Behavior Cloning Assisted Reinforcement Learning for Cable-Driven Continuum Space Robots in Sparse Reward Environments

Xianru Tian[1], Bo Xia[1], Junbo Tan[1], Bo Yuan[2], Zhiheng Li[1], and Xueqian Wang[1]

***Abstract*—Deep reinforcement learning (DRL) has emerged as a powerful tool for controlling cable-driven continuum space robots (CDCSRs), offering a solution that bypasses complex system modeling. However, DRL based on dense reward functions (DRLDR) requires meticulous tuning of the reward structure, whereas DRL based on sparse reward functions (DRLSR) exhibits limited decision-making abilities, particularly in the space environments. To avoid extensive fine-tuning and enhance the performance in controlling CDCSRs, we propose the behavior cloning assisted twin delayed deep deterministic policy gradient (BATD3), a novel algorithm that integrates behavior cloning (BC) with DRLSR. Firstly, a DRLSR-based control framework is developed, which reformulates the control problem as a Markov decision process (MDP). Building on this, the BATD3 algorithm is proposed, comprising two training phases: the prior phase to train the BC model using demonstrations; the formal phase to pre-fill the RL replay buffer with demonstrations and successful BC-environment interaction trajectories, and optimize the RL model with the assistance of BC. Finally, extensive experiments are conducted in the MuJoCo environment to assess the performance of BATD3 in controlling CDCSRs. The results highlight the effectiveness, generalization, stability, robustness and potential of BATD3, along with the practicality and feasibility of the DRLSR-based control framework for CDCSRs.**

***Index Terms*—Reinforcement Learning, Space Robotics and Automation, Imitation Learning.**

I. Introduction

In recent decades, cable-driven continuum space robots (CDCSRs) have garnered significant attention due to light weight, high flexibility and decoupled motor-machinery structure. These features make CDCSRs particularly promising for applications such as clearing up space debris [1]. However, the complex kinematics and dynamics models of CDCSRs pose significant challenges for effective control [2], [3]. Meanwhile, deep reinforcement learning (DRL) has made remarkable progress in the domain of robotic control, including rigid manipulators [4], bipedal robots [5] and humanoid hands [6]. In particular, model-free DRL eliminates the need for explicit environmental modeling, offering a approach to controlling CDCSRs. However, selecting an appropriate DRL algorithm for controlling CDCSRs remains a challenging endeavor.

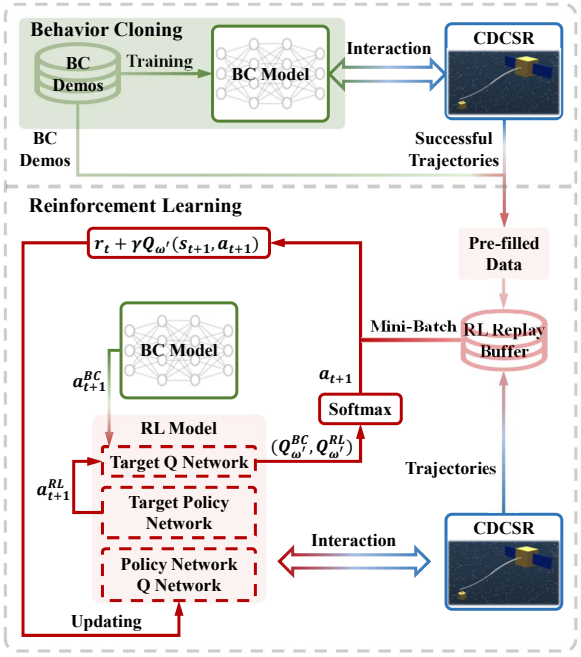

Fig. 1. Overview of BATD3. A DRLSR-based control framework is developed to reformulate the control problem as an MDP. Building on this framework, the BATD3 algorithm is introduced, consisting of two phases: 1) the prior phase to train the BC model; 2) the formal phase to pre-fill the replay buffer with demonstrations and successful BC-environment trajectories, and train the RL model with BC assistance.

DRL methods are broadly categorized into two types based on the reward structure: those using dense rewards (DRLDR) and those using sparse rewards (DRLSR). Compared to sparse rewards, dense rewards provide detailed feedback throughout the training process, often resulting in superior learning performance and efficiency. However, DRLDR has several notable limitations: 1) Its performance heavily depends on the design of the reward function, which requires careful tuning [7]. 2) The fine-tuning process demands domain-specific expertise, limiting accessibility to non-specialists. 3) In complex and uncertain space environments, system states are often subjected to disturbances, which can compromise the accuracy of dense reward functions and reduce the training stability.

In contrast, sparse reward functions avoid the complexities of reward shaping and the need for domain-specific knowledge, relying solely on whether the agent accomplishes the task. As a result, DRLSR is more accessible to non-specialists.

Manuscript received: January, 25, 2025; Revised April, 16, 2025; Accepted June 30, 2025.

This paper was recommended for publication by Editor Jens Kober upon evaluation of the Associate Editor and Reviewers' comments. This work was supported by the Natural Science Foundation of Shenzhen (No.JCYJ20230807111604008, No. JCYJ20240813112007010), the Natural Science Foundation of Guangdong Province (No.2024A1515010003), National Key Research and Development Program of China (No. 2022YFB4701400) and Cross-disciplinary Fund for Research and Innovation (No. JC2024002) of Tsinghua SIGS. (*Corresponding authors: Junbo Tan; Xueqian Wang. Co-first authors: Xianru Tian; Bo Xia.*)

[1]Xianru Tian, Bo Xia, Junbo Tan, Zhiheng Li, and Xueqian Wang are with the Shenzhen International Graduate School, Tsinghua University, Shenzhen 518055, China. txr23@mails.tsinghua.edu.cn; wang.xq@sz.tsinghua.edu.cn

[2]Bo Yuan is with School of Electrical Engineering and Computer Science, The University of Queensland, QLD 4072, Australia.

Digital Object Identifier (DOI): see top of this page.

However, sparse rewards only provide limited feedback, leading to suboptimal decision-making and learning, particularly for complex systems like CDCSRs. The complexity of CDCSR systems, the need to avoid extensive fine-tuning, and the challenge of enhancing decision-making capability of the agent collectively highlight the pivotal issue: developing an effective DRLSR method to control CDCSRs.

