Chinese Society of Aeronautics and Astronautics
& Beihang University

Chinese Journal of Aeronautics

cja@buaa.edu.cn
www.sciencedirect.com

FULL LENGTH ARTICLE

Demonstration-enhanced policy search for space multi-arm robot collaborative skill learning

Tian GAO [a], Chengfei YUE [a,*], Xiaozhe JU [a], Tao LIN [b]

[a] *Institute of Space Science and Applied Technology, Harbin Institute of Technology, Shenzhen 518055, China*
[b] *Research Center of Satellite Technology, Harbin Institute of Technology, Harbin 150001, China*

Received 22 February 2024; revised 7 April 2024; accepted 27 May 2024
Available online 19 August 2024

KEYWORDS

Space multi-arm collaboration;
Demonstrations;
Reinforcement Learning;
Probabilistic Movement Primitives;
Relative Entropy Policy Search;
Policy search mechanism

Abstract The increasing complexity of on-orbit tasks imposes great demands on the flexible operation of space robotic arms, prompting the development of space robots from single-arm manipulation to multi-arm collaboration. In this paper, a combined approach of Learning from Demonstration (LfD) and Reinforcement Learning (RL) is proposed for space multi-arm collaborative skill learning. The combination effectively resolves the trade-off between learning efficiency and feasible solution in LfD, as well as the time-consuming pursuit of the optimal solution in RL. With the prior knowledge of LfD, space robotic arms can achieve efficient guided learning in high-dimensional state-action space. Specifically, an LfD approach with Probabilistic Movement Primitives (ProMP) is firstly utilized to encode and reproduce the demonstration actions, generating a distribution as the initialization of policy. Then in the RL stage, a Relative Entropy Policy Search (REPS) algorithm modified in continuous state-action space is employed for further policy improvement. More importantly, the learned behaviors can maintain and reflect the characteristics of demonstrations. In addition, a series of supplementary policy search mechanisms are designed to accelerate the exploration process. The effectiveness of the proposed method has been verified both theoretically and experimentally. Moreover, comparisons with state-of-the-art methods have confirmed the outperformance of the approach.

1. Introduction

With the advancement of space exploration, space infrastructure such as space station and space telescope greatly benefits from on-orbit servicing technology, which aims to prolong the lifespan, expand the scale, and enhance the functionality of space assets. Space robots, as the primary performers of on-orbit servicing, will undertake more and more complex and critical tasks including space assembly, maintenance, and manufacturing, as shown in Fig. 1. To better adapt to these tasks,

* Corresponding author.
E-mail address: yuechengfei@hit.edu.cn (C. YUE).

https://doi.org/10.1016/j.cja.2024.08.018
1000-9361 © 2024 Chinese Society of Aeronautics and Astronautics. Production and hosting by Elsevier Ltd. All rights are reserved.
This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

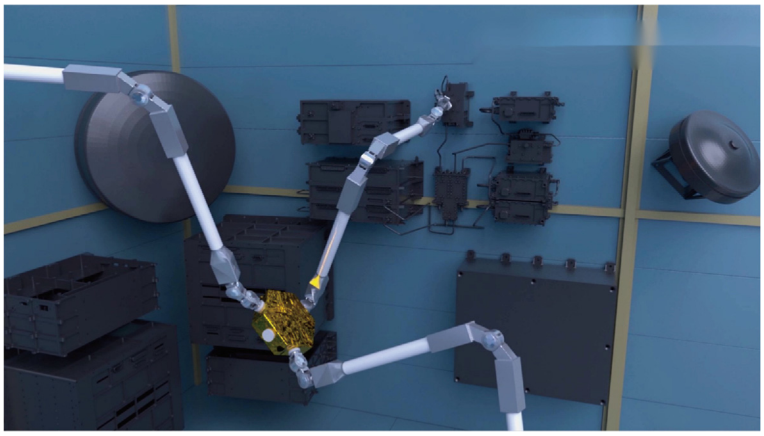


Fig. 1 Conceptual diagram of on-orbit operation of space multi-arm robot.

space robots are evolving from single-arm to multi-arm configurations for more dexterous operational capabilities. [1] For instance, the SpiderFab Bot, with its multi-arm flexibility, can achieve on-orbit assembly, fabrication, and integration.[2]

In space multi-arm robots, the collaborative operation between multiple arms has become a top priority. Traditional approaches to multi-arm collaboration typically treat it as a multi-arm planning problem. [3–5] Such researches include methods based on Artificial Potential Field, [6] sample-based methods, [7] optimization-based methods, [8] etc. These approaches aim to guide multi-arm to reach target locations or configurations while avoiding obstacles or self-collision. The robot system is usually treated as a closed-loop kinematic chain with the manipulated object and accounts for physical characteristics like redundancy, singularity, and joint angle/velocity limits. [9,10] Although these methods are effective in performing certain tasks, most of them require pre-programming according to specific requirements, lacking the adaptability to dynamic environments.[11] Therefore, the multi-arm system is increasingly developed with intelligence that learns cooperative skills and generalizes to new task scenarios. The skill learning of multi-arm robots relies on training data, which is commonly generated through interaction between the robotic arms and environment or provided by human experts. Based on this, two main categories can be distinguished: methods based on Reinforcement Learning and methods based on Learning from Demonstration.

In LfD, the human expert demonstrates or provides examples of a specific task to the robot. By imitating the expert's behaviors, the robot learns to perform similar tasks. Furthermore, the acquired skills can be transferred and generalized to new task scenarios. To better understand the generality and diversity of demonstration data, the probability-based methods, with their excellent modeling and representation capability, have been widely used. In recent work, these methods applied in multi-arm collaboration mainly include those based on Movement Primitives, such as Probabilistic Movement Primitives, [12] Dynamical Movement Primitives, [13] and those based on Gaussian Mixture Model. [14,15] However, as the number of robotic arms increases, an extensive amount of demonstration data is necessary to enhance their learning performance. Moreover, although these approaches exhibit great learning efficiency without iterative computations, they usually provide feasible solutions at the cost of sacrificing other performance metrics, which may not fulfill the complex task requirements.

In contrast, RL can yield optimal solutions through iterative learning. In RL, the robot interacts with the environment by trying different actions and observing feedback to obtain reward or punishment information, which guides the optimization of learned policy and improvement of future decision-making. [16] Some prior researches have focused on utilizing RL to learn the cooperative relationship between multi-arm, such as Proximal Policy Optimization, [17] Deep Q-Network, [18] Deep Deterministic Policy Gradient, [19,20] etc. However, the presence of multiple arms increases the dimensionality of the state-action space exponentially, which adds to the burden of exploration and results in lower learning efficiency. In order to overcome this limitation, some researchers have embarked on improvements to the input dataset. Chitnis et al. [21] employ skill parameterization to lead to faster learning, while Tamei et al. [22] represent high-dimensional data in a topological coordinate system, achieving dimensionality reduction. Other studies concentrate on refining the reward mechanisms and exploiting the structural characteristics of tasks. For instance, Ref. 23 uses intrinsic motivation to shape rewards to break through exploration bottlenecks, where the joint actions of multiple agents are rewarded rather than the individual action to encourage synergistic behavior between multiple arms. Refs. 12,24 respectively utilize the symmetry/asymmetry of dual-arm operations, where the motion of the master arm generates the symmetric/asymmetric movement of the slave arm. The common limitation of these approaches above is that they merely take advantage of ingenious designs to indirectly enhance learning performance, without directly altering the core learning mechanisms. This may result in a high dependency on specific environments and task setups, thus restricting their applicability to common scenarios. Meanwhile, iteration for obtaining optimal solutions remains the primary reason for decreased learning efficiency in RL.

The LfD + RL framework, with its outstanding performance, is becoming a new paradigm for multi-arm collaboration. With a small number of demonstrations, multi-arm can achieve more efficient guided learning of complicated collaborative skills with high-dimensional state-action space. At present, applications of the LfD + RL framework in multi-arm collaboration scenarios are less common. Depending on the predominance of the two learning strategies, it can be categorized into methods centered on RL and methods that prioritize LfD. The former takes RL as its core, with LfD providing training data input.[25–27] For example, in the work of Brys et al.,[28] expert demonstrations are utilized to bias exploration during the policy search process of RL, thereby accelerating the learning. The latter focuses on LfD and utilizes RL to enhance its performance as a further improvement.[29–31] For instance, Rajeswaran et al.[32] use Behavior Cloning for policy initialization and Natural Policy Gradient for policy optimization, realizing dexterous manipulation of a 24 Degrees of Freedom humanoid robotic hand. Nevertheless, all these methods face such a shared challenge that as the policy is updated to adapt to the environment, it tends to gradually lose the skill knowledge obtained from the initial LfD. In other words, the learned behaviors no longer preserve the characteristics of the demonstration. As a result, the role of LfD is greatly weakened, which is not what we expect.

In this paper, aiming to address the issue of exploring the high-dimensional state-action space of multi-arm cooperation, a demonstration-enhanced reinforcement learning approach is proposed to achieve rapid and effective learning. Meanwhile, it also focuses on reflecting and maintaining the characteristics captured from the initial demonstration. The proposed approach neutralizes the aforementioned two combination strategies. Specifically, ProMP is first employed to model and characterize the demonstration data probabilistically. The resulting probability distribution is used to initialize the policy. Secondly, in the RL stage, the REPS algorithm is modified to realize its search in continuous state-action space to align with ProMP. By utilizing the relative entropy constraint on the policy, the information loss between neighboring learning steps is restricted to preserve the skill knowledge acquired from the initial demonstration learning. In addition, three supplementary policy search mechanisms are proposed, namely revisiting, expanded search, and experience buffer mechanism, to further improve exploring efficiency.

