{\theta} - \boldsymbol{\theta}_k)^\top \nabla_{\boldsymbol{\theta}} J^c(\pi_{\boldsymbol{\theta}}) =: \mathcal{L}^c(\pi_{\boldsymbol{\theta}}), \tag{D28}$$

and

$$\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_k}) \approx (\boldsymbol{\theta} - \boldsymbol{\theta}_k)^\top \mathbf{H} (\boldsymbol{\theta} - \boldsymbol{\theta}_k), \tag{D29}$$

where $\mathbf{H}$ is Hessian matrix of $\bar{D}_{\mathrm{KL}}(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_k})$, i.e.,

$$\mathbf{H}[i,j] =: \frac{\partial^2}{\partial \boldsymbol{\theta}_i \partial \boldsymbol{\theta}_j} \mathbb{E}_{s \sim d^{\rho_0}_{\pi_{\boldsymbol{\theta}_k}}(\cdot)} \left[\mathrm{KL}(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}_k})[s] \right].$$

Then, TRPO-Lag updates the policy as follows,

$$(\pi_{k+1}, \lambda_{k+1}) = \arg\min_{\lambda \geq 0} \max_{\pi_{\boldsymbol{\theta}} \in \Pi_{\boldsymbol{\theta}}} \left\{ \mathcal{L}^r(\pi_{\boldsymbol{\theta}}) - \lambda \left(\mathcal{L}^c(\pi_{\boldsymbol{\theta}}) - b \right) \right\} |_{\boldsymbol{\theta} = \boldsymbol{\theta}_k, \lambda = \lambda_k}, \tag{D30}$$

where the policy parameter $\boldsymbol{\theta}$ satisfies the following condition,

$$(\boldsymbol{\theta} - \boldsymbol{\theta}_k)^\top \mathbf{H} (\boldsymbol{\theta} - \boldsymbol{\theta}_k) \leq \delta.$$

D.6 P3O

P3O solves the cumbersome constrained policy iteration via a single minimization of an equivalent unconstrained problem as follows,

$$\pi_{k+1} = \arg\min_{\pi \in \Pi_{\boldsymbol{\theta}}} \left\{ \mathbb{E}_{s \sim d^{\rho_0}_{\pi_k}(\cdot), a \sim \pi_k(\cdot|s)} \left[\frac{\pi(a|s)}{\pi_k(a|s)} A_{\pi_k}(s,a) \right] + \kappa B(\pi, b) \right\}, \tag{D31}$$

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

$$\begin{aligned} &B(\pi, b) \\ &= \max \left\{ 0, \mathbb{E}_{s \sim d^{\rho_0}_{\pi_k}(\cdot), a \sim \pi_k(\cdot|s)} \left[\frac{\pi(a|s)}{\pi_k(a|s)} A^c_{\pi_k}(s,a) \right] + (1-\gamma) \left(J^c(\pi_k) - b \right) \right\}. \end{aligned} \tag{D32}$$

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

For the practical implementation, P3O consider the following optimization objective:

$$\mathcal{L}_{\mathrm{P3O}}(\boldsymbol{\theta}) = \mathcal{L}^r_{\mathrm{P3O}}(\boldsymbol{\theta}) + \kappa \max \left\{ 0, \mathcal{L}^c_{\mathrm{P3O}}(\boldsymbol{\theta}) \right\}, \tag{D33}$$

where

$$\mathcal{L}^r_{\mathrm{P3O}}(\boldsymbol{\theta}) = \mathbb{E}\left[-\min\left\{ \frac{\pi_{\boldsymbol{\theta}}(a|s)}{\pi_k(a|s)} A_{\pi_k}(s,a), \mathrm{clip}\left(\frac{\pi_{\boldsymbol{\theta}}(a|s)}{\pi_{\boldsymbol{\theta}_k}(a|s)}, 1-\epsilon, 1+\epsilon \right) A_{\pi_k}(s,a) \right\} \right], \tag{D34}$$

$$\mathcal{L}^c_{\mathrm{P3O}}(\boldsymbol{\theta}) = \mathbb{E}\Bigg[\max\left\{ \frac{\pi_{\boldsymbol{\theta}}(a|s)}{\pi_k(a|s)} A^c_{\pi_k}(s,a), \mathrm{clip}\left(\frac{\pi_{\boldsymbol{\theta}}(a|s)}{\pi_{\boldsymbol{\theta}_k}(a|s)}, 1-\epsilon, 1+\epsilon \right) A^c_{\pi_k}(s,a) \right\} + (1-\gamma)\left(J^c(\pi_k) - b\right) \Bigg]. \tag{D35}$$

the notation $\mathbb{E}[\cdot]$ is short for $\mathbb{E}_{s\sim d^{\rho_0}_{\pi_k}(\cdot), a\sim\pi_k(\cdot|s)}[\cdot]$. All of the terms in (D33) can be estimated according to the samples selected by π_k.

