Survive on Planet Pandora: Robust Cross-Domain RL Under Distinct State-Action Representations

[1]Kuan-Chen Pan, [1]Ming-Hong Chen, [2]Xi Liu, [1]Ping-Chun Hsieh
[1]Department of Computer Science, National Yang Ming Chiao Tung University
[2]Applied Machine Learning, Meta AI, Menlo Park, CA, USA
`{penny644.cs11, mhchen1224.cs12, pinghsieh}@nycu.edu.tw, xiliu.tamu@gmail.com`

Abstract

Cross-domain reinforcement learning (CDRL) is meant to improve the data efficiency of RL by leveraging the data samples collected from a source domain to facilitate the learning in a similar target domain. Despite its potential, cross-domain transfer in RL is known to have two fundamental and intertwined challenges: (i) The source and target domains can have distinct representations (either in states or actions), and this makes direct transfer infeasible and thereby requires sophisticated inter-domain mappings; (ii) The domain similarity in RL is not easily identifiable a priori, and hence CDRL can be prone to negative transfer. In this paper, we propose to jointly tackle these two challenges through the lens of hybrid Q functions. Specifically, we propose QAvatar, which combines the Q functions from both the source and target domains with a proper weight decay function. Through this design, we characterize the convergence behavior of QAvatar and thereby show that QAvatar achieves robust transfer in the sense that it effectively leverages a source-domain Q function for knowledge transfer to the target domain, regardless of the quality of the source-domain model and domain similarity. Through extensive experiments, we demonstrate that QAvatar achieves superior transferability across domains on a variety of RL benchmark tasks, including locomotion and robot arm manipulation, even in the scenarios of potential negative transfer.

1 Introduction

Reinforcement learning (RL) has witnessed significant progress in various challenging domains, such as game playing [1, 2], robot control [3, 4], and language models [5], mainly due to the integration of general RL techniques with advancements in data collection and computation for large-scale training. However, data inefficiency of RL remains one significant obstacle to its deployment in many real-world applications, where online data collection is either costly (e.g., robotics and autonomous driving) or even hazardous (e.g., medical treatments). As one promising solution, cross-domain RL (CDRL) serves as a practical framework to improve the sample efficiency of RL from the perspective of transfer learning, which leverages the data or the pre-trained models from a source domain to enable knowledge transfer to the target domain, under the presumption that the data collection and model training are much less costly in the source domain (e.g., simulators).

A plethora of the existing CDRL methods focuses on knowledge transfer across environments that share the same state-action spaces but with different transition dynamics. This setting has been extensively studied from a variety of perspectives, such as domain randomization [6], learning similarity metrics [7], reward augmentation [8, 9], and data filtering [10]. Despite the above progress, to fully realize the promise of CDRL, there are two further fundamental challenges to tackle: (i) *Distinct state and/or action representations between domains*: To support flexible transfer across a wide variety of domains, the generic CDRL algorithms are required to address the discrepancies in the state and action representations between source and target domains. Take robot control as an example. One common scenario is to apply direct policy transfer across robot agents of different morphologies [11], which naturally leads to discrepancy in representations. This discrepancy significantly complicates the transfer of either data samples or learned source-domain models. (ii) *Unknown domain similarity and negative transfer*: Typical CDRL presumes that the source and target domains are sufficiently similar such that effective transfer is achievable. However, in practice, given that the data budget of the target domain is limited, it is rather difficult to determine a priori the similarity of a pair of domains, and this becomes even more challenging when the state-action representations of the two domains are distinct. Moreover, this issue can also be highlighted by the phenomenon of negative transfer [12, 13], where transfer learning from the source domain can have a negative impact on the target domain. As a consequence, despite that CDRL has been shown to succeed in various scenarios, without a proper design, the performance of CDRL could actually be much worse than the vanilla target-domain model learned without using any source knowledge beyond these good-case scenarios. Notably, to tackle (i), several approaches have been proposed to address such representation discrepancy by learning state-action correspondence, either in the typical RL [14] or unsupervised settings [11, 15]. However, these existing solutions are all oblivious to the issues of domain dissimilarity and negative transfer and therefore do not provide any performance guarantees. As a result, one fundamental research question about CDRL remains largely open: *How to achieve efficient and robust cross-domain transfer in RL across domains of distinct state-action representations with worst-case guarantees?*

In this paper, we answer the above question in the affirmative. Specifically, we revisit the cross-domain transfer problem in RL from the perspective of *mixing the source-domain and target-domain Q functions* and propose a new CDRL framework termed *QAvatar*, where an "*avatar*", as described in the movie *Avatar*, refers to a genetically engineered body that is created by combining human DNA with the DNA of the native inhabitants of the alien moon *Pandora*. These avatars allow humans on Earth to remotely control these bodies and quickly adapt to the toxic environment of the planet Pandora. By drawing an analogy between the cross-planet transfer of humans and the cross-domain transfer of models in RL, we propose to construct a QAvatar, which updates the target-domain policy based on the weighted combination of the learned target-domain Q function and the given source-domain Q function and learn the state-action correspondence by minimizing a cross-domain Bellman loss. To substantiate this idea, we first present a prototypical algorithm of QAvatar in the tabular setting and establish that QAvatar enjoys a nice upper bound on the sub-optimality under a properly designed weight decay function, regardless of the similarity between the source and target domains. This result also suggests that QAvatar can achieve improved sample efficiency of CDRL while preventing the potential negative transfer. Based on these findings, we further propose a practical implementation by integrating the QAvatar algorithm with a neural mapping function based on a normalizing flow model in learning the state-action correspondence.

The main contributions of this paper can be summarized as follows: 1) We propose the QAvatar framework that achieves knowledge transfer between two domains with distinct state and action spaces for improving sample efficiency. We then present a prototypical QAvatar algorithm and establish its convergence property, showing that QAvatar can improve sample efficiency while avoiding negative transfer. 2) We further substantiate the QAvatar framework by proposing a practical implementation with a normalizing-flow-based state-action mapping. This further demonstrates the compatibility of QAvatar with off-the-shelf methods for learning state-action correspondence. 3) Through extensive experiments and an ablation study, we show that QAvatar significantly outperforms the benchmark CDRL algorithms in various popular RL benchmark tasks, regardless of the quality of source-domain models and domain similarity.

2 Related Work

CDRL across domains with distinct state and action representations. The existing approaches can divided into three main categories: (i) *Manually designed latent mapping*: In [16] and [17], the trajectories are mapped manually and by sparse coding from the source domain and the target domain to a common latent space, respectively. The distance between latent states can then be calculated to find the correspondence of the states from the different domains. In [18], the correspondence of the states is found by dynamic time warping and the mapping function which can map the states from two domains to the latent space is found by the correspondence. (ii) *Learned inter-domain mapping*: In the literature [19, 11, 20, 15, 21], the inter-domain mapping is mainly learned by enforcing dynamics alignment (or termed dynamics cycle consistency in [11]), i.e., aligning the one-step transitions of the two domains. Additional properties have also been incorporated as auxiliary loss functions in learning the inter-domain mapping in the prior works, including domain cycle consistency [11, 20], effect cycle consistency [21], maximizing mutual information between states and embeddings [20], and alignment of target-domain rewards with the embeddings [20]. Moreover, as the state and action spaces are typically bounded sets and these methods directly map the data samples between the two domains, adversarial learning has been used to restrict the output range of the mapping functions [11, 15]. On the other hand, in [22], the state mapping function is found by Unsupervised Manifold Alignment [23]. Despite the above progress, the existing approaches all presume that the domains are sufficiently similar and do not have any performance guarantees (and hence can suffer from negative transfer in bad-case scenarios). By contrast, this paper proposes a robust CDRL method that can achieve transfer regardless of source-domain model quality or domain similarity with guarantees.

CDRL across domains with identical state and action representations. In CDRL, a variety of methods have been proposed for the case where source and target domains share the same state and action spaces but are subject to dynamics mismatch. (i) *Using the data samples from both source and target domains for policy learning*: One popular approach is to use the data from both domains for model updates [8, 9, 10]. For example, for compensating the discrepancy between domains in transition dynamics, [8] proposes to modify the reward function, which is learned by an auxiliary domain classifier that distinguishes between the source-domain and target-domain transitions. [9] handles the dynamics shift problem in offline RL by augmenting rewards in the source-domain dataset. [10] proposes to address dynamics mismatch by a value-guided data filtering scheme, which ensures selective sharing of the source-domain transitions based on the proximity of paired value targets. (ii) *Explicit domain similarity*: [7] proposes to selectively apply direct transfer of the source-domain policy to the target domain based on a learnable similarity metric, which is essentially the TD error of target domain trajectories with source Q function. Moreover, based on the policy invariant explicit shaping [24], [7] further uses the potential function as a bias term for selecting actions. (iii) *Using both Q-functions for the Q-learning updates*: Target Transfer Q-Learning [25] calculates the TD error by the source and target domains Q functions in order to select the TD target from the two Q functions. (iv) *Domain randomization*: To tackle sim-to-real transfer with dynamics mismatch, domain randomization [26, 6, 27, 28] and [28] collects data from multiple similar source domains with different configurations to learn a high-quality policy that can work robustly in a possibly unseen but similar target domain. Although many CDRL methods are applied when state and action representations are identical, they can't handle the negative transfer and lack the theoretical guarantees.

3 Preliminaries

In this section, we provide the problem formulation and basic building blocks of CDRL as well as the useful notation needed by subsequent sections. For a set $\mathcal{X}$, we let $\Delta(\mathcal{X})$ denote the set of probability distributions over $\mathcal{X}$. As in typical RL, we model each environment as an infinite-horizon discounted Markov decision process (MDP) denoted by $\mathcal{M} := (\mathcal{S}, \mathcal{A}, P, r, \gamma, \mu)$, where (i) $\mathcal{S}$ and $\mathcal{A}$ represent the state space and action space, (ii) $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ denotes the transition function, (iii) $r : \mathcal{S} \times \mathcal{A} \to [-R_{\max}, R_{\max}]$ is the reward function, (iv) $\gamma \in [0, 1)$ is the discounted factor, and (v) $\mu \in \Delta(\mathcal{S} \times \mathcal{A})$ denotes the initial state-action distribution. Notably, the use of an initial distribution over states and actions is a standard setting in the literature of natural policy gradient (NPG) [29, 30, 31, 32, 33]. Given any policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$, we use $\tau = (s_0, a_0, r_1, \cdots)$ to denote a (random) trajectory generated under π in $\mathcal{M}$, and the expected total discounted reward under π is defined as $V^{\pi}_{\mathcal{M}}(\mu) := \mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | \pi; s_0, a_0 \sim \mu]$. Moreover, as usual, we use $Q^{\pi}_{\mathcal{M}}(s, a)$ and $V^{\pi}_{\mathcal{M}}(s)$ to denote the Q function and value function of a policy π. We also define the state-action visitation distribution (also known as the occupancy measure in the MDP literature) of a policy π as $d^{\pi}(s, a) := (1 - \gamma)\big(\mu(s, a) + \sum_{t=1}^{\infty} \gamma^t \mathbb{P}(s_t = s, a_t = a; \pi)\big)$, for each (s, a).

Problem Formulation of Cross-Domain RL. In typical CDRL, the knowledge transfer involves two MDPs, namely the source-domain MDP $\mathcal{M}_{\text{src}} := (\mathcal{S}_{\text{src}}, \mathcal{A}_{\text{src}}, P_{\text{src}}, r_{\text{src}}, \gamma, \mu_{\text{src}})$ and the target-domain MDP $\mathcal{M}_{\text{tar}} := (S_{\text{tar}}, A_{\text{tar}}, P_{\text{tar}}, r_{\text{tar}}, \gamma, \mu_{\text{tar}})$[1]. Here we assume that the two MDPs share the

[1]Throughout this paper, we use the subscripts "src" and "tar" to represent the objects in the source domain and the target domain, respectively.

same discounted factor γ, which is rather mild. Moreover, the trajectories of the two domains are completely unpaired. Let Π_{tar} be the set of all stationary Markov policies for $\mathcal{M}_{\text{tar}}$. The goal of the RL agent is to learn a policy π^* in the target domain such that the expected total discounted reward is maximized, i.e., $\pi^* := \arg\max_{\pi\in\Pi_{\text{tar}}} V^{\pi}_{\mathcal{M}_{\text{tar}}}(\mu_{\text{tar}})$. To improve sample efficiency via knowledge transfer (compared to learning from scratch), in CDRL, the target-domain agent is granted access to $(\pi_{\text{src}}, Q_{\text{src}}, V_{\text{src}})$, which denotes a policy and the corresponding Q and value functions pre-trained in $\mathcal{M}_{src}$. Notably, we make no assumption on the quality of π_{src} (and hence π_{src} may not be optimal to $\mathcal{M}_{\text{src}}$), despite that π_{src} shall exhibit acceptable performance in practice.