To address these challenges, we propose the behavior cloning assisted twin delayed deep deterministic policy gradient (BATD3) algorithm, a novel DRLSR algorithm that integrates behavior cloning (BC) [8] with the twin delayed deep deterministic policy gradient (TD3) [9], as illustrated in Fig. 1. Firstly, a robust control framework based on DRLSR is developed, which reformulates the control problem as a Markov decision process (MDP). Building on this framework, the BATD3 algorithm is proposed, consisting of two training phases: the prior phase to train the BC model and the formal phase to train the RL model. Specifically, the formal phase includes two design components: 1) pre-filling demonstrations and successful BC-environment interaction trajectories into the RL replay buffer, which alleviates the absence of high-quality data at the beginning of RL training; 2) optimizing RL with the assistance of BC, avoiding the use of suboptimal actions generated by the policy network of RL. Finally, extensive experiments are conducted in the MuJoCo environment to evaluate the performance of BATD3 in controlling CDCSRs.

The primary contributions of this paper are as follows:

- **A CDCSR control framework based on DRLSR.** We present a novel control framework that reformulates CDCSR control as an MDP, enabling integration with DRLSR.
- **The BATD3 algorithm.** BATD3 leverages BC to pre-fill the replay buffer with high-quality data, providing appropriate guidance, and to assist in RL optimization, preventing the use of suboptimal actions.
- **Experimental validation.** Experiments in the MuJoCo environment validate the effectiveness, generalization, stability, robustness and potential of BATD3, alongside the practicality and feasibility of the DRLSR control framework.

The remainder of this paper is organized as follows. Section II reviews related work, including DRL for continuum robots and DRL with demonstrations for robots. Section III outlines the preliminaries, covering RL elements for CDCSRs and a brief introduction to BC. Section IV elaborates BATD3 in two training phases. Section V presents experimental results and analysis. This paper is concluded in Section VI with directions for future work.

II. Related Work

A. DRL for Continuum Robots

DRL has significantly advanced the control of continuum robots, which can be categorized into two primary methodologies: 1) **DRLDR Methods.** These methods have been widely adopted to enhance the control performance of pneumatical soft continuum robots. Specifically, Deep Q-Network (DQN) with experience replay has been used to complete open-loop tasks [3], while Deep Deterministic Policy Gradient (DDPG) [10] has demonstrated success in close-loop tasks [11]. Furthermore, this category of methods has gradually been extended to cable-driven continuum robots. For example, TD3 has been utilized to automatically optimize manipulability during trajectory tracking [12]. Additionally, multiagent DRLDR methods have been applied to control rigid robots with multiple degree of freedoms (DoFs) [13] and dual-arm CDCSRs [1]. 2) **DRLSR Methods.** Soft Actor-Critic (SAC) [14] combined with random network distillation (RND) and Actor-Critic Policy Gradient have been implemented to control CDCSRs [15], [16]. However, these two actor-critic methods oversimplify the control process: the former directly controls joints instead of cables, while the later defines the control problem in discrete state space and action space.

B. Imitation Learning for Robots

Although algorithms such as hindsight experience replay (HER) [17] and RND enhance the learning efficiency of DRLSR, their reliance on online interaction limits their effectiveness in robotic control. In contrast, imitation learning (IL) exhibits significant potential in controlling rigid robots. Depending on how IL integrates with RL, IL methods for robotic control can be categorized as follows: 1) **Pretraining Offline and Fine-Tuning Online.** This category utilizes IL, such as BC, to pretrain the policy network, and fine-tune it during the online learning stage. Techniques consist of combining n-step DDPG and L2 regularization [18], and augmenting the original loss function with the BC loss term [19], [20]. Furthermore, the combination of BC and inverse reinforcement learning (IRL) [21] has been extensively explored [22], [23]. Based on the pretrained BC policy, these methods utilizes IRL to derive the underlying reward function from demonstrations and to fine-tune the policy through online agent-environment interactions. 2) **Independent IL and RL.** This category is derived from RL with demonstrations, which commonly pre-fills the replay buffer with demonstrations to guide RL towards reasonable learning directions [24], [25]. Building upon this, an independent IL model integrates with RL to adjust actions during both exploration and exploitation stages, enhancing the decision-making capability of the agent [26]. However, this method has been merely leveraged on rigid robots, and its applicability to controlling complex cable-driven robots in the harsh space environment remains uncertain.

III. Preliminaries

A. Reinforcement Learning for CDCSRs

RL aims at accomplishing sequential decision-making tasks within the framework of an MDP, defined as a tuple: $(\mathcal{S},\mathcal{A},\mathcal{P},\mathcal{R},\gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state space and the action space, respectively; $\mathcal{P}$ is the state transition function $\mathcal{P}(s_{t+1}|s_t,a_t)$, representing the probability of transitioning from the state s_t at time step t to the next state s_{t+1} after taking action a_t; $\mathcal{R}$ and γ signify the reward function and the discount factor, respectively. Building upon the MDP tuple, the return is defined as $G_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k}$. Furthermore, the Q function is denoted as $Q(s_t,a_t) = \mathbb{E}[G_t \mid s_t,a_t]$, which represents the expectation of the return. RL optimizes the policy $\pi_\phi(\cdot|s)$ by

maximizing the expected return, which in turn leads to the learning of an optimal Q function $Q^{\star}(s_t,a_t)=\max_{\pi_\phi} Q(s_t,a_t)$. Regarding to CDCSRs, the state space, the action space and the reward function are defined as follows.