This paper is organized as follows: Section 2 introduces the preliminaries of ProMP. Building upon this foundation, Section 3 derives the REPS formulation in continuous state-action space, along with an introduction to three policy search mechanisms. In Section 4, the effectiveness and outperformance of the proposed approach are demonstrated as well as the comparisons with the state-of-the-art methods. Finally, the conclusive remarks are made in Section 5.

2. Preliminaries of ProMP

2.1. Probabilistic representation with ProMP

ProMP essentially serves as a probabilistic model to represent the diversity of the demonstration trajectories. Let us consider a scenario with n Degrees of Freedom (DoF). The trajectory of n DoFs can be represented by a sequence of state transition over time, denoted as $\boldsymbol{\xi} = \{\boldsymbol{Y}_t\}_{t=1}^{G}$, where $\boldsymbol{Y}_t = \left[(\boldsymbol{y}_{1,t})^{\mathrm{T}}, (\boldsymbol{y}_{2,t})^{\mathrm{T}}, \cdots, (\boldsymbol{y}_{n,t})^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbf{R}^{2n}$ is the joint state vector of all DoFs at time step t, $\boldsymbol{y}_{i,t} = [x_{i,t}, \dot{x}_{i,t}]^{\mathrm{T}} \in \mathbf{R}^2$ represents the position and velocity vector of the ith DoF at time step t, and G is the number of time steps. Using Linear Regression, the trajectory can be projected into a space of basis function as

$$\begin{cases} \boldsymbol{Y}_t = \boldsymbol{\Psi}_t \boldsymbol{\Omega} + \boldsymbol{\varepsilon}_Y = \begin{bmatrix} \boldsymbol{\Phi}_t & \mathbf{0} & \cdots & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Phi}_t & \mathbf{0} & \mathbf{0} \\ \vdots & \mathbf{0} & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \boldsymbol{\Phi}_t \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega}_1 \\ \boldsymbol{\omega}_2 \\ \vdots \\ \boldsymbol{\omega}_n \end{bmatrix} + \boldsymbol{\varepsilon}_Y \\ \boldsymbol{\Phi}_t = \begin{bmatrix} \boldsymbol{\phi}(t)^{\mathrm{T}} \\ \dot{\boldsymbol{\phi}}(t)^{\mathrm{T}} \end{bmatrix} \end{cases} \tag{1}$$

where $\boldsymbol{\Psi}_t \in \mathbf{R}^{2n \times Nn}$ denotes the basis function matrix, which is a unit block diagonal matrix of $\boldsymbol{\Phi}_t$. $\boldsymbol{\Phi}_t \in \mathbf{R}^{2 \times N}$ consists of the basis function vector $\boldsymbol{\phi}(t)$ and its derivation. $\boldsymbol{\phi}(t) = [\phi_1(t), \phi_2(t), \cdots, \phi_N(t)]^{\mathrm{T}} \in \mathbf{R}^N$ is the vector composed of basis functions, and N is the number of basis functions. Here, the basis function is chosen as the Gaussian kernel function to guarantee a smooth trajectory. $\boldsymbol{\Omega} \in \mathbf{R}^{Nn}$ represents the joint weight vector of all DoFs and $\boldsymbol{\omega}_i \in \mathbf{R}^N$ is the weight vector of the ith DoF. The potential error between approximation and actual data is introduced as $\boldsymbol{\varepsilon}_Y$, which follows a multivariate Gaussian distribution with the mean of zero, that is $\boldsymbol{\varepsilon}_Y \sim N(\mathbf{0}, \boldsymbol{\Sigma}_Y) \in \mathbf{R}^{Nn}$.

The variability caused by human experts results in different trajectories in each demonstration. In order to adjust the motion speed or scale time of the individual trajectory, ensuring temporal consistency across all demonstration trajectories, a phase variable z_t is introduced. As needed, it can be represented by an arbitrary monotonically increasing function with respect to time step t. Thus, the basis function can be written as

$$\phi_i(z_t) = \exp\left(-\frac{(z_t - c_i)^2}{2h}\right), \ i = 1, 2, \cdots, N \tag{2}$$

with h the kernel width of the Gaussian basis function and c_i denotes the center of the ith basis function.

Assuming that M demonstrations have been conducted, obtaining a set of demonstration trajectories, denoted as $\{\boldsymbol{\xi}^i\}_{i=1}^{M} = \left\{\left(\{\boldsymbol{Y}_t\}_{t=1}^{G}\right)^i\right\}_{i=1}^{M}$. To better capture the generality and diversity of the demonstration data, ProMP can be modeled as a multivariate Gaussian distribution. The probability of state $\boldsymbol{Y}_t$ at time step t given the weight vector $\boldsymbol{\Omega}$ condition can be expressed as

$$p(\boldsymbol{Y}_t|\boldsymbol{\Omega}) = N(\boldsymbol{Y}_t|\boldsymbol{\Psi}_t\boldsymbol{\Omega}, \boldsymbol{\Sigma}_Y) \tag{3}$$

2.2. Learning and reproduction with ProMP

ProMP characterizes the demonstration data as a form of weighted sum of nonlinear basis functions, which helps to better capture the characteristics of the demonstration motion and improve the flexibility and generalization of the model. Therefore, the weight will be emphasized. Using the Least Squares method, the weight of the ith DoF in Eq. (1) can be calculated as

$$\boldsymbol{\omega}_i = \left(\boldsymbol{\psi}^{\mathrm{T}}\boldsymbol{\psi} + \gamma\boldsymbol{I}\right)^{-1}\boldsymbol{\psi}^{\mathrm{T}}\boldsymbol{q}_i, \ i = 1, 2, \cdots, n \tag{4}$$

where $\boldsymbol{\psi} = [\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \cdots, \boldsymbol{\phi}_G]^{\mathrm{T}} \in \mathbf{R}^{G \times N}$ refers to the concatenation of basis function vectors of all time steps. $\boldsymbol{q}_i = [x_{i,1}, x_{i,2}, \cdots x_{i,G}]^{\mathrm{T}} \in \mathbf{R}^G$ represents the concatenation of states of the ith DoF of all time steps. γ is the ridge factor, which is usually set to a small value. In this research, γ is set to 1×10^{-12}.

Therefore, the weights of all demonstration trajectories can be obtained as $\boldsymbol{W} = \{\boldsymbol{\Omega}^i\}_{i=1}^{M}$. Similarly, these weight vectors also follow a multivariate Gaussian distribution, which can be expressed as

$$\pi(\boldsymbol{\Omega}) = N(\boldsymbol{\Omega}|\boldsymbol{\mu}_{\boldsymbol{\Omega}}, \boldsymbol{\Sigma}_{\boldsymbol{\Omega}}) \tag{5}$$

where $\boldsymbol{\mu}_{\boldsymbol{\Omega}} \in \mathbf{R}^{Nn}$ represents the mean vector of the weight distribution, and $\boldsymbol{\Sigma}_{\boldsymbol{\Omega}} \in \mathbf{R}^{Nn \times Nn}$ is the covariance matrix. The distribution parameters $\boldsymbol{\theta}_{\boldsymbol{\Omega}} = \{\boldsymbol{\mu}_{\boldsymbol{\Omega}}, \boldsymbol{\Sigma}_{\boldsymbol{\Omega}}\}$ can be learned from the data $\boldsymbol{W}$ using the principle of Maximum Likelihood Estimation (MLE) as

$$\begin{cases} \boldsymbol{\mu}_{\boldsymbol{\Omega}} = \frac{1}{M}\sum_{i=1}^{M}\boldsymbol{\Omega}^i \\ \boldsymbol{\Sigma}_{\boldsymbol{\Omega}} = \frac{1}{M}\sum_{i=1}^{M}\left(\boldsymbol{\Omega}^i - \boldsymbol{\mu}_{\boldsymbol{\Omega}}\right)\left(\boldsymbol{\Omega}^i - \boldsymbol{\mu}_{\boldsymbol{\Omega}}\right)^{\mathrm{T}} \end{cases} \tag{6}$$

According to the Bayesian principle, it can be easily derived the probability distribution of the state at time step t by combining Eq. (5) with Eq. (3) as follows

$$\begin{aligned} p(\boldsymbol{Y}_t;\boldsymbol{\theta}_{\boldsymbol{\Omega}}) &= \int p(\boldsymbol{Y}_t|\boldsymbol{\Omega})p(\boldsymbol{\Omega};\boldsymbol{\theta}_{\boldsymbol{\Omega}})\mathrm{d}\boldsymbol{\Omega} \\ &= \int N(\boldsymbol{Y}_t|\boldsymbol{\Psi}_t\boldsymbol{\Omega}, \boldsymbol{\Sigma}_Y)N(\boldsymbol{\Omega}|\boldsymbol{\mu}_{\boldsymbol{\Omega}}, \boldsymbol{\Sigma}_{\boldsymbol{\Omega}})\mathrm{d}\boldsymbol{\Omega} \\ &= N\left(\boldsymbol{Y}_t\middle|\boldsymbol{\Psi}_t\boldsymbol{\mu}_{\boldsymbol{\Omega}}, \boldsymbol{\Psi}_t\boldsymbol{\Sigma}_{\boldsymbol{\Omega}}\boldsymbol{\Psi}_t^{\mathrm{T}} + \boldsymbol{\Sigma}_Y\right) \end{aligned} \tag{7}$$

Furthermore, employing the principle of probability superposition, the probability distribution of the trajectory can be calculated as

$$p(\boldsymbol{\xi};\boldsymbol{\theta}_{\boldsymbol{\Omega}}) = \prod_{t=1}^{G} p(\boldsymbol{Y}_t;\boldsymbol{\theta}_{\boldsymbol{\Omega}}) \tag{8}$$

3. Demonstration-enhanced relative entropy policy search

3.1. Preliminaries of REPS

Let us begin with a basic Markov Decision Process (MDP). The MDP is a mathematical framework for describing sequential decision-making problems. This process is defined as a combination of state space S, action space A, and reward function R. Specifically, assuming an agent interacts with the environment and makes decisions based on the current state $s \in S$. It selects an action $a \in A$, and the environment provides feedback in the form of a reward $r \in R$. The action taken by the agent leads to a transition to the next state $s' \in S$. This process can be expressed as follows

$$\sum_{s \in S, a \in A} p^{\pi}(s)p^{\pi}(a|s)p(s'|s,a) = p^{\pi}(s') \tag{9}$$

where $p^{\pi}(s)$, $p^{\pi}(a|s)$ and $p(s'|s,a)$ represent the probability of being in state s under policy π, the probability of selecting action a in state s, and the probability of transitioning to the state s' after taking action a in state s, respectively.