In this paper, we focus on designing a robust CDRL algorithm in the sense that it effectively leverages a source-domain Q function Q_{src} for knowledge transfer to the target domain, regardless of the quality of Q_{src} and domain similarity.

Inter-Domain Mapping Functions. To address the discrepancy in state-action representations in CDRL, learning an inter-domain mapping function is one common building block of many CDRL algorithms. Specifically, there are a variety of ways to construct the mapping functions, such as handcrafted functions [16], encoders and decoders trained by cycle consistency [20] like cycle-GAN [34], neural networks trained by dynamics alignment of the MDPs [15]. Moreover, mapping functions have various candidate target spaces, such as a latent space, state or action spaces of the target domain (i.e., from $\mathcal{S}_{\text{src}}, \mathcal{A}_{\text{src}}$ to $\mathcal{S}_{\text{tar}}, \mathcal{A}_{\text{tar}}$), and state or action spaces of the source domain (i.e., from $\mathcal{S}_{\text{tar}}, \mathcal{A}_{\text{tar}}$ to $\mathcal{S}_{\text{src}}, \mathcal{A}_{\text{src}}$). For example, [15] proposed to learn two mapping functions $G_1 : \mathcal{S}_{\text{tar}} \to \mathcal{S}_{\text{src}}$ and $G_2 : \mathcal{A}_{\text{src}} \to \mathcal{A}_{\text{tar}}$ through dynamics alignment, which infers the unknown mapping between the unpaired trajectories of $\mathcal{M}_{\text{src}}$ and $\mathcal{M}_{\text{tar}}$ by aligning the one-step state transitions. Specifically, dynamics alignment can be implemented by minimizing the loss function defined as $L(G_1, G_2) = \mathbb{E}_{s_{\text{tar}}\sim\rho, s'_{\text{tar}}, s'_{\text{src}}}\big[\|s'_{\text{src}} - G_1(s'_{\text{tar}})\|_1\big]$, where s_{tar} is drawn from some target-domain state distribution ρ and $s'_{\text{tar}} \sim P_{\text{tar}}(\cdot|s_{\text{tar}}, G_2(a_{\text{src}}))$ with $a_{\text{src}} \sim \pi_{\text{src}}(\cdot|G_1(s_{\text{tar}}))$. However, this approach provides no performance guarantee as it can suffer from identification issue due to its unsupervised nature. By contrast, in this work, we propose to learn inter-domain state and action mapping functions in the form of $\phi : \mathcal{S}_{\text{tar}} \to \mathcal{S}_{\text{src}}$ and $\psi : \mathcal{A}_{\text{tar}} \to \mathcal{A}_{\text{src}}$ by leveraging a cross-domain Bellman-like loss function with guarantees, as described subsequently in Section 4.

Notation. Throughout this paper, for any real-valued function $h : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$, for any policy π, we use $h(s, \pi)$ and $\bar{h}(s, a; \pi)$ as the shorthand for $\mathbb{E}_{a\sim\pi(\cdot|s)}[h(s, a)]$ and $h(s, a) - \mathbb{E}_{a\sim\pi(\cdot|s)}[h(s, a)]$, respectively. For any real vector z and any $p \geq 1$, we use $\|z\|_p$ to denote the ℓ_p-norm of z.

4 Methodology

In this section, we first describe the prototypical framework of QAvatar in the tabular setting (i.e., $\mathcal{S}_{\text{tar}}$ and $\mathcal{A}_{\text{tar}}$ are finite) and establish convergence guarantees. We then extend this framework to a practical deep RL implementation.

4.1 The QAvatar Framework

The main idea of QAvatar is to utilize a weighted combination of a learned target-domain Q function and the given source-domain Q function for robust cross-domain knowledge transfer. In this way, QAvatar can enjoy improved sample efficiency in good-case scenarios (e.g., $\mathcal{M}_{\text{src}}$ and $\mathcal{M}_{\text{tar}}$ are similar) while avoiding potential negative transfer in other scenarios. Specifically, QAvatar consists of the following three major components:

- **Inter-domain mapping**: Under QAvatar , we propose to learn the inter-domain mappings $\phi : \mathcal{S}_{\text{tar}} \to \mathcal{S}_{\text{src}}$ and $\psi : \mathcal{A}_{\text{tar}} \to \mathcal{A}_{\text{src}}$ by minimizing a cross-domain Bellman-like loss function as

$$\mathcal{L}_{\text{CD}}(\phi, \psi; Q_{\text{src}}, \pi_{\text{tar}}, \mathcal{D}_{\text{tar}}) := \hat{\mathbb{E}}_{(s,a,r,s')\in\mathcal{D}_{\text{tar}}}\Big[\big|r+\gamma\mathbb{E}_{a'\sim\pi_{\text{tar}}}[Q_{\text{src}}(\phi(s'), \psi(a'))] - Q_{\text{src}}(\phi(s), \psi(a))\big|\Big], \tag{1}$$

  where Q_{src} is the pre-trained source-domain Q function and $\mathcal{D}_{\text{tar}} = \{(s, a, r, s')\}$ denotes a set of on-policy target-domain samples drawn under π_{tar}. Intuitively, the loss in (1) looks for a pair of mapping functions ϕ, ψ such that Q_{src} aligns as much with the target-domain transitions as possible. In the special case of $\mathcal{M}_{\text{src}} = \mathcal{M}_{\text{tar}}$ and ϕ, ψ being identity maps, (1) simply reduces to the standard loss function of temporal difference (TD) learning.

Algorithm 1 QAvatar

```
Require: Source-domain Q function $Q_{\text{src}}$, weight decay function $\alpha : \mathbb{N} \to [0,1]$, and $\eta \in (0, \frac{1-\gamma}{2}]$.
1: Initialize the state mapping function $\phi$, the action mapping function $\psi$, number of on-policy
   samples per iteration $N_{\text{tar}}$, and the target-domain policy $\pi^{(0)}$
2: for iteration $t = 1, \cdots, T$ do
3:     Sample $\mathcal{D}_{\text{tar}}^{(t)} = \{(s,a,r,s')\}$ of $N_{\text{tar}}^{(t)}$ on-policy samples using $\pi^{(t)}$ in the target domain
4:     Update $Q_{\text{tar}}$ by minimizing the TD loss in (2), i.e., $Q_{\text{tar}}^{(t)} \leftarrow \arg\min_{Q_{\text{tar}}} \mathcal{L}_{\text{TD}}(Q_{\text{tar}}; \pi^{(t)}, \mathcal{D}_{\text{tar}}^{(t)})$
5:     Update $\phi$ and $\psi$ by minimizing (1), i.e., $\phi^{(t)}, \psi^{(t)} \leftarrow \arg\min_{\phi,\psi} \mathcal{L}_{\text{CD}}(\phi, \psi; Q_{\text{src}}, \pi^{(t)}, \mathcal{D}_{\text{tar}}^{(t)})$.
6:     Update the target-domain policy by adapting NPG to CDRL as in (3).
7: end for
8: Return Target-domain policy $\pi_{\text{tar}}^{(T)} \sim \text{Uniform}(\{\pi^{(1)}, \cdots, \pi^{(T)}\})$.
```

- **Target-domain Q function**: To implement the idea of a hybrid Q function, QAvatar maintains a target-domain Q function Q_{tar}, which is essentially a critic of the current target-domain policy. Specifically, in each iteration t, Q_{tar} is obtained by a policy evaluation step via minimizing the standard TD loss for least-squares policy evaluation (LSPE) [35, 36, 37][2], i.e.,

$$\mathcal{L}_{\text{TD}}(Q_{\text{tar}}; \pi_{\text{tar}}, \mathcal{D}_{\text{tar}}) := \hat{\mathbb{E}}_{(s,a,r,s')\in\mathcal{D}_{\text{tar}}}\Big[\big|r + \gamma \mathbb{E}_{a'\sim\pi_{\text{tar}}}[Q_{\text{tar}}(s',a')] - Q_{\text{tar}}(s,a)\big|^2\Big], \qquad (2)$$

  where $\mathcal{D}_{\text{tar}} = \{(s,a,r,s')\}$ denotes on-policy target-domain samples.

- **NPG-like policy update with a weighted combination of Q functions**: The core idea of QAvatar is to leverage both Q_{src} and Q_{tar} to determine policy updates. In the tabular setting, inspired by [33] in the offline-to-online RL literature, we adapt the classic natural policy gradient (NPG) update [38], which takes an exponential-weight form on the Q function in the policy space (cf. [29, 39]), to the CDRL setting. In each iteration t,

$$\pi^{(t+1)}(a|s) \propto \pi^{(t)}(a|s) \exp\Big(\eta \cdot \big((1-\alpha(t))Q_{\text{tar}}^{(t)}(s,a) + \alpha(t) Q_{\text{src}}(\phi^{(t)}(s), \psi^{(t)}(a))\big)\Big), \qquad (3)$$

  where $\alpha : \mathbb{N} \to [0,1]$ is the weight decay function to be configured. Intuitively, $\alpha(t)$ shall be close to one for small t to achieve knowledge transfer from Q_{src} and gradually diminish with t to escape from potential negative transfer.

The pseudo code of QAvatar is provided in Algorithm 1.

Remark 1. In Line 8 of Algorithm 1, QAvatar outputs the final policy by choosing uniformly at random from the set of all intermediate polices. This is a standard procedure in the optimization literature to connect the average sub-optimality with the performance of output policy. In the experiments, we show that using the last-iterate policy is sufficient and performs well.

4.2 Performance Guarantees of QAvatar

In this section, we formally present the theoretical guarantee of QAvatar and thereby describe how to choose the proper decay parameter $\alpha(\cdot)$. Before stating the theorem, we first describe a useful definition on the coverage in terms of state-action distribution [33].

Definition 1 (Coverage). *Given a comparator policy $\pi^\dagger$ in $\mathcal{M}_{tar}$, we say that $\pi^\dagger$ has coverage $C_{\pi^\dagger}$ if for any policy $\pi \in \Pi_{tar}$, we have $\|d^{\pi^\dagger}/d^{\pi}\|_\infty \le C_{\pi^\dagger}$.*

Notably, one can verify that $C_{\pi^\dagger}$ is finite if $\|d^{\pi^\dagger}/\mu_{\text{tar}}\|_\infty$ is finite (given that $\|\mu_{\text{tar}}/d^{\pi}\|_\infty \le 1/(1-\gamma)$ for all π, by the definition of d^π), and this can be satisfied under an exploratory initial distribution with $\mu_{\text{tar}}(s,a) > 0$ for all (s,a), which is one standard assumption in the NPG literature [29, 30, 31, 32, 33]. Intuitively, the coverage is needed to enable direct comparison of the Bellman error between policies.

Assumption 1. *The initial distribution is exploratory, i.e., $\mu_{tar}(s,a) > 0$, for all s, a.*

[2]These works on LSPE are shown under linear function approximation, which includes the tabular setting as a special case by using one-hot feature vectors.

Definition 2 (TD Error). *For each state-action pair (s, a) and $t \in \mathbb{N}$, the TD error $\epsilon_{td}^{(t)}(s, a)$ is defined as $\epsilon_{td}^{(t)}(s, a) := \left|Q_{tar}^{(t)}(s, a) - r_{tar}(s, a) - \gamma\, \mathbb{E}_{s' \sim P_{tar}(\cdot|s,a), a' \sim \pi^{(t)}(\cdot|s')}[Q_{tar}^{(t)}(s', a')]\right|$.*

Definition 3 (Cross-Domain Bellman Error). *Given a source-domain Q_{src}, for each state-action pair (s, a) and $t \in \mathbb{N}$, the cross-domain Bellman error $\epsilon_{src,be}^{(t)}(s, a; Q_{src})$ is defined as $\epsilon_{src,be}^{(t)}(s, a; Q_{src}) := \left|Q_{src}(\phi^{(t)}(s), \psi^{(t)}(a)) - r_{tar}(s, a) - \gamma\, \mathbb{E}_{s' \sim P_{tar}(\cdot|s,a), a' \sim \pi^{(t)}(\cdot|s')}[Q_{src}(\phi^{(t)}(s'), \psi^{(t)}(a'))]\right|$.*

Below we use $\|\epsilon_{\text{src, be}}^{(t)}(Q_{\text{src}})\|_\infty$ and $\|\epsilon_{\text{td}}^{(t)}\|_\infty$ as shorthand for $\|\epsilon_{\text{src, be}}^{(t)}(\cdot, \cdot; Q_{\text{src}})\|_\infty$ and $\|\epsilon_{\text{td}}^{(t)}(\cdot, \cdot)\|_\infty$, and we use $\mu_{\text{tar,min}}$ as a shorthand for $\min_{s,a} \mu_{\text{tar}}(s, a)$. We are ready to present the main theoretical result, and the detailed proof is provided in Appendix B.