• **State space $\mathcal{S}$.** To describe the spatial relationship between the end effector of CDCSRs and the target point, the state s should include the end effector position $p_e\in\mathbb{R}^3$, the end effector velocity $v_e\in\mathbb{R}^3$, the target point position $p_{tar}\in\mathbb{R}^3$, along with the distance $d\in\mathbb{R}$ between the end effector and the target point. Considering the free-floating base of CDCSRs, the state s should also include the base posture $\bar{p}_b\in\mathbb{R}^7$ and the base velocity $\bar{v}_b\in\mathbb{R}^6$. Since the base is a rigid body, $\bar{p}_b$ is composed of the center position $p_b\in\mathbb{R}^3$ and the quaternion attitude $h_b\in\mathbb{R}^4$, while $\bar{v}_b$ encompasses the center velocity $v_b\in\mathbb{R}^3$ and the angular velocity $\omega_b\in\mathbb{R}^3$. Additionally, the state s contains movement distances of all cables $\delta\in\mathbb{R}^{4n}$, where n is the number of linkage segments in CDCSRs. Therefore, the state of CDCSRs is defined as:

$$s=(p_e,v_e,p_{tar},d,\bar{p}_b,\bar{v}_b,\delta)\in\mathbb{R}^{4n+23}. \tag{1}$$

• **Action space $\mathcal{A}$.** Given that all cables of CDCSRs are actuated by slide joints, and the action $a\in\mathbb{R}^{4n}$ represents the target positions of these slide joints. Especially, since cables can only generate tension but not thrust, every action element at time step t satisfies $a_{t,i}\in[-\eta_i,0]$, where $i=1\cdots4n$ and η_i signifies the maximum control signal of the i-th slide joint.

• **Reward function $\mathcal{R}$.** Due to the sparse success-based reward, the reward function $\mathcal{R}$ at time step t is defined as:

$$r_t=\Phi(d_{t+1}\leqslant\Delta d), \tag{2}$$

where $\Phi(\cdot)$ is the indicative function, and Δd is the success threshold.

B. Behavior Cloning

IL can be broadly categorized into two paradigms, BC and IRL. BC outperforms IRL in resource efficiency, training speed and reliance on environment interactions, making it ideal for enhancing the control performance of DRLSR.

BC utilizes supervised learning to derive a policy $\pi_\theta^{BC}(\cdot|s)$ by maximizing the likelihood as follows:

$$\max_{\theta}\mathop{\mathbb{E}}_{(s,a)\in\mathcal{D}^{BC}}\log\pi_\theta^{BC}(a|s)], \tag{3}$$

where $\mathcal{D}^{BC}$ is a dataset of demonstrations and contains entire state-action tuples (s,a) of demonstrations. For continuous action space $\mathcal{A}$, BC policy $\pi_\theta^{BC}(\cdot|s)$ is typically modeled as a Gaussian distribution $\mathbb{N}\big(\mu_\theta^{BC}(s),\sigma_\theta^{BC}(s)\big)$, where $\mu_\theta^{BC}(s)$ and $\sigma_\theta^{BC}(s)$ are the mean and standard deviation, respectively. Commonly, $\sigma_\theta^{BC}(s)$ is assumed to be a constant with no dependence on the policy parameter θ. Therefore, (3) can be derived as follows:

$$\min_{\theta}\quad L^{BC}(\theta)=\mathop{\mathbb{E}}_{(s,a)\in\mathcal{D}^{BC}}\|a-\mu_\theta^{BC}(s)\|_2^2. \tag{4}$$

Through iterative optimization of the objective in (4), the loss function $L^{BC}(\theta)$ decreases continuously towards 0. At convergence, $\mu_\theta^{BC}(s)$ closely approximates the action a, where the tuple (s,a) is from $\mathcal{D}^{BC}$. Additionally, BC converges with only a few demonstrations, with no need for online agent-environment interactions.

IV. Methods

This section details the proposed algorithm, BATD3, and highlights its advantages. The training process of BATD3 is divided into two phases: the prior phase to train the BC model, and the formal phase to pre-fill the replay buffer and train the RL model with the assistance of BC.

A. The Prior Phase: Training the BC Model

The primary goal of the BC model is to produce high quality data and assist in updating the RL model. We employ demonstrations to train BC and fix its parameters in the formal training phase. As a result, the BC model will not suffer degradation in distribution and catastrophic forgetting in the next phase, which are commonly observed in the pretrain-finetune frameworks. Additionally, as presented in Section III-B, BC exhibits high accuracy in approximating expert actions from demonstrations, making it highly effective in aiding the next training phase. Besides, BC requires only a few demonstrations, bypassing the need for online agent-environment interactions. Therefore, while BC may suffer from distributional mismatch [27], its high accuracy, sample efficiency and stability make it an excellent choice as the assisting model.

B. The Formal Phase: Training the RL Model

In contrast to on-policy RL algorithms, off-policy RL algorithms, such as DDPG, SAC and TD3, reuse transition data in the replay buffer when updating their policy networks and Q networks, resulting in significantly high sample efficiency. Among these algorithms, TD3 possesses a rapid training speed and a concise algorithm structure, making it particularly suitable for integration with BC. Additionally, random ensemble distillation (RED) [28] is utilized to improve the sample efficiency. Therefore, TD3 integrated with RED is adopted as the base RL backbone for BATD3.

To address the challenges of collecting valid data solely through interactions between DRLSR and the environment, the replay buffer $\mathcal{D}^{RL}$ is pre-filled with demonstrations in advance of training RL. Furthermore, given the prohibitive cost of collecting demonstrations in the space environment, high-quality data must be augmented based on these initial demonstrations. To achieve this, BC is employed to interact with the environment and gather successful trajectories, pre-filling the replay buffer. These trajectories, combined with the original demonstrations, can help mitigate the scarcity of superior data at the beginning of training.

The training process in this phase is divided into two stages: the exploration stage and the exploitation stage. In the exploration stage, RL interacts with the environment to explore unknown regions of CDCSRs' workspace and collect interaction trajectories. As training progresses, the quality of trajectories is ascending gradually. In the exploitation stage, BC is leveraged to assist in updating the RL model. Specifically, when constructing the TD target at each update step,

the BC mean network μ_θ^{BC} and the RL target policy network $\pi_{\phi'}^{RL}$ generate actions as follows:

$$\begin{cases} a_{t+1}^{BC} = \mu_\theta^{BC}(s_{t+1}), \\ a_{t+1}^{RL} = \pi_{\phi'}^{RL}(s_{t+1}). \end{cases} \tag{5}$$

