Cross-Domain Policy Adaptation by Capturing Representation Mismatch

Jiafei Lyu [1] **Chenjia Bai** [2] **Jingwen Yang** [3] **Zongqing Lu** [4 5] **Xiu Li** [1]

Abstract

It is vital to learn effective policies that can be transferred to different domains with dynamics discrepancies in reinforcement learning (RL). In this paper, we consider dynamics adaptation settings where there exists dynamics mismatch between the source domain and the target domain, and one can get access to sufficient source domain data, while can only have limited interactions with the target domain. Existing methods address this problem by learning domain classifiers, performing data filtering from a value discrepancy perspective, etc. Instead, we tackle this challenge from a *decoupled representation learning* perspective. We perform representation learning only in the target domain and measure the representation deviations on the transitions from the source domain, which we show can be a signal of dynamics mismatch. We also show that representation deviation upper bounds performance difference of a given policy in the source domain and target domain, which motivates us to adopt representation deviation as a reward penalty. The produced representations are not involved in either policy or value function, but only serve as a reward penalizer. We conduct extensive experiments on environments with kinematic and morphology mismatch, and the results show that our method exhibits strong performance on many tasks. Our code is publicly available at https://github.com/dmksjfl/PAR.

1. Introduction

Alice is interested in learning cooking. She bought a new set of cookware recently that is different from the one she used before. She expertly uses new cookware soon. As this example conveys, human beings are able to quickly transfer the learned policies to similar tasks. Such capability is also expected in reinforcement learning (RL) agents. Unfortunately, RL algorithms are known to require a vast number of interactions to learn meaningful policies (Silver et al., 2016; Lyu et al., 2023). A bare fact is that sometimes only limited interactions with the environment (*target domain*) are feasible because it may be expensive and time-consuming for a large number of interactions in scenarios like robotics (Cutler & How, 2015; Kober et al., 2013), autonomous driving (Kiran et al., 2020; Osinski et al., 2019), etc. Nevertheless, we may simultaneously have access to another structurally similar *source domain* where the experience is cheaper to gather, *e.g.*, a simulator. Since the source domain can be biased, a dynamics mismatch between the two domains may persist. It then necessitates developing algorithms that have good performance in the target domain, given the source domain with some dynamics discrepancies.

Note that there are numerous studies concerning policy adaptation, such as system identification (Yu et al., 2017; Clavera et al., 2018) and domain randomization (Slaoui et al., 2019; Tobin et al., 2017; Peng et al., 2017). These methods often rely on demonstrations from the target domain (Kim et al., 2019), the distributions from which the simulator parameters are sampled, a manipulable simulator (Chebotar et al., 2018), etc. We lift these requirements and consider learning policies with sufficient source domain data (either online or offline) and limited online interactions with the target domain. This setting is also referred to as *off-dynamics RL* (Eysenbach et al., 2021) or *online dynamics adaptation* (Xu et al., 2023). Existing methods tackle this problem by learning domain classifiers (Eysenbach et al., 2021), filtering source domain data that share similar value estimates with target domain data (Xu et al., 2023), etc.

In this paper, we study the cross-domain policy adaptation problem where only transition dynamics between the source domain and the target domain differ. The state space, action space, as well as the reward function, are kept unchanged. Unlike prior works, we address this issue from a representation learning perspective. Our motivation is that the dynamics mismatch between the source domain and the target domain can be captured by representation deviations of transitions from the two domains, which is grounded by our

[1]Tsinghua Shenzhen International Graduate School, Tsinghua University [2]Shanghai Artificial Intelligence Laboratory [3]Tencent IEG [4]School of Computer Science, Peking University [5]Beijing Academy of Artificial Intelligence. Correspondence to: Xiu Li <li.xiu@sz.tsinghua.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

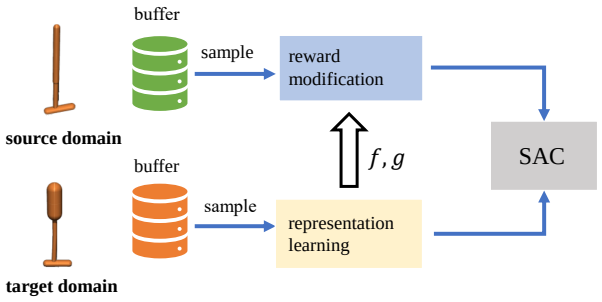


Figure 1. **Illustration of PAR**. We train encoders f, g merely with target domain data and utilize them to modify rewards from the source domain with measured representation deviations. Afterward, the downstream SAC algorithm can learn from transitions from both domains.

theoretical analysis. We further show concrete performance bounds given either online or offline source domain, where we observe that representation deviation upper bounds the performance difference of any given policy between the source domain and the target domain. Motivated by the theoretical findings, we deem that representation mismatch between two domains can be used as a reward penalizer to fulfill dynamics-aware policy adaptation.

For a practical usage, we propose **P**olicy **A**daptation by **R**epresentation mismatch, dubbed **PAR** algorithm. Our approach trains a state encoder and a state-action encoder *only in the target domain* to capture its latent dynamics structure, and then leverage the learned encoders to produce representations upon *transitions from the source domain*. We evaluate deviations between representations of the state-action pair and the next state, and use the resulting representation deviations to penalize source domain rewards, as depicted in Figure 1. Intuitively, the penalty is large if the transition deviates far from the target domain, and vice versa. In this way, the agent can benefit more from dynamics-consistent transitions and de-emphasize others. It is worth noting that the representation learning is decoupled from policy or value function training since the representations are not involved in them. Empirical results in environments with kinematic and morphology shifts show that our method notably beats previous strong baselines on many tasks in both online and offline source domain settings.

2. Related Work

Domain Adaptation in RL. Generalizing or transferring policies across varied domains remains a critical issue in RL, where domains may differ in terms of agent embodiment (Liu et al., 2022b; Zhang et al., 2021c), transition dynamics (Eysenbach et al., 2021; Viano et al., 2020), observation space (Gamrian & Goldberg, 2018; Bousmalis et al., 2018; Ge et al., 2022; Zhang et al., 2021b; Hansen et al., 2021), etc. We focus on policy adaptation under dynamics discrepancies between the two domains. Prior works mainly address this issue via system identification (Clavera et al., 2018; Zhou et al., 2019; Du et al., 2021; Xie et al., 2022), domain randomization (Slaoui et al., 2019; Mehta et al., 2019; Vuong et al., 2019; Jiang et al., 2023), meta-RL (Nagabandi et al., 2018; Raileanu et al., 2020; Arndt et al., 2019; Wu et al., 2022), or by leveraging expert demonstrations from the target domain (Kim et al., 2019; Hejna et al., 2020; Fickinger et al., 2022; Raychaudhuri et al., 2021). Though effective, these methods depend on a model of the environment, expert trajectories gathered in the target domain, or a proper choice of randomized parameters. In contrast, we dismiss these requirements and study dynamics adaptation problem (Xu et al., 2023) where only a small amount of online interactions with the target domain is allowed, and a source domain with sufficient data can be accessed. Under this setting, many approaches have been developed, such as directly optimizing the parameters of the simulator to calibrate the dynamics of the source domain (Farchy et al., 2013; Zhu et al., 2017; Collins et al., 2020; Chebotar et al., 2018; Ramos et al., 2019). However, it requires a manipulable simulator. There are attempts to use some expressive models to learn the dynamics change (Golemo et al., 2018; Hwangbo et al., 2019; Xiong et al., 2023), and action transformation methods that learn dynamics models of the two domains and utilize them to modify transitions from the source domain (Hanna et al., 2021; Desai et al., 2020), while it is difficult to learn accurate dynamics models (Malik et al., 2019; Lyu et al., 2022b). Another line of research trains domain classifiers and tries to close the dynamics gap by either reward modification (Eysenbach et al., 2021; Liu et al., 2022a), or importance weighting (Niu et al., 2022). A recent work (Xu et al., 2023) bridges the dynamics gap by selectively sharing transitions from the source domain that have similar value estimates as those in the target domain. Unlike these methods, we capture dynamics discrepancy by measuring representation mismatch. It is worth noting that in this work we only consider policy adaptation across domains that have the same state space and action space. Our method can also generalize to the setting where target domain has a different state space or action space by incorporating extra components or modules like prior works (Barekatain et al., 2019; You et al., 2022; Gui et al., 2023).

Representation Learning in RL. Representation learning is an important research topic in computer vision (Bengio et al., 2012; Kolesnikov et al., 2019; He et al., 2015). In the context of RL, representation learning is actively explored in image-based tasks (Kostrikov et al., 2020; Yarats et al., 2022; Liu et al., 2021; Cetin et al., 2022), aiming at extracting useful features from information-redundant images by contrastive learning (Srinivas et al., 2020; Eysenbach et al., 2022; Stooke et al., 2021; Zhu et al., 2020), MDP homo-

morphisms (van der Pol et al., 2020; Rezaei-Shoshtari et al., 2022), bisimulation (Ferns et al., 2011; Zhang et al., 2021a), self-predictive learning (Schwarzer et al., 2021; Tang et al., 2022; Kim et al., 2022), etc. Representation learning can also be found in model-based RL methods that rely on latent dynamics (Karl et al., 2016; Rafailov et al., 2020; Hafner et al., 2019; Hansen et al., 2022). In state-based tasks, it also spans in successor representation (Barreto et al., 2016; Fujimoto et al., 2021; Machado et al., 2023), learning state-action representations (Ota et al., 2020; Fujimoto et al., 2023), action representations (Whitney et al., 2020; Chandak et al., 2019) for improving sample efficiency, etc. We capture latent dynamics information by learning state-action representations, but we differ from previous approaches in that we use them for detecting dynamics mismatch.

3. Preliminaries

We formulate reinforcement learning (RL) problems as a Markov Decision Process (MDP), which can be specified by the 5-tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, P denotes the transition dynamics, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the scalar reward signal, and $\gamma \in [0, 1)$ is the discount factor. The objective of RL is to find a policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ that maximize the discounted cumulative return $\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)$. We consider access to a source domain $\mathcal{M}_{\text{src}} = (\mathcal{S}, \mathcal{A}, P_{\text{src}}, r, \gamma)$ and a target domain $\mathcal{M}_{\text{tar}} = (\mathcal{S}, \mathcal{A}, P_{\text{tar}}, r, \gamma)$ that share the state space and action space, and only differ in their transition dynamics. We assume the rewards are bounded, *i.e.*, $|r(s, a)| \leq r_{\max}, \forall\, s, a$.

In the rest of the paper, we specify the transition dynamics in a domain $\mathcal{M}$ as $P_{\mathcal{M}}$ (*e.g.*, $P_{\mathcal{M}_{\text{src}}}$ is the transition dynamics in the source domain). We denote $\rho^{\pi}_{\mathcal{M}}(s, a) := (1 - \gamma) \sum_{t=0}^{\infty} \gamma^t P^{\pi}_{\mathcal{M},t}(s)\pi(a|s)$ as the normalized probability that a policy π encounters the state action pair (s, a), and $P^{\pi}_{\mathcal{M},t}(s)$ is the probability that the policy π encounters the state s at timestep t in the domain $\mathcal{M}$. The expected return of a policy π in MDP $\mathcal{M}$ can then be simplified as $J_{\mathcal{M}}(\pi) = \mathbb{E}_{s,a\sim\rho^{\pi}_{\mathcal{M}}}[r(s, a)]$.

Notations: $I(X; Y)$ denotes the mutual information between two random variables X, Y. $\mathbb{H}(X)$ is the entropy of the random variable X. Δ is the probability simplex.

4. Dynamics Adaptation by Representation Mismatch

In this section, we start by theoretically unpacking the equivalence between the representation mismatch and the dynamics mismatch. We further show the performance bounds of a policy between the target domain and either online or offline source domain, where representation mismatch serves as the lower bound of the performance difference. Empowered by theoretical results, we leverage the representation mismatch to penalize source domain data and propose our practical algorithm for dynamics-aware policy adaptation.

4.1. Theoretical Analysis

Before moving to our theoretical results, we need to impose the following assumption, which can be generally satisfied in practice (*e.g.*, deep RL). We defer the detailed discussion on the rationality of this assumption to Section 6.3.

Assumption 4.1 (One-to-one Representation Mapping)**.** For any state-action pair (s, a) and its latent representation z, they construct a one-to-one mapping from the original state-action joint space $\mathcal{S} \times \mathcal{A}$ to the latent space $\mathcal{Z}$.

Our first result in Theorem 4.2 establishes a connection between mutual information and the representation deviation of transitions from different domains. Due to space limits, all proofs are deferred to Appendix A.

Theorem 4.2. *For any (s, a), denote its representation as z, and suppose $s'_{\text{src}} \sim P_{\mathcal{M}_{\text{src}}}(\cdot|s, a)$, $s'_{\text{tar}} \sim P_{\mathcal{M}_{\text{tar}}}(\cdot|s, a)$. Denote $h(z; s'_{\text{src}}, s'_{\text{tar}}) = I(z; s'_{\text{tar}}) - I(z; s'_{\text{src}})$, then we have measuring $h(z; s'_{\text{src}}, s'_{\text{tar}})$ is equivalent to measuring the representation deviation $D_{\text{KL}}(P(z|s'_{\text{tar}})\|P(z|s'_{\text{src}}))$.*

Remark. The defined function $h(z; s'_{\text{src}}, s'_{\text{tar}})$ measures the difference between the embedded target domain information and source domain information in z. This theorem illustrates that such a difference is equivalent to the KL-divergence between the distributions of z given source domain state and target domain state, respectively. Intuitively, $h(z; s'_{\text{src}}, s'_{\text{tar}})$ approaches 0 if the distribution of s'_{src} is close to that of s'_{tar}. If we enforce z to contain only target domain knowledge, $h(z; s'_{\text{src}}, s'_{\text{tar}})$ can be large if the dynamics mismatch between data from the two domains is large, incurring a large $D_{\text{KL}}(P(z|s'_{\text{tar}})\|P(z|s'_{\text{src}}))$. Naturally, one may think of using this representation deviation term as evidence of dynamics mismatch.

Below, we show that the representation deviation can *strictly* reflect the dynamics discrepancy between two domains.

Theorem 4.3. *Measuring the representation deviation between the source domain and the target domain is equivalent to measuring the dynamics mismatch between two domains. Formally, we can derive that $D_{\text{KL}}\left(P(z|s'_{\text{tar}})\|P(z|s'_{\text{src}})\right) = D_{\text{KL}}\left(P(s'_{\text{tar}}|z)\|P(s'_{\text{src}}|z)\right) + \mathbb{H}(s'_{\text{tar}}) - \mathbb{H}(s'_{\text{src}})$.*

The above theorem conveys the rationality of detecting dynamics shifts with the aid of the representation mismatch. This is appealing as representations can contain rich information and capture hidden features, and learning in the latent space is effective (Hansen et al., 2022). To see how representation mismatch affects the performance of the agent, we derive a novel performance bound of a policy given online target domain and *online source domain* in Theorem 4.4.

Theorem 4.4 (Online performance bound)**.** *Denote $\mathcal{M}_{\text{src}}$,*

$\mathcal{M}_{\text{tar}}$ as the source domain and the target domain, respectively, then the return difference of any policy π between $\mathcal{M}_{\text{src}}$ and $\mathcal{M}_{\text{tar}}$ is bounded:

$$J_{\mathcal{M}_{\text{tar}}}(\pi) - J_{\mathcal{M}_{\text{src}}}(\pi) \geq - \frac{\sqrt{2}\gamma r_{\max}}{(1-\gamma)^2} \underbrace{\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_{\text{src}}}}\left[\sqrt{D_{\text{KL}}(P(z|s'_{\text{src}})\|P(z|s'_{\text{tar}}))}\right]}_{(a)\text{: representation mismatch}} - \frac{\sqrt{2}\gamma r_{\max}}{(1-\gamma)^2} \underbrace{\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_{\text{src}}}}\left[\sqrt{|\mathbb{H}(s'_{\text{src}}) - \mathbb{H}(s'_{\text{tar}})|}\right]}_{(b)\text{: state distribution deviation}}.$$

Remark. The above bound indicates that the performance difference of a policy π in different domains is decided by the representation mismatch term (a), and the state distribution deviation term (b). Since both two domains are fixed, the entropy of their state distributions are constants, and term (b) is also a constant accordingly. Term (b) characterizes the inherent performance difference of a policy in two domains and vanishes if the two domains are identical.

Moreover, if the source domain is *offline* (*i.e.*, one can only have access to a static offline source domain dataset), we can derive a similar bound as shown below.