For each round, P3O chooses the parameter adaptively according to the following rule:

$$\kappa \leftarrow \min\left\{\rho\kappa, \kappa_{\max}\right\}, \tag{D36}$$

where $\rho > 1$ and $\kappa_{\max}$ is a positive scalar.

Finally, P3O updates the policy parameter as follows,

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \frac{\partial}{\partial \boldsymbol{\theta}} \mathcal{L}_{\mathrm{P3O}}(\boldsymbol{\theta}). \tag{D37}$$

D.7 IPO

IPO considers the objective with logarithmic barrier functions [64] to learn the safe policy. Concretely, IPO considers the following way to update policy,

$$\pi_{k+1} = \arg\max_{\pi\in\Pi_{\boldsymbol{\theta}}} \left\{ \mathcal{L}^r_{\mathrm{clip}}(\pi) + \phi(\pi) \right\}, \tag{D38}$$

where the clip objective is $\mathcal{L}^r_{\mathrm{clip}}(\pi)$, and $\phi(\pi)$ is the logarithm barrier function with respect to the CMDP problem,

$$\phi(\pi) = \frac{1}{m} \log\Big(b - J^c(\pi)\Big), \tag{D39}$$

where $m > 0$ is a hyper-parameter that needs to be tuned.

D.8 CPPO-PID

Appendix E Correctness Verification

E.1 Task List

Below are four safety environments that provide interesting tasks for Safe RL. Among them, three are popular safety environments, MuJoCo-Velocity [30], Safety-Gym [16], Bullet-Safety-Gym [23]. We test the 8 algorithms on 26 tasks in these safety environments to verified our re-implementations on ReDMan. The complete list of these tasks is shown in Tab. E14.

Environment	Category	Task
MuJoCo	Velocity	1. Ant-Velocity
MuJoCo	Velocity	2. Hopper-Velocity
MuJoCo	Velocity	3. Swimmer-Velocity
MuJoCo	Velocity	4. Walk2d-Velocity
Safety-Gym	Goal	5. PointGoal1
Safety-Gym	Goal	6. CarGoal1
Safety-Gym	Button	7. CarGoal1
Safety-Gym	Button	8. CarButton1
Safety-Gym	Goal	9. PointGoal2
Safety-Gym	Goal	10. PointButton2
Bullet-Safety-Gym	Circle	11. BallCircle
Bullet-Safety-Gym	Circle	12. CarCircle
Bullet-Safety-Gym	Circle	13. DroneCircle
Bullet-Safety-Gym	Circle	14. AntCircle
Bullet-Safety-Gym	Gather	15. BallGather
Bullet-Safety-Gym	Gather	16. CarGather
Bullet-Safety-Gym	Gather	17. DroneGather
Bullet-Safety-Gym	Gather	18. AntGather
Bullet-Safety-Gym	Reach	19. BallReach
Bullet-Safety-Gym	Reach	20. CarReach
Bullet-Safety-Gym	Reach	21. DroneReach
Bullet-Safety-Gym	Reach	22. AntReach
Bullet-Safety-Gym	Run	23. BallRun
Bullet-Safety-Gym	Run	24. CarRun
Bullet-Safety-Gym	Run	25. DroneRun
Bullet-Safety-Gym	Run	26. AntRun

Table E14 The complete list of all 30 tasks we test our implementations on.

E.2 Task Performance

we show the performance of our re-implementations on the tasks in Tab. E15, Tab. E16, Tab. E17, Tab. E18, Tab. E19, Tab. E20.

Table E15 Performance on MuJoCo-Velocity. We consider four MuJoCo environments where we attempt to train a robotic agent to move faster. We impose speed limits in our environments, which are calculated using 50% of the *undiscounted* speed attained by an unconstrained PPO agent after training for a million samples. In this table and the following tables, we specify the cost limit in the parenthesis after the task name.