Following that, we employ the target Q network $Q_{\omega'}(s,a)$ to evaluate the action values of these two actions: $Q_{\omega'}^{BC} = Q_{\omega'}(s_{t+1}, a_{t+1}^{BC})$ and $Q_{\omega'}^{RL} = Q_{\omega'}(s_{t+1}, a_{t+1}^{RL})$. Since the *softmax* function is continuous and differentiable, suitable for gradient descent during updating neural networks, a discrete probability distribution is constructed with this function to determine the action a_{t+1}:

$$\mathbf{P}(a_{t+1}) = \begin{cases} \dfrac{\exp(\rho Q_{\omega'}^{BC})}{\exp(\rho Q_{\omega'}^{BC}) + \exp(\rho Q_{\omega'}^{RL})}, & a_{t+1} = a_{t+1}^{BC}, \\ \dfrac{\exp(\rho Q_{\omega'}^{RL})}{\exp(\rho Q_{\omega'}^{BC}) + \exp(\rho Q_{\omega'}^{RL})}, & a_{t+1} = a_{t+1}^{RL}, \end{cases} \tag{6}$$

where ρ is a scaling factor of Q values. By randomly sampling from this probability distribution, the action a_{t+1} is determined and the TD target is computed as:

$$y_t = r_t + \gamma Q_{\omega'}(s_{t+1}, a_{t+1}). \tag{7}$$

Grounded in (7), the Q network and the policy network will be optimized progressively.

Utilizing BC to assist in updating RL is highly beneficial. During the initial period of this phase, the Q network may struggle to accurately evaluate Q values. However, the use of pre-filled high-quality data ensures that the batches used for updates can minimize the impact of inaccurate Q values. As training advances, the Q network's evaluation accuracy improves, allowing a_{t+1}^{BC} to serve as minimal guarantee for a_{t+1}. Since cables are composed of flexible materials and the space environment is weightless, the tension in cables tends to produce spikes and outliers, which causes RL to generate suboptimal a_{t+1}^{RL}. By replacing a_{t+1}^{RL} with a_{t+1}^{BC} based on the Q network's evaluation, the training process avoids using poor data, ultimately improving the training performance of RL.

V. Experiments and Results

This section presents a comprehensive description of the experiments to validate the effectiveness, generalization, robustness, stability and potential of BATD3. It includes the experiment setup, results and detailed analysis.

A. Experiment Setup

1) Experiment System: The simulation environment, developed in MuJoCo, is based on the mechanical structure of CDCSRs, as shown in Fig.2 (a). The CDCSR consists of a free-floating base and 12 links, with all links interconnected through a pair of cross-intersected hinge joints. These links are divided into two groups, forming two linkage segments, namely $n = 2$. In theory, CDCSRs possess two types of cables: linkage cables and actuating cables. Each segment is constrained and coupled by linkage cables, ensuring the equal angles between adjacent links. Therefore, all links in a segment behave as an integrated structure with 2 DoFs [29]. For simulation purposes, the effects of the linkage cables are modeled by adjusting the stiffness and damping of the joints, simplifying their physical representation. Meanwhile, each segment is driven by 4 actuating cables which are modeled by the tendon geometry in MuJoCo. The structure parameters are detailed in Table I, using standard units.

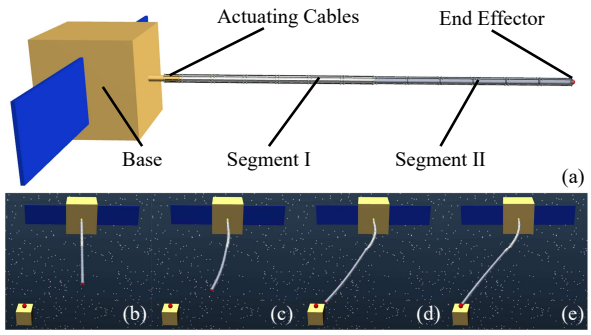

Fig. 2. The MuJoCo environment of CDCSRs. (a) illustrates the mechanical structure of CDCSRs. (b) to (e) illustrate the process of reaching the target.

TABLE I
Structure Parameters

Structure	Shape	Mechanical Parameters
Base	Box $l = 0.6$	$m = 500$ $I = \text{diag}(40, 35, 100)$
Link	Cylinder $l = 0.2$, $r = 0.02$	$m = 1.18$ $I = \text{diag}(0.016, 0.016, 0.00037)$
Joint	—	$stiffness = 2.00$ $damping = 1.05$

*All data use standard units.

Furthermore, the simulation time step is set to $\Delta T = 0.0025$s and the maximum episode length is configured as $L_{epi} = 250$. Additionally, the i-th slide joint's maximum control signal η_i is defined as $\eta_i = 5$, with $i = 1 \cdots 4n$ to ensure sufficient operational space.

2) Task Description: Since reaching the target point is the fundamental process in space debris cleanup, we perform this task across different operation spaces of CDCSRs to access the control performance of BATD3. Fig.2 (b) to (e) illustrates a successful execution of the task. Based on different task objectives, we define three categories of operation spaces:

- **Anchor Space.** For large-sized debris, CDCSRs entail to reach all parts of the debris to ensure the accuracy of manipulation. Therefore, the Anchor Space is defined as a cuboid space with fixed dimensions, representing the simplified large-sized debris, and the target points of demonstrations are located within this Space.
- **Floating Space.** Due to stochastic floating in the space environment, the position of small-sized debris may shift. Therefore, the agent must accomplish tasks within the neighborhood of original target points. Based on this, the Floating Space is defined as the neighborhood $\mathbf{U}(p_{tar}^D, \xi_F)$, where $\mathbf{U}$ signifies the neighborhood space, p_{tar}^D represents all target points of demonstrations and $\xi_F = 0.01$m denotes the radius of the neighborhood.