Theorem 4.5 (Offline performance bound)**.** *Denote the empirical policy distribution in the offline dataset D from source domain $\mathcal{M}_{\text{src}}$ as $\pi_D := \frac{\sum_D \mathbb{1}(s,a)}{\sum_D \mathbb{1}(s)}$, then the return difference of any policy π between the source domain $\mathcal{M}_{\text{src}}$ and the target domain $\mathcal{M}_{\text{tar}}$ is bounded:*

$$J_{\mathcal{M}_{\text{tar}}}(\pi) - J_{\mathcal{M}_{\text{src}}}(\pi) \geq - \frac{4 r_{\max}}{(1-\gamma)^2} \underbrace{\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\text{src}}}, P_{\mathcal{M}_{\text{src}}}}[D_{\text{TV}}(\pi_D||\pi)]}_{(a)\text{: policy deviation}} - \frac{\sqrt{2}\gamma r_{\max}}{(1-\gamma)^2} \underbrace{\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\text{src}}}}\left[\sqrt{D_{\text{KL}}(P(z|s'_{\text{src}})\|P(z|s'_{\text{tar}}))}\right]}_{(b)\text{: representation mismatch}} - \frac{\sqrt{2}\gamma r_{\max}}{(1-\gamma)^2} \underbrace{\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\text{src}}}}\left[\sqrt{|\mathbb{H}(s'_{\text{src}}) - \mathbb{H}(s'_{\text{tar}})|}\right]}_{(c)\text{: state distribution deviation}}.$$

Remark. This theorem also explicates the importance of the representation mismatch term (b) as a lower bound, similar to Theorem 4.4, but it additionally highlights the role of the policy deviation term (a). It is evident that controlling the policy deviation term counts with an offline source domain.

Theorem 4.4 and 4.5 motivate us to use the representation mismatch term as a reward penalty to encourage dynamics-consistent transitions, because it turns out that the core factor that affects the bound either with an online or offline source domain is the representation mismatch term.

4.2. Practical Algorithm

To acquire representations of the transitions, we train a state encoder $f_\psi(s)$ parameterized by ψ to produce z_1, the representation of the state s, along with a state-action encoder $g_\xi(z, a)$ parameterized by ξ that receives the state representation z_1 and action as inputs and outputs state-action representation z_2. By letting z_2 be close to the representation of the next state, we realize the latent dynamics consistency (Hansen et al., 2022; Ye et al., 2021). The objective function for learning these encoders gives:

$$\mathcal{L}(\psi, \xi) = \mathbb{E}_{(s,a,s')\sim D}\left[(g_\xi(f_\psi(s), a) - \texttt{SG}(f_\psi(s')))^2\right], \tag{1}$$

where D is the replay buffer, and $\texttt{SG}$ denotes stop gradient operator. Similar objectives are adopted in prior works (Ota et al., 2020; Fujimoto et al., 2023). A central difference is, that we only use the representations for measuring representation mismatch, instead of involving them in policy or value function training. One can also utilize a distinct objective, as long as it can embed the latent dynamics information (see Section 6). It is worth noting that both f and g are deterministic to fulfill Assumption 4.1.

Given insights from Theorem 4.2, we deem that the representations ought to embed more information of the target domain, and de-emphasize the source domain knowledge, such that the representation deviations can be a better proxy of dynamics shifts. This prompts us to train the state encoder and the state-action encoder *only in the target domain*, and evaluate the representation deviations upon *samples from the source domain*. We then penalize the source domain rewards with the calculated deviations, *i.e.*, for any transition $(s_{\text{src}}, a_{\text{src}}, r_{\text{src}}, s'_{\text{src}})$ from the source domain, we modify its reward to

$$\hat{r}_{\text{src}} = r_{\text{src}} - \beta \times \left[g_\xi(f_\psi(s_{\text{src}}), a_{\text{src}}) - f_\psi(s'_{\text{src}})\right]^2, \tag{2}$$

where $\beta \in \mathbb{R}$ is a hyperparameter. This penalty generally captures the representation mismatch between the source domain and the target domain. $g_\xi(f_\psi(s_{\text{src}}), a_{\text{src}})$ represents the state-action representation in the target domain since these domains share state space and action space, and f, g only encode target domain information. It approaches the representation of s'_{tar} that is incurred by $(s_{\text{src}}, a_{\text{src}})$ in the target domain. $f_\psi(s'_{\text{src}})$, instead, denotes the representation of s'_{src} from the source domain. A larger penalty will be allocated if the source domain data deviates too much from the dynamics of the target domain, and vice versa. Consequently, the agent can focus more on dynamics-consistent transitions and achieve better performance. Hence, such a penalty matches our theoretical results.

Formally, we introduce our novel method for cross-domain policy adaptation, **P**olicy **A**daptation by **R**epresentation Mismatch, tagged **PAR** algorithm. We use SAC (Haarnoja et al.,

2018) as the base algorithm, and aim at training value functions (*a.k.a.*, critics) $Q_{\theta_1}(s,a), Q_{\theta_2}(s,a)$ parameterized by θ_1, θ_2, and policy (*a.k.a.*, actor) $\pi_\phi(s)$ parameterized by ϕ. Denote $D_{\text{src}}, D_{\text{tar}}$ as the replay buffers of the source domain and the target domain, and let the rewards in D_{src} be corrected as $\hat{r}_{\text{src}}$, then the objective function for training the value functions gives:

$$\mathcal{L}_{\text{critic}} = \mathbb{E}_{(s,a,r,s')\sim D_{\text{src}}\cup D_{\text{tar}}}\left[(Q_{\theta_i}(s,a) - y)^2\right], \quad (3)$$

where $i \in \{1,2\}$ and y is the target value, which gives

$$y = r + \gamma\left[\min_{i=1,2} Q_{\theta'_i}(s',a') - \alpha \log \pi_\phi(a'|s')\right], \quad (4)$$

where $\theta'_i, i \in \{1,2\}$ are the parameters of the target networks, $\alpha \in \mathbb{R}_+$, and $a' \sim \pi_\phi(\cdot|s')$. The way PAR uses to update its policy depends on the condition of the source domain, *e.g.*, it can be either online or offline. We consider both conditions and discuss them below.

Online PAR. If the source domain is online, then the policy objective function gives:

$$\mathcal{L}^{\text{on}}_{\text{actor}} = \mathbb{E}_{\substack{s\sim D_{\text{src}}\cup D_{\text{tar}} \\ a\sim\pi_\phi(\cdot|s)}}\left[\min_{i=1,2} Q_{\theta_i}(s,a) - \alpha \log \pi_\phi(\cdot|s)\right]. \quad (5)$$

Offline PAR. Given an offline source domain, the deviation between the learned policy and the source domain behavior policy $\pi_{D_{\text{src}}}$ ought to be considered based on Theorem 4.5. We then incorporate a behavior cloning term into the objective function of the policy, similar to Fujimoto & Gu (2021). This term injects conservatism into policy learning on a fixed dataset and is necessary to mitigate the extrapolation error (Fujimoto et al., 2019), a challenge that is widely studied in offline RL (Levine et al., 2020; Kumar et al., 2020; Lyu et al., 2022c;a; Kostrikov et al., 2022). The policy objective function then yields:

$$\mathcal{L}^{\text{off}}_{\text{actor}} = \mathbb{E}_{\substack{(s,a)\sim D_{\text{src}} \\ \tilde{a}\sim\pi_\phi(\cdot|s)}}\left[-(a-\tilde{a})^2\right] + \lambda \times \mathcal{L}^{\text{on}}_{\text{actor}}, \quad (6)$$

where $\lambda = \nu / \frac{1}{N}\sum_{(s_j,a_j)} \min_{i=1,2} Q_{\theta_i}(s_j,a_j)$ is the normalization term that balances behavior cloning and maximizing the value function, $\nu \in \mathbb{R}_+$ is a hyperparameter, $\mathcal{L}^{\text{on}}_{\text{actor}}$ is the policy objective of online PAR in Equation 5. The behavior cloning term ensures that the learned policy is close to the data-collecting policy of the source domain dataset. We summarize in Algorithm 1 the abstracted pseudocode of PAR, and defer the full pseudocodes to Appendix C.

Algorithm 1 PAR (Abstracted Version)

Input: Source domain $\mathcal{M}_{\text{src}}$, target domain $\mathcal{M}_{\text{tar}}$, target domain interaction interval F, batch size N

Initialize policy π_ϕ, value functions $\{Q_{\theta_i}\}_{i=1,2}$ and target networks $\{Q_{\theta'_i}\}_{i=1,2}$, replay buffers $\{D_{\text{src}}, D_{\text{tar}}\}$
for i = 1, 2, ... **do**
  (*online*) Collect $(s_{\text{src}}, a_{\text{src}}, r_{\text{src}}, s'_{\text{src}})$ in $\mathcal{M}_{\text{src}}$ and store it, $D_{\text{src}} \leftarrow D_{\text{src}} \cup \{(s_{\text{src}}, a_{\text{src}}, r_{\text{src}}, s'_{\text{src}})\}$
  if $i\%\ F == 0$ **then**
    Interact with $\mathcal{M}_{\text{tar}}$ and get $(s_{\text{tar}}, a_{\text{tar}}, r_{\text{tar}}, s'_{\text{tar}})$. $D_{\text{tar}} \leftarrow D_{\text{tar}} \cup \{(s_{\text{tar}}, a_{\text{tar}}, r_{\text{tar}}, s'_{\text{tar}})\}$
  end if
  Sample N transitions from D_{tar}
  Train encoders in the target domain via Equation 1
  Sample N transitions from D_{src}
  Modify source domain rewards with Equation 2
  Update critics by minimizing Equation 3
  (*online*) Update actor by maximizing Equation 5
  (*offline*) Update actor by maximizing Equation 6
  Update target networks
end for

5. Experiments

In this section, we examine the effectiveness of our proposed method by conducting experiments on environments with kinematic and morphology discrepancies. We also extensively investigate the performance of our method under the offline source domain and different qualities of the offline datasets. Moreover, we empirically analyze the influence of the important hyperparameters in PAR.

5.1. Results with Online Source Domain

For the empirical evaluation of policy adaptation capabilities, we use four environments (*halfcheetah*, *hopper*, *walker*, *ant*) from OpenAI Gym (Brockman et al., 2016) as source domains and modify their dynamics following (Xu et al., 2023) to serve as target domains. The modifications include kinematic and morphology shifts, where we simulate broken joints of the robot by limiting the rotation angle of its joints (*i.e.*, kinematic shifts), and we clip the size of some limbs of the simulated robot to realize morphology shifts. Please see details of the environment setting in Appendix D.1.

We compare PAR against the following baselines: **SAC-tar** (Haarnoja et al., 2018), which trains the SAC agent merely in the target domain for 10^5 environmental steps; **DARC** (Eysenbach et al., 2021), which trains domain classifiers to estimate the dynamics discrepancy and leverage it to correct source domain rewards; **DARC-weight**, a variant of DARC that adopts the dynamics discrepancy term as importance sampling weights when updating critics; **VGDF** (Xu et al., 2023), a recent state-of-the-art method that filters transitions in the source domain that share similar value estimates as those in the target domain; **SAC-tune**, which trains the SAC agent in the source domain for 1M steps and fine-tunes it in the target domain with 10^5 transitions. For online experiments, we allow all algorithms to interact 1M environmental steps with the source domain, but only 10^5

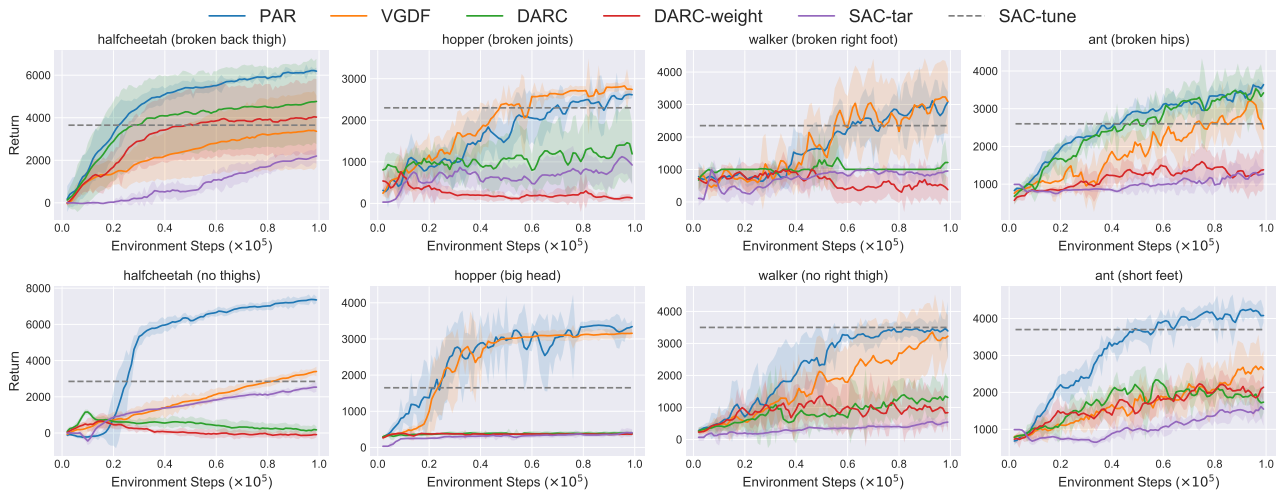


Figure 2. **Adaptation performance comparison when the source domain is online.** The curves depict the test performance of each algorithm *in the target domain* under kinematic shifts (*top*) and morphology shifts (*bottom*). The modification to the environment is specified in the parentheses of the task name. The solid lines are the average returns over 5 different random seeds and the shaded region captures the standard deviation. The dashed line of SAC-tune denotes its final performance after fine-tuning 10^5 steps.

steps in the target domain (*i.e.*, the target domain interaction interval $F = 10$). All algorithms are run with five random seeds. We defer implementation details to Appendix D.2.

We summarize the comparison results in Figure 2. Note that the evaluated environments are quite challenging, and baselines like DARC struggle for a good performance. Based on the curves, PAR outperforms SAC-tar on all of the tasks, indicating that our method successfully boosts the performance of the agent in the target domain by extracting useful knowledge from sufficient source domain data. Notably, PAR achieves the best performance on **6** out of 8 tasks, often surpassing baselines by a large margin. On the rest of the two tasks, PAR is able to achieve competitive performance against VGDF. PAR achieves **2x** sample efficiency compared to the best baseline method on tasks like *halfcheetah (no thighs)*, *ant (short feet)*, etc. Furthermore, PAR beats the fine-tuning method SAC-tune on **7** out of 8 tasks. These altogether illustrate the advantages of our method.

5.2. Evaluations under Offline Source Domain

There exist some circumstances where no real-time interaction with the source domain is available, but we have a previously gathered source domain dataset. We then investigate how our method behaves under this setting, and how the quality of the dataset affects the performance. To that end, we adopt datasets of the four environments (*halfcheetah*, *hopper*, *walker*, *ant*) from D4RL (Fu et al., 2020) "-v2" datasets with three quality levels (medium, medium-replay, medium-expert). This induces a total of **24** tasks.

We consider four baselines for comparison: **CQL-0** (Kumar et al., 2020), which trains a CQL agent solely in the source offline dataset and then directly deploys the learned policy in the target domain in a zero-shot manner; **CQL+SAC**, which updates the offline source domain data with the CQL loss function, while the online target domain data with the SAC loss; **H2O** (Niu et al., 2022), which trains domain classifiers to estimate the dynamics gap and use it as an importance sampling weight for the bellman error of data from the source domain dataset; **VGDF+BC** (Xu et al., 2023), which incorporates an additional behavior cloning term in vanilla VGDF, similar to PAR. All algorithms have a limited budget of 10^5 interactions with the target domain. The implementation details can be found in Appendix D.2.

We present the comparison results in Table 1. We observe that PAR also achieves superior performance given offline source domain datasets, surpassing baseline methods on **17** out of 24 tasks. It is worth mentioning that PAR is the only method that obtains meaningful performance on *halfcheetah (no thighs)* with medium-expert dataset, which is approximately **4x** the performance of the strongest baseline. PAR is also the only method that can generally gain better performance on many tasks with higher quality datasets, *e.g.*, despite that PAR has unsatisfying performance on *hopper (big head)* under medium-level source domain dataset, its performance given the medium-expert source domain dataset is good. Nevertheless, methods like VGDF and H2O have worse performance given medium-expert datasets compared to medium-replay or medium datasets. These collectively show the superiority of PAR and shed light on capturing representation mismatch for cross-domain policy adaptation.

Table 1. **Performance comparison when the source domain is offline**, *i.e.*, only static source domain datasets are available. We report the mean return in conjunction with standard deviation in the target domain under different dataset qualities of the source domain data (medium, medium-replay, medium-expert). The results are averaged over 5 varied random seeds. We **bold** and highlight the best cell.