The goal of MDP is to find a policy that allows the agent to take the optimal action in each state, maximizing the long-term cumulative reward as

$$J(\pi) = \sum_{s \in S, a \in A} p^{\pi}(s)p^{\pi}(a|s)r(s,a) \tag{10}$$

where $r(s,a)$ denotes the reward obtained by taking action a in state s. Here, the policy π can be represented as the probability of being in state s and taking action a. Hence, it is easy to derive the policy $\pi(s,a) = p^{\pi}(s)p^{\pi}(a|s)$.

For relative entropy policy search, on the one hand, it is necessary to search for an optimal policy that maximizes the reward, on the other hand, it is expected to restrict the information loss caused by policy updating during the learning process. Therefore, the relative entropy (i.e., Kullback-Leibler divergence) is utilized to measure the difference between the current policy and the previous policy, and constrain it within a boundary, which is given as

$$D_{KL}(\pi\| q) = \sum_{s \in S, a \in A} \pi(s,a)\log\frac{\pi(s,a)}{q(s,a)} \leqslant \varepsilon \tag{11}$$

where $D_{KL}(\pi\| q)$ represents the K-L divergence between the current policy $\pi(s,a)$ and the previous policy $q(s,a)$, and ε denotes the maximum relative entropy bound.

Based on the above description, REPS can be transformed into the following optimization problem.

$$\begin{aligned} &\max_{\pi} J(\pi) = \sum_{s \in S, a \in A} p^{\pi}(s)p^{\pi}(a|s)r(s,a) \\ &\text{s.t. } D_{KL}(\pi\| q) = \sum_{s \in S, a \in A} \pi(s,a)\log\tfrac{\pi(s,a)}{q(s,a)} \\ &\sum_{s \in S, a \in A} p^{\pi}(s)p^{\pi}(a|s)p(s'|s,a) = p^{\pi}(s') \\ &\sum_{s \in S, a \in A} p^{\pi}(s)p^{\pi}(a|s) = 1 \end{aligned} \tag{12}$$

where the last constraint is set to satisfy the normalization condition.

3.2. REPS derivation in continuous space

In the last section, the representation of REPS in discrete state-action space is introduced. In this section, its integration with ProMP will be further derived, as well as its formulation in continuous state-action space to adapt to ProMP.

In the discrete state-action space, the policy in MDP is defined as the probability of selecting action a given the state s, with the policy represented by a series of discrete probability values. While in the continuous state-action space, the MDP's action selection becomes sampling from the continuous space. In this case, the policy becomes a probability distribution over the sampled actions. Considering the weight probability distribution in ProMP (see Eq. (5)), we propose to treat the weight distribution as the policy for REPS, i.e., the weight distribution of ProMP is used to initialize the policy of REPS. Then, the action sampling is naturally transformed into the behavior of sampling weight samples in the weight distribution space. Therefore, the objective function in Eq. (10) can be rewritten as

$$J(\boldsymbol{\Omega}) = \sum_{i=1}^{\infty} \pi(\boldsymbol{\Omega}_i)r^{\pi}(\boldsymbol{\Omega}_i) \tag{13}$$

with $\boldsymbol{\Omega}_i$ the weight sample point. $\pi(\boldsymbol{\Omega}_i)$ denotes the probability of policy π sampling to $\boldsymbol{\Omega}_i$ and under this policy, sampling $\boldsymbol{\Omega}_i$ yields a reward $r^{\pi}(\boldsymbol{\Omega}_i)$.

Furthermore, the above formulation can be rewritten in an integral form as

$$J(\boldsymbol{\Omega}) = \int \pi(\boldsymbol{\Omega}) r^{\pi}(\boldsymbol{\Omega}) \mathrm{d}\boldsymbol{\Omega} \tag{14}$$

Similarly, the optimization problem in continuous space can be summarized as follows

$$\begin{aligned} &\max_{\pi} J(\boldsymbol{\Omega}) = \int \pi(\boldsymbol{\Omega}) r^{\pi}(\boldsymbol{\Omega}) \mathrm{d}\boldsymbol{\Omega} \\ &\text{s.t. } D_{KL}(\pi \| q) = \int \pi(\boldsymbol{\Omega}) \log \tfrac{\pi(\boldsymbol{\Omega})}{q(\boldsymbol{\Omega})} \mathrm{d}\boldsymbol{\Omega} \leqslant \varepsilon \\ &\int \pi(\boldsymbol{\Omega}) \mathrm{d}\boldsymbol{\Omega} = 1 \end{aligned} \tag{15}$$

where $q(\boldsymbol{\Omega})$ represents the policy distribution of the last step.

Subsequently, the Lagrange method is employed to solve the optimization problem in Eq. (15). Let us construct the Lagrange function as follows

$$\begin{aligned} L &= \left(\int \pi(\boldsymbol{\Omega}) r^{\pi}(\boldsymbol{\Omega}) \mathrm{d}\boldsymbol{\Omega}\right) + \eta\left(\varepsilon - \int \pi(\boldsymbol{\Omega}) \log \tfrac{\pi(\boldsymbol{\Omega})}{q(\boldsymbol{\Omega})} \mathrm{d}\boldsymbol{\Omega}\right) + \lambda\left(1 - \int \pi(\boldsymbol{\Omega}) d\boldsymbol{\Omega}\right) \\ &= \int \pi(\boldsymbol{\Omega}) \left(r^{\pi}(\boldsymbol{\Omega}) - \eta \log \tfrac{\pi(\boldsymbol{\Omega})}{q(\boldsymbol{\Omega})} - \lambda\right) \mathrm{d}\boldsymbol{\Omega} + \eta\varepsilon + \lambda \end{aligned} \tag{16}$$

where η and λ are the Lagrangian multipliers, which are positive.

By differentiating L with respect to $\pi(\boldsymbol{\Omega})$, yielding

$$\frac{\partial L}{\partial \pi(\boldsymbol{\Omega})} = r^{\pi}(\boldsymbol{\Omega}) - \eta \log \frac{\pi(\boldsymbol{\Omega})}{q(\boldsymbol{\Omega})} - \eta - \lambda \tag{17}$$

And setting $\frac{\partial L}{\partial \pi(\boldsymbol{\Omega})} = 0$, we obtain

$$\pi(\boldsymbol{\Omega}) = q(\boldsymbol{\Omega}) \exp\left(\frac{r^{\pi}(\boldsymbol{\Omega})}{\eta}\right) \exp\left(-1 - \frac{\lambda}{\eta}\right) \tag{18}$$

Furthermore, due to the constraint $\int \pi(\boldsymbol{\Omega}) \mathrm{d}\boldsymbol{\Omega} = 1$, considering the above equation, it can be derived as

$$\int q(\boldsymbol{\Omega}) \exp\left(\frac{r^{\pi}(\boldsymbol{\Omega})}{\eta}\right) \exp\left(-1 - \frac{\lambda}{\eta}\right) \mathrm{d}\boldsymbol{\Omega} = 1 \tag{19}$$

The above equation can be further rewritten as

$$\exp\left(-1 - \frac{\lambda}{\eta}\right) = \left(\int q(\boldsymbol{\Omega}) \exp\left(\frac{r^{\pi}(\boldsymbol{\Omega})}{\eta}\right) \mathrm{d}\boldsymbol{\Omega}\right)^{-1} \tag{20}$$

Substituting Eq. (20) into Eq. (18), the solution of Eq. (15) can be computed as follows

$$\pi(\boldsymbol{\Omega}) = \frac{q(\boldsymbol{\Omega}) \exp\left(\frac{r^{\pi}(\boldsymbol{\Omega})}{\eta}\right)}{\int q(\boldsymbol{\Omega}) \exp\left(\frac{r^{\pi}(\boldsymbol{\Omega})}{\eta}\right) \mathrm{d}\boldsymbol{\Omega}} \tag{21}$$