Theorem 1. *(Average Sub-Optimality) Under the QAvatar in Algorithm 1 and Assumption 1, the average sub-optimality over T iterations can be upper bounded as*

$$\frac{1}{T}\sum_{t=1}^{T}\Big(V^{\pi^*}(\mu_{tar}) - V^{\pi^{(t)}}(\mu_{tar})\Big)$$
$$\le \underbrace{\frac{2}{1-\gamma}\sqrt{\frac{\log(A)}{T}}}_{(a)} + \underbrace{\frac{C_0}{T}\sum_{t=1}^{T}\alpha(t)\|\epsilon_{src,\,be}^{(t)}(Q_{src})\|_\infty}_{(b)} + \underbrace{\frac{C_0}{T}\sum_{t=1}^{T}(1-\alpha(t))\|\epsilon_{td}^{(t)}\|_\infty}_{(c)}, \tag{4}$$

where $C_0 := 2\sqrt{C_{\pi^*}}/((1-\gamma)^2\mu_{tar,\,min})$.

Notably, in (4), the term (a) reflects the learning progress of NPG, the (b) reflects the effect of cross-domain transfer, and (c) indicates the error of policy evaluation for the target-domain policy. The term (c) reflects the sample complexity of the standard least-squares TD-based policy evaluation [35, 40, 36, 37] and can be made small with sufficient samples (i.e., sufficiently large $N_{\text{tar}}^{(t)}$).

Remark 2. Note that the proof of Theorem 1 bears some high-level resemblance with [33] as they also use NPG in their hybrid actor-critic (HAC) algorithm. That said, QAvatar is fundamentally different from HAC in two aspects: (i) QAvatar addresses cross-domain transfer while HAC focuses on using offline and online data from the same domain. (ii) QAvatar utilizes the hybrid Q function while HAC applies a hybrid squared error regression loss (i.e., the sum of TD errors calculated from both offline and online data).

Corollary 1. *By choosing $\alpha(t) = t^{-\beta}$ with $\beta \ge 1/2$, the policy returned by Algorithm 1 satisfies*

$$\mathbb{E}[V^{\pi^*}(\mu_{tar}) - V^{\pi_{tar}^{(T)}}(\mu_{tar})] \le \frac{2}{1-\gamma}\sqrt{\frac{\log(A)}{T}} + C_1 \max\Big\{\frac{2}{T^\beta}, \frac{2\ln(T)}{T}\Big\} + \frac{C_0}{T}\sum_{t=1}^{T}\|\epsilon_{td}^{(t)}\|_\infty,$$

where $C_1 := 2C_0 R_{\max}/(1-\gamma)$.

Based on the above, we can observe multiple features of the proposed algorithm:

- **Domain similarity is reflected by the cross-domain Bellman error**: Theorem 1 naturally reflects the domain similarity through the term (b) in (4), which involves cross-domain Bellman error. In fact, in the special case where $\mathcal{M}_{\text{tar}} = \mathcal{M}_{\text{src}}$ and Q_{src} is near-optimal, $\epsilon_{\text{src,be}}$ can be already close to zero in the first iteration.
- **QAvatar supports effective cross-domain knowledge transfer while avoiding potential negative transfer**: When effective knowledge transfer is possible (i.e., the cross-domain Bellman error $\epsilon_{\text{src,be}}$ is small), the term (b) in the upper bound (4) can be small even for small t (i.e., early training stage). On the other hand, if the two domains are fairly different such that negative transfer can likely happen, one can escape from this with the help of the weight decay $\alpha(t)$.
- **Weight decay function offers a flexible trade-off between transfer learning and target-domain RL**: As shown in Corollary 1, choosing $\alpha(t) = t^{-\beta}$ with any $\beta \ge 1/2$ is sufficient to bound the term (b) in (4). This resulting wide range of $\alpha(t)$ can serve as a flexible control of the trade-off between cross-domain transfer and the learning of the target domain *per se*.

4.3 Practical Implementation of QAvatar

We extend the QAvatar framework in Algorithm 1 to a practical deep RL implementation for continuous state and action spaces by applying the following design choices. The pseudo code is provided in Algorithm 2 in Appendix.

- **Learning the target-domain policy and the Q function.** To go beyond the tabular setting and handle continuous state and action spaces, we extend QAvatar by first connecting NPG with soft policy iteration (SPI) [41]. In the entropy-regularized RL setting, SPI has been shown to be a special case of NPG [42]. Based on this connection, we choose to integrate QAvatar with soft actor-critic (SAC) [41], i.e., updating the target-domain critic Q_{tar} by the critic loss of SAC and updating the target-domain policy $\pi^{(t)}$ by the SAC policy loss function with the weighted combination of Q_{tar} and Q_{src} of QAvatar . Regarding the weight decay function $\alpha(t)$, we follow the theoretical result and set $\alpha(t) = t^{-\beta}$ with $\beta \geq 1/2$.
- **Learning the inter-domain mapping functions with an augmented flow model.** Similar to the tabular setting, we learn the inter-domain mappings by minimizing the cross-domain Bellman loss. Notably, in practical RL problems, the state and action spaces are mostly bounded sets. As a result, we need to ensure that the outputs of the inter-domain mappings $\phi : \mathcal{S}_{\text{tar}} \rightarrow \mathcal{S}_{\text{src}}$ and $\psi : \mathcal{A}_{\text{tar}} \rightarrow \mathcal{A}_{\text{src}}$ fall within the feasible regions. As mentioned in Section 2, adversarial learning is widely adopted to solve this practical problem in the existing literature [19, 11, 15, 21]. However, we observe that adversarial learning could suffer from unstable training process in practice. Therefore, we use the method proposed by [43] and train a normalizing flow model that can map the outputs of the mapping functions to the feasible regions.

5 Experiments

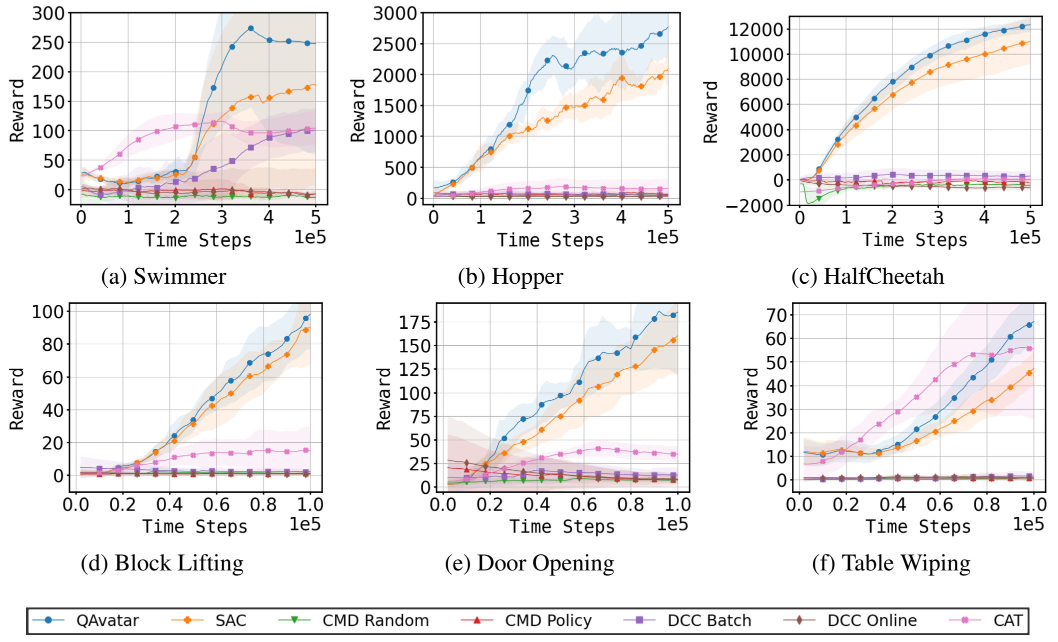
In this section, we show that QAvatar achieves effective cross-domain transfer amd improves the sample efficiency on various RL benchmark tasks. Moreover, we demonstrate that QAvatar can still perform well even with the existence of negative transfer between the source and target domains. Unless stated otherwise, all the results reported in this section are averaged over 5 random seeds.

5.1 Experimental Settings

Table 1: Dimensionalities of the source and target domains (“Src" and “Tar" represent the source domain and the target domain.

Environment	State		Action	
	Src	Tar	Src	Tar
Swimmer	8	10	2	3
Hopper	11	13	3	4
HalfCheetah	17	23	6	9
Ant	27	31	8	10
IP / Modified IDP	4	11	1	1
Block Lifting	42	47	8	7
Door Opening	46	51	8	7
Table Wiping	37	34	7	6

Baselines. We compare the performance of QAvatar with various recent CDRL benchmark algorithms under distinct state-action representations, including Dynamics Cycle-Consistency (DCC) [11], Cross-Morphology-Domain Policy Adaptation (CMD) [15], and Cross-domain Adaptive Transfer (CAT) [20]. For a fair comparison, all these benchmark methods and QAvatar use the same set of source-domain models (i.e., the policy and the corresponding Q-networks), which are pre-trained by using SAC in the source domain. Moreover, as the original DCC is implemented in the batch setting (i.e., a fixed number of trajectories are collected for learning the inter-domain mappings), we further consider a variant of DCC that can learn in the online setting (i.e., learning while collecting new trajectories iteratively). Accordingly, we call these two versions “DCC-Batch" and “DCC-Online" in our experiments, respectively. We also consider two versions of CMD as our baselines, namely “CMD-Random" and “CMD-Policy". Specifically, CMD-Random represents that CMD collects target-domain data with a random policy continuously, as in the default setting of the original paper [15]. However, CMD-Random can suffer if the collected trajectories are mostly of low return due to the random exploration. Accordingly, we also consider CMD-Policy, which collects target-domain data with the target-domain policy induced by the source-domain pre-trained policy and the current inter-domain mapping functions. For CAT and DCC, we use the implementation provided by the original papers



(a) Swimmer (b) Hopper (c) HalfCheetah

(d) Block Lifting (e) Door Opening (f) Table Wiping

Figure 1: The training curves of QAvatar and the benchmark methods: (a)-(c): Locomotion tasks in MuJoCo ; (d)-(f): Robot arm manipulation tasks in Robosuite. Note that the shading represents one standard deviation above and below the mean.

with no change. Moreover, we reproduce CMD by referring to the source code of DCC as they are similar and there is no CMD implementation available. Moreover, to demonstrate the sample efficiency of QAvatar , we also compare it with the standard SAC algorithm [41] that learns from scratch in the target domain and hence can serve as a baseline. The hyperparameters are provided in Appendix E.

Evaluation Environments. We evaluate QAvatar on two types of RL benchmark tasks: (i) *Locomotion*: We use the original MuJoCo environments, including Swimmer-v3, Hopper-v3, HalfCheetah-v3 and Ant-v3, as the source domains and modify them for the target domains [11], [10]. The details about the morphology is in Appendix E. Additionally, we use Inverted Pendulum-v2 as the source domain and use Inverted Double Pendulum-v2 as the target domain for evaluation in the scenario of negative transfer. (ii) *Robot arm manipulation*: We use the environments provided by Robosuite, a popular package for robot learning released by [44]. We evaluate our algorithm on three tasks, including block lifting, door opening and table wiping. For each task, we use the Panda robot arm as the source domain and set the UR5e robot arm as the target domain. Table 1 provides the dimensionalities of the state and action spaces in all the tasks.

5.2 Experimental Results

Does QAvatar improve data efficiency? As shown by the training curves in Figure 1, we observe that QAvatar achieves better data efficiency than SAC throughout the training process in all the MuJoCo and Robosuite tasks, despite that these tasks have rather different dimensionalities as shown in Table 1. CAT achieves moderate performance in Swimmer and Table Wiping but does not learn effectively in the other tasks. These appear reasonable as CAT has no performance guarantees and can suffer if the source and target are rather dissimilar, despite that CAT applies policy gradient with target-domain rewards to align the inter-domain mapping with the target domain. Regarding CMD, we observe that CMD cannot obtain good returns in most of the tasks, and these results are consistent with those in the original CMD paper. Moreover, in some environments like Ant, CMD appears very unstable due to its adversarial learning module for restricting the output of their mapping functions. DCC performs well only in Swimmer but rather poorly in all the other tasks (including HalfCheetah). This trend is similar to that in the original paper [11]. The rewards obtained by DCC in our experiments are slightly lower than the rewards shown in [11] despite that we try our best to reproduce their results. That said, the performance of QAvatar is still better than that of DCC in [11].

