DUAL-ROBUST CROSS-DOMAIN OFFLINE REINFORCEMENT LEARNING AGAINST DYNAMICS SHIFTS

Anonymous authors
Paper under double-blind review

ABSTRACT

Single-domain offline reinforcement learning (RL) often suffers from limited data coverage, while cross-domain offline RL handles this issue by leveraging additional data from other domains with dynamics shifts. However, existing studies primarily focus on train-time robustness (handling dynamics shifts from training data), neglecting the test-time robustness against dynamics perturbations when deployed in practical scenarios. In this paper, we investigate dual (both train-time and test-time) robustness against dynamics shifts in cross-domain offline RL. We first empirically show that the policy trained with cross-domain offline RL exhibits fragility under dynamics perturbations during evaluation, particularly when target domain data is limited. To address this, we introduce a novel robust cross-domain Bellman (RCB) operator, which enhances test-time robustness against dynamics perturbations while staying conservative to the out-of-distribution dynamics transitions, thus guaranteeing the train-time robustness. To further counteract potential value overestimation or underestimation caused by the RCB operator, we introduce two techniques, the dynamic value penalty and the Huber loss, into our framework, resulting in the practical **D**ual-**RO**bust **C**ross-domain **O**ffline RL (DROCO) algorithm. Extensive empirical results across various dynamics shift scenarios show that DROCO outperforms strong baselines and exhibits enhanced robustness to dynamics perturbations.

1 INTRODUCTION

Deep reinforcement learning (RL) (Sutton & Barto, 1999) has been a vital tool in various fields, such as embodied manipulation (Zakka et al., 2023; Shi et al., 2024) and natural language processing (Ouyang et al., 2022; Rafailov et al., 2023). The success of typical RL often relies on numerous online interactions with the environment. However, this *trial-and-error* manner can be costly or even risky when applied in the real world. Offline RL (Levine et al., 2020), instead, trains the policy with only a pre-logged offline dataset, eliminating the need for interactions with the environment. However, large-scale and diverse offline datasets are not always accessible in practice, and offline RL often struggle with a limited offline dataset. A line of recent studies (Wen et al., 2024; Lyu et al., 2025; Liu et al., 2022) has explored a paradigm known as Cross-Domain Offline RL. In this setting, data from the target domain is limited, but we have access to datasets from a relevant but distinct domain (the source domain), which may contain sufficient offline data. The goal of cross-domain offline RL is to utilize the datasets from both the source domain and the target domain to learn an effective policy for the target environment.

Although cross-domain offline RL is promising, simply merging the source domain dataset and target domain dataset for policy training induces policy divergence and suboptimal performance (Wen et al., 2024). The issue stems from the dynamics mismatch: the transition dynamics of the source domain may differ from that of the target domain. Recent advances tackle this issue by learning domain classifiers to estimate the dynamics gap (Liu et al., 2022), or by filtering source domain data based on mutual information (Wen et al., 2024) or optimal transport (Lyu et al., 2025). These works focus on enhancing the *train-time robustness* of the policy against dynamics shifts, that is, handling the source-target dynamics mismatch. However, they overlook the occurrence of potential dynamics shifts during deployment of the learned policy in real-world environments. For example, an RL policy for robotics manipulation is trained on data collected from a real robot (target domain data) and an imperfect simulator (source domain data). When the policy is deployed on the real robot,

the robot's physical components may degrade over time, causing the transition dynamics to deviate from that observed in the target domain dataset. Consequently, the policy's performance may deteriorate during deployment, highlighting the need for methods that ensure *test-time robustness*, that is, addressing the dynamics mismatch between the target and deployment environment.

In this paper, we initiate the investigation of dual (both train-time and test-time) robustness to dynamics shifts in cross-domain offline RL. We first empirically show that with limited target domain data, the learned policy could be highly fragile to test-time dynamics shifts. To address this issue, we propose **D**ual-**RO**bust **C**ross-domain **O**ffline RL (DROCO), bringing a new perspective on robustness specifically tailored for cross-domain offline RL, going beyond single-domain robust RL (Iyengar, 2005; Kuang et al., 2022). The core component of DROCO is a novel robust cross-domain Bellman (RCB) operator, which we theoretically prove enhances test-time robustness against dynamics perturbations while remaining conservative to the out-of-distribution (OOD) dynamics transitions (Liu et al., 2024a), thus guaranteeing train-time robustness. However, value overestimation or underestimation may occur when using the RCB operator. To mitigate this, we introduce two techniques, the dynamic value penalty and the Huber loss (Huber, 1973), to our framework, resulting in our practical DROCO algorithm. Our contributions are summarized as follows.