	Swimmer-v3(205.6)		Hopper-v3(1047)		Walker2d-v3(2410)		Ant-v3(2147.5)	
	Reward	Constraint	Reward	Constraint	Reward	Constraint	Reward	Constraint
CPO	8.23 ± 2.23	25.82 ± 2.84	161.43 ± 3.99	83.65 ± 2.27	550.75 ± 368.6	82.66 ± 3.71	1030.34 ± 1.65	98.07 ± 5.29
TRPO-Lag	-2.8 ± 5.65	19.71 ± 3.14	911.61 ± 201.04	71.94 ± 12.39	1077.69 ± 1.46	79.82 ± 1.28	1032.12 ± 13.44	52.33 ± 21.15
PPO-Lag	-5.42 ± 5.41	24.66 ± 1.83	1075.08 ± 12.3	84.04 ± 11.31	1083.62 ± 1.57	99.23 ± 23.5	1008.73 ± 27.28	25.46 ± 10.73
P3O	29.95 ± 7.65	48.91 ± 11.34	1084.42 ± 9.19	87.12 ± 8.5	1084.93 ± 60.06	107.5 ± 37.56	954.65 ± 48.06	19.56 ± 9.12
PCPO	22.21 ± 5.53	49.45 ± 5.44	183.46 ± 7.71	88.1 ± 10.14	473.73 ± 184.77	115.22 ± 44.29	987.66 ± 8.32	63.2 ± 48.94
FOCOPS	65.01 ± 9.65	102.01 ± 22.35	1078.97 ± 17.74	92.07 ± 28.85	1165.08 ± 74.63	174.15 ± 69.06	1034.57 ± 12.13	50.72 ± 13.43
CPPO-PID	2.98 ± 3.02	26.1 ± 5.94	1045.29 ± 27.67	47.57 ± 26.24	1082.06 ± 13.2	84.66 ± 7.67	1049.61 ± 13.76	80.16 ± 5.69
IPO	127.26 ± 0.87	351.76 ± 4.75	1226.56 ± 116.8	220.62 ± 114.17	1485.74 ± 61.94	481.84 ± 63.2	1422.43 ± 414.35	415.49 ± 379.83

Table E16 Performance on Safety-Gym.

	Safexp-PointGoal1-v0(25.0)		Safexp-PointButton1-v0(25.0)		Safexp-CarGoal1-v0(25.0)		Safexp-CarButton1-v0(25.0)	
	Reward	Constraint	Reward	Constraint	Reward	Constraint	Reward	Constraint
CPO	27.25 ± 0.11	44.34 ± 1.63	25.53 ± 1.77	100.74 ± 5.47	37.05 ± 0.23	52.92 ± 2.51	18.83 ± 1.82	160.06 ± 14.32
TRPO-Lag	12.82 ± 1.27	24.64 ± 2.24	2.98 ± 0.99	23.84 ± 3.4	24.38 ± 2.61	24.73 ± 1.93	0.45 ± 0.83	25.16 ± 2.14
PPO-Lag	15.7 ± 4.41	24.55 ± 6.72	4.49 ± 1.19	18.69 ± 3.96	19.02 ± 6.74	24.17 ± 9.84	0.93 ± 0.84	29.78 ± 12.71
P3O	13.83 ± 3.56	27.3 ± 5.96	2.28 ± 0.64	21.8 ± 6.1	18.58 ± 1.23	25.43 ± 3.31	0.2 ± 0.56	30.13 ± 10.03
PCPO	27.24 ± 0.23	53.03 ± 1.54	31.33 ± 0.49	131.04 ± 3.61	35.33 ± 1.32	56.84 ± 2.15	24.03 ± 2.21	274.06 ± 18.77
FOCOPS	23.0 ± 1.33	34.8 ± 5.1	4.79 ± 0.89	22.62 ± 6.3	18.42 ± 4.01	24.69 ± 6.79	1.19 ± 0.63	32.64 ± 13.28
CPPO-PID	2.84 ± 3.0	50.25 ± 32.53	0.04 ± 1.27	24.3 ± 21.7	1.82 ± 3.4	20.82 ± 16.04	1.01 ± 1.21	158.76 ± 83.54
IPO	25.59 ± 0.24	33.99 ± 1.29	7.83 ± 1.09	58.33 ± 2.7	27.38 ± 0.69	41.5 ± 2.07	3.66 ± 0.08	81.11 ± 2.55

Table E17 Performance on Bullet-Safety-Gym with agent Ant.