We conjecture that the undesired performance of CMD and DCC results from that they learn in an unsupervised manner and hence does not take target-domain rewards into account. Additionally, when we consider the time to threshold metric, our algorithm requires 268k fewer steps to achieve the threshold than SAC does in the best case. When we consider the asymptotic performance metric, our algorithm can obtain higher final rewards than SAC. The results of these two metrics are shown in the Appendix D.1 and D.2.

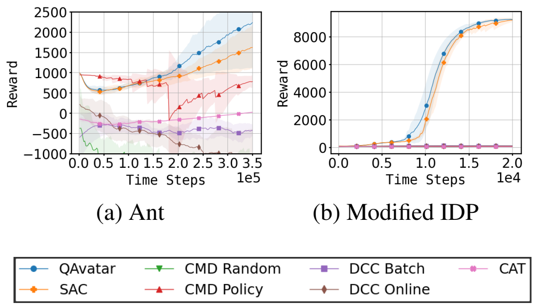


Figure 2: The training curves of QAvatar and the benchmark methods in the negative transfer cases.

Does QAvatar still perform robustly well when negative transfer is likely to happen? We construct negative transfer scenarios as follows: (i) We add an extra right-middle leg to the original Ant. Due to the fact that the original ant has no middle leg, the inter-domain action mapping cannot find a reasonable mapping to make this extra leg take a proper action. (ii) We use Inverted Pendulum (IP) as the source domain and Inverted Double Pendulum (IDP) as the target domain. Then, we modify the configuration of IDP by swapping the meaning of the actions "left" and "right" (termed "Modified IDP"). As a result, negative transfer shall easily occur if we deactivate the inter-domain action mapping ψ of QAvatar (such that QAvatar cannot learn by simply mapping to "left" in IDP to "right" in IP. As shown in Figure 2, we discover that QAvatar performs better than all the baselines (and even better than SAC) despite the negative transfer scenarios. This confirms the robustness of QAvatar offered by the use of hybrid Q function.

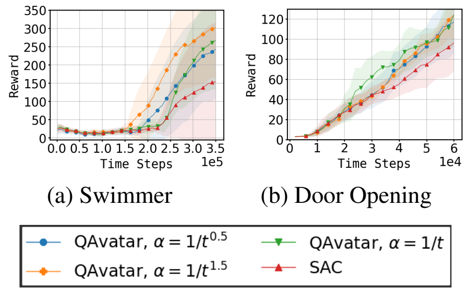


Figure 3: The training curves of QAvatar under different decay functions α.

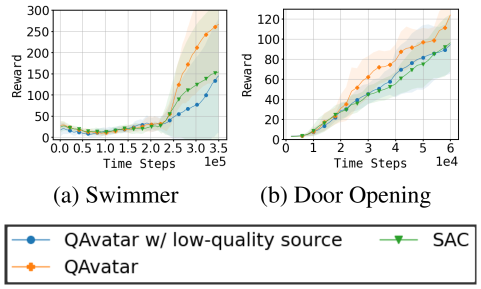


Figure 4: The training curves of QAvatar with a high-quality and a low-quality source model.

Is QAvatar sensitive to the decay function? We evaluate QAvatar with $\alpha(t)$ as $1/\sqrt{t}$, $1/t$, and $1/t^{1.5}$. As shown in Figure 3, the training curves of QAvatar are better than SAC and appear consistently favorable under all these choices of $\alpha(t)$.

Does QAvatar still perform robustly with a low-quality source-domain model? We further run QAvatar with low-quality source-domain Q networks, which are pre-trained only for 10k and 5k steps in Swimmer and Door Opening, respectively. As shown by Figure 4, we find that despite QAvatar is affected by the low-quality source model initially, it can quickly catch up and achieve total reward comparable to SAC. This appears consistent with the theoretical result in Theorem 1.

6 Concluding Remarks and Limitations

In this paper, we present QAvatar, the first CDRL method that can handle distinct state-action representations between domains with performance guarantees. Based on the idea of combining the source-domain and target-domain Q functions, QAvatar achieves robust knowledge transfer and tackles the negative transfer issue. Through extensive experiments, we show that QAvatar indeed serves as a promising and generic solution to cross-domain transfer in RL. One limitation of this work is that we follow the standard CDRL formulation and consider only one source domain and one target domain. Extending the idea of QAvatar to achieve knowledge transfer from multiple source and target domains is a promising future research direction.

References

[1] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533, 2015.

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

[3] Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3389–3396, 2017.

[4] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673, 2018.

[5] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems*, 35:27730–27744, 2022.

[6] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3803–3810, 2018.

[7] Ram Ananth Sreenivasan, Hyun-Rok Lee, Yeonjeong Jeong, Jongseong Jang, Dongsub Shim, and Chi-Guhn Lee. A learnable similarity metric for transfer learning with dynamics mismatch. In *PRL Workshop Series –Bridging the Gap Between AI Planning and Reinforcement Learning*, 2023.

[8] Benjamin Eysenbach, Shreyas Chaudhari, Swapnil Asawa, Sergey Levine, and Ruslan Salakhutdinov. Off-dynamics reinforcement learning: Training for transfer with domain classifiers. In *International Conference on Learning Representations*, 2020.

[9] Jinxin Liu, Zhang Hongyin, and Donglin Wang. DARA: Dynamics-Aware Reward Augmentation in Offline Reinforcement Learning. In *International Conference on Learning Representations*, 2022.

[10] Kang Xu, Chenjia Bai, Xiaoteng Ma, Dong Wang, Bin Zhao, Zhen Wang, Xuelong Li, and Wei Li. Cross-domain policy adaptation via value-guided data filtering. *Advances in Neural Information Processing Systems*, 36, 2023.

[11] Qiang Zhang, Tete Xiao, Alexei A Efros, Lerrel Pinto, and Xiaolong Wang. Learning cross-domain correspondence for control with dynamics cycle-consistency. In *International Conference on Learning Representations*, 2021.

[12] Karl Weiss, Taghi M Khoshgoftaar, and DingDing Wang. A survey of transfer learning. *Journal of Big data*, 3:1–40, 2016.

[13] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2009.

[14] Heng You, Tianpei Yang, Yan Zheng, Jianye Hao, and Matthew E Taylor. Cross-domain adaptive transfer reinforcement learning based on state-action correspondence. In *Uncertainty in Artificial Intelligence*, pages 2299–2309, 2022.

[15] Haiyuan Gui, Shanchen Pang, Shihang Yu, Sibo Qiao, Yufeng Qi, Xiao He, Min Wang, and Xue Zhai. Cross-domain policy adaptation with dynamics alignment. *Neural Networks*, 167:104–117, 2023.

[16] Haitham Bou Ammar and Matthew E Taylor. Reinforcement learning transfer via common subspaces. In *Adaptive and Learning Agents: International Workshop, ALA 2011, Held at AAMAS*, pages 21–36. Springer, 2012.

[17] Haitham B Ammar, Karl Tuyls, Matthew E Taylor, Kurt Driessens, and Gerhard Weiss. Reinforcement learning transfer via sparse coding. In *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*, volume 1, pages 383–390, 2012.

[18] Abhishek Gupta, Coline Devin, YuXuan Liu, Pieter Abbeel, and Sergey Levine. Learning invariant feature spaces to transfer skills with reinforcement learning. In *International Conference on Learning Representations*, 2017.

[19] Matthew E Taylor, Gregory Kuhlmann, and Peter Stone. Autonomous transfer for reinforcement learning. In *Proceedings of International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS)*, pages 283–290, 2008.

[20] You Heng, Tianpei Yang, Yan Zheng, Jianye Hao, and Matthew E. Taylor. Cross-domain adaptive transfer reinforcement learning based on state-action correspondence. In *Conference on Uncertainty in Artificial Intelligence*, 2022.

[21] Ruiqi Zhu, Tianhong Dai, and Oya Celiktutan. Cross domain policy transfer with effect cycle-consistency. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[22] Haitham Bou Ammar, Eric Eaton, Paul Ruvolo, and Matthew Taylor. Unsupervised cross-domain transfer in policy gradient reinforcement learning via manifold alignment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015.

[23] Chang Wang and Sridhar Mahadevan. Manifold alignment without correspondence. In *IJCAI*, volume 2, page 3, 2009.

[24] Paniz Behboudian, Yash Satsangi, Matthew E Taylor, Anna Harutyunyan, and Michael Bowling. Policy invariant explicit shaping: An efficient alternative to reward shaping. *Neural Computing and Applications*, pages 1–14, 2022.

[25] Yue Wang, Yuting Liu, Wei Chen, Zhi-Ming Ma, and Tie-Yan Liu. Target transfer Q-learning and its convergence analysis. *Neurocomputing*, 392:11–22, 2020.

[26] Aravind Rajeswaran, Sarvjeet Ghotra, Balaraman Ravindran, and Sergey Levine. Epopt: Learning robust neural network policies using model ensembles. In *International Conference on Learning Representations*, 2016.

[27] Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 8973–8979, 2019.

[28] Yuqing Du, Olivia Watkins, Trevor Darrell, Pieter Abbeel, and Deepak Pathak. Auto-tuned sim-to-real transfer. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1290–1296, 2021.

[29] Alekh Agarwal, Sham M Kakade, Jason D Lee, and Gaurav Mahajan. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *Journal of Machine Learning Research*, 22(98):1–76, 2021.

[30] Dongsheng Ding, Kaiqing Zhang, Tamer Basar, and Mihailo Jovanovic. Natural policy gradient primal-dual method for constrained markov decision processes. *Advances in Neural Information Processing Systems*, 33:8378–8390, 2020.

[31] Rui Yuan, Simon S Du, Robert M Gower, Alessandro Lazaric, and Lin Xiao. Linear convergence of natural policy gradient methods with log-linear policies. In *International Conference on Learning Representations*, 2022.

[32] Alekh Agarwal, Mikael Henaff, Sham Kakade, and Wen Sun. Pc-pg: Policy cover directed exploration for provable policy gradient learning. *Advances in Neural Information Processing Systems*, 33:13399–13412, 2020.

[33] Yifei Zhou, Ayush Sekhari, Yuda Song, and Wen Sun. Offline data enhanced on-policy policy gradient with provable guarantees. In *International Conference on Learning Representations*, 2024.

[34] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2223–2232, 2017.

[35] Michail G Lagoudakis and Ronald Parr. Model-free least-squares policy iteration. *Advances in Neural Information Processing Systems*, 14, 2001.

[36] Huizhen Yu and Dimitri P Bertsekas. Convergence results for some temporal difference methods based on least squares. *IEEE Transactions on Automatic Control*, 54(7):1515–1531, 2009.

[37] Alessandro Lazaric, Mohammad Ghavamzadeh, and Rémi Munos. Finite-sample analysis of least-squares policy iteration. *Journal of Machine Learning Research*, 13:3041–3074, 2012.

[38] Sham M Kakade. A natural policy gradient. *Advances in Neural Information Processing Systems*, 14, 2001.

[39] Lin Xiao. On the convergence rates of policy gradient methods. *Journal of Machine Learning Research*, 23(282):1–36, 2022.

[40] Michail G Lagoudakis, Ronald Parr, and Michael L Littman. Least-squares methods in reinforcement learning for control. In *Methods and Applications of Artificial Intelligence: Second Hellenic Conference on AI*, pages 249–260. Springer, 2002.

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

[42] Shicong Cen, Chen Cheng, Yuxin Chen, Yuting Wei, and Yuejie Chi. Fast global convergence of natural policy gradient methods with entropy regularization. *Operations Research*, 70(4):2563–2578, 2022.

[43] Janaka Brahmanage, Jiajing Ling, and Akshat Kumar. FlowPG: Action-constrained Policy Gradient with Normalizing Flows. *Advances in Neural Information Processing Systems*, 36, 2024.

[44] Yuke Zhu, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Abhishek Joshi, Soroush Nasiriany, and Yifeng Zhu. Robosuite: A modular simulation framework and benchmark for robot learning. *arXiv preprint arXiv:2009.12293*, 2020.

[45] Sham Kakade and John Langford. Approximately optimal approximate reinforcement learning. In *International Conference on Machine Learning*, pages 267–274, 2002.