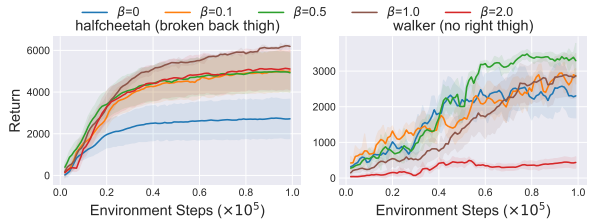
Dataset Type	Task Name	CQL-0	CQL+SAC	H2O	VGDF+BC	PAR (ours)
medium	halfcheetah (broken back thigh)	1128±156	3967±204	5450±194	4834±250	**5686**±603
medium	halfcheetah (no thighs)	361±29	1184±211	2863±209	3910±160	**5768**±117
medium	hopper (broken joints)	155±19	498±73	2467±323	2785±75	**2825**±112
medium	hopper (big head)	399±5	496±53	1451±480	**3060**±60	1450±143
medium	walker (broken right foot)	1453±412	1877±1040	3309±418	3000±388	**3683**±211
medium	walker (no right thigh)	975±131	1262±363	2225±546	**3293**±306	2899±841
medium	ant (broken hips)	1230±99	-1814±431	2704±253	1713±366	**3324**±72
medium	ant (short feet)	1839±137	-807±255	3892±85	3120±469	**4886**±97
medium-replay	halfcheetah (broken back thigh)	655±226	3868±295	5103±35	**5398**±360	5227±445
medium-replay	halfcheetah (no thighs)	398±63	575±619	3225±66	4271±162	**5161**±46
medium-replay	hopper (broken joints)	1018±6	686±60	2325±193	2242±1057	**2376**±777
medium-replay	hopper (big head)	365±7	556±222	**1854**±647	566±90	1336±419
medium-replay	walker (broken right foot)	156±175	1018±22	**3536**±431	2901±1101	3128±1084
medium-replay	walker (no right thigh)	337±189	1465±696	**4254**±207	2057±921	1249±706
medium-replay	ant (broken hips)	882±28	-1609±425	2497±190	2437±286	**2977**±186
medium-replay	ant (short feet)	1294±191	-1369±476	3782±382	4493±82	**4791**±102
medium-expert	halfcheetah (broken back thigh)	843±510	**4283**±180	4100±211	3580±1801	3741±378
medium-expert	halfcheetah (no thighs)	322±81	1669±439	1938±473	2740±297	**10517**±476
medium-expert	hopper (broken joints)	458±441	1147±595	2587±252	2144±938	**2838**±339
medium-expert	hopper (big head)	460±50	547±96	1156±574	2155±1182	**2676**±585
medium-expert	walker (broken right foot)	813±459	2431±782	2254±710	1540±926	**4211**±196
medium-expert	walker (no right thigh)	698±194	1547±346	2835±826	2047±1100	**4006**±1070
medium-expert	ant (broken hips)	321±373	304±1458	2178±799	1868±321	**3113**±501
medium-expert	ant (short feet)	1816±224	-812±105	3511±441	1821±516	**4902**±34

5.3. Parameter Study

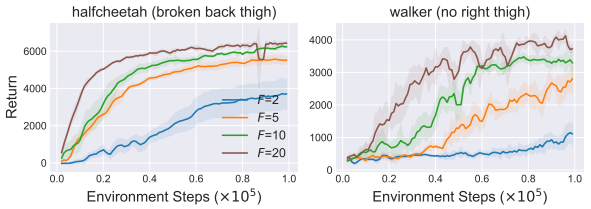
Now we investigate the influence of two critical hyperparameters in PAR, reward penalty coefficient β, and target domain interaction interval F. Considering the page limit, please check more experimental results in Appendix E.

Penalty coefficient β. β controls the scale of the measured representation mismatch. Intuitively, the agent will struggle for good performance if β is too large, and may fail to distinguish source domain samples with inconsistent dynamics if β is too small. To examine its impact, we conduct experiments on two tasks with online source domains, *halfcheetah (broken back thigh)* and *walker (no right thigh)*. We evaluate PAR across $\beta \in \{0, 0.1, 0.5, 1.0, 2.0\}$, and show the results in Figure 3(a). We find that setting $\beta = 0$ (*i.e.*, no representation mismatch penalty) usually incurs a worse final performance, especially on the *halfcheetah* task, verifying the necessity of the reward modification term. Figure 3(a) also illustrates that the optimal β can be task-dependent. We believe this is because different tasks have distinct inherent structures like rewards and state spaces. PAR exhibits some robustness to β, despite that employing a large β may incur a performance drop on some tasks, *e.g.*, on *walker* task.

Target domain interaction interval F. F decides how frequently the agent interacts with the target domain. Following Section 5.1, only 10^5 interactions with the target domain are permitted. We employ $F \in \{2, 5, 10, 20\}$, and summarize the results in Figure 3(b), which show that PAR



(a) Penalty coefficient β.



(b) Target domain interaction interval F.

Figure 3. **Parameter study of (a) reward penalty coefficient β, (b) target domain interaction interval F.** Results are averaged over 5 seeds and the shaded region denotes the standard deviation.

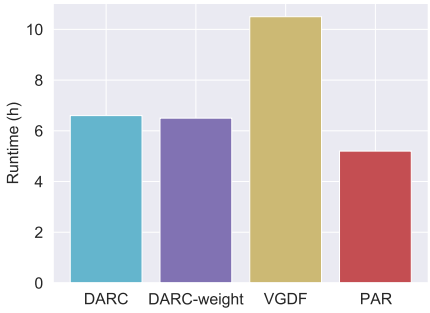


Figure 4. Runtime comparison of different methods.

generally benefits from more source domain data incurred by a larger F, indicating that PAR can exploit dynamics-consistent transitions and realize efficient policy adaptation to another domain. We simply use $F = 10$ by default.

5.4. Runtime Comparison

Furthermore, we compare the runtime of PAR against baselines. All methods are run on the *halfcheetah (broken back thigh)* task on a single GPU. The results in Figure 4 show that PAR is highly efficient in runtime thanks to training in the latent space with one state encoder and one state-action encoder. DARC and its variant have slightly larger training costs. VGDF consumes the most training time because it trains an ensemble of dynamics models in the original state space following model-based RL (Janner et al., 2019).

6. Discussions

In this section, we provide discussions on whether the performance of PAR can be largely affected if we use another representation learning objective, and why PAR beats DARC. We also explain the validity of the assumption. We believe these make a better understanding of our method.

6.1. PAR with a Different Objective

We investigate how PAR behaves with a varying representation learning objective against Equation 1. Such an objective needs to learn the latent dynamics information as well. To that end, we consider the following objective where g now receives true state s (instead of its representation) and action a as inputs, and no stop gradient operator is required:

$$\mathcal{L}'(\psi, \xi) = \mathbb{E}_{(s,a,s')\sim D}\left[(g_\xi(s,a) - f_\psi(s'))^2\right]. \quad (7)$$

Importantly, both f and g are optimized with this objective. Equation 7 also guarantees latent dynamics consistency. We tag this variant as PAR-B. To see how PAR-B competes against vanilla PAR, we conduct experiments on four tasks with kinematic and morphology mismatch. We report their final mean performance in the target domain in Figure 5, where only marginal return difference is observed between PAR and PAR-B, implying that another objective can also be valid as long as it embeds latent dynamics information.

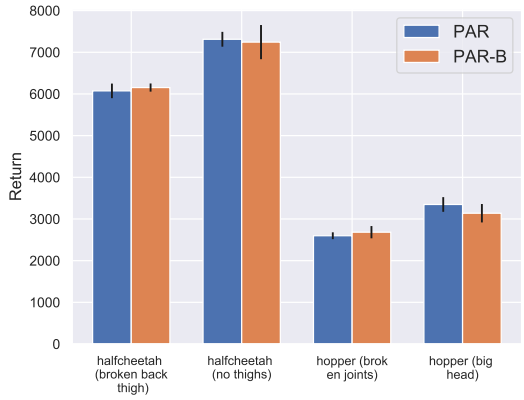


Figure 5. Performance comparison between PAR and PAR-B.

6.2. Why PAR Outperforms DARC?

It is vital to address why PAR significantly outperforms DARC on numerous online tasks given that DARC also corrects source domain rewards (check Figure 2). We stress that DARC learns *domain classifiers* by leveraging both source domain data and target domain data and estimates the dynamics gap, which can be interpreted as how likely the measured source domain transition belongs to the target domain. However, if the transition deviates far from the target domain, the estimated gap $\log \frac{P_{\mathcal{M}_{\text{tar}}}(s'|s,a)}{P_{\mathcal{M}_{\text{src}}}(s'|s,a)}$ can be large and negatively affect the policy learning, which is similar in spirit to DARC's overly pessimistic issue that is criticized by Xu et al. (2023). PAR, instead, captures representation mismatch by training encoders only with target domain data and evaluating representation deviations upon source domain data. We claim that PAR produces more appropriate reward penalties.

To verify our claim, we log the reward penalties calculated by DARC and PAR, and summarize the results in Figure 6. The reward penalty of PAR is large at first, while it decreases with more interactions, meaning that PAR uncovers more dynamics-consistent samples from the source domain. Note that the penalty by PAR tends to converge to a small number (not 0). However, the penalty from DARC is inconsistent on two tasks, *i.e.*, it approaches 0 on *halfcheetah* task while becomes large on *walker* task. The results clearly indicate that capturing representation mismatch is a better choice.

6.3. On the Rationality of the Assumption

In Assumption 4.1, we assume a one-to-one mapping between $\mathcal{S} \times \mathcal{A}$ and $\mathcal{Z}$. One-to-one mapping mathematically indicates that the mapping is *injective* (not necessarily sur-

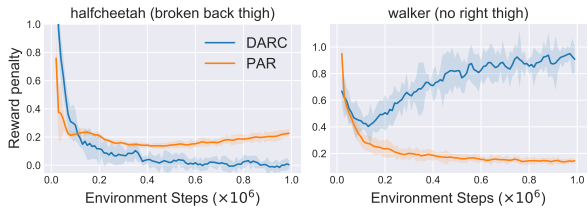


Figure 6. **Reward penalty comparison between DARC and PAR.** We record the average reward penalty across 5 seeds when training each method. The shaded region denotes the standard deviation.

jective). That is, we only require there exists one unique z in the latent space corresponding to the specific (s, a) tuple. To satisfy this assumption, we first employ a deterministic state encoder f and state-action encoder g for representation learning, i.e., f constructs a deterministic mapping from $\mathcal{S}$ to $\mathcal{Z}$ and g is a deterministic mapping from $\mathcal{S} \times \mathcal{A}$ to $\mathcal{Z}$. It remains to decide whether the mapped representation is unique. Note that it is the user's choice of what representation learning approach and which latent representation space to use. One can surely choose a representation method and representation space to let the assumption hold. With our adopted representation learning formula in Equation 1, it is less likely that two distinct (s, a) tuples are mapped into the same latent vector because that indicates they share the same dynamics transition information (since Equation 1 realizes latent dynamics consistency). To further mitigate this concern, the dimension of the state-action representation in PAR is much larger (it is set to be 256 as shown in Table 2 in the appendix) than the input state vector and action vector. We believe these explain the rationality of the assumption.

7. Conclusion and Limitations

In this paper, we study how to effectively adapt policies to another domain with dynamics discrepancy. We propose a novel algorithm, Policy Adaptation by Representation Mismatch (PAR), which captures the representation mismatch between the source domain and the target domain, and employs the resulting representation deviation to compensate source domain rewards. Our method is motivated and supported by rigorous theoretical analysis. Experimental results demonstrate that PAR achieves strong performance and outperforms recent strong baselines under scenarios like kinematic shifts and morphology mismatch, regardless of whether the source domain is online or offline.

Despite the effectiveness of our method, we have to admit that there exist some limitations of our work. First, one may need to decide the best β manually in practice. Second, PAR behaves less satisfyingly given some (not all of them) medium-replay source domain datasets, suggesting that it may be hard for PAR to handle datasets with large diversities. For future work, it is interesting to design mechanisms to adaptively tune β, and enable PAR to consistently acquire a good performance provided datasets with large diversities.

Acknowledgements

This work was supported by the STI 2030-Major Projects under Grant 2021ZD0201404 and the NSFC under Grant 62250068. This work was done when Jiafei Lyu worked as an intern at Tencent IEG. The authors thank Liangpeng Zhang for providing advice on the draft of this work. The authors also would like to thank the anonymous reviewers for their valuable comments on our manuscript.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Arndt, K., Hazara, M., Ghadirzadeh, A., and Kyrki, V. Meta reinforcement learning for sim-to-real domain adaptation. In *IEEE International Conference on Robotics and Automation*, 2019.

Barekatain, M., Yonetani, R., and Hamaya, M. Multipolar: Multi-source policy aggregation for transfer reinforcement learning between diverse environmental dynamics. *arXiv preprint arXiv:1909.13111*, 2019.

Barreto, A., Dabney, W., Munos, R., Hunt, J. J., Schaul, T., Silver, D., and Hasselt, H. V. Successor features for transfer in reinforcement learning. *ArXiv*, abs/1606.05312, 2016.

Bengio, Y., Courville, A. C., and Vincent, P. Representation learning: A review and new perspectives. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35: 1798–1828, 2012.

Bousmalis, K., Irpan, A., Wohlhart, P., Bai, Y., Kelcey, M., Kalakrishnan, M., Downs, L., Ibarz, J., Pastor, P., Konolige, K., Levine, S., and Vanhoucke, V. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. In *IEEE International Conference on Robotics and Automation*, 2018.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. OpenAI Gym. *ArXiv*, abs/1606.01540, 2016.

Buckman, J., Hafner, D., Tucker, G., Brevdo, E., and Lee, H. Sample-Efficient Reinforcement Learning with Stochas-

tic Ensemble Value Expansion. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Cetin, E., Ball, P. J., Roberts, S., and Çeliktutan, O. Stabilizing off-policy deep reinforcement learning from pixels. In *International Conference on Machine Learning*, 2022.

Chandak, Y., Theocharous, G., Kostas, J. E., Jordan, S. M., and Thomas, P. S. Learning action representations for reinforcement learning. In *International Conference on Machine Learning*, 2019.

Chebotar, Y., Handa, A., Makoviychuk, V., Macklin, M., Issac, J., Ratliff, N. D., and Fox, D. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *International Conference on Robotics and Automation*, 2018.

Clavera, I., Nagabandi, A., Fearing, R. S., Abbeel, P., Levine, S., and Finn, C. Learning to adapt: Meta-learning for model-based control. *ArXiv*, abs/1803.11347, 2018.

Collins, J. J., Brown, R., Leitner, J., and Howard, D. Traversing the reality gap via simulator tuning. *ArXiv*, abs/2003.01369, 2020.

Csiszár, I. and Körner, J. *Information theory: coding theorems for discrete memoryless systems*. Cambridge University Press, 2011.

Cutler, M. and How, J. P. Efficient Reinforcement Learning for Robots using Informative Simulated Priors. In *IEEE International Conference on Robotics and Automation*, 2015.

Desai, S., Durugkar, I., Karnan, H., Warnell, G., Hanna, J., and Stone, P. An imitation from observation approach to transfer learning with dynamics mismatch. In *Neural Information Processing Systems*, 2020.

Du, Y., Watkins, O., Darrell, T., Abbeel, P., and Pathak, D. Auto-tuned sim-to-real transfer. In *IEEE International Conference on Robotics and Automation*, 2021.

Eysenbach, B., Chaudhari, S., Asawa, S., Levine, S., and Salakhutdinov, R. Off-dynamics reinforcement learning: Training for transfer with domain classifiers. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=eqBwg3AcIAK.

Eysenbach, B., Zhang, T., Salakhutdinov, R., and Levine, S. Contrastive learning as goal-conditioned reinforcement learning. *ArXiv*, abs/2206.07568, 2022.

Farchy, A., Barrett, S., MacAlpine, P., and Stone, P. Humanoid robots learning to walk faster: from the real world to simulation and back. In *Adaptive Agents and Multi-Agent Systems*, 2013.

Ferns, N., Panangaden, P., and Precup, D. Bisimulation metrics for continuous markov decision processes. *SIAM Journal on Computing*, 40(6):1662–1714, 2011.

Fickinger, A., Cohen, S., Russell, S., and Amos, B. Cross-domain imitation learning via optimal transport. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=xP3cPq2hQC.

Fu, J., Kumar, A., Nachum, O., Tucker, G., and Levine, S. D4RL: Datasets for Deep Data-Driven Reinforcement Learning. *ArXiv*, abs/2004.07219, 2020.

Fujimoto, S. and Gu, S. S. A Minimalist Approach to Offline Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Fujimoto, S., Meger, D., and Precup, D. Off-Policy Deep Reinforcement Learning without Exploration. In *International Conference on Machine Learning (ICML)*, 2019.