	SafetyAntRun-v0(25.0)		SafetyAntCircle-v0(25.0)		SafetyAntReach-v0(25.0)		SafetyAntGather-v0(25.0)	
	Reward	Constraint	Reward	Constraint	Reward	Constraint	Reward	Constraint
CPO	1787.14 ± 117.91	32.06 ± 5.34	633.87 ± 65.76	81.81 ± 9.12	36.34 ± 0.01	52.55 ± 0.03	0.74 ± 0.53	0.25 ± 0.06
TRPO-Lag	2274.23 ± 6.87	23.89 ± 0.38	494.43 ± 62.5	25.81 ± 3.66	22.26 ± 0.04	24.21 ± 0.02	0.74 ± 0.53	0.25 ± 0.06
PPO-Lag	2139.49 ± 33.15	10.96 ± 8.96	197.0 ± 55.92	19.27 ± 14.6	16.35 ± 0.12	20.78 ± 0.17	1.48 ± 0.46	0.21 ± 0.08
P3O	2161.45 ± 42.51	23.25 ± 0.39	341.08 ± 77.31	18.62 ± 6.14	15.99 ± 0.03	25.46 ± 0.05	2.5 ± 0.96	0.21 ± 0.08
PCPO	1519.33 ± 226.48	58.42 ± 51.41	353.08 ± 65.9	138.64 ± 63.84	34.33 ± 0.03	55.31 ± 0.04	3.95 ± 0.19	0.45 ± 0.1
FOCOPS	2261.37 ± 6.42	17.3 ± 5.44	218.78 ± 75.52	32.77 ± 17.84	16.38 ± 0.16	24.92 ± 0.05	10.12 ± 2.62	1.11 ± 0.18
CPPO-PID	2040.92 ± 374.26	15.01 ± 10.81	749.73 ± 104.48	19.28 ± 7.33	1.11 ± 0.05	24.99 ± 0.25	1.48 ± 0.46	0.21 ± 0.08
IPO	2050.19 ± 9.61	26.36 ± 1.27	755.53 ± 103.44	26.23 ± 1.67	26.25± 0.62	38.71± 0.34	4.91 ± 0.24	0.46 ± 0.04

Table E18 Performances on Bullet-Safety-Gym with agent Ball.

	SafetyBallRun-v0(25.0)		SafetyBallCircle-v0(25.0)		SafetyBallReach-v0(25.0)		SafetyBallGather-v0(25.0)	
	Reward	Constraint	Reward	Constraint	Reward	Constraint	Reward	Constraint
CPO	305.54 ± 262.19	28.19 ± 4.09	516.05 ± 10.63	25.02 ± 0.34	36.34 ± 0.01	52.55 ± 0.03	24.22 ± 0.48	0.58 ± 0.07
TRPO-Lag	296.85 ± 392.82	65.5 ± 60.53	596.27 ± 134.49	23.97 ± 0.51	22.26 ± 0.04	24.21 ± 0.02	24.22 ± 0.48	0.58 ± 0.07
PPO-Lag	574.19 ± 3.27	16.54 ± 2.3	428.04 ± 107.86	5.07 ± 6.6	16.35 ± 0.12	20.78 ± 0.17	19.63 ± 1.36	0.4 ± 0.04
P3O	574.43 ± 5.52	24.46 ± 1.28	596.43 ± 40.66	26.5 ± 1.7	15.99 ± 0.03	25.46 ± 0.05	21.82 ± 0.16	0.43 ± 0.07
PCPO	642.32 ± 139.1	131.88 ± 17.63	284.34 ± 139.48	64.04 ± 28.4	34.33 ± 0.03	55.31 ± 0.04	20.0 ± 4.47	0.97 ± 0.4
FOCOPS	576.68 ± 3.54	17.03 ± 1.74	535.54 ± 8.3	19.73 ± 5.06	16.38 ± 0.16	24.92 ± 0.05	21.63 ± 1.04	0.96 ± 0.21
CPPO-PID	558.42 ± 7.2	0.0 ± 0.0	766.7 ± 22.35	45.16 ± 9.39	1.11 ± 0.05	24.99 ± 0.25	19.63 ± 1.36	0.4 ± 0.04
IPO	489.38 ± 3.06	25.79 ± 0.29	790.14 ± 122.23	41.77 ± 15.78	26.25± 0.62	38.71± 0.34	24.05 ± 0.6	0.61 ± 0.03

Table E19 Performance on Bullet-Safety-Gym with agent Car.