[46] Alekh Agarwal, Nan Jiang, Sham M Kakade, and Wen Sun. Reinforcement learning: Theory and algorithms. *CS Dept., UW Seattle, Seattle, WA, USA, Tech. Rep*, 32:96, 2019.

[47] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

A Supporting Lemmas

Lemma 1 (Performance difference lemma)**.** *For any two policies π and π', for any state s, we have*

$$V^{\pi'}(\mu) - V^{\pi}(\mu) = \frac{1}{1-\gamma}\mathbb{E}_{s,a\sim d^{\pi'}}[A^{\pi}(s,a)],$$

where $A^{\pi}(s,a) := Q^{\pi}(s,a) - V^{\pi}(s)$ is the advantage function.

Proof. This can be directly obtained from Lemma 6.1 in [45]. □

Lemma 2 ([33], Lemma 6)**.** *Suppose $f^{(t)}$ and $\pi^{(t)}$ denote the value function and the policies at iteration t. Then, for any $\eta \leq \frac{1-\gamma}{2}$ and policy π^*, we have*

$$\sum_t \mathbb{E}_{s,a\sim d^{\pi^*}}[f^{(t)}(s,a) - f^{(t)}(s,\pi^{(t)}(s))] \leq \frac{2}{1-\gamma}\sqrt{\log(A)T}$$

Lemma 3 ([46], Chapter 4)**.** *Let $\tau = (s_0, a_0, s_1, a_1, \cdots)$ denote the (random) trajectory generated under a policy π in an infinite-horizon MDP $\mathcal{M}$. For any function $f : \mathcal{S}\times\mathcal{A}\to\mathbb{R}$, we have*

$$\mathbb{E}_{\tau}\left[\sum_{t=0}^{\infty}\gamma^t f(s_t,a_t)\right] = \frac{1}{1-\gamma}\,\mathbb{E}_{(s,a)\sim d^{\pi}}\,\big[f(s,a)\big]. \tag{5}$$

Lemma 4 (Importance Ratio)**.** *Given a fixed policy π and a fixed state-action pair (s,a), let $p_k(s,a)$ denote the probability of reaching (s,a) under an initial distribution d^{π} and policy π after k time steps. Then, for any $k\in\mathbb{N}$, we have*

$$\frac{p_k(s,a)}{d^{\pi}(s,a)} \leq \frac{1}{(1-\gamma)\mu(s,a)}. \tag{6}$$

Proof. To begin with, recall the definition of d^{π} as

$$d^{\pi}(s,a) := (1-\gamma)\Big(\mu(s,a) + \sum_{t=1}^{\infty}\gamma^t P(s_t = s, a_t = a;\pi)\Big) \equiv \sum_{t=0}^{\infty}\gamma^t P(s_t = s, a_t = a;\pi). \tag{7}$$

Let $s_{\text{next},k}$ and $a_{\text{next},k}$ denote the state and action after k time steps. Then, we can write down $p_k(s,a)$:

$$\begin{aligned}
p_k(s,a) &= \sum_{(s_0,a_0)}\mathbb{P}(s_{\text{next},k} = s, a_{\text{next},k} = a|s_0,a_0;\pi)d^{\pi}(s_0,a_0) && (8)\\
&= \sum_{(s_0,a_0)}\mathbb{P}(s_{\text{next},k} = s, a_{\text{next},k} = a|s_0,a_0;\pi)\cdot(1-\gamma)\cdot\sum_{t=0}^{\infty}\gamma^t\,\mathbb{P}(s_t = s_0, a_t = a_0;\pi) && (9)\\
&= (1-\gamma)\cdot\sum_{t=0}^{\infty}\gamma^t\sum_{s_0,a_0}\mathbb{P}(s_{\text{next},k} = s, a_{\text{next},k} = a|s_0,a_0;\pi)\cdot\mathbb{P}(s_t = s_0, a_t = a_0;\pi) && (10)\\
&= (1-\gamma)\sum_{t=0}^{\infty}\gamma^t\,\mathbb{P}(s_{t+k} = s, a_{t+k} = a;\pi) && (11)
\end{aligned}$$

Then, we have

$$\begin{aligned}
\frac{p_k(s,a)}{d^{\pi}(s,a)} &= \frac{(1-\gamma)\sum_{t=0}^{\infty}\gamma^t\,\mathbb{P}(s_{t+k} = s; a_{t+k} = a;\pi)}{(1-\gamma)\sum_{t=0}^{\infty}\gamma^t\,\mathbb{P}(s_t = s, a_t = a;\pi)} && (12)\\
&= \frac{\sum_{t=0}^{\infty}\gamma^t\,\mathbb{P}(s_{t+k} = s, a_{t+k} = a;\pi)}{\sum_{t=0}^{\infty}\gamma^t\,\mathbb{P}(s_t = s, a_t = a;\pi)} && (13)\\
&\leq \frac{\sum_{t=0}^{\infty}\gamma^t}{\sum_{t=0}^{\infty}\gamma^t\,\mathbb{P}(s_t = s;\pi)} && (14)\\
&= \frac{1}{1-\gamma}\cdot\frac{1}{\sum_{t=0}^{\infty}\gamma^t\,\mathbb{P}(s_t = s;\pi)} && (15)
\end{aligned}$$

where (14) holds by $\mathbb{P}(s_{t+k} = s, a_{t+k} = a; \pi) \le 1$ and (15) holds by taking the sum of an infinite geometric sequence. By $\sum_{t=0}^{\infty} \gamma^t \, \mathbb{P}(s_t = s, a_t = a; \pi) = \mu_{\text{tar}}(s) + \sum_{t=1}^{\infty} \gamma^t \, \mathbb{P}(s_t = s, a_t = a; \pi)$, we have

$$\frac{1}{1-\gamma} \cdot \frac{1}{\sum_{t=0}^{\infty} \gamma^t \, \mathbb{P}(s_t = s, a_t = a; \pi)} = \frac{1}{1-\gamma} \cdot \frac{1}{\mu(s,a) + \sum_{t=1}^{\infty} \gamma^t \, \mathbb{P}(s_t = s, a_t = a; \pi)} \tag{16}$$

$$\le \frac{1}{(1-\gamma)\mu(s,a)} \tag{17}$$

where (17) holds by $\sum_{t=1}^{\infty} \gamma^t \, \mathbb{P}(s_t = s, a_t = a; \pi) \ge 0$. □

B Proof of Theorem 1

Recall that for any policy π, we use d^{π} to denote the discounted state-action visitation distribution under policy π in the target domain.

Lemma 5. *Under Algorithm 1, for any $t \in \mathbb{N}$, we have*

$$\begin{aligned}&\mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \left[\left(\bar{f}^t(s,a) - A^{\pi^t}(s,a) \right)^2 \right] \\ &\le \frac{4}{(1-\gamma)^2 \mu_{\textit{tar, min}}^2} \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \left[\left((1-\alpha(t))\epsilon_{\textit{td}}^{(t)}(s,a) + \alpha(t)\epsilon_{\textit{src,be}}^{(t)}(s,a;Q_{\textit{src}}) \right)^2 \right]\end{aligned} \tag{18}$$

Proof. Recall the definitions that $\bar{f}^{(t)}(s,a) := f^{(t)}(s,a) - f^{(t)}(s,\pi^{(t)}(s))$ and $A^{\pi^{(t)}}(s,a) := Q^{\pi^{(t)}}(s,a) - Q^{\pi^{(t)}}(s,\pi^{(t)}(s))$. Then, we have

$$\mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \left[\left(\bar{f}^{(t)}(s,a) - A^{\pi^{(t)}}(s,a) \right)^2 \right] \tag{19}$$

$$= \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \left[\left(f^{(t)}(s,a) - f^{(t)}(s,\pi^{(t)}(s)) - Q^{\pi^{(t)}}(s,a) + Q^{\pi^{(t)}}(s,\pi^{(t)}(s)) \right)^2 \right] \tag{20}$$

$$\le \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \left[2\left(f^{(t)}(s,a) - Q^{\pi^{(t)}}(s,a) \right)^2 + 2\left(Q^{\pi^{(t)}}(s,\pi^{(t)}(s)) - f^{(t)}(s,\pi^{(t)}(s)) \right)^2 \right] \tag{21}$$

where (21) holds by the fact that $(x+y)^2 \le 2x^2 + 2y^2$ for any $x, y \in \mathbb{R}$. Then, by linearity of expectation, we obtain

$$\mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \left[2\left(f^{(t)}(s,a) - Q^{\pi^{(t)}}(s,a) \right)^2 + 2\left(Q^{\pi^{(t)}}(s,\pi^{(t)}(s)) - f^{(t)}(s,\pi^{(t)}(s)) \right)^2 \right] \tag{22}$$

$$= \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \left[2\left(f^{(t)}(s,a) - Q^{\pi^{(t)}}(s,a) \right)^2 \right] + \mathbb{E}_{s\sim d^{\pi^{(t)}}} \left[2\left(Q^{\pi^{(t)}}(s,\pi^{(t)}(s)) - f^{(t)}(s,\pi^{(t)}(s)) \right)^2 \right] \tag{23}$$

$$= \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \left[2\left(f^{(t)}(s,a) - Q^{\pi^{(t)}}(s,a) \right)^2 \right] + \mathbb{E}_{s\sim d^{\pi^{(t)}}} \left[2\left[\mathbb{E}_{a'\sim\pi^{(t)}(s)} \left[Q^{\pi^{(t)}}(s,a') - f^{(t)}(s,a') \right] \right]^2 \right] \tag{24}$$

$$\le \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \left[2\left(f^{(t)}(s,a) - Q^{\pi^{(t)}}(s,a) \right)^2 \right] + \mathbb{E}_{(s,a')\sim d^{\pi^{(t)}}} \left[2\left(Q^{\pi^{(t)}}(s,a') - f^{(t)}(s,a') \right)^2 \right] \tag{25}$$

$$\le 4\mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \left[\left(f^{(t)}(s,a) - Q^{\pi^{(t)}}(s,a) \right)^2 \right] \tag{26}$$

where (25) holds by Jensen's inequality. Then, we proceed to derive an upper bound on $\mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \left[\left(f^{(t)}(s,a) - Q^{\pi^{(t)}}(s,a) \right)^2 \right]$. By the definition of $f^{(t)} := (1-\alpha(t))Q_{\text{tar}}^{(t)}(s,a) + \alpha(t)Q_{\text{src}}(\phi^{(t)}(s), \psi^{(t)}(a))$, we have

$$\mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \left[\left(f^{(t)}(s,a) - Q^{\pi^{(t)}}(s,a) \right)^2 \right] \tag{27}$$

$$= \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \left[\left((1-\alpha(t))Q_{\text{tar}}^{(t)}(s,a) + \alpha(t)Q_{\text{src}}(\phi^{(t)}(s), \psi^{(t)}(a)) - Q^{\pi^{(t)}}(s,a) \right)^2 \right] \tag{28}$$

$$\begin{aligned}= \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \Bigg[\Big(&(1-\alpha(t))\left(Q_{\text{tar}}^{(t)}(s,a) - r_{\text{tar}}(s,a) + r_{\text{tar}}(s,a) \right) \\ &+ \alpha(t)\left(Q_{\text{src}}(\phi^{(t)}(s), \psi^{(t)}(a)) - r_{\text{tar}}(s,a) + r_{\text{tar}}(s,a) \right) - Q^{\pi^{(t)}}(s,a) \Big)^2 \Bigg]\end{aligned} \tag{29}$$

$$= \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}}\Bigg[\Big(\big(1-\alpha(t)\big)\big(Q_{\text{tar}}^{(t)}(s,a) - r_{\text{tar}}(s,a) + r_{\text{tar}}(s,a) - \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{tar}}^{(t)}(s',a')] + \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{tar}}^{(t)}(s',a')]\big) + \alpha(t)\Big(Q_{\text{src}}(\phi^{(t)}(s),\psi^{(t)}(a)) - r_{\text{tar}}(s,a) + r_{\text{tar}}(s,a) - \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{src}}(\phi^{(t)}(s'),\psi^{(t)}(a'))] + \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{src}}(\phi^{(t)}(s'),\psi^{(t)}(a'))]\Big) - Q^{\pi^{(t)}}(s,a)\Big)^2\Bigg] \tag{30}$$