As for the solution of the Lagrangian multiplier η of the K-L constraint, substituting Eqs. (18) and (21) into Eq. (16), the dual function can be obtained as

$$\begin{aligned} g(\eta) &= \eta \int \frac{q(\boldsymbol{\Omega}) \exp\left(\frac{r^{\pi}(\boldsymbol{\Omega})}{\eta}\right)}{\int q(\boldsymbol{\Omega}) \exp\left(\frac{r^{\pi}(\boldsymbol{\Omega})}{\eta}\right) \mathrm{d}\boldsymbol{\Omega}} \mathrm{d}\boldsymbol{\Omega} + \eta\varepsilon + \lambda \\ &= \eta + \eta\varepsilon + \lambda \\ &= \eta \log\left(\exp\left(1 + \tfrac{\lambda}{\eta}\right) \exp(\varepsilon)\right) \\ &= \eta \log\left(\int q(\boldsymbol{\Omega}) \exp\left(\tfrac{r^{\pi}(\boldsymbol{\Omega})}{\eta} + \varepsilon\right) \mathrm{d}\boldsymbol{\Omega}\right) \end{aligned} \tag{22}$$

The optimal Lagrangian multiplier η^* is determined by minimizing the dual function, which can be expressed as

$$\eta^* = \arg\min_{\eta} g(\eta) \tag{23}$$

3.3. Demonstration-enhanced relative entropy policy search algorithm

The analytical solution for the policy distribution $\pi(\boldsymbol{\Omega})$ has been given as shown in Eq. (21), however, calculating such an integral solution proves to be challenging. Therefore, an approximating calculation is adopted, which replaces the integral over a sum of samples. It is observed that the solution takes the following form

$$\pi(\boldsymbol{\Omega}) \propto q(\boldsymbol{\Omega}) \exp\left(\frac{r^{\pi}(\boldsymbol{\Omega})}{\eta}\right) \tag{24}$$

where the symbol $\propto$ indicates the proportional relationship.

Therefore, the Weighted Least Squares (WLS) method is considered to update the policy. Specifically, each rollout under the current policy is assigned a weight factor, then the Least Squares are employed to estimate the distribution parameters of the new policy. It is important to note that the "weight factor" in WLS is different from the "weight" in ProMP. Suppose that K rollouts are carried out under the current policy, i.e., K weight samples are sampled from the current weight distribution space, denoted as $\{\boldsymbol{\Omega}_k\}_{k=1}^{K}$, yielding K rewards denoted as $\{r^{\pi}(\boldsymbol{\Omega}_k)\}_{k=1}^{K}$, then the weight factor can be calculated as

$$m_k = q(\boldsymbol{\Omega}_k) \exp\left(\frac{r^{\pi}(\boldsymbol{\Omega}_k)}{\eta^*}\right), \ k = 1, 2, \cdots, K \tag{25}$$

Rewriting the formulation in Eq. (4) into a weighted form, the mean of the new policy distribution can be calculated as

$$\boldsymbol{\mu}_{\boldsymbol{\Omega}}^{\pi} = \left(\boldsymbol{\varphi}^{\mathrm{T}} \boldsymbol{M} \boldsymbol{\varphi} + \gamma \boldsymbol{I}\right)^{-1} \boldsymbol{\varphi}^{\mathrm{T}} \boldsymbol{M} \boldsymbol{Q} \tag{26}$$

where $\boldsymbol{\varphi} = \left[\boldsymbol{\psi}_1^{\mathrm{T}}, \boldsymbol{\psi}_2^{\mathrm{T}}, \cdots, \boldsymbol{\psi}_K^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbf{R}^{GnK \times Nn}$ represents the concatenation of basis function matrices of all time steps for all samples. And $\boldsymbol{Q} = \left[\boldsymbol{q}_1^{\mathrm{T}}, \boldsymbol{q}_2^{\mathrm{T}}, \cdots, \boldsymbol{q}_K^{\mathrm{T}}\right]^{\mathrm{T}} \in \mathbf{R}^{GnK}$ represents the concatenation of states of all time steps for all samples. $\boldsymbol{M} \in \mathbf{R}^{GnK \times GnK}$ is the block diagonal weighting matrix containing the weight $\{m_k\}_{k=1}^{K}$, which can be expressed as

$$\begin{cases} \boldsymbol{M} = \begin{bmatrix} \boldsymbol{D}_1 & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{D}_2 & \cdots & \boldsymbol{0} \\ \vdots & \vdots & & \vdots \\ \boldsymbol{0} & \boldsymbol{0} & \cdots & \boldsymbol{D}_K \end{bmatrix} \\ \boldsymbol{D}_k = \operatorname{diag}\left(\underbrace{m_k, \cdots, m_k}_{Gn}\right), \ k = 1, 2, \cdots, K \end{cases} \tag{27}$$

Meanwhile, employing the Weighted Maximum Likelihood Estimation (WMLE), the covariance of the new policy distribution can be calculated as

$$\boldsymbol{\Sigma}_{\boldsymbol{\Omega}}^{\pi} = \frac{1}{f} \sum_{k=1}^{K} m_k \left(\boldsymbol{\Omega}_k - \boldsymbol{\mu}_{\boldsymbol{\Omega}}^{\pi}\right)\left(\boldsymbol{\Omega}_k - \boldsymbol{\mu}_{\boldsymbol{\Omega}}^{\pi}\right)^{\mathrm{T}} \tag{28}$$

where f denotes a regularization factor, which can be computed as

$$f = \left(\sum_{k=1}^{K} m_k\right)^2 - \sum_{k=1}^{K} (m_k)^2 \tag{29}$$

So far, the demonstration-enhanced REPS algorithm can be summarized in Algorithm 1. After obtaining the optimal policy distribution, by utilizing the reproduction capability of ProMP, substitute it into Eq. (7) to obtain the state distribution at time step t, denoted as $p\left(\boldsymbol{Y}_t^*;\boldsymbol{\theta}_{\boldsymbol{\Omega}}^*\right) = N\left(\boldsymbol{Y}_t\middle|\boldsymbol{\Psi}_t\boldsymbol{\mu}_{\boldsymbol{\Omega}}^{\pi^*},\boldsymbol{\Psi}_t\boldsymbol{\Sigma}_{\boldsymbol{\Omega}}^{\pi^*}\boldsymbol{\Psi}_t^{\mathrm{T}}+\boldsymbol{\Sigma}_Y\right)$. Subsequently, by applying Eq. (8), the trajectory distribution can be derived as $p\left(\boldsymbol{\xi}^*;\boldsymbol{\theta}_{\boldsymbol{\Omega}}^*\right)$. Generally, the mean trajectory is sampled as the final reproduced trajectory, that is $\left\{\boldsymbol{\Psi}_t\boldsymbol{\mu}_{\boldsymbol{\Omega}}^{\pi^*}\right\}_{t=1}^{G}$.

Algorithm 1. Demonstration-enhanced REPS algorithm

```
Input: Demonstration trajectories $\{\xi^i\}_{i=1}^M$, the maximum relative entropy bound $\varepsilon$
Initialize: Perform Gaussian Linear Regression on the demonstration data, yielding the weight set $\{\boldsymbol{\Omega}^i\}_{i=1}^M$ using Eqs. (1) and (4)
Perform MLE on the weight data to obtain the initial policy distribution $\pi_l(\boldsymbol{\Omega})$ using Eq. (6)
while the mean reward not converged do
    Rollouts: Sample $K$ data points $\{\boldsymbol{\Omega}_k\}_{k=1}^K$ from the current policy distribution $\pi_l(\boldsymbol{\Omega})$ and compute the rewards $\{r^{\pi_l}(\boldsymbol{\Omega}_k)\}_{k=1}^K$
    Policy evaluation: Optimize the dual function $g(\eta)$ to obtain the optimal Lagrangian multiplier $\eta^*$ using Eq. (23)
    Compute the weight factor $m_k$ for each rollout using Eq. (25)
    Normalize the weight factor such that $\sum_{k=1}^K m_k = 1$
    Policy improvement: Perform WLS to compute the mean $\boldsymbol{\mu}_{\boldsymbol{\Omega}}^{\pi_{l+1}}$ of new policy distribution $\pi_{l+1}(\boldsymbol{\Omega})$ using Eq. (26)
    Perform WMLE to compute the covariance $\boldsymbol{\Sigma}_{\boldsymbol{\Omega}}^{\pi_{l+1}}$ of new policy distribution $\pi_{l+1}(\boldsymbol{\Omega})$ using Eq. (28)
    Testing: Compute the reward corresponding to the mean of the current policy distribution $\{r^{\pi_{l+1}}(\boldsymbol{\mu}_{\boldsymbol{\Omega}}^{\pi_{l+1}})\}_{k=1}^K$
    $l \leftarrow l+1$
end
Output: The optimal policy distribution $\pi^*(\boldsymbol{\Omega}) = N(\boldsymbol{\Omega}|\boldsymbol{\mu}_{\boldsymbol{\Omega}}^{\pi^*},\boldsymbol{\Sigma}_{\boldsymbol{\Omega}}^{\pi^*})$
```

3.4. Three policy search mechanisms

In order to achieve faster policy search, three supplementary search mechanisms are designed, namely the revisiting, expanded search, and experience buffer mechanisms. The details are given as follows.