	SafetyCarRun-v0(25.0)		SafetyCarCircle-v0(25.0)		SafetyCarReach-v0(25.0)		SafetyCarGather-v0(25.0)	
	Reward	Constraint	Reward	Constraint	Reward	Constraint	Reward	Constraint
CPO	660.59 ± 81.95	27.26 ± 0.85	516.05 ± 10.63	25.02 ± 0.34	8.68 ± 1.31	43.85 ± 1.83	23.1 ± 1.24	1.04 ± 0.1
TRPO-Lag	730.46 ± 2.82	25.45 ± 0.87	596.27 ± 134.49	23.97 ± 0.51	1.88 ± 0.23	23.68 ± 2.0	23.1 ± 1.24	1.04 ± 0.1
PPO-Lag	728.65 ± 2.35	21.05 ± 8.66	428.04 ± 107.86	5.07 ± 6.6	0.64 ± 0.16	35.23 ± 4.14	9.23 ± 2.39	0.4 ± 0.05
P3O	733.02 ± 0.1	25.78 ± 0.61	596.43 ± 40.66	26.5 ± 1.7	1.2 ± 0.21	25.48 ± 4.27	9.6 ± 2.26	0.4 ± 0.04
PCPO	402.96 ± 105.44	149.17 ± 138.0	284.34 ± 139.48	64.04 ± 28.4	6.22 ± 0.38	48.56 ± 4.14	20.53 ± 3.56	1.56 ± 0.38
FOCOPS	732.45 ± 1.13	12.99 ± 2.64	535.54 ± 8.3	19.73 ± 5.06	1.36 ± 1.23	31.13 ± 6.59	13.47 ± 4.42	0.91 ± 0.17
CPPO-PID	731.67 ± 4.24	42.12 ± 51.86	766.7 ± 22.35	45.16 ± 9.39	3.02 ± 0.32	27.74 ± 1.96	9.23 ± 2.39	0.4 ± 0.05
IPO	710.75 ± 0.55	58.16 ± 1.51	790.14 ± 122.23	41.77 ± 15.78	6.54 ± 0.16	27.79 ± 1.66	10.7 ± 2.83	0.71 ± 0.08

Table E20 Performance on Bullet-Safety-Gym with agent Drone.

	SafetyDroneRun-v0(25.0)		SafetyDroneCircle-v0(25.0)		SafetyDroneReach-v0(25.0)		SafetyCarGather-v0(25.0)	
	Reward	Constraint	Reward	Constraint	Reward	Constraint	Reward	Constraint
CPO	660.59 ± 81.95	27.26 ± 0.85	615.91 ± 99.46	51.44 ± 8.02	23.13 ± 5.68	20.08 ± 0.69	8.28 ± 0.24	1.08 ± 0.32
TRPO-Lag	730.46 ± 2.82	25.45 ± 0.87	809.99 ± 84.58	23.51 ± 1.5	20.22 ± 3.92	21.57 ± 1.42	7.32 ± 0.66	0.85 ± 0.12
PPO-Lag	728.65 ± 2.35	21.05 ± 8.66	336.91 ± 194.8	16.07 ± 1.97	6.51 ± 0.7	18.66 ± 1.01	5.02 ± 0.8	1.28 ± 0.08
P3O	733.02 ± 0.1	25.78 ± 0.61	302.52 ± 46.36	21.56 ± 0.79	4.09 ± 0.21	18.91 ± 0.36	5.59 ± 0.31	1.07 ± 0.22
PCPO	402.96 ± 105.44	149.17 ± 138.0	333.92 ± 89.46	80.82 ± 43.12	26.48 ± 3.21	26.44 ± 1.46	6.24 ± 1.13	0.99 ± 0.59
FOCOPS	732.45 ± 1.13	12.99 ± 2.64	312.17 ± 213.26	25.9 ± 3.78	15.45 ± 7.66	24.6 ± 0.46	4.83 ± 0.7	2.28 ± 1.97
CPPO-PID	731.67 ± 4.24	42.12 ± 51.86	697.05 ± 91.0	25.82 ± 4.45	15.2 ± 3.83	21.86 ± 0.9	5.02 ± 0.8	1.28 ± 0.08
IPO	710.75 ± 0.55	58.16 ± 1.51	871.97 ± 147.5	24.43 ± 1.2	18.71 ± 7.84	23.07 ± 1.16	6.84 ± 0.75	0.91 ± 0.05

References

[1] Okamura, A.M., Smaby, N., Cutkosky, M.R.: An overview of dexterous manipulation. In: Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065), vol. 1, pp. 255–262 (2000). IEEE