$$= \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}}\Bigg[\Big(\big(1-\alpha(t)\big)\big(Q_{\text{tar}}^{(t)}(s,a) - r_{\text{tar}}(s,a) - \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{tar}}^{(t)}(s',a')]\big) + \alpha(t)\Big(Q_{\text{src}}(\phi^{(t)}(s),\psi^{(t)}(a)) - r_{\text{tar}}(s,a) - \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{src}}(\phi^{(t)}(s'),\psi^{(t)}(a'))]\Big) + \big(1-\alpha(t)\big)\gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{tar}}^{(t)}(s',a')] + \alpha(t)\gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{src}}(\phi^{(t)}(s'),\psi^{(t)}(a'))] + r_{\text{tar}}(s,a) - Q^{\pi^{(t)}}(s,a)\Big)^2\Bigg] \tag{31}$$

$$= \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}}\Bigg[\Big(\big(1-\alpha(t)\big)\Big(Q_{\text{tar}}^{(t)}(s,a) - r_{\text{tar}}(s,a) - \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{tar}}^{(t)}(s',a')]\Big) + \alpha(t)\Big(Q_{\text{src}}(\phi^{(t)}(s),\psi^{(t)}(a)) - r_{\text{tar}}(s,a) - \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{src}}(\phi^{(t)}(s'),\psi^{(t)}(a'))]\Big) + \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[f^{(t)}(s',a')] + r_{\text{tar}}(s,a) - Q^{\pi^{(t)}}(s,a)\Big)^2\Bigg] \tag{32}$$

where we obtain (29) by adding the dummy terms $\big(1-\alpha(t)\big)\big(-r_{\text{tar}}(s,a)+r_{\text{tar}}(s,a)\big)$ and $\alpha(t)\big(-r_{\text{tar}}(s,a)+r_{\text{tar}}(s,a)\big)$ to the inner part of (28), (30) is obtained by adding $\big(1-\alpha(t)\big)\big(-\gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{tar}}^{(t)}(s',a')] + \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{tar}}^{(t)}(s',a')]\big)$ and $\alpha(t)\big(-\gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{src}}(\phi^{(t)}(s'),\psi^{(t)}(a'))] + \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{src}}(\phi^{(t)}(s'),\psi^{(t)}(a'))]\big)$ to the inner part of (29), (31) holds by rearranging the terms in (30), and (32) holds by the definition of $f^{(t)}$. Then, by adding $\gamma\,\mathbb{E}_{\substack{s''\sim P_{\text{tar}}(\cdot|s,a)\\ a''\sim\pi^{(t)}(\cdot|s'')}}[Q^{\pi^{(t)}}(s'',a'')] - \gamma\,\mathbb{E}_{\substack{s''\sim P_{\text{tar}}(\cdot|s,a)\\ a''\sim\pi^{(t)}(\cdot|s'')}}[Q^{\pi^{(t)}}(s'',a'')]$ to the inner part of (32), we can rewrite (32) as

$$\mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}}\Bigg[\Big(\big(1-\alpha(t)\big)\Big(Q_{\text{tar}}^{(t)}(s,a) - r_{\text{tar}}(s,a) - \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{tar}}^{(t)}(s',a')]\Big) + \alpha(t)\Big(Q_{\text{src}}(\phi^{(t)}(s),\psi^{(t)}(a)) - r_{\text{tar}}(s,a) - \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{src}}(\phi^{(t)}(s'),\psi^{(t)}(a'))]\Big) + \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[f^{(t)}(s',a')] + r_{\text{tar}}(s,a) - Q^{\pi^{(t)}}(s,a) + \gamma\,\mathbb{E}_{\substack{s''\sim P_{\text{tar}}(\cdot|s,a)\\ a''\sim\pi^{(t)}(\cdot|s'')}}[Q^{\pi^{(t)}}(s'',a'')] - \gamma\,\mathbb{E}_{\substack{s''\sim P_{\text{tar}}(\cdot|s,a)\\ a''\sim\pi^{(t)}(\cdot|s'')}}[Q^{\pi^{(t)}}(s'',a'')]\Big)^2\Bigg] \tag{33}$$

$$= \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \Bigg[\bigg| \big(1-\alpha(t)\big) \Big(Q^{(t)}_{\text{tar}}(s,a) - r_{\text{tar}}(s,a) - \gamma \, \mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a) \\ a'\sim\pi^{(t)}(\cdot|s')}} [Q^{(t)}_{\text{tar}}(s',a')] \Big)$$
$$+ \alpha(t) \Big(Q_{\text{src}}(\phi^{(t)}(s), \psi^{(t)}(a)) - r_{\text{tar}}(s,a) - \gamma \, \mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a) \\ a'\sim\pi^{(t)}(\cdot|s')}} [Q_{\text{src}}(\phi^{(t)}(s'), \psi^{(t)}(a'))] \Big)$$
$$+ \gamma \, \mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a) \\ a'\sim\pi^{(t)}(\cdot|s')}} [f^{(t)}(s',a')] + r_{\text{tar}}(s,a) - Q^{\pi^{(t)}}(s,a) \tag{34}$$
$$+ \gamma \, \mathbb{E}_{\substack{s''\sim P_{\text{tar}}(\cdot|s,a) \\ a''\sim\pi^{(t)}(\cdot|s'')}} [Q^{\pi^{(t)}}(s'',a'')] - \gamma \, \mathbb{E}_{\substack{s''\sim P_{\text{tar}}(\cdot|s,a) \\ a''\sim\pi^{(t)}(\cdot|s'')}} [Q^{\pi^{(t)}}(s'',a'')] \bigg|^2 \Bigg]$$

$$\leq \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \Bigg[\bigg(\Big| \big(1-\alpha(t)\big) \Big(Q^{(t)}_{\text{tar}}(s,a) - r_{\text{tar}}(s,a) - \gamma \, \mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a) \\ a'\sim\pi^{(t)}(\cdot|s')}} [Q^{(t)}_{\text{tar}}(s',a')] \Big) \Big|$$
$$+ \Big| \alpha(t) \Big(Q_{\text{src}}(\phi^{(t)}(s), \psi^{(t)}(a)) - r_{\text{tar}}(s,a) - \gamma \, \mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a) \\ a'\sim\pi^{(t)}(\cdot|s')}} [Q_{\text{src}}(\phi^{(t)}(s'), \psi^{(t)}(a'))] \Big) \Big|$$
$$+ \Big| \gamma \, \mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a) \\ a'\sim\pi^{(t)}(\cdot|s')}} [f^{(t)}(s',a')] + r_{\text{tar}}(s,a) - Q^{\pi^{(t)}}(s,a) \Big| \tag{35}$$
$$+ \Big| \gamma \, \mathbb{E}_{\substack{s''\sim P_{\text{tar}}(\cdot|s,a) \\ a''\sim\pi^{(t)}(\cdot|s'')}} [Q^{\pi^{(t)}}(s'',a'')] - \gamma \, \mathbb{E}_{\substack{s''\sim P_{\text{tar}}(\cdot|s,a) \\ a''\sim\pi^{(t)}(\cdot|s'')}} [Q^{\pi^{(t)}}(s'',a'')] \Big| \bigg)^2 \Bigg]$$

$$\leq \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \Bigg[\bigg(\big(1-\alpha(t)\big) \underbrace{\Big| Q^{(t)}_{\text{tar}}(s,a) - r_{\text{tar}}(s,a) - \gamma \, \mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a) \\ a'\sim\pi^{(t)}(\cdot|s')}} [Q^{(t)}_{\text{tar}}(s',a')] \Big|}_{=:\epsilon^{(t)}_{\text{td}}(s,a)}$$
$$+ \alpha(t) \underbrace{\Big| \Big(Q_{\text{src}}(\phi^{(t)}(s), \psi^{(t)}(a)) - r_{\text{tar}}(s,a) - \gamma \, \mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a) \\ a'\sim\pi^{(t)}(\cdot|s')}} [Q_{\text{src}}(\phi^{(t)}(s'), \psi^{(t)}(a'))] \Big) \Big|}_{=:\epsilon^{(t)}_{\text{src,be}}(s,a;Q_{\text{src}})} \tag{36}$$
$$+ \Big| \gamma \, \mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a) \\ a'\sim\pi^{(t)}(\cdot|s')}} [f^{(t)}(s',a')] - \gamma \, \mathbb{E}_{\substack{s''\sim P_{\text{tar}}(\cdot|s,a) \\ a''\sim\pi^{(t)}(\cdot|s'')}} [Q^{\pi^{(t)}}(s'',a'')] \Big|$$
$$+ \underbrace{\Big| r_{\text{tar}}(s,a) - Q^{\pi^{(t)}}(s,a) + \gamma \, \mathbb{E}_{\substack{s''\sim P_{\text{tar}}(\cdot|s,a) \\ a''\sim\pi^{(t)}(\cdot|s'')}} [Q^{\pi^{(t)}}(s'',a'')] \Big|}_{=0} \bigg)^2 \Bigg]$$

$$= \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \Bigg[\bigg(\big(1-\alpha(t)\big) \epsilon^{(t)}_{\text{td}}(s,a) + \alpha(t) \epsilon^{(t)}_{\text{src,be}}(s,a;Q_{\text{src}})$$
$$+ \gamma \Big| \mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a) \\ a'\sim\pi^{(t)}(\cdot|s'')}} \Big[f^{(t)}(s',a') - Q^{\pi^{(t)}}(s',a') \Big] \Big| \bigg)^2 \Bigg] \tag{37}$$

$$= \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}} \Bigg[\bigg(\big(1-\alpha(t)\big) \epsilon^{(t)}_{\text{td}}(s,a) + \alpha(t) \epsilon^{(t)}_{\text{src,be}}(s,a;Q_{\text{src}})$$
$$+ \gamma \, \mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a) \\ a'\sim\pi^{(t)}(\cdot|s'')}} \Big[\big| f^{(t)}(s',a') - Q^{\pi^{(t)}}(s',a') \big| \Big] \bigg)^2 \Bigg] \tag{38}$$

where (34) holds by the fact that $x^2 = |x|^2$, (35) holds by triangle inequality, (36) by the facts that $0 \leq \alpha(t) \leq 1$ and $0 \leq 1 - \alpha(t) \leq 1$, (37) holds by coupling (s', a') and (s'', a'') and applying Bellman expectation equation as well as the definitions that $\epsilon^{(t)}_{\text{td}}(s,a) := \big| Q^{(t)}_{\text{tar}}(s,a) - r_{\text{tar}}(s,a) - \gamma \, \mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a) \\ a'\sim\pi^{(t)}(\cdot|s')}} [Q^{(t)}_{\text{tar}}(s',a')] \big|$ and $\epsilon^{(t)}_{\text{src,be}}(s,a;Q_{\text{src}}) := \big| Q_{\text{src}}(\phi^{(t)}(s), \psi^{(t)}(a)) - r_{\text{tar}}(s,a) - \gamma \, \mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a) \\ a'\sim\pi^{(t)}(\cdot|s')}} [Q_{\text{src}}(\phi^{(t)}(s'), \psi^{(t)}(a'))] \big|$. By recursively applying the procedure from (27) to (38) to $\left|f^{(t)}(s',a') - Q^{\pi^{(t)}}(s',a')\right|$, we obtain a bound on $\mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}}\left[\left(f^{(t)}(s,a) - Q^{\pi^{(t)}}(s,a)\right)^2\right]$ as follows:

$$\mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}}\left[\left(f^{(t)}(s,a) - Q^{\pi^{(t)}}(s,a)\right)^2\right] \tag{39}$$

$$\begin{aligned}\le \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}}\Bigg[\bigg(&\big(1-\alpha(t)\big)\epsilon^{(t)}_{\text{td}}(s,a) + \alpha(t)\epsilon^{(t)}_{\text{src,be}}(s,a;Q_{\text{src}})\\ &+ \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}\left[\left|f^{(t)}(s',a') - Q^{\pi^{(t)}}(s',a')\right|\right]\bigg)^2\Bigg]\end{aligned} \tag{40}$$

$$\begin{aligned}\le \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}}\Bigg[\bigg(&\big(1-\alpha(t)\big)\epsilon^{(t)}_{\text{td}}(s,a) + \alpha(t)\epsilon^{(t)}_{\text{src,be}}(s,a;Q_{\text{src}})\\ &+ \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}\Big[\big(1-\alpha(t)\big)\epsilon^{(t)}_{\text{td}}(s',a') + \alpha(t)\epsilon^{(t)}_{\text{src,be}}(s',a';Q_{\text{src}})\\ &\quad+ \mathbb{E}_{\substack{s''\sim P_{\text{tar}}(\cdot|s',a')\\ a''\sim\pi^{(t)}(\cdot|s'')}}\left[\left|f^{(t)}(s'',a'') - Q^{\pi^{(t)}}(s'',a'')\right|\right]\Big]\bigg)^2\Bigg]\end{aligned} \tag{41}$$

$$\begin{aligned}\le \mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}}\Bigg[\bigg(&\big(1-\alpha(t)\big)\epsilon^{(t)}_{\text{td}}(s,a) + \alpha(t)\epsilon^{(t)}_{\text{src,be}}(s,a;Q_{\text{src}})\\ &+ \frac{1}{(1-\gamma)\mu_{\text{tar,min}}}\Big(\gamma\big(1-\alpha(t)\big)\epsilon^{(t)}_{\text{td}}(s,a) + \gamma\alpha(t)\epsilon^{(t)}_{\text{src,be}}(s,a;Q_{\text{src}})\\ &+ \gamma^2\big(1-\alpha(t)\big)\epsilon^{(t)}_{\text{td}}(s,a) + \gamma^2\alpha(t)\epsilon^{(t)}_{\text{src,be}}(s,a;Q_{\text{src}}) + \cdots\Big)\bigg)^2\Bigg]\end{aligned} \tag{42}$$

$$\le \frac{1}{(1-\gamma)^4\mu^2_{\text{tar,min}}}\mathbb{E}_{(s,a)\sim d^{\pi^{(t)}}}\left[\left(\big(1-\alpha(t)\big)\epsilon^{(t)}_{\text{td}}(s,a) + \alpha(t)\epsilon^{(t)}_{\text{src,be}}(s,a;Q_{\text{src}})\right)^2\right] \tag{43}$$

where (41) holds by applying the procedure from (27) to (38) to $f^{(t)}(s',a') - Q^{\pi^{(t)}}(s',a')$, (42) holds by applying the procedure from (27) to (38) to all the subsequent time steps and using importance sampling with the importance ratio bound in Lemma 4 and then using the same dummy variables (s,a) for all the subsequent state-action pairs, and (43) holds by taking the sum of an infinite geometric sequence. □