- We empirically demonstrate the fragility of cross-domain offline RL to test-time dynamics shifts and initiate the study of dual robustness in this setting, contributing new perspectives to the field.
- We introduce a novel RCB operator which is theoretically proven to achieve dual robustness against dynamics shifts. We further introduce dynamic value penalty and Huber loss to mitigate value overestimation or underestimation, yielding our practical algorithm, DROCO.
- Extensive experiments across diverse dynamics shift scenarios including kinematic and morphology shifts demonstrate that DROCO outperforms strong baselines and exhibits significant robustness against various test-time dynamics perturbations.

2 Preliminaries

We consider a Markov Decision Process (MDP) (Puterman, 1990) which is defined by the six-tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, r, \rho, \gamma)$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition dynamics, $\Delta(\cdot)$ is the probability simplex, $r(s,a) : \mathcal{S} \times \mathcal{A} \to [-r_{\max}, r_{\max}]$ is the reward function, ρ is the initial state distribution, and γ is the discount factor. The objective of RL is to learn a policy $\pi : \mathcal{S} \to \Delta(\mathcal{A})$ that maximizes the expected discounted cumulative return $\mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$. We define $Q^\pi(s,a) := \mathbb{E}_\pi\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) | s_0 = s, a_0 = a\right]$ and $V^\pi(s) := \mathbb{E}_{a\sim\pi(\cdot|s)}\left[Q^\pi(s,a)\right]$.

Cross-Domain RL. In cross-domain RL, we have access to a *source domain* MDP $\mathcal{M}_{\text{src}} = (\mathcal{S}, \mathcal{A}, P_{\text{src}}, r, \rho, \gamma)$ and a *target domain* MDP $\mathcal{M}_{\text{tar}} = (\mathcal{S}, \mathcal{A}, P_{\text{tar}}, r, \rho, \gamma)$. The only difference between the two domains is the transition dynamics, as considered by previous works (Wen et al., 2024; Lyu et al., 2025). In the offline setting, only a target domain dataset $\mathcal{D}_{\text{tar}}$ and a source domain dataset $\mathcal{D}_{\text{src}}$ are available. We aim to leverage the mixed dataset $\mathcal{D}_{\text{src}} \cup \mathcal{D}_{\text{tar}}$ to learn a well-performing agent in the target domain.

Enhancing Robustness in RL. Robust RL aims to optimize the worst-case policy performance to enhance the robustness against environmental perturbations. Different from standard RL, robust RL applies the following *robust Bellman operator* for Bellman backup:

$$\mathcal{T}_{\text{robust}} Q(s,a) = r(s,a) + \gamma \inf_{\mathcal{M}\in\mathcal{M}_\epsilon} \mathbb{E}_{s'\sim P_{\mathcal{M}}(\cdot|s,a)}\left[\max_{a'\in\mathcal{A}} Q(s',a')\right],$$

where $\mathcal{M}_\epsilon$ is the dynamics uncertainty set under some distributional distance metric. If we choose Wasserstein distance (Villani et al., 2008) as the distance metric, then $\mathcal{M}_\epsilon$ is the Wasserstein uncertainty set:

$$\mathcal{M}_\epsilon = \{\widehat{\mathcal{M}} : \mathcal{W}\left(P_{\mathcal{M}}(\cdot|s,a), P_{\widehat{\mathcal{M}}}(\cdot|s,a)\right) \le \epsilon\}, \tag{1}$$

where $\mathcal{W}\left(P_{\mathcal{M}}(\cdot|s,a), P_{\widehat{\mathcal{M}}}(\cdot|s,a)\right) = \inf_{\gamma\in\Gamma(P_{\mathcal{M}}, P_{\widehat{\mathcal{M}}})} \mathbb{E}_{s'_1, s'_2\sim\gamma}\left[d(s'_1, s'_2)\right]$ is the Wasserstein distance between $P_{\mathcal{M}}(\cdot|s,a)$ and $P_{\widehat{\mathcal{M}}}(\cdot|s,a)$, $\Gamma(\cdot,\cdot)$ is the joint distribution, and $d(\cdot,\cdot)$ is an element-wise distance metric such as the Euclidean distance.

3 Is Cross-Domain Offline RL Sensitive to Test-Time Dynamics Perturbations?

To motivate our approach, we conduct an empirical study on the sensitivity of cross-domain offline RL to test-time perturbations. Our key finding is that cross-domain offline RL could be *highly sensitive* to test-time dynamics perturbations, especially when limited target domain data is given. Therefore, enhancing test-time robustness is crucial for cross-domain offline RL.