Fujimoto, S., Meger, D., and Precup, D. A deep reinforcement learning approach to marginalized importance sampling with the successor representation. In *International Conference on Machine Learning*, 2021.

Fujimoto, S., Chang, W.-D., Smith, E. J., Gu, S. S., Precup, D., and Meger, D. For sale: State-action representation learning for deep reinforcement learning. *ArXiv*, abs/2306.02451, 2023.

Gamrian, S. and Goldberg, Y. Transfer learning for related reinforcement learning tasks via image-to-image translation. In *International Conference on Machine Learning*, 2018.

Ge, Y., Macaluso, A., Li, E. L., Luo, P., and Wang, X. Policy adaptation from foundation model feedback. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022.

Golemo, F., Taïga, A. A., Courville, A. C., and Oudeyer, P.-Y. Sim-to-real transfer with neural-augmented robot simulation. In *Conference on Robot Learning*, 2018.

Gui, H., Pang, S., Yu, S., Qiao, S., Qi, Y., He, X., Wang, M., and Zhai, X. Cross-domain policy adaptation with dynamics alignment. *Neural Networks*, 167:104–117, 2023.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., et al. Soft Actor-Critic Algorithms and Applications. *arXiv preprint arXiv:1812.05905*, 2018.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. Learning latent dynamics for

planning from pixels. In *International conference on machine learning*, 2019.

Hanna, J. P., Desai, S., Karnan, H., Warnell, G. A., and Stone, P. Grounded action transformation for sim-to-real reinforcement learning. *Machine Learning*, 110:2469 – 2499, 2021.

Hansen, N., Jangir, R., Sun, Y., Alenyà, G., Abbeel, P., Efros, A. A., Pinto, L., and Wang, X. Self-supervised policy adaptation during deployment. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=o_V-MjyyGV_.

Hansen, N., Wang, X., and Su, H. Temporal Difference Learning for Model Predictive Control. In *International Conference on Machine Learning*, 2022.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

Hejna, D. J., Abbeel, P., and Pinto, L. Hierarchically decoupled imitation for morphological transfer. *ArXiv*, abs/2003.01709, 2020.

Hwangbo, J., Lee, J., Dosovitskiy, A., Bellicoso, D., Tsounis, V., Koltun, V., and Hutter, M. Learning agile and dynamic motor skills for legged robots. *Science Robotics*, 4, 2019.

Janner, M., Fu, J., Zhang, M., and Levine, S. When to Trust Your Model: Model-Based Policy Optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.

Jiang, Y., Li, C., Dai, W., Zou, J., and Xiong, H. Variance reduced domain randomization for reinforcement learning with policy gradient. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 46:1031–1048, 2023.

Karl, M., Sölch, M., Bayer, J., and van der Smagt, P. Deep variational bayes filters: Unsupervised learning of state space models from raw data. *ArXiv*, abs/1605.06432, 2016.

Kim, K., Gu, Y., Song, J., Zhao, S., and Ermon, S. Domain adaptive imitation learning. In *International Conference on Machine Learning*, 2019.

Kim, K., Ha, J., and Kim, Y. Self-predictive dynamics for generalization of vision-based reinforcement learning. In *International Joint Conference on Artificial Intelligence*, 2022.

Kingma, D. P. and Ba, J. Adam: A Method for Stochastic Optimization. In *International Conference on Learning Representation*, 2015. URL http://arxiv.org/abs/1412.6980.

Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Sallab, A. A. A., Yogamani, S. K., and P'erez, P. Deep reinforcement learning for autonomous driving: A survey. *IEEE Transactions on Intelligent Transportation Systems*, 23: 4909–4926, 2020.

Kober, J., Bagnell, J. A., and Peters, J. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32:1238 – 1274, 2013.

Kolesnikov, A., Beyer, L., Zhai, X., Puigcerver, J., Yung, J., Gelly, S., and Houlsby, N. Big transfer (bit): General visual representation learning. In *European Conference on Computer Vision*, 2019.

Kostrikov, I., Yarats, D., and Fergus, R. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *ArXiv*, abs/2004.13649, 2020.

Kostrikov, I., Nair, A., and Levine, S. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=68n2s9ZJWF8.

Kumar, A., Zhou, A., Tucker, G., and Levine, S. Conservative Q-Learning for Offline Reinforcement Learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Levine, S., Kumar, A., Tucker, G., and Fu, J. Offline Reinforcement Learning: Tutorial, Review, and Perspectives on Open Problems. *ArXiv*, abs/2005.01643, 2020.

Liu, G., Zhang, C., Zhao, L., Qin, T., Zhu, J., Jian, L., Yu, N., and Liu, T.-Y. Return-based contrastive representation learning for reinforcement learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=_TM6rT7tXke.

Liu, J., Hongyin, Z., and Wang, D. DARA: Dynamics-aware reward augmentation in offline reinforcement learning. In *International Conference on Learning Representations*, 2022a. URL https://openreview.net/forum?id=9SDQB3b68K.

Liu, X., Pathak, D., and Kitani, K. M. Revolver: Continuous evolutionary models for robot-to-robot policy transfer. In *International Conference on Machine Learning*, 2022b.

Luo, Y., Xu, H., Li, Y., Tian, Y., Darrell, T., and Ma, T. Algorithmic framework for model-based deep reinforcement learning with theoretical guarantees. In

International Conference on Learning Representations, 2019. URL https://openreview.net/forum?id=BJe1E2R5KX.

Lyu, J., aicheng Gong, Wan, L., Lu, Z., and Li, X. State advantage weighting for offline RL. In *3rd Offline RL Workshop: Offline RL as a "Launchpad"*, 2022a. URL https://openreview.net/forum?id=2rOD_UQfvl.

Lyu, J., Li, X., and Lu, Z. Double check your state before trusting it: Confidence-aware bidirectional offline model-based imagination. In *Advances in Neural Information Processing Systems*, 2022b. URL https://openreview.net/forum?id=3e3IQMLDSLP.

Lyu, J., Ma, X., Li, X., and Lu, Z. Mildly conservative q-learning for offline reinforcement learning. In *Neural Information Processing Systems*, 2022c.

Lyu, J., Wan, L., Lu, Z., and Li, X. Off-policy rl algorithms can be sample-efficient for continuous control via sample multiple reuse. *ArXiv*, abs/2305.18443, 2023.

Machado, M. C., Barreto, A., Precup, D., and Bowling, M. Temporal abstraction in reinforcement learning with the successor representation. *Journal of Machine Learning Research*, 24(80):1–69, 2023.

Malik, A., Kuleshov, V., Song, J., Nemer, D., Seymour, H., and Ermon, S. Calibrated model-based deep reinforcement learning. In *International Conference on Machine Learning*, 2019.

Mehta, B., Diaz, M., Golemo, F., Pal, C. J., and Paull, L. Active domain randomization. *ArXiv*, abs/1904.04762, 2019.

Nagabandi, A., Clavera, I., Liu, S., Fearing, R. S., Abbeel, P., Levine, S., and Finn, C. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. *arXiv preprint arXiv:1803.11347*, 2018.

Niu, H., Sharma, S., Qiu, Y., Li, M., Zhou, G., HU, J., and Zhan, X. When to trust your simulator: Dynamics-aware hybrid offline-and-online reinforcement learning. In *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=zXE8iFOZKw.

Osinski, B., Jakubowski, A., Milos, P., Ziecina, P., Galias, C., Homoceanu, S., and Michalewski, H. Simulation-based reinforcement learning for real-world autonomous driving. In *IEEE International Conference on Robotics and Automation*, 2019.

Ota, K., Oiki, T., Jha, D., Mariyama, T., and Nikovski, D. Can increasing input dimensionality improve deep reinforcement learning? In *International conference on machine learning*, 2020.

Pan, F., He, J., Tu, D., and He, Q. Trust the Model When It Is Confident: Masked Model-based Actor-Critic. In *Advances in Neural Information Processing Systems*, 2020.

Peng, X. B., Andrychowicz, M., Zaremba, W., and Abbeel, P. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

Qiao, Z., Lyu, J., and Li, X. The primacy bias in model-based rl. *ArXiv*, abs/2310.15017, 2023.

Rafailov, R., Yu, T., Rajeswaran, A., and Finn, C. Offline reinforcement learning from images with latent space models. In *Conference on Learning for Dynamics & Control*, 2020.

Raileanu, R., Goldstein, M., and Szlam, A. Fast adaptation to new environments via policy-dynamics value functions. In *International Conference on Machine Learning*, 2020.

Ramos, F. T., Possas, R., and Fox, D. Bayessim: adaptive domain randomization via probabilistic inference for robotics simulators. *ArXiv*, abs/1906.01728, 2019.

Raychaudhuri, D. S., Paul, S., Vanbaar, J., and Roy-Chowdhury, A. K. Cross-domain imitation from observations. In *International Conference on Machine Learning*, 2021.

Rezaei-Shoshtari, S., Zhao, R., Panangaden, P., Meger, D., and Precup, D. Continuous MDP homomorphisms and homomorphic policy gradient. In *Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=Adl-fs-8OzL.

Schwarzer, M., Anand, A., Goel, R., Hjelm, R. D., Courville, A., and Bachman, P. Data-Efficient Reinforcement Learning with Self-Predictive Representations. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=uCQfPZwRaUu.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., van den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T. P., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. Mastering the Game of Go with Deep Neural Networks and Tree Search. *Nature*, 529:484–489, 2016.

Slaoui, R. B., Clements, W. R., Foerster, J. N., and Toth, S. Robust domain randomization for reinforcement learning. *ArXiv*, abs/1910.10537, 2019.

Srinivas, A., Laskin, M., and Abbeel, P. CURL: Contrastive Unsupervised Representations for Reinforcement Learning. In *International Conference on Machine Learning*, 2020.

Stooke, A., Lee, K., Abbeel, P., and Laskin, M. Decoupling representation learning from reinforcement learning. In *International Conference on Machine Learning*, 2021.

Tang, Y., Guo, Z. D., Richemond, P. H., Pires, B. Á., Chandak, Y., Munos, R., Rowland, M., Azar, M. G., Lan, C. L., Lyle, C., Gyorgy, A., Thakoor, S., Dabney, W., Piot, B., Calandriello, D., and Vaĺko, M. Understanding self-predictive learning for reinforcement learning. In *International Conference on Machine Learning*, 2022.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. Domain randomization for transferring deep neural networks from simulation to the real world. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.

van der Pol, E., Worrall, D. E., van Hoof, H., Oliehoek, F. A., and Welling, M. Mdp homomorphic networks: Group symmetries in reinforcement learning. In *Neural Information Processing Systems*, 2020.

Viano, L., ting Huang, Y., Kamalaruban, P., and Cevher, V. Robust inverse reinforcement learning under transition dynamics mismatch. In *Neural Information Processing Systems*, 2020.

Vuong, Q. H., Vikram, S., Su, H., Gao, S., and Christensen, H. I. How to pick the domain randomization parameters for sim-to-real transfer of reinforcement learning policies? *ArXiv*, abs/1903.11774, 2019.

Whitney, W., Agarwal, R., Cho, K., and Gupta, A. Dynamics-aware embeddings. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BJgZGeHFPH.

Wu, Z., Xie, Y., Lian, W., Wang, C., Guo, Y., Chen, J., Schaal, S., and Tomizuka, M. Zero-shot policy transfer with disentangled task representation of meta-reinforcement learning. In *IEEE International Conference on Robotics and Automation*, 2022.

Xie, A., Sodhani, S., Finn, C., Pineau, J., and Zhang, A. Robust policy learning over multiple uncertainty sets. In *International Conference on Machine Learning*, 2022.

Xiong, Z., Beck, J., and Whiteson, S. Universal morphology control via contextual modulation. In *International Conference on Machine Learning*, 2023.

Xu, K., Bai, C., Ma, X., Wang, D., Zhao, B., Wang, Z., Li, X., and Li, W. Cross-domain policy adaptation via value-guided data filtering. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=qdM260dXsa.

Yarats, D., Fergus, R., Lazaric, A., and Pinto, L. Mastering Visual Continuous Control: Improved Data-Augmented Reinforcement Learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=_SJ-_yyes8.

Ye, W., Liu, S., Kurutach, T., Abbeel, P., and Gao, Y. Mastering Atari Games with Limited Data. In *Advances in Neural Information Processing Systems*, 2021.

You, H., Yang, T., Zheng, Y., Hao, J., and E Taylor, M. Cross-domain adaptive transfer reinforcement learning based on state-action correspondence. In *Uncertainty in Artificial Intelligence*, 2022.

Yu, W., Liu, C. K., and Turk, G. Preparing for the unknown: Learning a universal policy with online system identification. *ArXiv*, abs/1702.02453, 2017.

Zhang, A., McAllister, R. T., Calandra, R., Gal, Y., and Levine, S. Learning invariant representations for reinforcement learning without reconstruction. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=-2FCwDKRREu.

Zhang, G., Zhong, L., Lee, Y., and Lim, J. J. Policy transfer across visual and dynamics domain gaps via iterative grounding. *ArXiv*, abs/2107.00339, 2021b.

Zhang, Q., Xiao, T., Efros, A. A., Pinto, L., and Wang, X. Learning cross-domain correspondence for control with dynamics cycle-consistency. In *International Conference on Learning Representations*, 2021c. URL https://openreview.net/forum?id=QIRlze3I6hX.

Zhou, W., Pinto, L., and Gupta, A. Environment probing interaction policies. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=ryl8-3AcFX.

Zhu, J., Xia, Y., Wu, L., Deng, J., gang Zhou, W., Qin, T., and Li, H. Masked contrastive representation learning for reinforcement learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45:3421–3433, 2020.

Zhu, S., Kimmel, A., Bekris, K. E., and Boularias, A. Fast model identification via physics engines for data-efficient policy search. In *International Joint Conference on Artificial Intelligence*, 2017.

A. Missing Proofs

In this section, we formally present all the missing proofs from the main text. For better readability, we restate theorems in the appendix. We also need some lemmas, which can be found in Appendix B.

A.1. Proof of Theorem 4.2

Theorem A.1. *For any (s, a), denote its representation as z, and suppose $s'_{\text{src}} \sim P_{\mathcal{M}_{\text{src}}}(\cdot|s,a)$, $s'_{\text{tar}} \sim P_{\mathcal{M}_{\text{tar}}}(\cdot|s,a)$. Denote $h(z; s'_{\text{src}}, s'_{\text{tar}}) = I(z; s'_{\text{tar}}) - I(z; s'_{\text{src}})$, then we have measuring $h(z; s'_{\text{src}}, s'_{\text{tar}})$ is equivalent to measuring the representation deviation $D_{\text{KL}}(P(z|s'_{\text{tar}})\|P(z|s'_{\text{src}}))$.*

Proof. By the definition of mutual information, we have

$$
\begin{aligned}
h(z; s'_{\text{src}}, s'_{\text{tar}}) &= I(z; s'_{\text{tar}}) - I(z; s'_{\text{src}}) \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z, s'_{\text{tar}}) \log \frac{P(z, s'_{\text{tar}})}{P(z)P(s'_{\text{tar}})} dz ds'_{\text{tar}} - \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z, s'_{\text{src}}) \log \frac{P(z, s'_{\text{src}})}{P(z)P(s'_{\text{src}})} dz ds'_{\text{src}} \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z, s'_{\text{tar}}) \log \frac{P(z|s'_{\text{tar}})}{P(z)} dz ds'_{\text{tar}} - \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z, s'_{\text{src}}) \log \frac{P(z|s'_{\text{src}})}{P(z)} dz ds'_{\text{src}} \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}}\int_{\mathcal{S}} P(z, s'_{\text{tar}}, s'_{\text{src}}) \log \frac{P(z|s'_{\text{tar}})}{P(z)} dz ds'_{\text{tar}} ds'_{\text{src}} - \int_{\mathcal{Z}}\int_{\mathcal{S}}\int_{\mathcal{S}} P(z, s'_{\text{src}}, s'_{\text{tar}}) \log \frac{P(z|s'_{\text{src}})}{P(z)} dz ds'_{\text{src}} ds'_{\text{tar}} \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}}\int_{\mathcal{S}} P(z, s'_{\text{tar}}, s'_{\text{src}}) \log \frac{P(z|s'_{\text{tar}})}{P(z|s'_{\text{src}})} dz ds'_{\text{tar}} ds'_{\text{src}} \\
&= D_{\text{KL}}\left(P(z|s'_{\text{tar}})\|P(z|s'_{\text{src}})\right). \qquad \text{(By the definition of Kullback–Leibler divergence)}
\end{aligned}
$$