Theorem 1. *(Average Sub-Optimality) Under the QAvatar in Algorithm 1 and Assumption 1, the average sub-optimality over T iterations can be upper bounded as*

$$\begin{aligned}&\frac{1}{T}\sum_{t=1}^{T}\left(V^{\pi^*}(\mu_{tar}) - V^{\pi^{(t)}}(\mu_{tar})\right)\\ &\le \underbrace{\frac{2}{1-\gamma}\sqrt{\frac{\log(A)}{T}}}_{(a)} + \underbrace{\frac{C_0}{T}\sum_{t=1}^{T}\alpha(t)\|\epsilon^{(t)}_{src,\,be}(Q_{src})\|_\infty}_{(b)} + \underbrace{\frac{C_0}{T}\sum_{t=1}^{T}(1-\alpha(t))\|\epsilon^{(t)}_{td}\|_\infty}_{(c)},\end{aligned} \tag{4}$$

where $C_0 := 2\sqrt{C_{\pi^*}}/((1-\gamma)^2\mu_{tar,\,min})$.

Proof. We start by providing an upper bound on the sub-optimality gap $V^{\pi^*}(\mu_{\text{tar}}) - V^{\pi^{(t)}}(\mu_{\text{tar}})$ at each iteration. Recall that d^{π}_{tar} denotes the discounted state-action visitation distribution of policy π in the target domain. Note that

$$V^{\pi^*}(\mu_{\text{tar}}) - V^{\pi^{(t)}}(\mu_{\text{tar}}) \tag{44}$$

$$= \frac{1}{1-\gamma}\mathbb{E}_{(s,a)\sim d^{\pi^*}_{\text{tar}}}\left[A^{\pi^{(t)}}(s,a)\right] \tag{45}$$

$$= \frac{1}{1-\gamma}\mathbb{E}_{(s,a)\sim d^{\pi^*}_{\text{tar}}}\left[\bar{f}^{(t)}(s,a) - \bar{f}^{(t)}(s,a) + A^{\pi^{(t)}}(s,a)\right] \tag{46}$$

$$= \frac{1}{1-\gamma}\mathbb{E}_{(s,a)\sim d_{\text{tar}}^{\pi^*}}\Big[\bar{f}^{(t)}(s,a)\Big] + \frac{1}{1-\gamma}\mathbb{E}_{(s,a)\sim d_{\text{tar}}^{\pi^*}}\Big[-\bar{f}^{(t)}(s,a) + A^{\pi^{(t)}}(s,a)\Big] \tag{47}$$

$$\leq \frac{1}{1-\gamma}\mathbb{E}_{(s,a)\sim d_{\text{tar}}^{\pi^*}}\Big[\bar{f}^{(t)}(s,a)\Big] + \frac{1}{1-\gamma}\sqrt{\mathbb{E}_{(s,a)\sim d_{\text{tar}}^{\pi^*}}\Big[\big(-\bar{f}^{(t)}(s,a) + A^{\pi^{(t)}}(s,a)\big)^2\Big]}, \tag{48}$$

where (45) holds by the performance difference lemma (cf. Lemma 1), (46) is obtained by adding $\bar{f}^t(s,a) - \bar{f}^t(s,a)$, (47) is obtained by rearranging the terms in (46), and (48) holds by Jensen's inequality. By the fact that $\|\frac{d^{\pi^*}}{d^{\pi^{(t)}}}\|_\infty \leq C$, we have

$$\frac{1}{1-\gamma}\mathbb{E}_{(s,a)\sim d^{\pi^*}}\Big[\bar{f}^{(t)}(s,a)\Big] + \frac{1}{1-\gamma}\sqrt{\mathbb{E}_{s,a\sim d^{\pi^*}}\Big[\big(-\bar{f}^{(t)}(s,a) + A^{\pi^{(t)}}(s,a)\big)^2\Big]} \tag{49}$$

$$\leq \frac{1}{1-\gamma}\mathbb{E}_{(s,a)\sim d^{\pi^*}}\Big[\bar{f}^{(t)}(s,a)\Big] + \frac{1}{1-\gamma}\sqrt{C\cdot\mathbb{E}_{s,a\sim d^{\pi^{(t)}}}\Big[\big(-\bar{f}^{(t)}(s,a) + A^{\pi^{(t)}}(s,a)\big)^2\Big]}. \tag{50}$$

Recall the definitions of $\epsilon_{\text{td}}^{(t)}(s,a)$ and $\epsilon_{\text{src, be}}^{(t)}(s,a;Q_{\text{src}})$ as

$$\epsilon_{\text{td}}^{(t)}(s,a) := \big|Q_{\text{tar}}^{(t)}(s,a) - r_{\text{tar}}(s,a) - \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{tar}}^{(t)}(s',a')]\big|, \tag{51}$$

$$\epsilon_{\text{src,be}}^{(t)}(s,a;Q_{\text{src}}) := \big|Q_{\text{src}}(\phi^{(t)}(s),\psi^{(t)}(a)) - r_{\text{tar}}(s,a) - \gamma\,\mathbb{E}_{\substack{s'\sim P_{\text{tar}}(\cdot|s,a)\\ a'\sim\pi^{(t)}(\cdot|s')}}[Q_{\text{src}}(\phi^{(t)}(s'),\psi^{(t)}(a'))]\big|. \tag{52}$$

Recall that we also define

$$\|\epsilon_{\text{td}}^{(t)}\|_\infty := \max_{(s,a)\in\mathcal{S}\times\mathcal{A}} \epsilon_{\text{td}}^{(t)}(s,a), \tag{53}$$

$$\|\epsilon_{\text{src, be}}^{(t)}(Q_{\text{src}})\|_\infty := \max_{(s,a)\in\mathcal{S}\times\mathcal{A}} \epsilon_{\text{src, be}}^{(t)}(s,a;Q_{\text{src}}). \tag{54}$$

We are ready to put everything together and establish the cumulative sub-optimality. By taking the summation of (50) over iterations, we have

$$\sum_{t=1}^{T}\mathbb{E}_{s\sim\mu_{\text{tar}}}\Big[V^{\pi^*}(s) - V^{\pi^{(t)}}(s)\Big] \tag{55}$$

$$\leq \sum_{t=1}^{T}\frac{1}{1-\gamma}\mathbb{E}_{s,a\sim d^{\pi^*}}\Big[\bar{f}^{(t)}(s,a)\Big] + \sum_{t=1}^{T}\frac{1}{1-\gamma}\sqrt{C\mathbb{E}_{s,a\sim d^{\pi^{(t)}}}\Big[\big(-\bar{f}^{(t)}(s,a) + A^{\pi^{(t)}}(s,a)\big)^2\Big]} \tag{56}$$

$$\begin{aligned}\leq \frac{2}{1-\gamma}\sqrt{\log(A)T} &+ \frac{2\sqrt{C}}{(1-\gamma)^2\mu_{\text{tar,min}}}\sum_{t=1}^{T}\alpha(t)\|\epsilon_{\text{src, be}}^{(t)}(Q_{\text{src}})\|_\infty \\ &+ \frac{2\sqrt{C}}{(1-\gamma)^2\mu_{\text{tar,min}}}\sum_{t=1}^{T}(1-\alpha(t))\|\epsilon_{\text{td}}^{(t)}\|_\infty.\end{aligned} \tag{57}$$

where (56) follows directly from (50) and (57) holds by Lemma 5 and Lemma 2. □

C Implementation Details of QAvatar

C.1 Pseudo Code of the Practical Implementation of QAvatar

In this section, we provide the pseudo code of the practical version of QAvatar in Algorithm 2.

C.2 Inter-Domain Mapping Network Augmented With a Normalizing Flow Model

Algorithm 2 Practical Implementation of QAvatar

```
Require: Source-domain Q-network $Q_{\text{src}}$, value function $V_{\text{src}}$, and the decay function $\alpha : \mathbb{R} \to [0, 1]$.
1: Initialize the state mapping function $\phi$, the action mapping function $\psi$, the target-domain policy
   network $\pi$, and entropy coefficient $\beta$
2: for iteration $t = 1, \cdots, T$ do
3:     Sample $\mathcal{D}_{\text{tar}}^{(t)} = \{(s, a, r, s')\}$ of $N_{\text{tar}}$ samples using $\pi^{(t)}$ in the target domain
4:     Update the target-domain $\{Q_{\text{tar},1}, Q_{\text{tar},2}\}$ by SAC's critic loss:
       $Q_{\text{tar},j}^{(t)} = \arg\min_{Q_{\text{tar}}} \hat{\mathbb{E}}_{(s,a,r,s') \in \mathcal{D}_{\text{tar}}^{(t)}} \Big[ \big| r + \gamma \mathbb{E}_{a' \sim \pi^{(t)}} \big[ Q_{\text{tar}}(s', a') - \beta \log(\pi(a'|s')) \big] - Q_{\text{tar}}(s, a) \big|^2 \Big].$   (58)
5:     Update the state mapping function $\phi$ and action mapping function $\psi$ by minimizing
6:     the following loss
       $\phi^{(t)}, \psi^{(t)} = \arg\min_{\phi,\psi} \hat{\mathbb{E}}_{(s,a,r,s') \in \mathcal{D}_{\text{tar}}^{(t)}} \Big[ \big| r + \gamma V_{\text{src}}(\phi(s')) - Q_{\text{src}}(\phi(s), \psi(a)) \big| \Big].$   (59)
7:     Update the target-domain policy $\pi$
       $\pi^{(t+1)} = \arg\min_{\pi} \hat{\mathbb{E}}_{(s,a,r,s') \in \mathcal{D}_{\text{tar}}^{(t)}, a' \sim \pi^{(t)}(\cdot|s)} \Big[ \beta \log \pi(a'|s) - f^{(t)}(s, a') \Big],$   (60)
       $f^{(t)}(s, a') = (1 - \alpha(t)) \min_{j=1,2} Q_{\text{tar},j}^{(t)}(s, a') + \alpha(t) Q_{\text{src}}(\phi^{(t)}(s), \psi^{(t)}(a')).$   (61)
8: end for
```

As mentioned in Section 4, we use the flow model to map the outputs of the mapping functions to the feasible regions. The way to integrate these two components is shown in Figure 5.

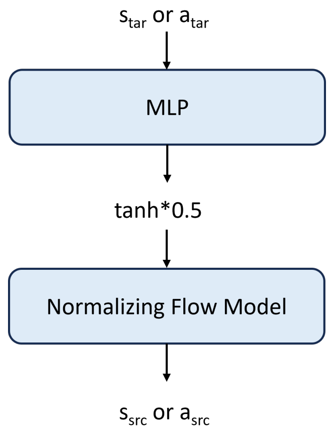


Figure 5: Integration of the mapping function and the normalizing flow model.