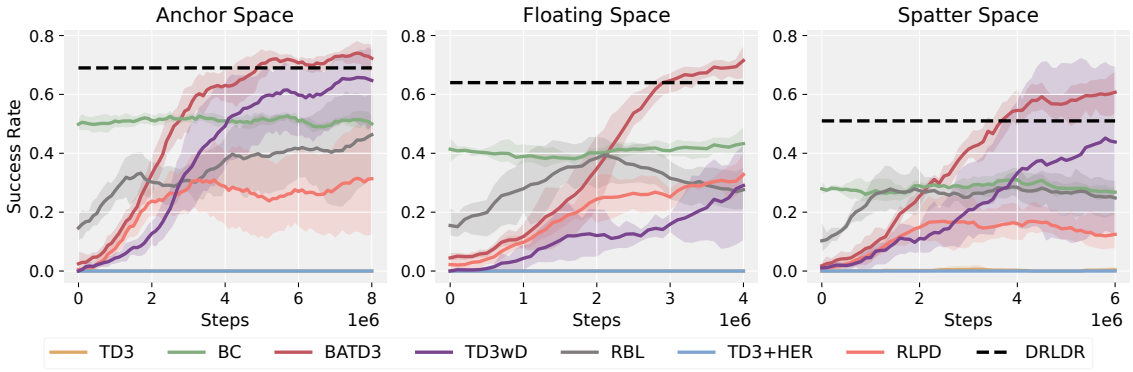

Fig. 3. Training results of comparison experiments. Across all operation spaces, BATD3 outperforms all baselines in the training process.

TABLE II
EVALUATION RESULTS OF COMPARISON EXPERIMENTS

Operation Space		BC	BATD3	TD3wD	RBL	TD3	TD3+HER	RLPD
Anchor Space	κ	0.54	**0.76**	0.37	0.26	0.00	0.00	0.34
	l_{epi}	179.4 ± 67.3	$\mathbf{144.6 \pm 63.2}$	194.2 ± 73.7	215.5 ± 59.8	250.0 ± 0.0	250.0 ± 0.0	210.9 ± 59.8
Floating Space	κ	0.37	**0.79**	0.31	0.32	0.00	0.00	0.31
	l_{epi}	203.1 ± 64.3	$\mathbf{135.8 \pm 63.7}$	210.6 ± 63.3	204.8 ± 67.3	250.0 ± 0.0	250.0 ± 0.0	211.8 ± 59.6
Spatter Space	κ	0.25	**0.65**	0.37	0.35	0.00	0.00	0.16
	l_{epi}	216.6 ± 61.0	$\mathbf{151.5 \pm 74.6}$	195.3 ± 71.9	204.1 ± 65.2	250.0 ± 0.0	250.0 ± 0.0	227.6 ± 54.1

• **Spatter Space.** When cleaning up large-sized debris, small debris may detach due to collision. To ensure CDCSRs can capture these scattered debris pieces, the Spatter Space is defined as $\mathbf{U}(p^D_{tar}, \xi_S)$, with $\xi_S = 0.15$m, which can fully envelope the Anchor Space.

Experiments are conducted in these operation spaces and the agent is trained using 10 demonstrations. Additionally, both the end effector and the target point are modeled as spheres, with their radii $r_e = 0.01$m and $r_{tar} = 0.018$m, respectively. The distance threshold Δd is defined as $\Delta d = 0.03$m, satisfying $r_e + r_{tar} \leqslant \Delta d$, which ensures the end effector can physically reach the target point. At the beginning of experiments, CDCSRs remain stationary, with all driving cables situated at their initial positions and the base is located at the origin of the world coordinate system.

3) Network Architecture: The BC policy network is implemented as an MLP with three hidden layers, each consisting of 256 neurons with Tanh as the activation function. The RL policy network and Q network are constructed as MLPs with identical hidden layers but use ReLU as the activation function. The key hyperparameters are configured as follows: discount factor $\gamma = 0.99$, learning rate $lr = 1 \times 10^{-4}$, batch size $B = 256$, soft updating factor $\beta = 0.01$ used in updating target networks, noise clip parameter $c = 0.3$ leveraged in the clip function of TD3, softmax temperature parameter $\rho = 10$, large ensemble size $E = 5$ and number of critic targets $Z = 2$ used in RED.

B. Comparison Experiments

1) Baseline Description: BATD3 is evaluated against several well-established algorithms. Among them, BC and TD3 are their original formulations. TD3+HER combines TD3 with HER, an effective approach to improving the sample efficiency in robotic tasks with sparse rewards. TD3 with demonstrations (TD3wD) integrates demonstrations into the replay buffer of TD3. Additionally, TD3 regularized with the BC loss function (RBL) incorporates the BC loss, shown in (4), into original loss function $L^{RL}(\phi)$ of RL's Q network to improve the decision-making ability. Besides, RBL employs the soft Q-filtering [22] to adaptively balance the proportion between $L^{BC}(\theta)$ and $L^{RL}(\phi)$. Besides, Reinforcement Learning with Prior Data (RLPD) [25] is a state-of-the-art DRLSR algorithm that has demonstrated strong performance across various robotic control tasks. All baselines leverage TD3 combined with RED as their core RL backbone.

2) Comparison Results: Comparison experiments are conducted across three operation spaces. Fig.3 illustrates the training curves of success rate κ, with the success rate of DRLDR included as a standard reference. Each curve represents the mean performance over 5 random seeds, with the shaded region illustrating the 95% confidence interval. Furthermore, 100 evaluation episodes are conducted for all models to evaluate their effectiveness. Similar to the exploration stage of the formal phase, only the RL model interacts with the environment during conducting evaluation experiments. The corresponding success rate κ and episode length l_{epi} are presented in Table II. As illustrated by Fig.3: i) The success rates of TD3 and TD3+HER remain at zero throughout the training process. Although HER exhibits remarkable enhancement in utilizing DRLSR to control rigid robots, it is unsuitable for controlling CDCSRs. ii) Due to the BC loss function $L^{BC}(\theta)$, RBL demonstrates the fastest initial training speed. However, its overall performance is limited by BC, with its success rate failing to surpass that of BC and even declining in the later stage of training. iii) RLPD shows inferior performance across all three operation spaces. Since RLPD enforces the use of 50% demonstration data in every training update, the limited