[2] Arunachalam, S.P., Silwal, S., Evans, B., Pinto, L.: Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation. arXiv preprint arXiv:2203.13251 (2022)

[3] Allshire, A., MittaI, M., Lodaya, V., Makoviychuk, V., Makoviichuk, D., Widmaier, F., Wüthrich, M., Bauer, S., Handa, A., Garg, A.: Transferring dexterous manipulation from gpu simulation to a remote real-world trifinger. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 11802–11809 (2022). IEEE

[4] Wu, Y.-H., Wang, J., Wang, X.: Learning generalizable dexterous manipulation from human grasp affordance. In: Conference on Robot Learning, pp. 618–629 (2023). PMLR

[5] Andrychowicz, O.M., Baker, B., Chociej, M., Jozefowicz, R., McGrew, B., Pachocki, J., Petron, A., Plappert, M., Powell, G., Ray, A., *et al.*: Learning dexterous in-hand manipulation. The International Journal of Robotics Research **39**(1), 3–20 (2020)

[6] Chen, Y., Yang, Y., Wu, T., Wang, S., Feng, X., Jiang, J., McAleer, S.M., Dong, H., Lu, Z., Zhu, S.-C.: Towards human-level bimanual dexterous manipulation with reinforcement learning. arXiv preprint arXiv:2206.08686 (2022)

[7] Billard, A., Kragic, D.: Trends and challenges in robot manipulation. Science **364**(6446), 8414 (2019)

[8] Kim, U., Jung, D., Jeong, H., Park, J., Jung, H.-M., Cheong, J., Choi, H.R., Do, H., Park, C.: Integrated linkage-driven dexterous anthropomorphic robotic hand. Nature communications **12**(1), 1–13 (2021)

[9] Kumar, V., Todorov, E., Levine, S.: Optimal control with learned local models: Application to dexterous manipulation. In: 2016 IEEE International Conference on Robotics and Automation (ICRA), pp. 378–383 (2016). IEEE

[10] Bircher, W.G., Dollar, A.M., Rojas, N.: A two-fingered robot gripper with large object reorientation range. In: 2017 IEEE International Conference on Robotics and Automation (ICRA), pp. 3453–3460 (2017). IEEE

[11] Rahman, N., Carbonari, L., D'Imperio, M., Canali, C., Caldwell, D.G., Cannella, F.: A dexterous gripper for in-hand manipulation. In: 2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM), pp. 377–382 (2016). IEEE

[12] OpenAI, Akkaya, I., Andrychowicz, M., Chociej, M., Litwin, M., McGrew, B., Petron, A., Paino, A., Plappert, M., Powell, G., Ribas, R., Schneider, J., Tezak, N., Tworek, J., Welinder, P., Weng, L., Yuan, Q., Zaremba, W., Zhang, L.: Solving rubik's cube with a robot hand. CoRR **abs/1910.07113** (2019)

[13] Chen, T., Xu, J., Agrawal, P.: A system for general in-hand object re-orientation. In: Conference on Robot Learning, pp. 297–307 (2022). PMLR

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

[15] Brunke, L., Greeff, M., Hall, A.W., Yuan, Z., Zhou, S., Panerati, J., Schoellig, A.P.: Safe learning in robotics: From learning-based control to safe reinforcement learning. Annual Review of Control, Robotics, and Autonomous Systems **5**, 411–444 (2022)

[16] Ray, A., Achiam, J., Amodei, D.: Benchmarking safe exploration in deep reinforcement learning. arXiv preprint arXiv:1910.01708 **7**, 1 (2019)

[17] Dulac-Arnold, G., Mankowitz, D., Hester, T.: Challenges of real-world reinforcement learning. arXiv preprint arXiv:1904.12901 (2019)

[18] Breazeal, C., Siegel, M., Berlin, M., Gray, J., Grupen, R., Deegan, P., Weber, J., Narendran, K., McBean, J.: Mobile, dexterous, social robots for mobile manipulation and human-robot interaction. In: ACM SIGGRAPH 2008 New Tech Demos, pp. 1–1 (2008)

[19] Alshiekh, M., Bloem, R., Ehlers, R., Könighofer, B., Niekum, S., Topcu, U.: Safe reinforcement learning via shielding. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32 (2018)

[20] Stooke, A., Achiam, J., Abbeel, P.: Responsive safety in reinforcement learning by pid lagrangian methods. In: International Conference on Machine Learning, pp. 9133–9143 (2020). PMLR