3.4.1. Revisiting mechanism

During the sampling process, the approach of summing samples to replace integral results in a limited number of samples, which may not cover the entire sample space. Consequently, the samples obtained from the current policy distribution may not necessarily reflect the optimal gradient direction for policy update, as evidenced by the rewards of the current policy being smaller than those from the previous policy. For this case, the revisiting mechanism is designed. It is inspired by the pruning technology in Neural Network, which is used for pruning unnecessary channels, reducing model parameters and connections. [33] Specifically, if the mean reward of the updated policy in the current step is smaller than that of the previous step, the current policy will be discarded, and the search process will instead continue from the previous policy. This mechanism can be expressed as

$$\pi_{l+1}(\boldsymbol{\Omega}) = \begin{cases} \pi_{l+1}(\boldsymbol{\Omega}), & r(\boldsymbol{\mu}_{\boldsymbol{\Omega}}^{\pi_{l+1}}) \geqslant r(\boldsymbol{\mu}_{\boldsymbol{\Omega}}^{\pi_l}) \\ \pi_l(\boldsymbol{\Omega}), & \text{else} \end{cases} \tag{30}$$

3.4.2. Expanded search mechanism

In the later stage of policy search, as the reward converges, the covariance of the policy distribution used for sampling will also gradually converge to a smaller value, which indicates a narrowing of the search scope. Consequently, the improvement of reward slows down, increasing the risk of getting trapped in the local optima. To address this issue, an influence factor is introduced into the process of updating the covariance to appropriately expand the search scope. Therefore, the Eq. (28) can be rewritten as

$$\boldsymbol{\Sigma}_{\boldsymbol{\Omega}}^{\pi} = \alpha\left(\frac{1}{f}\sum_{k=1}^{K} m_k\left(\boldsymbol{\Omega}_k - \boldsymbol{\mu}_{\boldsymbol{\Omega}}^{\pi}\right)\left(\boldsymbol{\Omega}_k - \boldsymbol{\mu}_{\boldsymbol{\Omega}}^{\pi}\right)^{\mathrm{T}}\right) \tag{31}$$

where the influence factor α is usually set to 1.0–1.5 in the later stage of the search when the variation of reward is less than a certain threshold.

3.4.3. Experience buffer mechanism

Although LfD provides a good policy initialization for RL, the demonstration data is not fully utilized in the policy search process, which can provide effective search direction. Additionally, due to the limited sample data that cannot cover the entire sample space, the previous policy's sample data may be superior to the current one. Based on these two considerations, we propose to incorporate the experience of both demonstration data and previously excellent samples into the search process to guide policy improvement. Therefore, the Eq. (31) can be further rewritten as

$$\begin{aligned} \boldsymbol{\Sigma}_{\boldsymbol{\Omega}}^{\pi} &= \alpha(1-\beta)\tfrac{1}{f}\sum_{k=1}^{K} m_k\left(\boldsymbol{\Omega}_k - \boldsymbol{\mu}_{\boldsymbol{\Omega}}^{\pi}\right)\left(\boldsymbol{\Omega}_k - \boldsymbol{\mu}_{\boldsymbol{\Omega}}^{\pi}\right)^{\mathrm{T}} \\ &+\alpha\beta\sum_{\boldsymbol{\Omega}_j\in\boldsymbol{\rho}_{\mathrm{D}}\cup\boldsymbol{\rho}_{\mathrm{P}}}\left(\boldsymbol{\Omega}_j - \boldsymbol{\mu}_{\boldsymbol{\Omega}}^{j}\right)\left(\boldsymbol{\Omega}_j - \boldsymbol{\mu}_{\boldsymbol{\Omega}}^{j}\right)^{\mathrm{T}} \end{aligned} \tag{32}$$

where $\boldsymbol{\rho}_{\mathrm{D}}$ and $\boldsymbol{\rho}_{\mathrm{P}}$ represent the demonstration dataset and the collection of previously excellent samples, respectively, and $\boldsymbol{\mu}_{\boldsymbol{\Omega}}^{j}$ denotes the mean vector of their union. $\beta \in [0,1]$ is the mixing factor.

Here, it is necessary to distinguish between the proposed experience buffer and the experience replay in Deep Reinforcement Learning. [34] The motivation behind experience replay is primarily to break the correlation among training data and make it satisfy the independent and identically distributed (i.i.d.) assumption, so that to train the network more fully and quickly. Therefore, the data in the experience pool is required to be general and not necessarily the optimal sample data. Conversely, the proposed experience buffer mechanism only retains the previously optimal sample data to guide the best search direction.

In conclusion, the complete learning process is shown in Fig. 2. In this process, ProMP is responsible for providing the initial policy, while REPS searches for policy estimation and improvement. The updated policy obtained from the current episode is used for the next iteration. In addition, three mechanisms are introduced to expedite the search process.

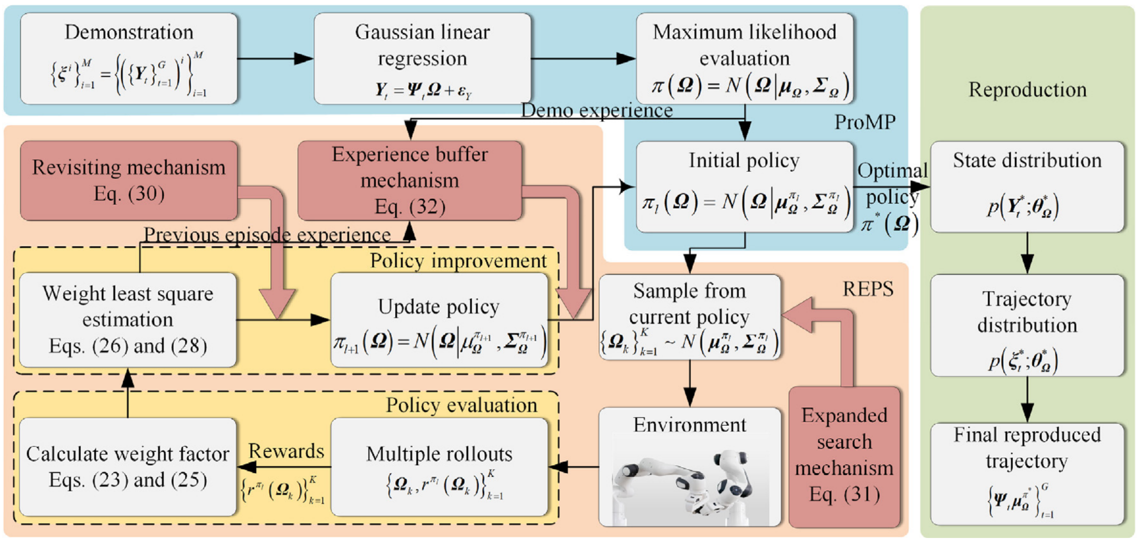

Demonstration
$\{\xi^i\}_{i=1}^M = \left\{\left(\{Y_t\}_{t=1}^G\right)^i\right\}_{i=1}^M$
Gaussian linear regression
$Y_t = \Psi_t \Omega + \varepsilon_Y$
Maximum likelihood evaluation
$\pi(\Omega) = N(\Omega|\mu_\Omega, \Sigma_\Omega)$
Demo experience
ProMP
Reproduction
Revisiting mechanism Eq. (30)
Experience buffer mechanism Eq. (32)
Initial policy
$\pi_l(\Omega) = N(\Omega|\mu_\Omega^{\pi_l}, \Sigma_\Omega^{\pi_l})$
Optimal policy $\pi^*(\Omega)$
State distribution
$p(Y_t^*;\theta_\Omega^*)$
Previous episode experience
Policy improvement
Weight least square estimation Eqs. (26) and (28)
Update policy
$\pi_{l+1}(\Omega) = N(\Omega|\mu_\Omega^{\pi_{l+1}}, \Sigma_\Omega^{\pi_{l+1}})$
Sample from current policy
$\{\Omega_k\}_{k=1}^K \sim N(\mu_\Omega^{\pi_l}, \Sigma_\Omega^{\pi_l})$
REPS
Trajectory distribution
$p(\xi_t^*;\theta_\Omega^*)$
Policy evaluation
Calculate weight factor Eqs. (23) and (25)
Rewards
$\{r^{\pi_l}(\Omega_k)\}_{k=1}^K$
Multiple rollouts
$\{\Omega_k, r^{\pi_l}(\Omega_k)\}_{k=1}^K$
Environment
Expanded search mechanism Eq. (31)
Final reproduced trajectory
$\{\Psi_t \mu_\Omega^{\pi^*}\}_{t=1}^G$

Fig. 2 Complete learning process.

Finally, the satisfactory trajectory can be reproduced through the optimal policy.

4. Experimental results and analysis

In this section, simulation will be conducted to verify the effectiveness of the proposed approach considering the task scenario of dual-arm collaboration as an example. The necessity of the demonstration-enhanced process is firstly validated through scenarios with and without LfD. Then, a comparison will be made between our approach with and without the search mechanisms to demonstrate their superiority. Finally, a comparison with the state-of-the-art methods will be performed to illustrate the outperformance of our method.

4.1. Dual-arm collaboration experiment

During the LfD stage, the pose DoFs at the end-effector of dual-arm are selected, including both positions and orientations. We perform $M = 30$ demonstrations on two Franka Panda robotic arms and record their end-effector trajectories. These trajectories are then approximated using $N = 30$ Gaussian basis functions. In the policy search stage, $K = 100$ samples are drawn from the policy distribution in each policy update episode and the maximum K-L divergence bond ε is empirically set to 0.5.[35] The details of dual-arm collaboration, including dual-arm grasping and dual-arm assembly, correspond to on-orbit operation scenarios of space multi-arm robot, as shown below.