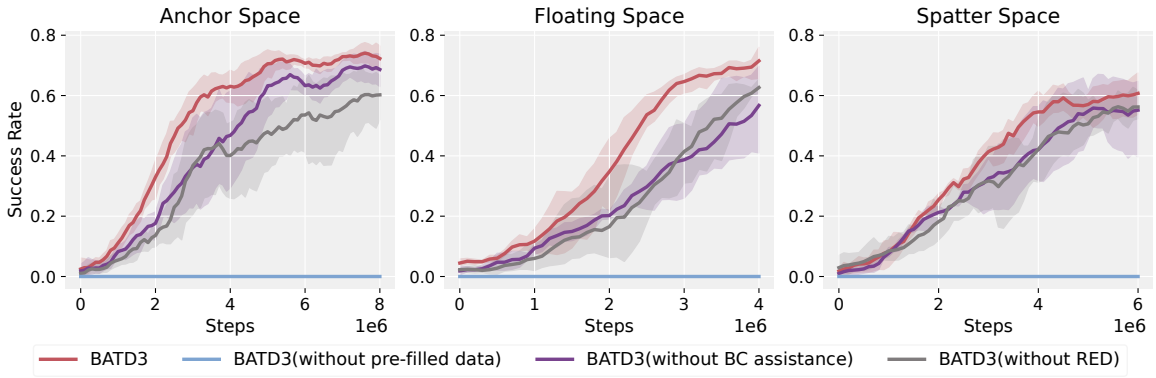

Fig. 4. Training results of ablation experiments. The results validate the pivotal roles of pre-filled data, BC assistance, and RED.

number of demonstrations may be overexploited. In addition, the utilization of environment-interaction data is relatively insufficient, which limits data diversity and negatively affects the overall training process. iv) In both the Anchor Space and the Spatter Space, BATD3 and TD3wD achieve the best training performance. In comparison, BATD3 exhibits superior characteristics, including faster training speed, higher final success rate and narrower confidence interval compared to TD3wD, underscoring its advantages across multiple aspects. Moreover, BATD3 outperforms DRLDR in all operation spaces. As indicated by Table II, BATD3 achieves the shortest episode length compared to all baselines, indicating its capability to complete tasks in minimal cycles.

Given the operational requirements for CDCSRs to efficiently clean up debris in the space environment, the control method must facilitate rapid deployment, precise operation, and swift task completion. Moreover, it must demonstrate sufficient stability to maintain high performance under different random conditions. BATD3 fulfills these requirements by offering a faster learning process, higher success rate, shorter task cycle and narrower confidence interval compared to all baselines. Therefore, these attributes highlight the strong applicability and effectiveness of BATD3 for controlling CDCSRs.

C. Ablation Experiments

Ablation experiments are conducted by separately removing BC assistance, pre-filled data and RED, to validate the contribution of each design component and the improvement in sample efficiency brought by RED. As shown in Fig. 4: 1) Removing BC assistance leads to a slower learning speed, a reduced final success rate and a wider confidence interval, particularly in the Floating Space. This underscores the role of BC assistance in improving sample efficiency, enhancing learning performance and ensuring the stability of BATD3. 2) Without pre-filled data, the success rate remains at zero throughout the training process. This indicates that, without the guidance of high-quality data, the agent is incapable of extracting meaningful information from the space environment and thus trapped in ineffective optimization, resulting in erroneous decision-making even with the assistance of BC. 3) The absence of RED results in a decreased success rate and a slower learning speed. This demonstrates that RED has indeed improved the sample efficiency while also maintaining a relatively high update-to-data ratio, which significantly enhances training effectiveness.

These results affirm the pivotal importance of BC assistance, pre-filled data and RED. Pre-filled data reduces ineffective exploration during BATD3 training, which is especially crucial in the complex and resource-constrained space environments. Meanwhile, BC assistance enhances the performance and stability for controlling CDCSRs, while RED significantly contributes to improving sampling efficiency.

D. Validation of Dependence on Demonstration Quantity

Additional experiments are conducted to investigate the dependence on demonstration quantity, using BATD3, TD3wD, and BC with different numbers of demonstrations. As described in Section V-A, the extents of the Floating Space and the Spatter Space are affected by the number of demonstrations. In contrast, the Anchor Space is a cuboid space with fixed dimensions. Therefore, to mitigate the potential impact of fluctuations in the extents of operation spaces, these experiments are solely conducted in the Anchor Space. As illustrated by Fig. 5: 1) Reducing the number of demonstrations significantly degrades the performance of TD3wD and BC. In contrast, BATD3 exhibits a moderate decline with different numbers of demonstrations. 2) Across all demonstration quantities, BATD3 outperforms TD3wD and BC in both the success rate and the confidence interval during training. These results highlight BATD3's low dependence on demonstration quantity, which validates that BATD3 possesses sufficient stability for controlling CDCSRs in space environments, where acquiring demonstrations is particularly challenging.

E. Generalization and Robustness Validation

1) Generalization Validation: Although the task objectives of the three operation spaces are different, the process of cleaning up space debris commonly contains various types of objectives. Therefore, a model trained in one operation space should be capable of generalizing effectively to others. To assess this, models are evaluated over 100 episodes in different operation spaces. The results, presented in Table III, reveal the following: i) Due to the distinct shape of the Anchor Space compared to the Floating Space and the Spatter Space, the performance of the model trained in the Anchor Space exhibits a certain extent of decline. However, compared to

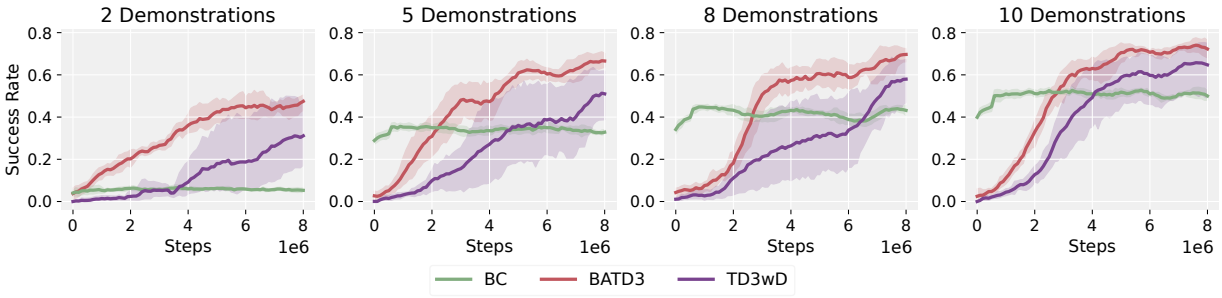

Fig. 5. Validation of dependence on demonstration quantity. These results confirm that BATD3 is more suitable for tasks with limited demonstrations.