We then can conclude that measuring the defined function $h(z; s'_{\text{src}}, s'_{\text{tar}})$ is equivalent to measuring the KL-divergence between $P(z|s'_{\text{tar}})$ and $P(z|s'_{\text{src}})$, which is the deviation of representations given the source domain state and target domain state, respectively. Note that the definition of the KL-divergence already involves expectations over s'_{src} and s'_{tar}. While one can also write $\mathbb{E}_{s'_{\text{src}}, s'_{\text{tar}}}[D_{\text{KL}}\left(P(z|s'_{\text{tar}})\|P(z|s'_{\text{src}})\right)]$ and it should not affect the result. □

A.2. Proof of Theorem 4.3

Theorem A.2. *Measuring the representation deviation between the source domain and the target domain is equivalent to measuring the dynamics mismatch between two domains. Formally, we can derive that $D_{\text{KL}}\left(P(z|s'_{\text{tar}})\|P(z|s'_{\text{src}})\right) = D_{\text{KL}}\left(P(s'_{\text{tar}}|z)\|P(s'_{\text{src}}|z)\right) + \mathbb{H}(s'_{\text{tar}}) - \mathbb{H}(s'_{\text{src}})$.*

Proof. We would like to establish a connection between the representation deviations in the two domains and the dynamics discrepancies between the two domains. We achieve this by rewriting the defined function $h(z; s'_{\text{src}}, s'_{\text{tar}})$ as follows,

$$
\begin{aligned}
h(z; s'_{\text{src}}, s'_{\text{tar}}) &= I(z; s'_{\text{tar}}) - I(z; s'_{\text{src}}) \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z, s'_{\text{tar}}) \log \frac{P(z, s'_{\text{tar}})}{P(z)P(s'_{\text{tar}})} dz ds'_{\text{tar}} - \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z, s'_{\text{src}}) \log \frac{P(z, s'_{\text{src}})}{P(z)P(s'_{\text{src}})} dz ds'_{\text{src}} \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z, s'_{\text{tar}}) \log \frac{P(s'_{\text{tar}}|z)}{P(s'_{\text{tar}})} dz ds'_{\text{tar}} - \int_{\mathcal{Z}}\int_{\mathcal{S}} P(z, s'_{\text{src}}) \log \frac{P(s'_{\text{src}}|z)}{P(s'_{\text{src}})} dz ds'_{\text{src}} \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}}\int_{\mathcal{S}} P(z, s'_{\text{tar}}, s'_{\text{src}}) \log \frac{P(s'_{\text{tar}}|z)}{P(s'_{\text{tar}})} dz ds'_{\text{tar}} ds'_{\text{src}} - \int_{\mathcal{Z}}\int_{\mathcal{S}}\int_{\mathcal{S}} P(z, s'_{\text{src}}, s'_{\text{tar}}) \log \frac{P(s'_{\text{src}}|z)}{P(s'_{\text{src}})} dz ds'_{\text{src}} ds'_{\text{tar}} \\
&= \int_{\mathcal{Z}}\int_{\mathcal{S}}\int_{\mathcal{S}} P(z, s'_{\text{tar}}, s'_{\text{src}}) \log \frac{P(s'_{\text{tar}}|z)}{P(s'_{\text{src}}|z)} dz ds'_{\text{tar}} ds'_{\text{src}} - \int_{\mathcal{S}} P(s'_{\text{tar}}) \log P(s'_{\text{tar}}) ds'_{\text{tar}} \\
&\qquad + \int_{\mathcal{S}} P(s'_{\text{src}}) \log P(s'_{\text{src}}) ds'_{\text{src}} \\
&= D_{\text{KL}}\left(P(s'_{\text{tar}}|z)\|P(s'_{\text{src}}|z)\right) + \mathbb{H}(s'_{\text{tar}}) - \mathbb{H}(s'_{\text{src}}).
\end{aligned}
$$

One can see that the defined function is also connected to the dynamics discrepancy term $D_{\text{KL}}(P(s'_{\text{tar}})|z)\|P(s'_{\text{src}}|z)$. It also correlates to two entropy terms. Nevertheless, we observe that the source domain and the target domain are specified

and fixed, and their state distributions are also fixed, indicating that the entropy terms are constants. Then by using the conclusion from Theorem 4.2, we have

$$\underbrace{D_{\mathrm{KL}}\left(P(z|s'_{\mathrm{tar}})\|P(z|s'_{\mathrm{src}})\right)}_{\text{representation deviation}} = \underbrace{D_{\mathrm{KL}}\left(P(s'_{\mathrm{tar}}|z)\|P(s'_{\mathrm{src}}|z)\right)}_{\text{dynamics deviation}} + \underbrace{\mathbb{H}(s'_{\mathrm{tar}}) - \mathbb{H}(s'_{\mathrm{src}})}_{\text{constants}}. \tag{8}$$

Hence, we conclude that measuring representation deviations between two domains is equivalent to measuring the dynamics mismatch. □

A.3. Proof of Theorem 4.4

Theorem A.3 (Online performance bound)**.** *Denote $\mathcal{M}_{\mathrm{src}}$, $\mathcal{M}_{\mathrm{tar}}$ as the source domain and the target domain, respectively, then the return difference of any policy π between $\mathcal{M}_{\mathrm{src}}$ and $\mathcal{M}_{\mathrm{tar}}$ is bounded:*

$$J_{\mathcal{M}_{\mathrm{tar}}}(\pi) - J_{\mathcal{M}_{\mathrm{src}}}(\pi) \geq -\frac{\sqrt{2}\gamma r_{\max}}{(1-\gamma)^2}\underbrace{\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_{\mathrm{src}}}}\left[\sqrt{D_{\mathrm{KL}}(P(z|s'_{\mathrm{src}})\|P(z|s'_{\mathrm{tar}}))}\right]}_{(a)\text{:representation mismatch}} - \frac{\sqrt{2}\gamma r_{\max}}{(1-\gamma)^2}\underbrace{\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_{\mathrm{src}}}}\left[\sqrt{|\mathbb{H}(s'_{\mathrm{src}}) - \mathbb{H}(s'_{\mathrm{tar}})|}\right]}_{(b)\text{:state distribution deviation}}.$$

Proof. To show this theorem, we reiterate the Assumption 4.1 we made in the main text, *i.e.*, the state-action pair (s, a) and its corresponding representation z are a one-to-one mapping from the original space $\mathcal{S} \times \mathcal{A}$ to the latent space $\mathcal{Z}$. This indicates that we could construct a pseudo probability distribution given the representation z that is the same as the transition dynamics probability in the system, *i.e.*, $P(s'_{\mathrm{src}}|z) = P(s'_{\mathrm{src}}|s,a) = P_{\mathcal{M}_{\mathrm{src}}}(\cdot|s,a), P(s'_{\mathrm{tar}}|z) = P(s'_{\mathrm{tar}}|s,a) = P_{\mathcal{M}_{\mathrm{tar}}}(\cdot|s,a), \forall\, s, a$.

Recall that the value function $V(s)$ estimates the expected return given the state s, and state-action value function $Q(s,a)$ estimates the expected return given the state s and action a. Since the rewards are bounded, we have $|V(s)| \leq \frac{r_{\max}}{1-\gamma}, |Q(s,a)| \leq \frac{r_{\max}}{1-\gamma}, \forall\, s, a$. We denote value function under policy π and MDP $\mathcal{M}$ as $V^{\pi}_{\mathcal{M}}(s), Q^{\pi}_{\mathcal{M}}(s,a)$, respectively.

By using Lemma B.1, we have

$$\begin{aligned}
J_{\mathcal{M}_{\mathrm{src}}}(\pi) - J_{\mathcal{M}_{\mathrm{tar}}}(\pi) &= \frac{\gamma}{1-\gamma}\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_{\mathrm{src}}}(s,a)}\left[\int_{s'} P_{\mathcal{M}_{\mathrm{src}}}(s'|s,a)V^{\pi}_{\mathcal{M}_{\mathrm{tar}}}(s')ds' - \int_{s'} P_{\mathcal{M}_{\mathrm{tar}}}(s'|s,a)V^{\pi}_{\mathcal{M}_{\mathrm{tar}}}(s')ds'\right] \\
&= \frac{\gamma}{1-\gamma}\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_{\mathrm{src}}}(s,a)}\left[\int_{s'} (P_{\mathcal{M}_{\mathrm{src}}}(s'|s,a) - P_{\mathcal{M}_{\mathrm{tar}}}(s'|s,a))V^{\pi}_{\mathcal{M}_{\mathrm{tar}}}(s')ds'\right] \\
&\leq \frac{\gamma}{1-\gamma}\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_{\mathrm{src}}}(s,a)}\left|\int_{s'} (P_{\mathcal{M}_{\mathrm{src}}}(s'|s,a) - P_{\mathcal{M}_{\mathrm{tar}}}(s'|s,a))V^{\pi}_{\mathcal{M}_{\mathrm{tar}}}(s')ds'\right| \\
&\leq \frac{\gamma}{1-\gamma}\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_{\mathrm{src}}}(s,a)}\left[\int_{s'} |P_{\mathcal{M}_{\mathrm{src}}}(s'|s,a) - P_{\mathcal{M}_{\mathrm{tar}}}(s'|s,a)| \times \left|V^{\pi}_{\mathcal{M}_{\mathrm{tar}}}(s')\right| ds'\right] \\
&\leq \frac{\gamma r_{\max}}{(1-\gamma)^2}\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_{\mathrm{src}}}(s,a)}\left[\int_{s'} |P_{\mathcal{M}_{\mathrm{src}}}(s'|s,a) - P_{\mathcal{M}_{\mathrm{tar}}}(s'|s,a)|\, ds'\right] \\
&= \frac{2\gamma r_{\max}}{(1-\gamma)^2}\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_{\mathrm{src}}}(s,a)}\left[D_{\mathrm{TV}}\left(P_{\mathcal{M}_{\mathrm{src}}}(s'|s,a)\|P_{\mathcal{M}_{\mathrm{tar}}}(s'|s,a)\right)\right] \\
&= \frac{2\gamma r_{\max}}{(1-\gamma)^2}\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_{\mathrm{src}}}(s,a)}\left[D_{\mathrm{TV}}\left(P(s'_{\mathrm{src}}|z)\|P(s'_{\mathrm{tar}}|z)\right)\right] \\
&\leq \frac{2\gamma r_{\max}}{(1-\gamma)^2}\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_{\mathrm{src}}}(s,a)}\left[\sqrt{\frac{1}{2}D_{\mathrm{KL}}\left(P(s'_{\mathrm{src}}|z)\|P(s'_{\mathrm{tar}}|z)\right)}\right] \qquad (i) \\
&\leq \frac{\sqrt{2}\gamma r_{\max}}{(1-\gamma)^2}\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_{\mathrm{src}}}(s,a)}\left[\sqrt{D_{\mathrm{KL}}\left(P(z|s'_{\mathrm{src}})\|P(z|s'_{\mathrm{tar}})\right)}\right] \\
&\quad + \frac{\sqrt{2}\gamma r_{\max}}{(1-\gamma)^2}\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_{\mathrm{src}}}(s,a)}\left[\sqrt{|\mathbb{H}(s'_{\mathrm{src}}) - \mathbb{H}(s'_{\mathrm{tar}})}\right],
\end{aligned}$$

where $D_{\mathrm{TV}}(p,q)$ denotes the total variation distance between two distribution p and q, the inequality (i) is due to the Pinsker's inequality (Csiszár & Körner, 2011), and the last step is by using Equation 8 and the triangle inequality. Then we conclude the proof. □

A.4. Proof of Theorem 4.5

Theorem A.4 (Offline performance bound)**.** *Denote the empirical policy distribution in the offline dataset D from source domain $\mathcal{M}_{\rm src}$ as $\pi_D := \frac{\sum_D \mathbb{1}(s,a)}{\sum_D \mathbb{1}(s)}$, then the return difference of any policy π between the source domain $\mathcal{M}_{\rm src}$ and the target domain $\mathcal{M}_{\rm tar}$ is bounded:*

$$J_{\mathcal{M}_{\rm tar}}(\pi) - J_{\mathcal{M}_{\rm src}}(\pi) \geq -\frac{4r_{\max}}{(1-\gamma)^2} \underbrace{\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\rm src}}, P_{\mathcal{M}_{\rm src}}}[D_{\rm TV}(\pi_D || \pi)]}_{(a):\text{policy deviation}} - \frac{\sqrt{2}\gamma r_{\max}}{(1-\gamma)^2} \underbrace{\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\rm src}}} \left[\sqrt{D_{\rm KL}(P(z|s'_{\rm src}) \| P(z|s'_{\rm tar}))}\right]}_{(b):\text{representation mismatch}}$$
$$- \frac{\sqrt{2}\gamma r_{\max}}{(1-\gamma)^2} \underbrace{\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\rm src}}} \left[\sqrt{|\mathbb{H}(s'_{\rm src}) - \mathbb{H}(s'_{\rm tar})|}\right]}_{(c):\text{state distribution deviation}}.$$

Proof. Since it is infeasible to directly interact with the source domain, and we have the empirical policy distribution π_D in the offline dataset, we bound the performance difference by involving the term $J_{\mathcal{M}_{\rm src}}(\pi_D)$ We have

$$J_{\mathcal{M}_{\rm tar}}(\pi) - J_{\mathcal{M}_{\rm src}}(\pi) = \underbrace{(J_{\mathcal{M}_{\rm tar}}(\pi) - J_{\mathcal{M}_{\rm src}}(\pi_D))}_{(a)} + \underbrace{(J_{\mathcal{M}_{\rm src}}(\pi_D) - J_{\mathcal{M}_{\rm src}}(\pi))}_{(b)}. \tag{9}$$

The term (a) depicts the performance of the learned policy in the target domain against the performance of the data-collecting policy in the offline dataset, and the term (b) measures the performance deviation between the learned policy and the behavior policy in the source domain. We first bound term (b). By using Lemma B.3, we have

$$\begin{aligned}
J_{\mathcal{M}_{\rm src}}(\pi_D) - J_{\mathcal{M}_{\rm src}}(\pi) &= \frac{1}{1-\gamma}\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\rm src}}(s,a), s'\sim P_{\mathcal{M}_{\rm src}}(\cdot|s,a)} \left[\mathbb{E}_{a'\sim\pi_D}\left[Q^{\pi}_{\mathcal{M}_{\rm src}}(s',a')\right] - \mathbb{E}_{a'\sim\pi}\left[Q^{\pi}_{\mathcal{M}_{\rm src}}(s',a')\right]\right] \\
&\geq -\frac{1}{1-\gamma}\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\rm src}}(s,a), s'\sim P_{\mathcal{M}_{\rm src}}(\cdot|s,a)} \left|\mathbb{E}_{a'\sim\pi_D}\left[Q^{\pi}_{\mathcal{M}_{\rm src}}(s',a')\right] - \mathbb{E}_{a'\sim\pi}\left[Q^{\pi}_{\mathcal{M}_{\rm src}}(s',a')\right]\right| \\
&\geq -\frac{1}{1-\gamma}\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\rm src}}(s,a), s'\sim P_{\mathcal{M}_{\rm src}}(\cdot|s,a)} \left|\sum_{a'\in\mathcal{A}} (\pi_D(a'|s') - \pi(a'|s')) Q^{\pi}_{\mathcal{M}_{\rm src}}(s',a')\right| \\
&\geq -\frac{r_{\max}}{(1-\gamma)^2}\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\rm src}}(s,a), s'\sim P_{\mathcal{M}_{\rm src}}(\cdot|s,a)} \left|\sum_{a'\in\mathcal{A}} (\pi_D(a'|s') - \pi(a'|s'))\right| \\
&= -\frac{2r_{\max}}{(1-\gamma)^2}\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\rm src}}(s,a), s'\sim P_{\mathcal{M}_{\rm src}}(\cdot|s,a)} \left[D_{\rm TV}(\pi_D(\cdot|s') \| \pi(\cdot|s')\right].
\end{aligned}$$

It remains to bound term (a). By using Lemma B.2, we have

$$\begin{aligned}
J_{\mathcal{M}_{\rm tar}}(\pi) - J_{\mathcal{M}_{\rm src}}(\pi_D) &= -\frac{1}{1-\gamma}\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\rm src}}(s,a)} \left[\mathbb{E}_{s'_{\rm src}\sim P_{\mathcal{M}_{\rm src}}, a'\sim\pi_D}[Q^{\pi}_{\mathcal{M}_{\rm tar}}(s'_{\rm src},a')] - \mathbb{E}_{s'_{\rm tar}\sim P_{\mathcal{M}_{\rm tar}}, a'\sim\pi}[Q^{\pi}_{\mathcal{M}_{\rm tar}}(s'_{\rm tar},a')]\right] \\
&= -\frac{1}{1-\gamma}\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\rm src}}(s,a)} \Bigg[\underbrace{(\mathbb{E}_{s'_{\rm src}\sim P_{\mathcal{M}_{\rm src}}, a'\sim\pi_D}[Q^{\pi}_{\mathcal{M}_{\rm tar}}(s'_{\rm src},a')] - \mathbb{E}_{s'_{\rm src}\sim P_{\mathcal{M}_{\rm src}}, a'\sim\pi}[Q^{\pi}_{\mathcal{M}_{\rm tar}}(s'_{\rm src},a')])}_{(c)} \\
&\qquad + \underbrace{(\mathbb{E}_{s'_{\rm src}\sim P_{\mathcal{M}_{\rm src}}, a'\sim\pi}[Q^{\pi}_{\mathcal{M}_{\rm tar}}(s'_{\rm src},a')] - \mathbb{E}_{s'_{\rm tar}\sim P_{\mathcal{M}_{\rm tar}}, a'\sim\pi}[Q^{\pi}_{\mathcal{M}_{\rm tar}}(s'_{\rm tar},a')])}_{(d)}\Bigg].
\end{aligned}$$