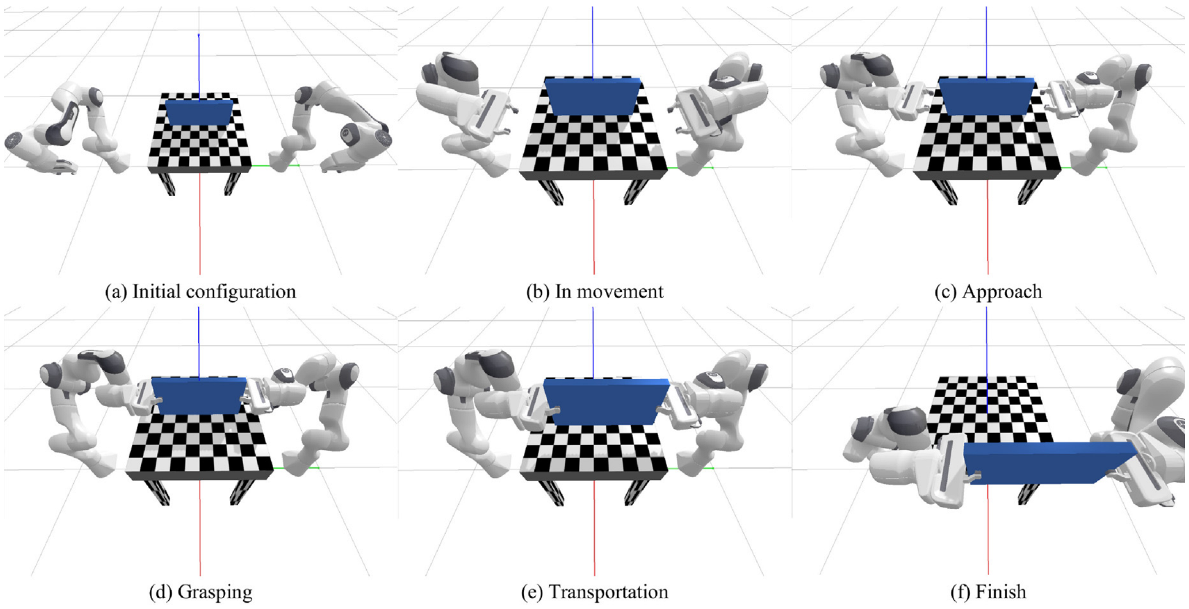


Fig. 3 Snapshot of dual-arm grasping.

4.1.1. Dual-arm grasping

In this task scenario, dual-arm is required to simultaneously grasp a rectangular board for transportation cooperation while maintaining a steady orientation of the board. Accordingly, the reward function can be defined as follows

$$r = \exp\left(-w\left(\| \boldsymbol{\theta}_{\mathrm{d}} - \boldsymbol{\theta}_{\mathrm{arm1}} \|^2 + \| \boldsymbol{\theta}_{\mathrm{d}} - \boldsymbol{\theta}_{\mathrm{arm2}} \|^2\right)\right) \tag{33}$$

where $\boldsymbol{\theta}_{\mathrm{arm1}}$ and $\boldsymbol{\theta}_{\mathrm{arm2}}$ represent the attitude angle vectors of two robotic arms' end-effectors respectively, and $\boldsymbol{\theta}_{\mathrm{d}}$ denotes the desired attitude of the board. The parameter w is adjustable to control the rate of reward transition.

The learned dual-arm grasping behavior is shown in Fig. 3, where the two arms first approach the two sides of the board for grasping, and then move the board synchronously while maintaining the desired attitude. The analysis results of the dual-arm grasping and dual-arm assembly are presented in Fig. 4, respectively. Fig. 4(a) illustrates the transition of the orientation quaternion of the manipulated board, showing a consistently maintained deviation at an order of magnitude of 10^{-3}.

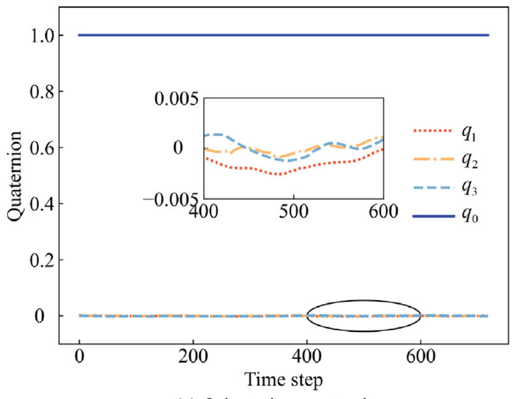
4.1.2. Dual-arm assembly

In this case, dual-arm is required to grip a part of the assembly respectively and then insert the protrusion into the groove to complete the insertion operation. This cooperation demands accurate gripping and insertion of the two robotic arms. Consequently, the reward function can be designed as follows

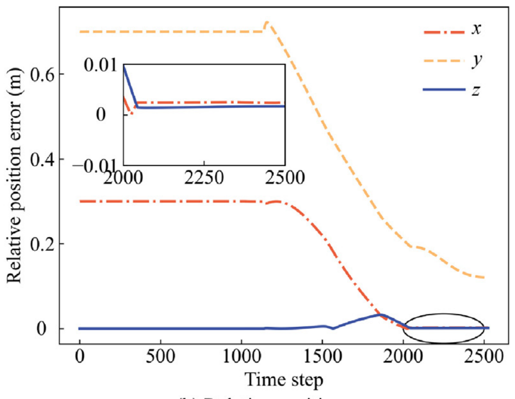
$$r = -w\left(\left(\| \boldsymbol{x}_{\mathrm{d}_1}^{t_1} - \boldsymbol{x}_{\mathrm{arm1}}^{t_1} \|^2 + \| \boldsymbol{x}_{\mathrm{d}_2}^{t_1} - \boldsymbol{x}_{\mathrm{arm2}}^{t_1} \|^2\right) + \left(\| \boldsymbol{x}_{\mathrm{d}}^{t_2} - \boldsymbol{x}_{\mathrm{arm1}}^{t_2} \|^2 + \| \boldsymbol{x}_{\mathrm{d}}^{t_2} - \boldsymbol{x}_{\mathrm{arm2}}^{t_2} \|^2\right)\right) \tag{34}$$

where $\boldsymbol{x}_{\mathrm{d}_1}^{t_1}$ and $\boldsymbol{x}_{\mathrm{d}_2}^{t_1}$ respectively represent the desired positions for gripping the groove part and the protrusion part at the expected time t_1, while $\boldsymbol{x}_{\mathrm{arm1}}^{t_1}$ and $\boldsymbol{x}_{\mathrm{arm2}}^{t_1}$ denote the actual positions of the two arms' end-effectors at time t_1. $\boldsymbol{x}_{\mathrm{d}}^{t_2}$ represents the desired docking position of the assembly at the expected time t_2, while $\boldsymbol{x}_{\mathrm{arm1}}^{t_2}$ and $\boldsymbol{x}_{\mathrm{arm2}}^{t_2}$ denote the actual positions of the end-effectors of two arms at time t_2. Similarly, w is defined as the rate parameter.

The obtained collaborative behavior of dual-arm is given in Fig. 5. From the snapshot, it is observed that the two arms



(a) Orientation quaternion



(b) Relative position error

Fig. 4 Transition of pose of manipulated objects.

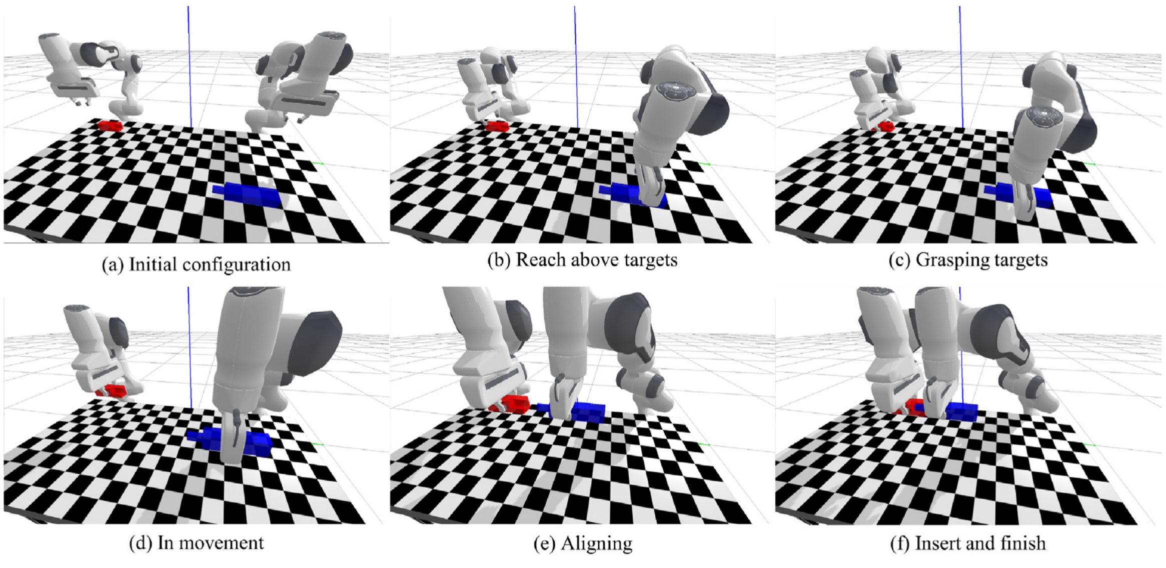


Fig. 5 Snapshot of dual-arm assembly.

grasp assembly parts respectively and complete the insertion operation in the air, with the protrusion part being accurately inserted into the groove part. Meanwhile, the transition of relative position error between two assembly parts is provided in Fig. 4(b), where the docking accuracy reaches the order of magnitude of 10^{-3} m.

Aiming to explain the importance of the demonstration-enhanced process, the reward values are extracted in the scenarios with and without LfD respectively. The comparative results are shown in Fig. 6. With the help of prior knowledge acquired from LfD, the reward can converge from a higher level from the beginning, which significantly reduces the number of learning steps in the RL phase. This undoubtedly effectively improves learning efficiency.