[21] Gu, S., Kuba, J.G., Wen, M., Chen, R., Wang, Z., Tian, Z., Wang, J., Knoll, A., Yang, Y.: Multi-agent constrained policy optimisation. arXiv preprint arXiv:2110.02793 (2021)

[22] Yuan, Z., Hall, A.W., Zhou, S., Brunke, L., Greeff, M., Panerati,

J., Schoellig, A.P.: safe-control-gym: a unified benchmark suite for safe learning-based control and reinforcement learning. arXiv preprint arXiv:2109.06325 (2021)

[23] Gronauer, S.: Bullet-safety-gym: Aframework for constrained reinforcement learning (2022)

[24] Yang, L., Ji, J., Dai, J., Zhang, Y., Li, P., Pan, G.: Cup: A conservative update policy algorithm for safe reinforcement learning. arXiv preprint arXiv:2202.07565 (2022)

[25] Liu, Z., Guo, Z., Cen, Z., Zhang, H., Tan, J., Li, B., Zhao, D.: On the robustness of safe reinforcement learning under observational perturbations. arXiv preprint arXiv:2205.14691 (2022)

[26] Zhang, L., Shen, L., Yang, L., Chen, S., Yuan, B., Wang, X., Tao, D., et al.: Penalized proximal policy optimization for safe reinforcement learning. Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI) (2022)

[27] Yang, T.-Y., Rosca, J., Narasimhan, K., Ramadge, P.J.: Projection-based constrained policy optimization. arXiv preprint arXiv:2010.03152 (2020)

[28] Al-Rfou, R., Alain, G., Almahairi, A., Angermueller, C., Bahdanau, D., Ballas, N., Bastien, F., Bayer, J., Belikov, A., Belopolsky, A., et al.: Theano: A python framework for fast computation of mathematical expressions. arXiv e-prints, 1605 (2016)

[29] Zhang, Y., Vuong, Q., Ross, K.: First order constrained optimization in policy space. Advances in Neural Information Processing Systems **33**, 15338–15349 (2020)

[30] Todorov, E., Erez, T., Tassa, Y.: Mujoco: A physics engine for model-based control. In: 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5026–5033 (2012). IEEE

[31] Ray, A., Achiam, J., Amodei, D.: Benchmarking safe exploration in deep reinforcement learning. arXiv preprint arXiv:1910.01708 **7**, 1 (2019)

[32] Leike, J., Martic, M., Krakovna, V., Ortega, P.A., Everitt, T., Lefrancq, A., Orseau, L., Legg, S.: Ai safety gridworlds. arXiv preprint arXiv:1711.09883 (2017)

[33] Zhang, Y., Vuong, Q., Ross, K.: First order constrained optimization in policy space. Advances in Neural Information Processing Systems **33**, 15338–15349 (2020)

[34] Xu, M., Liu, Z., Huang, P., Ding, W., Cen, Z., Li, B., Zhao, D.: Trustworthy reinforcement learning against intrinsic vulnerabilities: Robustness, safety, and generalizability. arXiv preprint arXiv:2209.08025 (2022)

[35] Gu, S., Yang, L., Du, Y., Chen, G., Walter, F., Wang, J., Yang, Y., Knoll, A.: A review of safe reinforcement learning: Methods, theory and applications. arXiv preprint arXiv:2205.10330 (2022)

[36] Achiam, J., Held, D., Tamar, A., Abbeel, P.: Constrained policy optimization. In: International Conference on Machine Learning, pp. 22–31 (2017). PMLR

[37] Yang, T.-Y., Rosca, J., Narasimhan, K., Ramadge, P.J.: Projection-based constrained policy optimization. arXiv preprint arXiv:2010.03152 (2020)

[38] Chow, Y., Nachum, O., Duenez-Guzman, E., Ghavamzadeh, M.: A lyapunov-based approach to safe reinforcement learning. Advances in neural information processing systems **31** (2018)

[39] Stooke, A., Achiam, J., Abbeel, P.: Responsive safety in reinforcement learning by pid lagrangian methods. In: International Conference on Machine Learning, pp. 9133–9143 (2020). PMLR

[40] Bohg, J., Morales, A., Asfour, T., Kragic, D.: Data-driven grasp synthesis—a survey. IEEE Transactions on robotics **30**(2), 289–309 (2013)

[41] Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., Levine, S.: Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In: Conference on Robot Learning, pp. 1094–1100 (2020). PMLR