We bound term (c) as follows:

$$\begin{aligned}(c) &= \mathbb{E}_{s'_{\text{src}}\sim P_{\mathcal{M}_{\text{src}}}}\left[\sum_{a'\in\mathcal{A}}(\pi_D(a'|s'_{\text{src}})-\pi(a'|s'_{\text{src}}))Q^{\pi}_{\mathcal{M}_{\text{tar}}}(s'_{\text{src}},a')\right]\\ &\le \mathbb{E}_{s'_{\text{src}}\sim P_{\mathcal{M}_{\text{src}}}}\left[\sum_{a'\in\mathcal{A}}|\pi_D(a'|s'_{\text{src}})-\pi(a'|s'_{\text{src}})|\times|Q^{\pi}_{\mathcal{M}_{\text{tar}}}(s'_{\text{src}},a')|\right]\\ &\le \frac{2r_{\max}}{1-\gamma}\mathbb{E}_{s'\sim P_{\mathcal{M}_{\text{src}}}}\left[D_{\text{TV}}(\pi_D(\cdot|s')\|\pi(\cdot|s')\right].\end{aligned}$$

Finally, we bound term (d).

$$\begin{aligned}(d) &= \mathbb{E}_{s'\sim P_{\mathcal{M}_{\text{src}}},a'\sim\pi}[Q^{\pi}_{\mathcal{M}_{\text{tar}}}(s',a')]-\mathbb{E}_{s'\sim P_{\mathcal{M}_{\text{tar}}},a'\sim\pi}[Q^{\pi}_{\mathcal{M}_{\text{tar}}}(s',a')]\\ &= \mathbb{E}_{a'\sim\pi}\left[\int_{\mathcal{S}}(P_{\mathcal{M}_{\text{src}}}(s'|s,a)-P_{\mathcal{M}_{\text{tar}}}(s'|s,a))Q^{\pi}_{\mathcal{M}_{\text{tar}}}(s',a')ds'\right]\\ &\le \mathbb{E}_{a'\sim\pi}\left[\int_{\mathcal{S}}|P_{\mathcal{M}_{\text{src}}}(s'|s,a)-P_{\mathcal{M}_{\text{tar}}}(s'|s,a)|\times|Q^{\pi}_{\mathcal{M}_{\text{tar}}}(s',a')|ds'\right]\\ &\le \frac{r_{\max}}{1-\gamma}\left[\int_{\mathcal{S}}|P_{\mathcal{M}_{\text{src}}}(s'|s,a)-P_{\mathcal{M}_{\text{tar}}}(s'|s,a)|ds'\right]=\frac{2r_{\max}}{1-\gamma}\left[D_{\text{TV}}(P_{\mathcal{M}_{\text{src}}}(\cdot|s,a)\|P_{\mathcal{M}_{\text{tar}}}(\cdot|s,a))\right]\end{aligned}$$

Then, we get the bound for term (a):

$$\begin{aligned}J_{\mathcal{M}_{\text{tar}}}(\pi)-J_{\mathcal{M}_{\text{src}}}(\pi_D) \ge &-\frac{2r_{\max}}{(1-\gamma)^2}\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\text{src}}}(s,a),s'\sim P_{\mathcal{M}_{\text{src}}}}\left[D_{\text{TV}}(\pi_D(\cdot|s')\|\pi(\cdot|s')\right]\\ &-\frac{2r_{\max}}{(1-\gamma)^2}\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\text{src}}}(s,a)}\left[D_{\text{TV}}(P_{\mathcal{M}_{\text{src}}}(\cdot|s,a)\|P_{\mathcal{M}_{\text{tar}}}(\cdot|s,a))\right].\end{aligned}$$

Combining the bounds for term (a) and term (b), and we have

$$\begin{aligned}J_{\mathcal{M}_{\text{tar}}}(\pi)-J_{\mathcal{M}_{\text{src}}}(\pi) \ge &-\frac{4r_{\max}}{(1-\gamma)^2}\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\text{src}}}(s,a),s'\sim P_{\mathcal{M}_{\text{src}}}}\left[D_{\text{TV}}(\pi_D(\cdot|s')\|\pi(\cdot|s')\right]\\ &-\frac{2r_{\max}}{(1-\gamma)^2}\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\text{src}}}(s,a)}\left[D_{\text{TV}}(P_{\mathcal{M}_{\text{src}}}(\cdot|s,a)\|P_{\mathcal{M}_{\text{tar}}}(\cdot|s,a))\right].\end{aligned}$$

Following the same procedure in the proof of Theorem 4.4 in Appendix A.3, we convert the dynamics discrepancy term into the representation mismatch term, incurring the following bounds:

$$\begin{aligned}J_{\mathcal{M}_{\text{tar}}}(\pi)-J_{\mathcal{M}_{\text{src}}}(\pi) \ge &-\frac{4r_{\max}}{(1-\gamma)^2}\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\text{src}}}(s,a),s'\sim P_{\mathcal{M}_{\text{src}}}}\left[D_{\text{TV}}(\pi_D(\cdot|s')\|\pi(\cdot|s')\right]\\ &-\frac{\sqrt{2}r_{\max}}{(1-\gamma)^2}\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\text{src}}}(s,a)}\left[\sqrt{D_{\text{KL}}(P_{\mathcal{M}_{\text{src}}}(\cdot|s,a)\|P_{\mathcal{M}_{\text{tar}}}(\cdot|s,a))}\right]\\ &-\frac{\sqrt{2}\gamma r_{\max}}{(1-\gamma)^2}\mathbb{E}_{\rho^{\pi_D}_{\mathcal{M}_{\text{src}}}(s,a)}\left[\sqrt{|\mathbb{H}(s'_{\text{src}})-\mathbb{H}(s'_{\text{tar}})|}\right].\end{aligned}$$

□

B. Useful Lemmas

Lemma B.1 (Telescoping lemma)**.** *Denote $\mathcal{M}_1=(\mathcal{S},\mathcal{A},P_1,r,\gamma)$ and $\mathcal{M}_2=(\mathcal{S},\mathcal{A},P_2,r,\gamma)$ as two MDPs that only differ in their transition dynamics. Then for any policy π, we have*

$$J_{\mathcal{M}_1}(\pi)-J_{\mathcal{M}_2}(\pi)=\frac{\gamma}{1-\gamma}\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_1}(s,a)}\left[\mathbb{E}_{s'\sim P_1}[V^{\pi}_{\mathcal{M}_2}(s')]-\mathbb{E}_{s'\sim P_2}[V^{\pi}_{\mathcal{M}_2}(s')]\right]. \tag{10}$$

Proof. This is Lemma 4.3 in (Luo et al., 2019), please check the proof there. □

Lemma B.2 (Extended telescoping lemma)**.** *Denote $\mathcal{M}_1 = (\mathcal{S}, \mathcal{A}, P_1, r, \gamma)$ and $\mathcal{M}_2 = (\mathcal{S}, \mathcal{A}, P_2, r, \gamma)$ as two MDPs that only differ in their transition dynamics. Suppose we have two policies π_1, π_2, we can reach the following conclusion:*

$$J_{\mathcal{M}_1}(\pi_1) - J_{\mathcal{M}_2}(\pi_2) = \frac{1}{1-\gamma}\mathbb{E}_{\rho^{\pi_1}_{\mathcal{M}_1}(s,a)}\left[\mathbb{E}_{s'\sim P_1, a'\sim\pi_1}[Q^{\pi_2}_{\mathcal{M}_2}(s',a')] - \mathbb{E}_{s'\sim P_2, a'\sim\pi_2}[Q^{\pi_2}_{\mathcal{M}_2}(s',a')]\right]. \tag{11}$$

Proof. This is Lemma C.2 in (Xu et al., 2023), please check the proof there. □

Lemma B.3. *Denote $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \gamma)$ as the underlying MDP. Suppose we have two policies π_1, π_2, then the performance difference of these policies in the MDP gives:*

$$J_{\mathcal{M}}(\pi_1) - J_{\mathcal{M}}(\pi_2) = \frac{1}{1-\gamma}\mathbb{E}_{\rho^{\pi_1}_{\mathcal{M}}(s,a), s'\sim P}\left[\mathbb{E}_{a'\sim\pi_1}[Q^{\pi_2}_{\mathcal{M}}(s',a')] - \mathbb{E}_{a'\sim\pi_2}[Q^{\pi_2}_{\mathcal{M}}(s',a')]\right]. \tag{12}$$

Proof. Similar to (Luo et al., 2019), we use a telescoping sum to prove the result. Denote W_j as the expected return when deploying π_1 in the MDP $\mathcal{M}$ for the first j steps and then switching to policy π_2, *i.e.*,

$$W_j = \mathbb{E}_{\substack{t<j: s_t\sim P, a_t\sim\pi_1 \\ t\geq j: s_t\sim P, a_t\sim\pi_2}}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t, a_t)\right]. \tag{13}$$

By definition, it is easy to find that $W_0 = J_{\mathcal{M}}(\pi_2)$ and $W_\infty = J_{\mathcal{M}}(\pi_2)$. Next, we express the value difference as the following form:

$$J_{\mathcal{M}}(\pi_1) - J_{\mathcal{M}}(\pi_2) = \sum_{t=0}^{\infty}(W_{k+1} - W_j). \tag{14}$$

The above term can be simplified as:

$$W_{j+1} - W_j = \gamma^j \mathbb{E}_{s_{j-1}\sim P, a_{j-1}\sim\pi_1}\left[\mathbb{E}_{s_j\sim P, a_j\sim\pi_1}[Q^{\pi_2}_{\mathcal{M}}(s_j,a_j)] - \mathbb{E}_{s_j\sim P, a_j\sim\pi_2}[Q^{\pi_2}_{\mathcal{M}}(s_j,a_j)]\right]. \tag{15}$$

Plug it back into Equation 14, and we have

$$\begin{aligned}
J_{\mathcal{M}}(\pi_1) - J_{\mathcal{M}}(\pi_2) &= \sum_{t=0}^{\infty}(W_{k+1} - W_j) \\
&= \sum_{j=0}^{\infty}\gamma^j \mathbb{E}_{\rho^{\pi_1}_{\mathcal{M}}, s'\sim P}\left[\mathbb{E}_{a'\sim\pi_1}[Q^{\pi_2}_{\mathcal{M}}(s',a')] - \mathbb{E}_{a'\sim\pi_2}[Q^{\pi_1}_{\mathcal{M}}(s',a')]\right] \\
&= \frac{1}{1-\gamma}\mathbb{E}_{\rho^{\pi_1}_{\mathcal{M}}, s'\sim P}\left[\mathbb{E}_{a'\sim\pi_1}[Q^{\pi_2}_{\mathcal{M}}(s',a')] - \mathbb{E}_{a'\sim\pi_2}[Q^{\pi_1}_{\mathcal{M}}(s',a')]\right].
\end{aligned}$$

□

C. Pseudocodes

In this section, we provide detailed pseudocodes for online PAR and offline PAR, as shown in Algorithm 2 and Algorithm 3.

D. Experimental Details and Hyperparameter Setup

In this section, we describe the detailed experimental setup as well as the hyperparameter setup used in this work. To ensure reproducibility, we include the codes of PAR in the supplementary material, and will open source our codes upon acceptance.

D.1. Environment Setting

We adopt the same environments proposed in (Xu et al., 2023) without any modification. We employ four widely used environments from OpenAI Gym (Brockman et al., 2016), *HalfCheetah-v2*, *Hopper-v2*, *Walker2d-v2*, *Ant-v3*, as source domains. To simulate dynamics discrepancies, we consider kinematic shifts and morphology shifts between the source domain and the target domain. This results in a total of 8 target domains. Kinematic shifts indicate that some joints of the

Algorithm 2 Policy Adaptation by Representation Mismatch (online version)

Input: Source domain $\mathcal{M}_{\text{src}}$, target domain $\mathcal{M}_{\text{tar}}$, target domain interaction interval F, batch size N, source domain interaction maximum step $T_{\max}$, reward penalty coefficient β, temperature (for SAC) α, target update rate τ.

Initialize policy π_ϕ, value functions $\{Q_{\theta_i}\}_{i=1,2}$ and target networks $\{Q_{\theta'_i}\}_{i=1,2}$, source domain replay buffer $D_{\text{src}} \leftarrow \emptyset$, target domain replay buffer $D_{\text{tar}} \leftarrow \emptyset$. Initialize the state encoder f and state-action encoder g with parameters ψ, ξ, respectively
for i = 1, 2, ..., $T_{\max}$ **do**
  Collect transition $(s_{\text{src}}, a_{\text{src}}, r_{\text{src}}, s'_{\text{src}})$ with policy π_ϕ in $\mathcal{M}_{\text{src}}$
  Store the transition in D_{src}, $D_{\text{src}} \leftarrow D_{\text{src}} \cup \{(s_{\text{src}}, a_{\text{src}}, r_{\text{src}}, s'_{\text{src}})\}$
  if $i\%\ F == 0$ **then**
    Given s_{tar} in $\mathcal{M}_{\text{tar}}$, execute a_{tar} using the policy π_ϕ and get $(s_{\text{tar}}, a_{\text{tar}}, r_{\text{tar}}, s'_{\text{tar}})$
    Store the transition in the replay buffer, $D_{\text{tar}} \leftarrow D_{\text{tar}} \cup \{(s_{\text{tar}}, a_{\text{tar}}, r_{\text{tar}}, s'_{\text{tar}})\}$
  end if
  Sample N transitions $d_{\text{tar}} = \{(s_j, a_j, r_j, s'_j)\}_{j=1}^N$ from D_{tar}
  Train encoders f, g in the target domain by minimizing: $\frac{1}{N}\sum_{d_{\text{tar}}}\left[(g_\xi(f_\psi(s), a) - \texttt{SG}(f_\psi(s')))^2\right]$
  Sample N transitions $d_{\text{src}} = \{(s_j, a_j, r_j, s'_j)\}_{j=1}^N$ from D_{src}
  Modify source domain rewards into $\hat{r}_{\text{src}} = r_{\text{src}} - \beta \times [g_\xi(f_\psi(s_{\text{src}}), a_{\text{src}}) - f_\psi(s'_{\text{src}})]^2$
  Calculate target values $y = r + \gamma\left[\min_{i=1,2} Q_{\theta'_i}(s', a') - \alpha \log \pi_\phi(a'|s')\right], a' \sim \pi_\phi(\cdot|s')$
  Update critics by minimizing $\frac{1}{2N}\sum_{d_{\text{src}}\cup d_{\text{tar}}}(Q_{\theta_i} - y)^2, i \in \{1, 2\}$
  Update actor by maximizing $\frac{1}{2N}\sum_{d_{\text{src}}\cup d_{\text{tar}}, a\sim\pi_\phi(\cdot|s)}\left[\min_{i=1,2} Q_{\theta_i}(s, a) - \alpha \log \pi_\phi(\cdot|s)\right]$
  Update target networks: $\theta'_i \leftarrow \tau\theta_i + (1-\tau)\theta'_i$
end for

simulated robot are *broken*, while morphology mismatch means that there are some morphological differences between the simulated robots in the two domains. We explicate the detailed modifications below.

halfcheetah (broken back thigh): The rotation angle of the joint on the thigh of the Cheetah robot's back leg is modified from $[-0.52, 1.05]$ to $[-0.0052, 0.0105]$.

hopper (broken joints): The rotation angles of the head joint and the foot joint are modified from $[-150, 0]$, $[-45, 45]$ to $[-0.15, 0]$, $[-18, 18]$, respectively.

walker (broken right foot): The rotation angle of the foot joint on the robot's right leg is modified from $[-45, 45]$ to $[-0.45, 0.45]$.