D Additional Experimental Results

D.1 Final Rewards

In this section, we show the asymptotic performance of all baselines and our algorithm. In the MuJoCo environments except for Ant and Inverted Double Pendulum, we train all the target-domain models for 500k steps. In Ant and Inverted Double Pendulum, we train all the target-domain models for 350k and 20k steps, respectively. In Robosuite environments, we train all the target-domain models for 20k steps. The asymptotic performances of all baselines and our algorithm are shown in the following tables.

Table 2: Final rewards of QAvatar and all baselines in the MuJoCo environments.

Algorithm	Swimmer	Hopper	HalfCheetah	Ant	Modified IDP
QAvatar	**247 ± 187**	**2762 ± 440**	**12316 ± 586**	**2234 ± 1112**	**9241 ± 62**
SAC	177 ± 168	2086 ± 257	10986 ± 1822	1620 ± 527	9212 ± 152
CMD Random	-14 ± 4	39 ± 28	-427 ± 84	-1713 ± 475	50 ± 7
CMD Policy	-7 ± 5	59 ± 46	-253 ± 344	778 ± 144	72 ± 12
DCC Batch	100 ± 37	48 ± 27	292 ± 418	-430 ± 138	115 ± 8
DCC Online	-7 ± 44	30 ± 16	-631 ± 185	-1240 ± 838	95 ± 6
CAT	104 ± 28	154 ± 156	46 ± 250	17 ± 27	41 ± 10

Table 3: Final rewards of QAvatar and all baselines in the Robosuite environments

Environment	Block Lifting	Door Opening	Table Wiping
QAvatar	**98.0 ± 21.1**	**185.2 ± 66.9**	**67.1 ± 9.1**
SAC	90.3 ± 23.4	160.1 ± 40.3	47.2 ± 7.1
CMD Random	1.2 ± 0.5	7.4 ± 0.9	0.7 ± 0.5
CMD Policy	0.9 ± 0.6	7.8 ± 6.4	0.8 ± 0.4
DCC Policy	1.4 ± 1.1	18.7 ± 16.0	1.0 ± 1.1
DCC Online	0.6 ± 0.2	8.2 ± 4.7	0.9 ± 0.7
CAT	15.0 ± 14.3	34.7 ± 8.4	55.5 ± 29.7

D.2 Time To Threshold

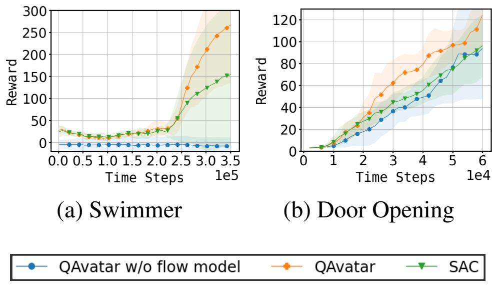
In the following table, we discover that QAvatar uses the less data to reach the threshold than SAC does. In Hopper, QAvatar only needs half the amount of data SAC needs to reach the goal.

Table 4: Time to threshold of QAvatar and all baselines

Environment	Threshold	QAvatar	SAC	SAC / QAvatar
Swimmer	175	284K	484K	1.70
Hopper	2000	218K	486K	2.23
HalfCheetah	10000	288K	400K	1.39
Ant	1600	254K	344K	1.35
Block Lifting	85	90K	94K	1.04
Door Opening	150	80K	94K	1.18
Table Wiping	45	74K	96K	1.30
Inverted Double Pendulum	9000	16K	18K	1.13

D.3 Ablation Study: Deactivating the Flow Model

As mentioned above, we use a normalizing flow model to restrict the output range of the mapping functions in the feasible regions. In this experiment, we disable the flow model and evaluate QAvatar in Swimmer and Door Opening. In Figure 6, QAvatar without a flow model performs worse than QAvatar with a flow model. In Swimmer, the ewma values of rewards obtained by QAvatar without the flow model isn't higher than 50. Additionally, although the ewma values of rewards obtained by it are higher than 100 in Door Opening, it has to spend more time attaining high rewards than QAvatar with the flow model does.



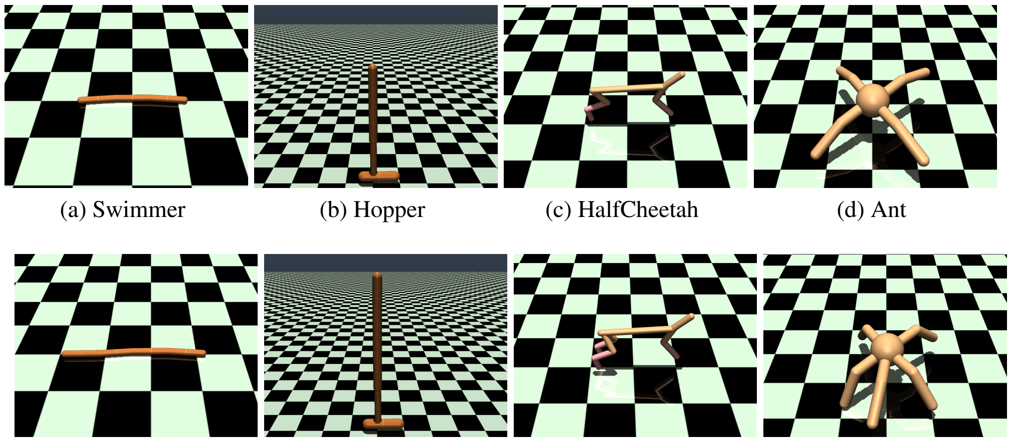
(a) Swimmer (b) Door Opening

Figure 6: Ablation Study: QAvatar without the flow model

E Configuration Details of The Experiments

E.1 MuJoCo Environments

As mentioned in Section 5, the source domains of our experiments are the original MuJoCo environments such as Swimmer-v3, Hopper-v3, HalfCheetah-v3 and Ant-v3. The target domains are the modified MuJoCo environments such as Swimmer with four limbs, Hopper with an extra thigh, HalfCheetah with three legs and Ant with five legs. The environments are shown in Figure 7.

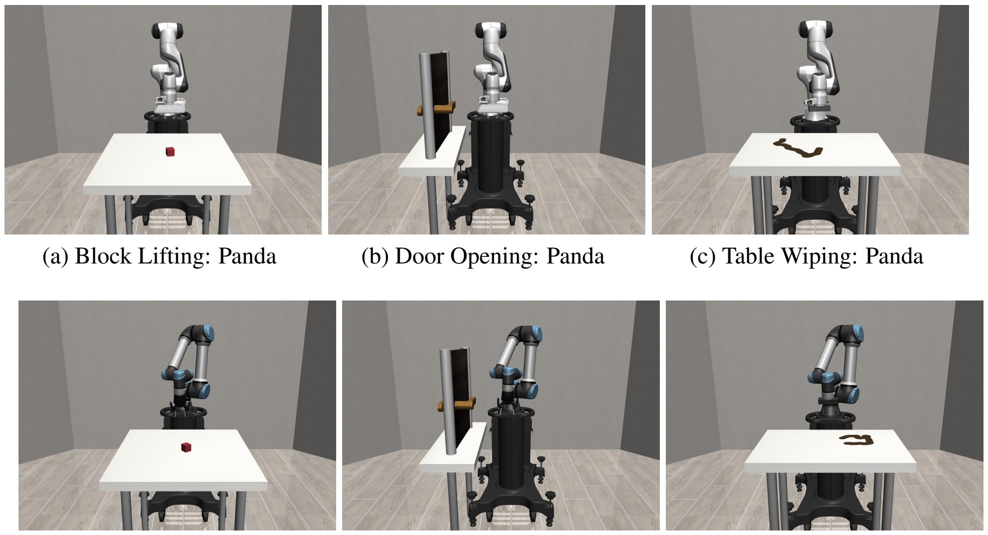

(a) Swimmer (b) Hopper (c) HalfCheetah (d) Ant

(e) Four-limb Swimmer (f) Three-thigh Hopper (g) Three-leg HaldCheetah (h) Five-leg Ant

Figure 7: The environments of the source domains and the target domains. (a)-(d): Source domains – Original MuJoCo environments. (e)-(h): Target domains – Modified MuJoCo environments.

E.2 Robosuite Environments

Robosuite is a popular robot learning package. We evaluate QAvatar on three tasks, including block lifting, door opening, and table wiping. For each task, we consider cross-domain transfer from controlling a Panda robot arm to controlling a UR5e robot arm. These three tasks are illustrated in Figure 8.


(a) Block Lifting: Panda (b) Door Opening: Panda (c) Table Wiping: Panda

(d) Block Lifting: UR5e (e) Door Opening: UR5e (f) Table Wiping: UR5e

Figure 8: The environments of the source domains and the target domains. (a)-(c)The source domains: control Panda to solve the tasks. (d)-(f)The target domains: control UR5e to solve the tasks.

E.3 The Implementation Details of Baselines

SAC. The implementation of SAC used in our experiments is released by stable-baselines3 [47]. The settings of all hyperparameters except for the discouted factor γ follows the default settings of SAC in the documentation of stable-baselines3. The discouted factor is set as 0.9999 in Swimmer-v3 and 0.99 in all other MuJoCo environments, which follows the setting shown in Hugging Face. As for in the Robosuite environments, we set the discouted factor to 0.9.

CMD. We implement CMD by ourselves according to the pseudocode of CMD shown in its original paper [15]. We follow the setting of the hyperparameters which is revealed in its original paper. Additionally, we change CMD from collecting the fixed amount of data to collecting data continuously for a fair comparison. As for the source model, we use the same model used in our algorithm.

DCC. We use the original implementation of [11] (`https://github.com/sjtuzq/Cycle_Dynamics`) with their default setting [11]. For a fair comparison, we use the same source model used in QAvatar and change DCC from collecting the fixed amount of data to collecting data continuously.

CAT. We use the authors' implementation (`https://github.com/TJU-DRL-LAB/transfer-and-multi-task-reinforcement-learning/tree/main/Single-agent%20Transfer%20RL/Cross-domain%20Transfer/CAT`) and use PPO as the target-domain base algorithm following the original paper. For a fair comparison, we use the same source model used in QAvatar . The hyperparameters are shown in the following table and "n epochs" means the number of epochs when optimizing the surrogate loss.

Table 5: A list of candidate hyperparameters for Robosuite and MuJoCo.

Parameter	MuJoCo	Robosuite
learning rate	0.0001, 0.0003, 0.0004, 0.0008	0.0001, 0.0003
length of rollouts	500, 2000 (50, 100 for Modified IDP)	2000
batch size	50, 100 (20, 25 for Modified IDP)	50, 100, 200
entropy coefficient (ent. coef.)	0.01, 0.002	0.01, 0.002
n epochs	10, 20	5, 10
num. of hidden layer of encoder/decoder	1	1
num. of hidden layer of actor/critic	2	2
hidden layer size	256	256

Table 6: Final hyperparameters chosen for each environment.

	learning rate	len. of rollouts	batch size	ent. coef.	n epochs
Swimmer	0.0003	500	50	0.01	10
Hopper	0.0008	2000	100	0.002	20
HalfCheetah	0.0001	500	50	0.002	10
Ant	0.0004	500	50	0.002	10
InvertedDoublePendulum	0.001	100	20	0.01	20
Robosuite	0.0003	2000	100	0.01	10

E.4 Detailed Configuration of QAvatar

The base algorithm, SAC, is implemented by stable-baselines3 [47]. As for the compute resource, we use NVIDIA GeForce RTX 3090 to do the experiments. Finishing the whole training process including training the source-domain model, target-domain model and flow model once needs 44 hours in the MuJoCo environments and 39 hours in the Robosuite environments. The Hyperparameters of QAvatar are shown in the following two tables. The consider functions of decay functions are $1/\sqrt{t}, 1/t, 1/t^2$ and $1/t^3$ and the final decay functions chosen for each environments are shown in the table 8. The settings of hyperparameters such as critic/actor learning rate, batch size, buffer size and discounted factor are same as SAC.

Table 7: A list of hyperparameters of QAvatar .

Parameter	Value
critic/actor learning rate	0.0003
state mapping function learning rate	0.01
action mapping function learning rate	0.01
batch size	256
replay buffer size	10^6
optimizer	Adam
number of hidden layer of mapping functions	1
hidden layer size	256

Table 8: A list of environment-specific hyperparameters of QAvatar .

Environment	Decay Function α
Swimmer-v3	$1/t$
Hopper-v3	$1/t^2$
HalfCheetah-v3	$1/t^3$
Ant-v3	$1/\sqrt{t}$
InvertedDoublePendulum-v2	$1/t$
Block Lifting	$1/t$
Door Opening	$1/t$
Table Wiping	$1/t$