[42] James, S., Ma, Z., Arrojo, D.R., Davison, A.J.: Rlbench: The robot learning benchmark & learning environment. IEEE Robotics Autom. Lett. **5**(2), 3019–3026 (2020)

[43] Zhu, Y., Wong, J., Mandlekar, A., Martín-Martín, R.: robosuite: A modular simulation framework and benchmark for robot learning. CoRR **abs/2009.12293** (2020)

[44] Williams, G., Aldrich, A., Theodorou, E.A.: Model predictive path integral control using covariance variable importance sampling. CoRR **abs/1509.01149** (2015)

[45] Charlesworth, H.J., Montana, G.: Solving challenging dexterous manipulation tasks with trajectory optimisation and reinforcement learning. In: International Conference on Machine Learning, pp. 1496–1506 (2021). PMLR

[46] Qin, Y., Su, H., Wang, X.: From one hand to multiple hands: Imitation learning for dexterous manipulation from single-camera teleoperation. CoRR **abs/2204.12490** (2022)

[47] Qin, Y., Wu, Y., Liu, S., Jiang, H., Yang, R., Fu, Y., Wang, X.: Dexmv: Imitation learning for dexterous manipulation from human videos. CoRR **abs/2108.05877** (2021)

[48] Chen, Y., Yang, Y., Wu, T., Wang, S., Feng, X., Jiang, J., McAleer, S.M., Dong, H., Lu, Z., Zhu, S.: Towards human-level bimanual dexterous manipulation with reinforcement learning. CoRR **abs/2206.08686** (2022)

[49] Makoviychuk, V., Wawrzyniak, L., Guo, Y., Lu, M., Storey, K., Macklin, M., Hoeller, D., Rudin, N., Allshire, A., Handa, A., et al.: Isaac gym: High performance gpu-based physics simulation for robot learning. arXiv preprint arXiv:2108.10470 (2021)

[50] Calli, B., Singh, A., Bruce, J., Walsman, A., Konolige, K., Srinivasa, S., Abbeel, P., Dollar, A.M.: Yale-cmu-berkeley dataset for robotic manipulation research. The International Journal of Robotics Research **36**(3), 261–268 (2017)

[51] Xiang, F., Qin, Y., Mo, K., Xia, Y., Zhu, H., Liu, F., Liu, M., Jiang, H., Yuan, Y., Wang, H., *et al.*: Sapien: A simulated part-based interactive environment. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 11097–11107 (2020)

[52] Sharma, D., Tokas, K., Puri, A., Sharda, K.: Shadow hand. Journal of Advance Research in Applied Science **1**(1), 04–07 (2014)

[53] Qi, C.R., Su, H., Mo, K., Guibas, L.J.: Pointnet: Deep learning on point sets for 3d classification and segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 652–660 (2017)

[54] Afram, A., Janabi-Sharifi, F.: Theory and applications of hvac control systems–a review of model predictive control (mpc). Building and Environment **72**, 343–355 (2014)

[55] Carlson, T., Demiris, Y.: Increasing robotic wheelchair safety with collaborative control: Evidence from secondary task experiments. In: 2010 IEEE International Conference on Robotics and Automation, pp. 5582–5587 (2010). IEEE

[56] Bi, Z.M., Luo, C., Miao, Z., Zhang, B., Zhang, W., Wang, L.: Safety assurance mechanisms of collaborative robotic systems in manufacturing.

Robotics and Computer-Integrated Manufacturing **67**, 102022 (2021)

[57] Achiam, J., Held, D., Tamar, A., Abbeel, P.: Constrained policy optimization. In: International Conference on Machine Learning, pp. 22–31 (2017). PMLR

[58] Liu, Y., Ding, J., Liu, X.: Ipo: Interior-point policy optimization under constraints. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, pp. 4940–4947 (2020)

[59] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347 (2017)

[60] Boyd, S., Boyd, S.P., Vandenberghe, L.: Convex Optimization, pp. 561–623. Cambridge university press, ??? (2004)

[61] Ziegler, J.G., Nichols, N.B., et al.: Optimum settings for automatic controllers. trans. ASME **64**(11) (1942)

[62] Ang, K.H., Chong, G., Li, Y.: Pid control system analysis, design, and technology. IEEE transactions on control systems technology **13**(4), 559–576 (2005)

[63] Han, J.: From pid to active disturbance rejection control. IEEE transactions on Industrial Electronics **56**(3), 900–906 (2009)

[64] Nemirovski, A.: Interior point polynomial time methods in convex programming. Lecture notes **42**(16), 3215–3224 (2004)