TABLE III
GENERALIZATION VALIDATION

Training Space	Validation Space	Data Category	κ	l_{epi}
Anchor Space	**Floating Space**	Original	0.79	135.8 ± 63.7
		Current	0.49	183.2 ± 70.6
	Spatter Space	Original	0.65	151.5 ± 74.6
		Current	0.33	200.4 ± 71.5
Floating Space	**Anchor Space**	Original	0.76	144.6 ± 63.2
		Current	0.65	156.1 ± 71.1
	Spatter Space	Original	0.65	151.5 ± 74.6
		Current	0.63	152.4 ± 77.3
Spatter Space	**Anchor Space**	Original	0.76	144.6 ± 63.2
		Current	0.65	158.1 ± 70.0
	Floating Space	Original	0.79	135.8 ± 63.7
		Current	0.67	151.7 ± 71.1

TABLE IV
ROBUSTNESS VALIDATION

Parameter		Anchor Space		Floating Space		Spatter Space	
		κ	l_{epi}	κ	l_{epi}	κ	l_{epi}
σ_a	**0**	0.76	144.6 ± 63.2	0.79	135.8 ± 63.7	0.65	151.5 ± 74.6
	0.5	0.75	148.2 ± 62.4	0.79	137.7 ± 62.8	0.65	153.5 ± 73.1
	1.0	0.75	151.5 ± 60.7	0.77	144.1 ± 62.7	0.63	159.4 ± 71.6
	1.5	0.74	155.9 ± 59.2	0.72	155.7 ± 62.8	0.59	168.5 ± 69.9
	2.0	0.71	165.3 ± 57.6	0.70	164.8 ± 60.7	0.57	172.7 ± 67.2
σ_b	**0**	0.76	144.6 ± 63.2	0.79	135.8 ± 63.7	0.65	151.5 ± 74.6
	0.01	0.74	147.1 ± 64.7	0.76	139.4 ± 66.6	0.65	153.0 ± 73.7
	0.02	0.71	151.0 ± 67.0	0.70	147.4 ± 70.9	0.62	156.5 ± 75.0
	0.03	0.68	155.2 ± 68.9	0.67	152.0 ± 72.6	0.61	161.2 ± 73.7
p	**0**	0.76	144.6 ± 63.2	0.79	135.8 ± 63.7	0.65	151.5 ± 74.6
	0.2	0.76	144.5 ± 63.2	0.79	135.9 ± 63.7	0.65	151.4 ± 74.7
	0.4	0.76	144.5 ± 63.3	0.79	136.2 ± 63.5	0.65	151.2 ± 74.8
	0.6	0.75	144.5 ± 64.3	0.79	136.4 ± 63.2	0.64	152.5 ± 75.3
	0.8	0.75	148.4 ± 63.4	0.77	143.0 ± 64.6	0.62	156.6 ± 75.0
	1.0	0.00	250.0 ± 0.0	0.00	250.0 ± 0.0	0.00	250.0 ± 0.0

the data in Table II, BATD3 remains better than baselines in the Floating Space and as competitive as them in the Spatter Space. ii) The performance of models trained in the Floating Space and the Spatter Space manifests only a slight decline, emphasizing BATD3's ability to generalize across different operation spaces.

2) Robustness Validation: The harsh space environment demands that BATD3 possesses sufficient robustness to resist noise and communication failures. To evaluate this, BATD3 is tested over 100 episodes under three different scenarios: i) Action Noise, represented as Gaussian noise $\Delta a_t \sim \mathbb{N}(0, \sigma_a)$; ii) Base Position Noise, modeled as $\Delta p_{b,t} \sim \mathbb{N}(0, \sigma_b)$; iii) Packet Loss, simulated by discarding s_{t+1} with a probability p and setting $s_{t+1} = s_t$. Results of these scenarios are presented in Table IV. These results illustrate that despite the presence of significant noise or a high probability of packet loss, the performance of BATD3 merely exhibits a minimal decrease, remaining under 20%.

Consequently, BATD3 not only possesses adequate generalization in cleaning multiple types of debris, but also maintains high robustness against noise disturbances and communication failures.

F. Validation of Non-cooperative Target Capture

To further evaluate the capability of BATD3 in addressing complex scenarios such as non-cooperative target capture, the target-following task is conducted in this section. In this task, the end effector continues to follow the target, ensuring the distance between them remains below the distance threshold Δd. As the BATD3 agent can only observe the current position of the target without access to its motion pattern or any form of information exchange, the task represents a typical non-cooperative target capture scenario.

Visualization of this target-following task is presented in Fig.6 (b) to (e), the white line and the red line correspond to motion trajectories of the end-effector and the target point, respectively. These subfigures demonstrate that BATD3 effectively controls the end effector to closely track the target as it moves downward along a straight-line path. Besides, the position errors and attitude errors of the free-floating base are presented in Fig. 6 (a), which illustrates that the position error and the attitude error are respectively maintained below 3 mm and 0.05 rad throughout the entire target-following process. These results indicate that BATD3 ensures adequate base stability in the target-following tasks of CDCSRs.

In summary, BATD3 shows promising potential in controlling CDCSRs during non-cooperative target capture missions. This experiment further substantiates the superior control effectiveness of BATD3.

VI. CONCLUSION

This paper addresses the challenges in controlling CDCSRs through DRLSR, by introducing the behavior cloning assisted twin delayed deep deterministic policy gradient (BATD3) algorithm. Specifically, the control process of CDCSRs is firstly formulated as an MDP to seamlessly integrate with DRLSR. Build upon this framework, we present BATD3, which includes two training phases: the prior phase to train

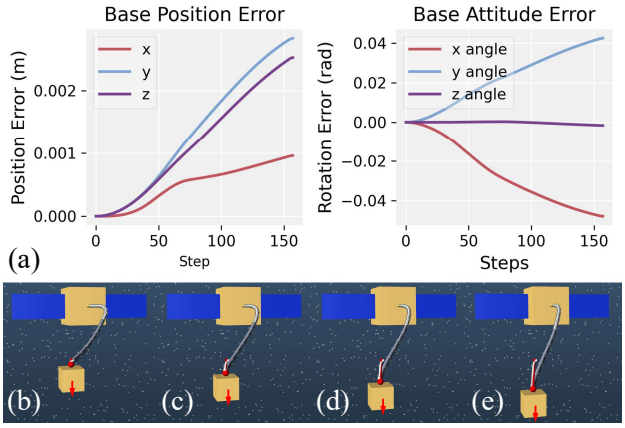

Fig. 6. Experiments of the target-following task. The end effector tracks the target as it moves linearly downward.