ant (broken hips): The rotation angle of the joints on the hip of two legs are modified from $[-30, 30]$ to $[-0.3, 0.3]$

halfcheetah (no thighs): The sizes of the back thigh and the forward thigh are reduced as shown below:

```
# back thigh
<geom fromto="0 0 0 -0.0001 0 -0.0001" name="bthigh" size="0.046" type="capsule"/>
<body name="bshin" pos="-0.0001 0 -0.0001">
# forward thigh
<geom fromto="0 0 0 0.0001 0 0.0001" name="fthigh" size="0.046" type="capsule"/>
<body name="fshin" pos="0.0001 0 0.0001">
```

hopper (big head): The head size of the robot is modified as shown below:

```
# head size
<geom friction="0.9" fromto="0 0 1.45 0 0 1.05" name="torso_geom" size="0.125" type="
    capsule"/>
```

walker (no right foot): The thigh on the right leg of the robot is modified as the following:

```
# right leg
<body name="thigh" pos="0 0 1.05">
  <joint axis="0 -1 0" name="thigh_joint" pos="0 0 1.05" range="-150 0" type="hinge"/>
```

Algorithm 3 Policy Adaptation by Representation Mismatch (offline version)

Input: Source domain $\mathcal{M}_{\text{src}}$, target domain $\mathcal{M}_{\text{tar}}$, target domain interaction interval F, batch size N, maximum gradient step $T_{\max}$, reward penalty coefficient β, normalization coefficient ν, temperature (for SAC) α, target update rate τ. Source domain offline dataset $\mathcal{D}_{\text{off}}$

Initialize policy π_ϕ, value functions $\{Q_{\theta_i}\}_{i=1,2}$ and target networks $\{Q_{\theta'_i}\}_{i=1,2}$, source domain replay buffer $D_{\text{src}} \leftarrow D_{\text{off}}$, target domain replay buffer $D_{\text{tar}} \leftarrow \emptyset$. Initialize the state encoder f and state-action encoder g with parameters ψ, ξ, respectively
for i = 1, 2, ..., $T_{\max}$ **do**
  if $i\%\ F == 0$ **then**
    Given s_{tar} in $\mathcal{M}_{\text{tar}}$, execute a_{tar} using the policy π_ϕ and get $(s_{\text{tar}}, a_{\text{tar}}, r_{\text{tar}}, s'_{\text{tar}})$
    Store the transition in the replay buffer, $D_{\text{tar}} \leftarrow D_{\text{tar}} \cup \{(s_{\text{tar}}, a_{\text{tar}}, r_{\text{tar}}, s'_{\text{tar}})\}$
  end if
  Sample N transitions $d_{\text{tar}} = \{(s_j, a_j, r_j, s'_j)\}_{j=1}^N$ from D_{tar}
  Train encoders f, g in the target domain by minimizing: $\frac{1}{N}\sum_{d_{\text{tar}}}\left[(g_\xi(f_\psi(s), a) - \texttt{SG}(f_\psi(s')))^2\right]$
  Sample N transitions $d_{\text{src}} = \{(s_j, a_j, r_j, s'_j)\}_{j=1}^N$ from D_{src}
  Modify source domain rewards into $\hat{r}_{\text{src}} = r_{\text{src}} - \beta \times [g_\xi(f_\psi(s_{\text{src}}), a_{\text{src}}) - f_\psi(s'_{\text{src}})]^2$
  Calculate target values $y = r + \gamma\left[\min_{i=1,2} Q_{\theta'_i}(s', a') - \alpha \log \pi_\phi(a'|s')\right], a' \sim \pi_\phi(\cdot|s')$
  Update critics by minimizing $\frac{1}{2N}\sum_{d_{\text{src}}\cup d_{\text{tar}}}(Q_{\theta_i} - y)^2, i \in \{1, 2\}$
  Update actor by maximizing $\frac{\lambda}{2N}\sum_{d_{\text{src}}\cup d_{\text{tar}}, a\sim\pi_\phi(\cdot|s)}\left[\min_{i=1,2} Q_{\theta_i}(s, a) - \alpha \log \pi_\phi(\cdot|s)\right] - \frac{1}{N}\sum_{d_{\text{src}}}(a - \tilde{a})^2$, where $\tilde{a} \sim \pi_\phi(\cdot|s), \lambda = \nu / \frac{1}{2N}\sum_{d_{\text{src}}\cup d_{\text{tar}}} \min_{i=1,2} Q_{\theta_i}(s,a)$
  Update target networks: $\theta'_i \leftarrow \tau\theta_i + (1-\tau)\theta'_i$
end for

```
  <geom friction="0.9" fromto="0 0 1.05 0 0 1.045" name="thigh_geom" size="0.05" type="
      capsule"/>
  <body name="leg" pos="0 0 0.35">
    <joint axis="0 -1 0" name="leg_joint" pos="0 0 1.045" range="-150 0" type="hinge"/>
    <geom friction="0.9" fromto="0 0 1.045 0 0 0.3" name="leg_geom" size="0.04" type="
        capsule"/>
    <body name="foot" pos="0.2 0 0">
      <joint axis="0 -1 0" name="foot_joint" pos="0 0 0.3" range="-45 45" type="hinge"/>
      <geom friction="0.9" fromto="-0.0 0 0.3 0.2 0 0.3" name="foot_geom" size="0.06" type="
          capsule"/>
    </body>
  </body>
</body>
```

ant (short feet): The size of the ant robot's feet on its front two legs are modified into the following parameters:

```
# leg 1
<geom fromto="0.0 0.0 0.0 0.1 0.1 0.0" name="left_ankle_geom" size="0.08" type="capsule"/>
# leg 2
<geom fromto="0.0 0.0 0.0 -0.1 0.1 0.0" name="right_ankle_geom" size="0.08" type="capsule"
    />
```

For more details, one could check the `xml` files in the supplementary material.

D.2. Implementation Details

In this subsection, we provide implementation details as well as an introduction of the baselines adopted in this work and the PAR algorithm. When the source domain is online, we consider the following baselines for comparison: SAC-tar (Haarnoja et al., 2018), DARC (Eysenbach et al., 2021), DARC-weight, VGDF (Xu et al., 2023), and SAC-tune. We list the detailed implementation details of these methods below.

SAC-tar: We train an SAC agent solely in the target domain for 10^5 environmental steps, with its hyperparameter setup specified in Table 2. We do not use the temperature auto-tuned SAC, but keep the temperature coefficient $\alpha = 0.2$ fixed.

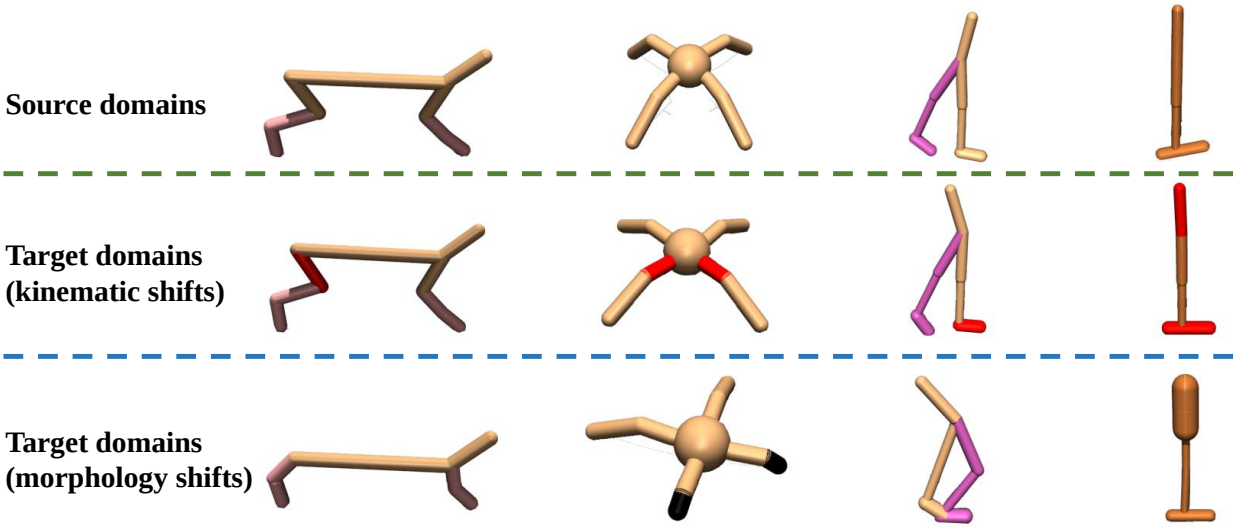


Figure 7. **Graphical illustration of the evaluated environments.** The source domains (*top*) are well-functional simulated robots from Gym, the target domains either have kinematic shifts (*middle*) or morphology shifts (*bottom*) compared to the source domains.

This applies to all of the other baselines.

SAC-tune: We train an SAC agent first in the source domain for 10^6 environmental steps, and then fine-tune its policy by interacting with the target domain for another 10^5 timesteps. We use the same hyperparameters as SAC-tar.

DARC: We follow the original paper to train two domain classifiers $q_{\theta_{\text{SAS}}}(\text{target}|s_t, a_t, s_{t+1}), q_{\theta_{\text{SA}}}(\text{target}|s_t, a_t)$ parameterized by θ_{SAS} and θ_{SA}, respectively. These domain classifiers are optimized via the cross-entropy loss:

$$\mathcal{L}(\theta_{\text{SAS}}) = \mathbb{E}_{\mathcal{D}_{\text{tar}}}\left[\log q_{\theta_{\text{SAS}}}(\text{target}|s_t, a_t, s_{t+1})\right] + \mathbb{E}_{\mathcal{D}_{\text{src}}}\left[\log(1 - q_{\theta_{\text{SAS}}}(\text{target}|s_t, a_t, s_{t+1}))\right],$$
$$\mathcal{L}(\theta_{\text{SA}}) = \mathbb{E}_{\mathcal{D}_{\text{tar}}}\left[\log q_{\theta_{\text{SA}}}(\text{target}|s_t, a_t)\right] + \mathbb{E}_{\mathcal{D}_{\text{src}}}\left[\log(1 - q_{\theta_{\text{SA}}}(\text{target}|s_t, a_t))\right],$$

where $D_{\text{src}}, D_{\text{tar}}$ are replay buffers of the source domain and the target domain, respectively. Following the original paper, we use Gaussian standard deviation $\sigma = 1$ for training the domain classifiers. We experimentally find that DARC is quite sensitive to the standard deviation σ, *e.g.*, if one sets $\sigma = 0.1$, DARC exhibits very poor performance on almost all tasks. We do not modify this value and keep it fixed across all runs. Then, DARC compensates the source domain rewards by estimating the dynamics gap with the form: $\log \frac{P_{\mathcal{M}_{\text{tar}}}(s_{t+1}|s_t,a_t)}{P_{\mathcal{M}_{\text{src}}}(s_{t+1}|s_t,a_t)}$. By approximating this term with the trained encoders, DARC estimates the reward correction term δr by

$$\delta r(s_t, a_t) = -\log \frac{q_{\theta_{\text{SAS}}}(\text{target}|s_t, a_t, s_{t+1}) q_{\theta_{\text{SA}}}(\text{source}|s_t, a_t)}{q_{\theta_{\text{SAS}}}(\text{source}|s_t, a_t, s_{t+1}) q_{\theta_{\text{SA}}}(\text{target}|s_t, a_t)}. \tag{16}$$

The source domain rewards are formally modified via:

$$\hat{r}_{\text{src}}^{\text{DARC}} = r_{\text{src}}(s_t, a_t) - \beta \times \delta_t, \tag{17}$$

where $\beta \in \mathbb{R}$ is the reward penalty coefficient. Note that Equation 17 is slightly different from the original paper in the following aspects: (1) DARC paper adopts $\beta = 1$ by default and does not tune this hyperparameter, while we search the best β across $\{0.1, 0.5, 1.0, 2.0\}$ for online experiments; (2) we adopt a negative reward correction term, *i.e.*, $r - \delta r$, whereas DARC paper has the form $r + \delta r$. We use this form to ensure consistency between PAR and DARC in reward correction, and for the benefit of reward penalty comparison illustrated in Figure 6 and Figure 11. We provide DARC's the best hyperparameter β for each task in Table 3. We adopt the default hyperparameter setup from the authors (https://github.com/google-research/google-research/tree/master/darc). Moreover, we follow the instruction in Appendix E of the DARC paper and warmup the algorithm without δr for the first 10^5 steps.

DARC-weight: This variant generally resembles vanilla DARC, except that it does not perform reward correction for the

source domain data, but utilizes that estimated dynamics gap as an importance sampling term for critic updates, *i.e.*,

$$\mathcal{L}_{\text{critic}} = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}_{\text{src}}}\left[\omega(s,a,s')\left(Q_{\theta_i}(s,a)-y\right)^2\right], i\in\{1,2\}, \tag{18}$$

where

$$\omega(s,a,s') = \frac{q_{\theta_{\text{SAS}}}(\text{target}|s,a,s')q_{\theta_{\text{SA}}}(\text{source}|s,a)}{q_{\theta_{\text{SAS}}}(\text{source}|s,a,s')q_{\theta_{\text{SA}}}(\text{target}|s,a)}. \tag{19}$$

To ensure that the importance sampling weight $\omega(s,a,s')$ lies in a valid range, we clip its value to the range of $[1e^{-4},1]$ for training stability.

VGDF: VGDF is constructed based on the theoretical results that the performance bound of a policy in the source domain and the target domain is controlled by value difference, *i.e.*,

$$J_{\mathcal{M}_{\text{tar}}}(\pi) \geq J_{\mathcal{M}_{\text{src}}}(\pi) - \frac{\gamma}{1-\gamma}\mathbb{E}_{\rho^{\pi}_{\mathcal{M}_{\text{src}}}}\left[\left|\mathbb{E}_{P_{\mathcal{M}_{\text{src}}}}[V^{\pi}_{\mathcal{M}_{\text{tar}}}(s')] - \mathbb{E}_{P_{\mathcal{M}_{\text{tar}}}}[V^{\pi}_{\mathcal{M}_{\text{tar}}}(s')]\right|\right]. \tag{20}$$

Then, VGDF decides to filter samples in the source domain that share similar value estimates as those in the target domain. To that end, it trains an ensemble of dynamics model akin to model-based RL (Janner et al., 2019; Pan et al., 2020; Qiao et al., 2023; Buckman et al., 2018) in the *original state-action space* of the target domain to predict the next state that follows the transition dynamics of the target domain given source domain data $(s_{\text{src}}, a_{\text{src}})$. Then it measures the mean and variance of the value ensemble $\{Q(s'_i,a'_i)\}_{i=1}^{M}$ to construct a Gaussian distribution, where s'_i is the predicted next state, a'_i is sampled from the policy, and M is the ensemble size. After that, the authors utilize rejection sampling to select a fixed percentage of source domain data with the highest likelihood estimation and share them with the target domain. We use the official implementation of VGDF (https://github.com/Kavka1/VGDF) without any modification. We set the data selection ratio in VGDF as 25%. VGDF also adopts SAC as the base algorithm, with its temperature set to be 0.2. VGDF trains an extra exploration policy for better exploration (while PAR does not). Following the original implementation, we warm-start VGDF by disabling rejection sampling (*i.e.*, accept all transitions from the source domain) for the first 10^5 timesteps. We generally can reproduce the reported performance of VGDF.

PAR: Different from the above methods, PAR detects the dynamics mismatch by capturing the representation mismatch, *i.e.*, the representation deviation between the source domain state-action pair and its subsequent next state using the state encoder f and state-action encoder g trained only in the target domain. Note that different from VGDF, we only train *one single* state encoder along with *one single* state-action encoder, which we find can already incur satisfying performance and suitable reward penalty. The encoders are updated via Equation 1. We compensate the source domain rewards by measuring the representation deviation, as shown in Equation 2. The detailed hyperparameter setup for PAR is available in Table 2. We use the same batch size of source domain data and target domain data for training. Since the representation deviation is strongly correlated with the environment itself, *e.g.*, the state space, the action space, and the reward function, we believe it is understandable that the best penalty coefficient β differs among different evaluated online tasks. We sweep across $\beta\in\{0.1,0.5,1.0,2.0\}$ and report the adopted β for each task in Table 3.

When the source domain is offline, we consider baseline methods of CQL-0 (Kumar et al., 2020), CQL+SAC, H2O (Niu et al., 2022), and VGDF+BC (Xu et al., 2023). The corresponding implementation details can be found below.

CQL-0: CQL is a well-known offline RL algorithm. CQL-0 denotes that we train a CQL agent merely on the offline source domain dataset and transfer the learned policy to the target domain in a zero-shot manner. We use the public implementation of CQL (https://github.com/tinkoff-ai/CORL) and train it for 10^6 gradient steps.