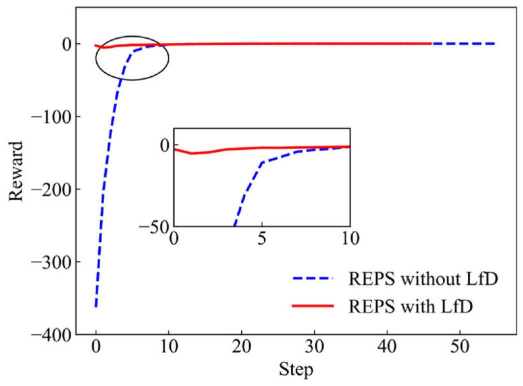


Fig. 6 Transition of reward with and without LfD.

4.2. Validation of policy search mechanisms

The experiment contrasting scenarios with and without the implementation of policy search mechanisms is conducted and the comparative results are given in Fig. 7.

Fig. 7(a) presents the transition of reward values corresponding to the mean of distribution in each policy update episode. It is evident that all proposed supplementary search mechanisms exhibit a greater improvement in search efficiency compared to the standard policy search method, as evidenced by a faster increase in reward values. Moreover, the combined utilization of all three mechanisms yields the best search performance, followed by the revisiting, expanded search, and experience buffer in descending order. Fig. 7(b) depicts the transition of the samples' mean rewards and their 95% confidence intervals in each policy update episode. The results align with those shown in Fig. 7(a). In addition, a repeated experiment with 50 tests is carried out to eliminate the randomness. The termination condition is defined as achieving a given threshold for the reward value, and the maximum number of learning steps is set to 300. Fig. 7(c) illustrates the statistical results. It is easily obtained that the proposed search mecha-

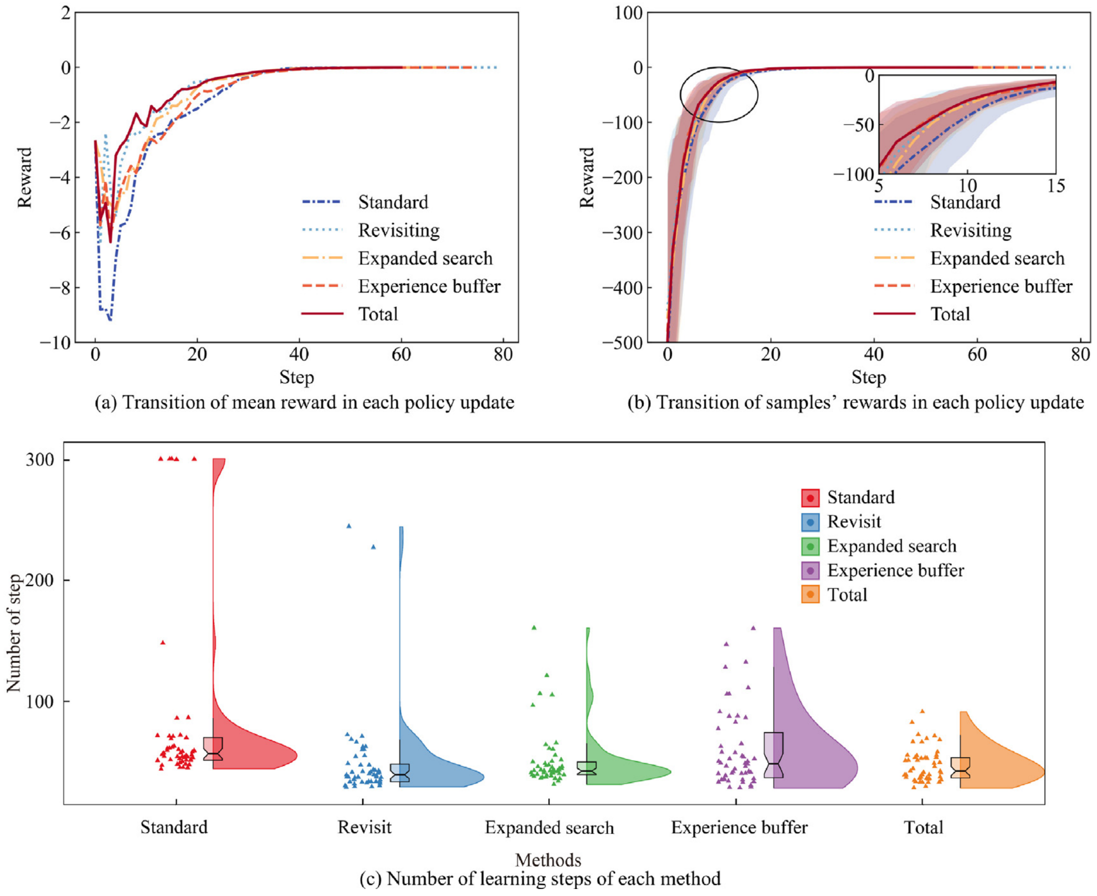


Fig. 7 Comparative results with and without supplementary search mechanisms.

nisms require fewer learning steps to achieve the same level of learning performance compared to the standard search method. Moreover, the distribution of these numbers is more concentrated, showing no significant outliers. This suggests improved stability in policy search with the assistance of these mechanisms. Additionally, it is worth noting that the standard approach exhibits failed cases with a success rate of 90% (as depicted by the presence of 5 data points of 300 steps). In contrast, all mechanisms demonstrate a 100% success rate without any instances of failure.

4.3. Comparison with other methods

In order to highlight the superiority of the proposed method, a comparison is made with two other state-of-the-art approaches, namely Demo Augmented Policy Gradient (DAPG) [32] and Probabilistic Motion Primitives based Trajectory Planning (PROMPT). [36] PROMPT employs a combination of ProMP and Stochastic Gradient method. [37] It generates the noisy trajectories using the learned distribution from ProMP, and then calculates stochastic gradients to determine the direction of optimization for the next learning step. DAPG utilizes Behavior Cloning to initialize the nodes of the Neural Network and further enhances performance through training the neural network using Natural Policy Gradients. Both methods are based on the LfD + RL framework.

Taking the dual-arm assembly as the task scenario, learning is conducted using the two aforementioned approaches as well as our method. Similarly, the termination condition is set as reaching a specific threshold for the reward value, with a maximum number of 300 learning steps. The comparative results are illustrated in Fig. 8, which provide the transition of rewards and average position errors in each policy update episode, respectively. It is observed that though PROMPT shows a comparable speed to the proposed method, it tends to get trapped in the local optima (as depicted by the reward of PROMPT fails to achieve the given condition, even until reaching the maximum number of 300 learning steps without any further increase). The DAPG is able to complete learning, but its learning efficiency is inferior compared to the proposed method. The learning of DAPG is completed around the 260th step. In contrast, the proposed method achieves the fastest improvement of reward value and reaches the termination condition in the shortest time (around 50 steps). This can be attributed to the proposed policy search mechanisms.

In addition, the learned motion trajectories of the dual-arm end-effectors with the proposed method and DAPG are respectively plotted in Fig. 9, where the blue curves represent the initial trajectories obtained from LfD, while the red curves represent the learned trajectory obtained through the LfD + RL framework. The red star markers indicate the desired positions of the dual-arm end-effectors. By comparing

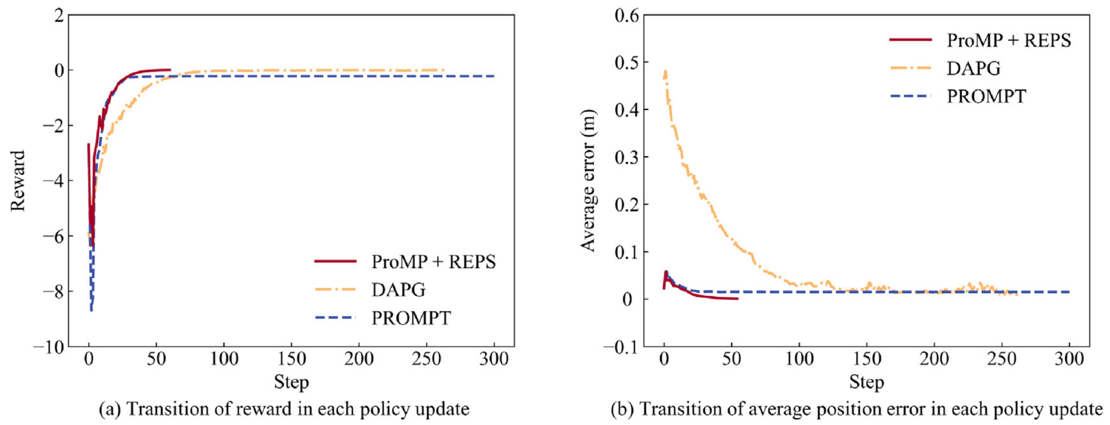


Fig. 8 Comparative results with other methods.