BC with demonstrations; the formal phase to pre-fill the replay buffer with demonstrations and successful BC-environment interaction trajectories, and train RL with the assistance of BC for eliminating suboptimal actions generated by the RL policy network. Finally, extensive experiments are conducted in the MuJoCo simulation environment. According to the experiments, the outstanding effectiveness compared to baselines, predominant generalization across different operation spaces, low dependence on demonstration quantity, remarkable robustness against disturbance and potential in performing non-cooperative target capture missions are validated.

Future research directions could involve incorporating visual imitation learning, developing DRLSR algorithms tailored to multi-arm CDCSR systems, and implementing our algorithm on physical robotic platforms.

REFERENCES

[1] D. Jiang, Z. Cai, H. Peng, and Z. Wu, "Coordinated control based on reinforcement learning for dual-arm continuum manipulators in space capture missions," *Journal of Aerospace Engineering*, vol. 34, no. 6, p. 04021087, 2021.

[2] Y. Chen, T. Li, W. Cui, G. Tuo, X. Liu, and Y. Wang, "A deep reinforcement learning based efficient optimization solution method for inverse kinematics of hyper-redundant robot," in *2022 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 501–506, 2022.

[3] S. Satheeshbabu, N. K. Uppalapati, G. Chowdhary, and G. Krishnan, "Open loop position control of soft continuum arm using deep reinforcement learning," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 5133–5139, 2019.

[4] Y.-H. Wu, Z.-C. Yu, C.-Y. Li, M.-J. He, B. Hua, and Z.-M. Chen, "Reinforcement learning in dual-arm trajectory planning for a free-floating space robot," *Aerospace Science and Technology*, vol. 98, p. 105657, 2020.

[5] Z. Li, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Reinforcement learning for versatile, dynamic, and robust bipedal locomotion control," *The International Journal of Robotics Research*, 2024.

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

[7] A. Y. Ng, D. Harada, and S. Russell, "Policy invariance under reward transformations: Theory and application to reward shaping," in *Icml*, pp. 278–287, 1999.

[8] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," *Advances in neural information processing systems*, 1988.

[9] S. Fujimoto, H. van Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proceedings of the 35th International Conference on Machine Learning*, Proceedings of Machine Learning Research, pp. 1587–1596, PMLR, 2018.

[10] T. Lillicrap, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[11] S. Satheeshbabu, N. K. Uppalapati, T. Fu, and G. Krishnan, "Continuous control of a soft continuum arm using deep reinforcement learning," in *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*, pp. 497–503, 2020.

[12] H. Yang, X. Li, D. Meng, X. Wang, and B. Liang, "Manipulability optimization of redundant manipulators using reinforcement learning," *Industrial Robot: the international journal of robotics research and application*, no. 5, pp. 830–840, 2023.

[13] G. Ji, J. Yan, J. Du, W. Yan, J. Chen, Y. Lu, J. Rojas, and S. S. Cheng, "Towards safe control of continuum manipulator using shielded multiagent reinforcement learning," *IEEE Robotics and Automation Letters*, pp. 7461–7468, 2021.

[14] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proceedings of the 35th International Conference on Machine Learning*, Proceedings of Machine Learning Research, pp. 1861–1870, 2018.

[15] C. Yang, J. Yang, X. Wang, and B. Liang, "Control of space flexible manipulator using soft actor-critic and random network distillation," in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 3019–3024, 2019.

[16] C. Frazelle, J. Rogers, I. Karamouzas, and I. Walker, "Optimizing a continuum manipulator's search policy through model-free reinforcement learning," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5564–5571, 2020.

[17] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba, "Hindsight experience replay," *Advances in neural information processing systems*, 2017.

[18] T. Hester, M. Vecerik, O. Pietquin, M. Lanctot, T. Schaul, B. Piot, D. Horgan, J. Quan, A. Sendonaris, I. Osband, *et al.*, "Deep q-learning from demonstrations," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, 2018.

[19] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Overcoming exploration in reinforcement learning with demonstrations," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6292–6299, 2018.

[20] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," *arXiv preprint arXiv:1709.10087*, 2017.

[21] A. Y. Ng, S. Russell, *et al.*, "Algorithms for inverse reinforcement learning.," in *Icml*, p. 2, 2000.

[22] S. Haldar, V. Mathur, D. Yarats, and L. Pinto, "Watch and match: Supercharging imitation with regularized optimal transport," in *Conference on Robot Learning*, pp. 32–43, PMLR, 2023.

[23] S. Haldar, J. Pari, A. Rai, and L. Pinto, "Teach a robot to fish: Versatile imitation from one minute of demonstrations," *arXiv preprint arXiv:2303.01497*, 2023.

[24] M. Vecerik, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. Riedmiller, "Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards," *arXiv preprint arXiv:1707.08817*, 2017.

[25] P. J. Ball, L. Smith, I. Kostrikov, and S. Levine, "Efficient online reinforcement learning with offline data," in *Proceedings of the 40th International Conference on Machine Learning*, Proceedings of Machine Learning Research, pp. 1577–1594, PMLR, 2023.

[26] H. Hu, S. Mirchandani, and D. Sadigh, "Imitation bootstrapped reinforcement learning," *arXiv preprint arXiv:2311.02198*, 2023.

[27] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, Proceedings of Machine Learning Research, pp. 627–635, 2011.

[28] X. Chen, C. Wang, Z. Zhou, and K. Ross, "Randomized ensembled double q-learning: Learning fast without a model," *arXiv preprint arXiv:2101.05982*, 2021.

[29] T. Liu, W. Xu, T. Yang, and Y. Li, "A hybrid active and passive cable-driven segmented redundant manipulator: Design, kinematics, and planning," *IEEE/ASME Transactions on Mechatronics*, pp. 930–942, 2021.