We adopt the `hopper-v2` task from MuJoCo (Todorov et al., 2012) as our target domain, and the full-size `hopper-expert-v2` dataset from D4RL (Fu et al., 2020) as the target domain dataset. To simulate dynamics shifts in the source domain, we create a modified `hopper-v2` environment with *kinematic shifts* (called `hopper-kinematic-v2`) by constraining the robot's joint rotation range. For the source domain dataset, we train an expert-level SAC (Haarnoja et al., 2018) policy and collect 1M samples in `hopper-kinematic-v2` environment with it. To examine the test-time robustness of cross-domain offline RL, we first train a policy using IGDF (Wen et al., 2024) on the full-size source and target domain datasets for 1M steps. We then evaluate the trained policy under four conditions: (1) the original target environment (*clean*), and (2-4) kinematic perturbations with three levels (*easy*, *medium*, *hard*) following Lyu et al. (2024b). As shown by the blue curve in Figure 1, the policy demonstrates vulnerability to intense dynamics shifts, with performance degradation of 40.9% (*medium*) and 72.4% (*hard*) compared to the *clean* environment.

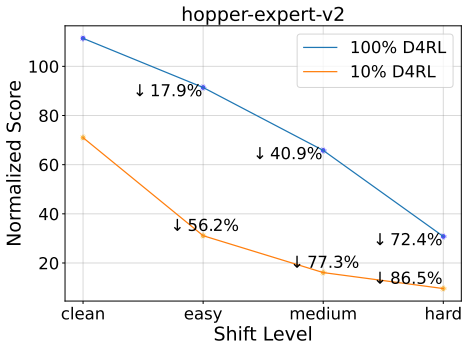


Figure 1: Performance comparison with different dataset sizes under dynamics perturbations.

To better mimic the challenges when target domain data is limited in cross-domain offline RL, we construct a reduced target domain dataset by sampling only 10% of the `hopper-expert-v2` dataset. Our experiments reveal that the policy trained with this limited target data (while retaining full source domain data) is significantly more vulnerable to dynamics perturbations. As illustrated by the orange curve in Figure 1, performance degradation intensifies across all shift levels compared to the full-data case, demonstrating substantially reduced test-time robustness.

We attribute this phenomenon to the discrepancy between the true dynamics and the observed dynamics in the target domain dataset, whose magnitude inversely correlates with the dataset size. This discrepancy causes the policy to overfit to the dataset dynamics, thereby reducing its robustness to dynamics perturbations. These results highlight the necessity of enhancing test-time robustness against dynamics shifts for cross-domain offline RL, which we address in the following section.

4 Dual-Robust Cross-Domain Offline RL

In this section, we present our solution for fulfilling dual-robustness for cross-domain offline RL. We first define the robust cross-domain Bellman (RCB) operator and additionally give a practical version of it. We then show that dual-robustness can be achieved by applying the RCB operator solely on the source domain data. Finally, we present our practical algorithm, DROCO.

4.1 Robust Cross-Domain Bellman Operator

Definition 4.1 (RCB operator)**.** *The robust cross-domain Bellman (RCB) operator $\mathcal{T}_{\mathrm{RCB}}$ is defined as*

$$\mathcal{T}_{\mathrm{RCB}}Q = \begin{cases} r + \gamma \mathbb{E}_{s' \sim P_{\mathcal{M}}} \left[\max_{a' \sim \hat{\mu}(\cdot|s')} Q(s', a') \right], & \text{if } \mathcal{M} = \mathcal{M}_{\mathrm{tar}} \\ r + \gamma \inf_{\widehat{\mathcal{M}} \in \mathcal{M}_\epsilon} \mathbb{E}_{s' \sim P_{\widehat{\mathcal{M}}}} \left[\max_{a' \sim \hat{\mu}(\cdot|s')} Q(s', a') \right], & \text{if } \mathcal{M} = \mathcal{M}_{\mathrm{src}}, \end{cases} \tag{2}$$

where $\hat{\mu}(\cdot|\cdot)$ is the behavior policy, and $\max_{a' \sim \hat{\mu}(\cdot|s')} Q(s', a')$ denotes taking maximum over actions in the support of $\hat{\mu}(\cdot|s')$, i.e., $\max_{a' \in \mathcal{A} \text{ s.t. } \hat{\mu}(a'|s')>0} Q(s', a')$.

In Equation 2, we assume that the source and target domain datasets share the same behavior policy $\hat{\mu}$, following (Wen et al., 2024). Note that this assumption is **only for notational simplicity.** Even if it does not hold, we could replace $\hat{\mu}$ with the respective behavior policies without affecting our analysis. The basic idea behind the RCB operator is that if the triplet (s, a, s') comes from the target domain dataset, we use the standard in-sample Bellman operator (Kostrikov et al., 2021; Xu et al., 2023) for backup to enhance the performance; while if the data are sampled from the source domain dataset, we apply the in-sample robust Bellman operator (which integrates in-sample learning into the robust Bellman operator) to achieve dual robustness to dynamics shifts, which we discuss later.