CQL+SAC: This baseline leverages both offline source domain data and online target domain transitions for learning a policy. Since learning from offline data requires conservatism, while learning from online samples does not, we train critics by updating source domain data with the CQL loss while the online target domain data with the SAC loss, *i.e.*,

$$\mathcal{L}_{\text{critic}} = \mathbb{E}_{D_{\text{src}}\cup D_{\text{tar}}}\left[(Q_{\theta_i}(s,a)-y)^2\right] + \beta_{\text{CQL}}\left(\mathbb{E}_{s\sim D_{\text{src}},\tilde{a}\sim\pi_{\phi}(\cdot|s)}[Q_{\theta_i}(s,\tilde{a})] - \mathbb{E}_{D_{\text{src}}}[Q_{\theta_i}(s,a)]\right), i\in\{1,2\}, \tag{21}$$

where β_{CQL} is the hyperparameter, and we use $\beta_{\text{CQL}}=10.0$ (which is the same as CQL-0). Note that we sample the same batch size (128) of both source domain data and target domain for update at each gradient step. We train CQL+SAC for 10^6 gradient steps, with 10^5 interactions with the target domain.

H2O: H2O can be viewed as an offline version of *DARC-weight* algorithm, which also trains domain classifiers to estimate dynamics gap, which further serves as importance sampling weights for critic optimization. It additionally combines CQL

loss to inject conservatism. To be specific, its critic objective function can be written as:

$$\begin{aligned}\mathcal{L}_{\text{critic}} = \mathbb{E}_{D_{\text{tar}}}\left[(Q_{\theta_i}(s,a)-y)^2\right] + \mathbb{E}_{D_{\text{src}}}\left[\omega(s,a,s')(Q_{\theta_i}(s,a)-y)^2\right] \\ + \beta_{\text{CQL}}\left(\mathbb{E}_{s\sim D_{\text{src}},\tilde{a}\sim\pi_\phi(\cdot|s)}[Q_{\theta_i}(s,\tilde{a})] - \mathbb{E}_{D_{\text{src}}}[Q_{\theta_i}(s,a)]\right), i\in\{1,2\},\end{aligned} \tag{22}$$

where $\omega(s,a,s')$ is evaluated as in Equation 19. We use the default configurations of hyperparameters specified in the official codebase (https://github.com/t6-thu/H2O), except that we use $\beta_{\text{CQL}} = 10.0$ since we experimentally find that the recommended $\beta_{\text{CQL}} = 0.01$ from the authors incurs very poor performance on all of the datasets. We train H2O for 10^6 gradient steps, and allow the policy to gather data in the target domain every 10 steps.

VGDF+BC: VGDF+BC generally has the same hyperparameter setup as VGDF, except that we incorporate a behavior cloning term into its policy training, akin to offline PAR. Its actor generally follows the same way of updating as Equation 6. We take the results of VGDF+BC on six medium-level datasets from its paper directly, where one can see that it actually uses different hyperparameter setups on different tasks, *i.e.*, ν. For other datasets, we set $\nu = 5$ by following the instructions from the original paper. We train VGDF+BC for 10^6 gradient steps, with 10^5 interactions with the target domain.

PAR: Offline PAR differs from its online variant in that an additional behavior cloning term is involved. We generally adopt fixed reward penalty coefficient β and normalization parameter ν across all tasks, as depicted in Table 3. We train PAR for 10^6 gradient steps, and let it interact with the target domain every 10 gradient steps.

D.3. Hyperparameter Setup

In this part, we list the detailed hyperparameter setup for PAR and baseline methods in Table 2. We also provide the adopted key hyperparameters given both the online source domain and the offline source domain in Table 3.

E. Extended Experimental Results

In the main text, we cannot include all of our experiments due to the space limit. In this section, we provide more experimental results concerning on parameter study (*i.e.*, more experiments with the online source domain and results given the offline source domain), and the reward penalty comparison between DARC and PAR. We believe these are helpful to better understand the effect of the key hyperparameters in PAR and further validate the advantages of PAR against DARC.

E.1. Wider Parameter Study

We first include more results of the parameter study of reward penalty coefficient β and target domain interaction interval F, as illustrated in Figure 8(a) and Figure 8(b), respectively. Note that the results are conducted over the online PAR algorithm. For the reward penalty coefficient β, setting $\beta = 0$ makes PAR degenerate into VGDF with the data selection ratio 0%. We find that setting $\beta = 0$ (*i.e.*, no reward modification for source domain data) generally does not incur a good performance, which is consistent with the results in Figure 3(a) of the main text. Note that these tasks also have different optimal β.

As for the target domain interaction interval F, one can see that on other tasks, larger F also results in a better performance. This indicates that the amount of source domain data is critical for PAR, and more data from the source domain can boost the performance of PAR in the target domain.

Next, we investigate how the hyperparameters affect the performance of PAR given an offline source domain. We are interested in the reward penalty coefficient β and the normalization parameter ν. We sweep across $\beta \in \{0, 0.1, 0.5, 1.0, 2.0\}$ and $\nu \in \{1.0, 2.5, 5.0, 10.0\}$. We summarize the analysis and experimental results below.

Reward penalty coefficient β in offline PAR. We consider two quality levels, *medium* and *medium-expert*, from D4RL datasets as the source domain, and run experiments on four tasks, two with kinematic shifts (*walker (broken right foot)*, *ant (broken hips)*) and two with morphology shifts (*halfcheetah (no thighs)*, *hopper (big head)*). We present the results in Figure 9. We find that setting $\beta = 0$ incurs worse performance on most of the tasks, regardless of whether medium-level datasets or medium-expert-level datasets are provided. This further demonstrates the necessity of reward modification by PAR and highlights the effectiveness of our method. We also observe that the performance of offline PAR is generally better with higher quality datasets. Meanwhile, it still holds that the best penalty coefficient β differs in different environments. For all of our offline experiments, we set $\beta = 1$ by default and do not tune this value.

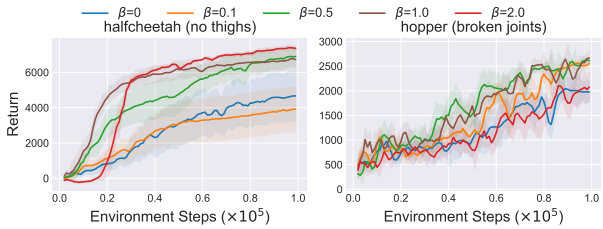
Normalization coefficient ν in offline PAR. ν controls the balance between the behavior cloning term and the term that

Table 2. Detailed hyperparameter setup for PAR and baseline methods on the evaluated tasks.

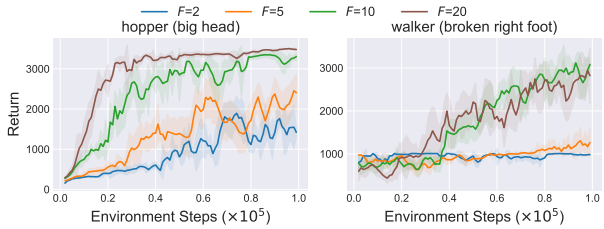
Hyperparameter	**Value**
Shared	
Actor network	$(256, 256)$
Critic network	$(256, 256)$
Batch size	256 for SAC-tar, CQL-0, and 128 for others
Learning rate	3×10^{-4}
Optimizer	Adam (Kingma & Ba, 2015)
Discount factor	0.99
Replay buffer size	10^6
Warmup steps	0 for PAR and 10^5 for others
Nonlinearity	ReLU
Target update rate	5×10^{-3}
Temperature coefficient	0.2
Maximum log std	2
Minimum log std	-20
Target domain interaction interval	10
DARC, DARC-weight, H2O	
Classifier Network	$(256, 256)$
CQL-0, CQL+SAC, H2O	
CQL penalty coefficient β_{CQL}	10.0
VGDF, VGDF+BC	
Dynamics model network	$(200, 200, 200, 200, 200)$
Ensemble size	7
Data selection ratio	25%
Normalization coefficient ν	5.0
VGDF	
Exploration policy network	$(256, 256)$
PAR	
Encoder Network	$(256, 256)$
Representation dimension	256
Nomralization coefficient ν	5.0

Table 3. Adopted hyperparameters for PAR and baseline method DARC on evaluated environments.

Task Name	PAR (online) β	DARC (online) β	PAR (offline) β	PAR (offline) ν
halfcheetah (broken back thigh)	1.0	2.0	1.0	5.0
halfcheetah (no thighs)	2.0	0.5	1.0	5.0
hopper (broken joints)	0.5	2.0	1.0	5.0
hopper (big head)	0.5	1.0	1.0	5.0
walker (broken right foot)	0.5	1.0	1.0	5.0
walker (no right thigh)	0.5	1.0	1.0	5.0
ant (broken hips)	0.1	1.0	1.0	5.0
ant (short feet)	0.1	0.1	1.0	5.0

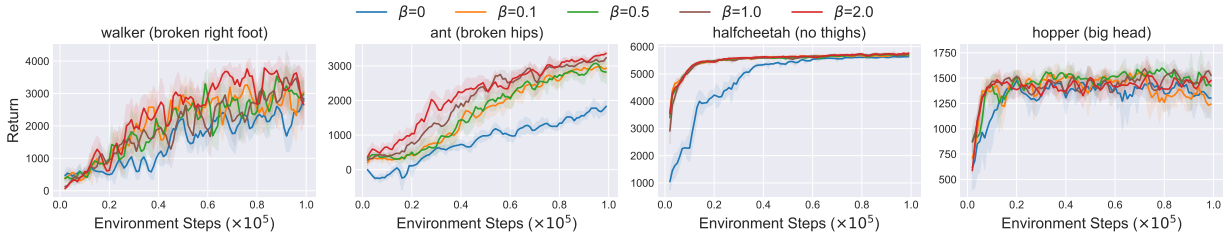


(a) Penalty coefficient β.

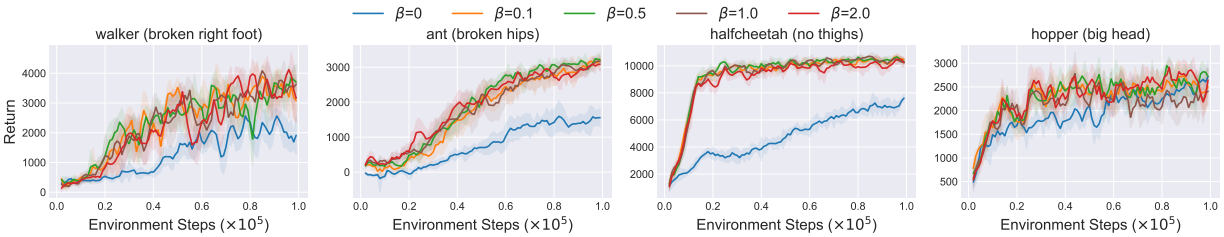


(b) Target domain interaction interval F.

Figure 8. **Extended parameter study of (a) reward penalty coefficient β, (b) target domain interaction interval F on wider environments.** These curves show the performance of the policy in the target domain. The reported results are averaged across 5 different random seeds and the shaded region represents the standard deviation.



(a) Comparison of different reward penalty coefficient β given medium-level datasets.



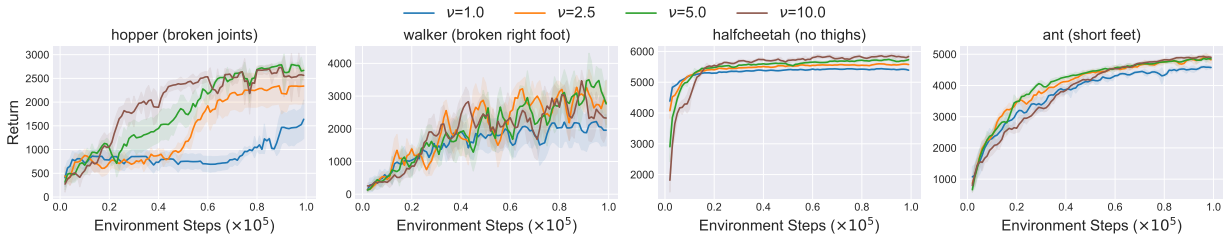
(b) Comparison of different reward penalty coefficient β given medium-expert-level datasets.

Figure 9. **Extended parameter study of reward penalty coefficient β given medium, medium-expert level source domain datasets.** We report the average return over 5 different runs along with the standard deviation of the policy in the target domain.

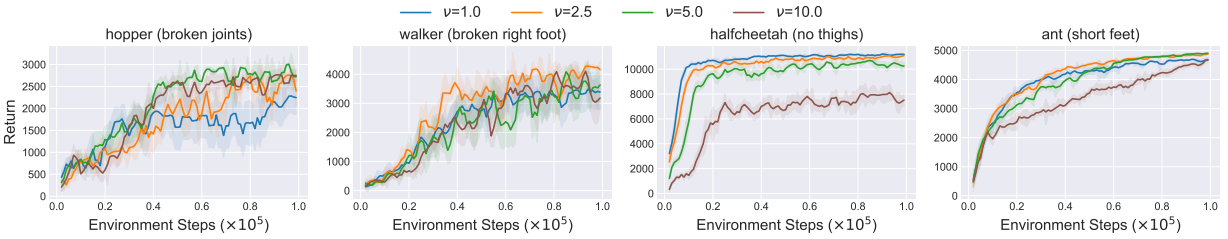
maximizes the Q-value. Intuitively, a larger ν tends to bring more conservatism, encouraging the learned policy to be close to the behavior policy in the source domain dataset. We run experiments on four tasks that are made up of two tasks with kinematic shifts (*hopper (broken joints)*, *walker (broken right foot)*) and two tasks with morphology shifts (*halfcheetah (no thighs)*, *ant (short feet)*). The results can be found in Figure 10. We can see that PAR is robust to ν on tasks like *walker (broken right foot)*. While it turns out that on tasks like *hopper (broken joints)*, a too small ν is not preferred, and on tasks like *halfcheetah (no thighs)*, a too large ν does not ensure good performance. We therefore adopt $\nu = 5.0$ for all of the offline experiments to seek a trade-off.

E.2. Wider Evidence on Reward Penalty Comparison between DARC and PAR

In this part, we provide more evidence that PAR is better than DARC. Since DARC is an online algorithm, we conduct experiments with the online source domains. We run experiments on wider environments, including two environments with kinematic mismatch (*hopper (broken joints)*, *walker (broken right foot)*) and two tasks with morphology mismatch (*halfcheetah (no thighs)*, *ant (short feet)*). We summarize the comparison results in Figure 11. Based on the curves, we find that on many tasks (*e.g.*, *halfcheetah (no thighs)*), the reward penalty given by DARC is quite large and even becomes larger with more environment steps, indicating that DARC can be overly pessimistic and the classifiers may fail to produce suitable reward penalties to compensate source domain data. On the *hopper (broken joints)* task, however, DARC gives penalties



(a) Comparison of different normalization coefficient ν given medium-level datasets.



(b) Comparison of different normalization coefficient ν given medium-expert-level datasets.

Figure 10. **Extended parameter study of normalization coefficient ν under medium, medium-expert level source domain datasets.** The results depict the mean performance of the policy in the target domain with 5 different random seeds. The shaded region is the standard deviation.

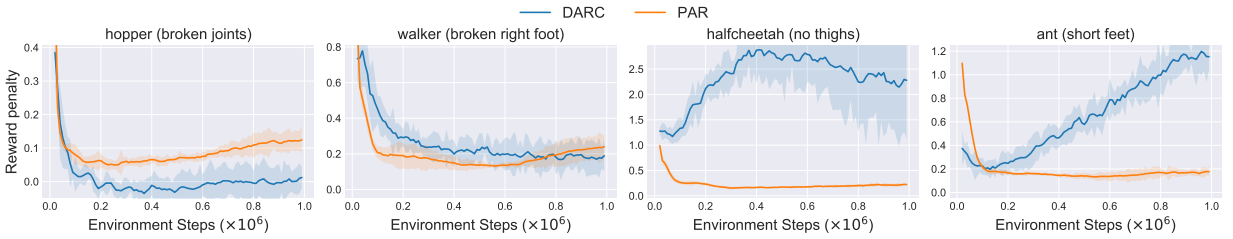


Figure 11. **Extensive comparison on reward penalty produced by DARC and PAR.** We record the mean reward penalty calculated in a sampled batch during the training of each algorithm. The environmental steps here denote the step in the source domain. The reported curves are further averaged across 5 independent runs and the shaded region captures the standard deviation.

that quite approach 0, and fails to inform the agent of the dynamics difference between the source domain and the target domain. Instead, we observe that on all of the evaluated tasks, the reward penalty given by PAR gradually converges to a small number (but not 0). Despite that at the initial stage, the collected samples from the source domain and the target domain may have large discrepancies, our method can fully exploit these data and successfully find dynamics-consistent behaviors and transitions later on. We believe these further verify that PAR is a better choice than DARC.

F. Compute Infrastructure

In Table 4, we list the compute infrastructure that we use to run all of the algorithms.

Table 4. Compute infrastructure.

CPU	GPU	Memory
AMD EPYC 7452	RTX3090×8	288GB