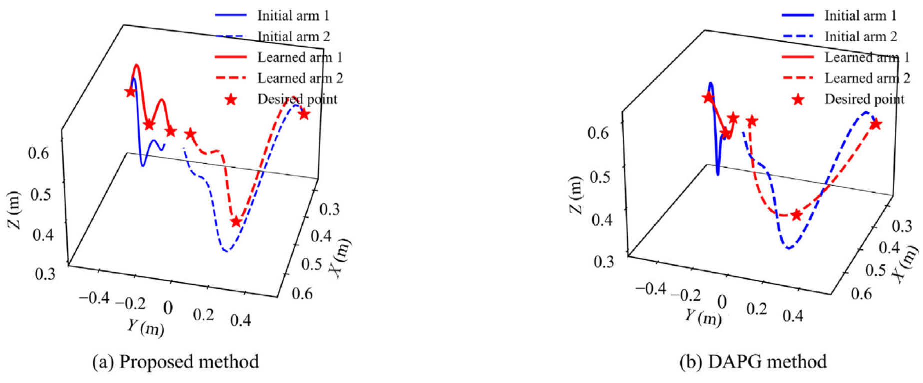


Fig. 9 Learned motion trajectories of dual-arm end-effectors with different methods.

the trajectories obtained from the two methods, it is found that the end-effector trajectory learned by the proposed method inherits the shape of the initial LfD trajectory. This alignment indicates that the proposed method can maintain the prior knowledge captured from the initial teaching. This is attributed to the constraint of the maximum relative entropy bound. In contrast, in other LfD + RL methods such as DAPG, the final learned trajectory is able to reach the desired positions, but the trajectory shape differs significantly from the initial teaching trajectory, which does not fully reflect the characteristics of LfD. This further demonstrates the outperformance of the proposed method.

5. Conclusions

In this paper, an LfD + RL approach is proposed for the space multi-arm collaborative skill learning to achieve fast and effective learning in high-dimensional state-action space. During the LfD phase, the utilization of ProMP allows for the better probabilistic representation of the demonstration data and provides input for the initial policy of RL, while the use of REPS in the RL phase helps reduce the information loss caused by policy update, effectively preserving the prior knowledge acquired from LfD. In addition, a series of supplementary policy search mechanisms are designed to accelerate the learning speed. Both theoretical and experimental analyses demonstrate the effectiveness of the proposed method in multi-arm collaborative skill learning.

CRediT authorship contribution statement

Tian GAO: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Conceptualization. **Chengfei YUE:** Writing – review & editing, Funding acquisition, Conceptualization. **Xiaozhe JU:** Writing – review & editing, Methodology, Investigation, Conceptualization. **Tao LIN:** Writing – review & editing, Software, Methodology.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This study was co-supported by the National Natural Science Foundation of China (No. 12372045), the Guangdong Basic and Applied Basic Research Foundation, China (No. 2023B1515120018), and the Shenzhen Science and Technology Program, China (No. JCYJ20220818102207015).

References

1. Xue ZH, Liu JG, Wu CC, et al. Review of in-space assembly technologies. *Chin J Aeronaut* 2021;**34**(11):21–47.
2. Hoyt RP. SpiderFab: an architecture for self-fabricating space systems. *Proceedings of the AIAA SPACE 2013 conference and exposition*. Reston: AIAA; 2013.
3. Yue CF, Lin T, Zhang X, et al. Hierarchical path planning for multi-arm spacecraft with general translational and rotational locomotion mode. *Sci China Technol Sci* 2023;**66**(4):1180–91.
4. Xu WF, Yan L, Hu ZH, et al. Area-oriented coordinated trajectory planning of dual-arm space robot for capturing a tumbling target. *Chin J Aeronaut* 2019;**32**(9):2151–63.
5. Yue CF, Zhang X, Wang HX, et al. Three-dimensional path planning of on-orbit manipulation robot based on neighborhood continuation search. *J Astronaut* 2022;**43**:203–13 [Chinese].
6. Zhang H, Zhu YF, Liu XF, et al. Analysis of obstacle avoidance strategy for dual-arm robot based on speed field with improved artificial potential field algorithm. *Electronics* 2021;**10**(15):1850.
7. Chen XY, You XJ, Jiang JC, et al. Trajectory planning of dual-robot cooperative assembly. *Machines* 2022;**10**(8):689.
8. Larsen L, Kim J. Path planning of cooperating industrial robots using evolutionary algorithms. *Robot Comput Integr Manuf* 2021;**67**:102053.
9. Xian Z, Lertkultanon P, Pham QC. Closed-chain manipulation of large objects by multi-arm robotic systems. *IEEE Robot Autom Lett* 2017;**2**(4):1832–9.
10. Zhan Q, He YH, Chen M. Collision avoidance of cooperative dual redundant manipulators. *Chin J Aeronaut* 2003;**16**(2):117–22.
11. Lin T, Yue CF, Liu ZR, et al. Modular multi-level replanning tamp framework for dynamic environment. *IEEE Robot Autom Lett* 2024;**9**(5):4234–41.
12. Amadio F, Colomé A, Torras C. Exploiting symmetries in reinforcement learning of bimanual robotic tasks. *IEEE Robot Autom Lett* 2019;**4**(2):1838–45.
13. Lu ZY, Wang N, Shi DH. DMPs-based skill learning for redundant dual-arm robotic synchronized cooperative manipulation. *Complex Intell Syst* 2022;**8**(4):2873–82.
14. Hu HP, Zhao ZL, Yang XS, et al. A learning from demonstration method for robotic assembly with a dual-sub-6-DoF parallel robot. *2021 WRC symposium on advanced robotics and automation (WRC SARA)*. Piscataway: IEEE Press; 2021. p. 73–8.
15. Silvério J, Rozo L, Calinon S, et al. Learning bimanual end-effector poses from demonstrations using task-parameterized dynamical systems. *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. Piscataway: IEEE Press; 2015. p. 464–70.
16. Zhang Q, Xie ZW, Cao BS, et al. A policy iteration method for improving robot assembly trajectory efficiency. *Chin J Aeronaut* 2023;**36**(3):436–48.
17. Tang WX, Cheng C, Ai HP, et al. Dual-arm robot trajectory planning based on deep reinforcement learning under complex environment. *Micromachines* 2022;**13**(4):564.
18. Liu D, Cao JF, Lei XK. Slabstone installation skill acquisition for dual-arm robot based on reinforcement learning. *2019 IEEE international conference on robotics and biomimetics (ROBIO)*. Piscataway: IEEE Press; 2019. p. 1298–305.
19. Tang MY, Yue XF, Zuo Z, et al. Coordinated motion planning of dual-arm space robot with deep reinforcement learning. *2019 IEEE international conference on unmanned systems (ICUS)*. Piscataway: IEEE Press; 2019. p. 469–73.
20. Liu LY, Liu QY, Song Y, et al. A collaborative control method of dual-arm robots based on deep reinforcement learning. *Appl Sci* 2021;**11**(4):1816.
21. Chitnis R, Tulsiani S, Gupta S, et al. Efficient bimanual manipulation using learned task schemas. *2020 IEEE international conference on robotics and automation (ICRA)*. Piscataway: IEEE Press; 2020. p. 1149–55.
22. Tamei T, Matsubara T, Rai A, et al. Reinforcement learning of clothing assistance with a dual-arm robot. *2011 11th IEEE-RAS international conference on humanoid robots*. Piscataway: IEEE Press; 2011. p. 733–38.

23. Chitnis R, Tulsiani S, Gupta S, et al. Intrinsic motivation for encouraging synergistic behavior. arXiv preprint:2002.05189; 2020.
24. Ureche LP, Billard A. Constraints extraction from asymmetrical bimanual tasks and their use in coordinated behavior. *Robot Auton Syst* 2018;**103**:222–35.
25. Hester T, Vecerik M, Pietquin O, et al. Deep q-learning from demonstrations. *Proceedings of the AAAI conference on artificial intelligence*; 2018.
26. Vecerik M, Sushkov O, Barker D, et al. A practical approach to insertion with variable socket position using deep reinforcement learning. *2019 international conference on robotics and automation (ICRA)*. Piscataway: IEEE Press; 2019. p. 754–60.
27. Zhu YK, Wang ZY, Merel J, et al. Reinforcement and imitation learning for diverse visuomotor skills. arXiv preprint:1802.09564; 2018.
28. Brys T, Harutyunyan A, Suay HB, et al. Reinforcement learning from demonstration through shaping. *Proceedings of the 24th international conference on artificial intelligence*; 2015. p. 3352–8.
29. Stark S, Peters J, Rueckert E. Experience reuse with probabilistic movement primitives. *2019 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. Piscataway: IEEE Press; 2019. p. 1210–17.
30. Ewerton M, Arenz O, Maeda G, et al. Learning trajectory distributions for assisted teleoperation and path planning. *Front Robot AI* 2019;**6**:89.
31. Ross S, Gordon GJ, Bagnell JA. A reduction of imitation learning and structured prediction to no-regret online learning. *Proceedings of the fourteenth international conference on artificial intelligence and statistics*; 2011. p. 627–35.
32. Rajeswaran A, Kumar V, Gupta A, et al. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. arXiv preprint:1709.10087; 2017.
33. Reed R. Pruning algorithms-a survey. *IEEE Trans Neural Netw* 1993;**4**(5):740–7.
34. Fang M, Li Y, Cohn T. Learning how to active learn: A deep reinforcement learning approach. arXiv preprint:1708.02383; 2017.
35. Colomé A, Torras C. Dual reps: a generalization of relative entropy policy search exploiting bad experiences. *IEEE Trans Robot* 2017;**33**(4):978–85.
36. Loew T, Bandyopadhyay T, Williams J, et al. Prompt: probabilistic motion primitives based trajectory planning. *Robotics: science and systems foundation*; 2021.
37. Kalakrishnan M, Chitta S, Theodorou E, et al. Stomp: stochastic trajectory optimization for motion planning. *2011 IEEE international conference on robotics and automation*. Piscataway: IEEE Press; 2011. p. 4569–74.