We now characterize the dynamic programming property of the RCB operator and give the following proposition. All proofs are deferred to Appendix B.

Proposition 4.1 (γ-contraction)**.** *The RCB operator is a γ-contraction operator in the complete state-action space $(\mathbb{R}^{|\mathcal{S}\times\mathcal{A}|}, \|\cdot\|_\infty)$ where $\|\cdot\|_\infty$ denotes the ℓ_∞ norm, i.e., $\|\mathcal{T}_{RCB}Q_1 - \mathcal{T}_{RCB}Q_2\|_\infty \leq \gamma\|Q_1 - Q_2\|_\infty$ for any Q-functions Q_1 and Q_2.*

Proposition 4.1 presents that the RCB operator is a γ-contraction in the tabular MDP setting. However, directly applying the RCB operator for backup is unrealistic, since we are not available to the uncertainty set $\mathcal{M}_\epsilon$, given that the source environment is a blackbox. To handle this issue, we introduce the following dual reformulation of Equation 2 under Wasserstein distance measure.

Proposition 4.2 (Dual Reformulation)**.** *Let $\mathcal{M}_\epsilon$ be the Wasserstein uncertainty set defined by Equation 1, then the term $\inf_{\widehat{\mathcal{M}}\in\mathcal{M}_\epsilon} \mathbb{E}_{s'\sim P_{\widehat{\mathcal{M}}}}\left[\max_{a'\sim\hat{\mu}(\cdot|s')} Q(s', a')\right]$ in Equation 2 is equivalent to*

$$\mathbb{E}_{s'\sim P_{\mathcal{M}}}\left[\inf_{\bar{s}} \max_{a'\sim\hat{\mu}(\cdot|\bar{s})} Q(\bar{s}, a')\right], \quad s.t. \quad d(s', \bar{s}) \leq \epsilon.$$

Proposition 4.2 provides a solution for transforming the intractable dynamics disturbance into the tractable state perturbations. Based on Proposition 4.2, we propose the practical RCB operator.

Definition 4.2 (Practical RCB operator)**.** *The practical RCB operator $\widehat{\mathcal{T}}_{\text{RCB}}$ is defined as*

$$\widehat{\mathcal{T}}_{\text{RCB}}Q = \begin{cases} r + \gamma\mathbb{E}_{s'\sim P_{\mathcal{M}}}\left[\max\limits_{a'\sim\hat{\mu}(\cdot|s')} Q(s', a')\right], & \text{if } \mathcal{M} = \mathcal{M}_{\text{tar}} \\ r + \gamma\mathbb{E}_{s'\sim P_{\mathcal{M}}}\left[\inf_{\bar{s}\in U_\epsilon(s')} \max\limits_{a'\sim\hat{\mu}(\cdot|\bar{s})} Q(\bar{s}, a')\right], & \text{if } \mathcal{M} = \mathcal{M}_{\text{src}} \end{cases}$$

where $U_\epsilon(s') = \{\bar{s} \in \mathcal{S} \,|\, d(s', \bar{s}) \leq \epsilon\}$ is the state uncertainty set.

The key distinction between $\widehat{\mathcal{T}}_{\text{RCB}}$ and $\mathcal{T}_{\text{RCB}}$ lies in their Bellman target computation for source domain data. While $\mathcal{T}_{\text{RCB}}$ requires the dynamics uncertainty set $\mathcal{M}_\epsilon$ that is typically unavailable, $\widehat{\mathcal{T}}_{\text{RCB}}$ solely relies on the state uncertainty set $U_\epsilon(s')$. Since s' is observable in the source domain dataset, $U_\epsilon(s')$ can be constructed through noise perturbations of s'. This makes $\widehat{\mathcal{T}}_{\text{RCB}}$ more feasible for Bellman backup than $\mathcal{T}_{\text{RCB}}$, and the subsequent analyses are based on $\widehat{\mathcal{T}}_{\text{RCB}}$. Moreover, the following proposition shows that $\widehat{\mathcal{T}}_{\text{RCB}}$ still possesses the same favorable property as $\mathcal{T}_{\text{RCB}}$, i.e., $\widehat{\mathcal{T}}_{\text{RCB}}$ remains a γ-contraction.

Proposition 4.3 (γ-contraction)**.** *The practical RCB operator is a γ-contraction operator in the space $(\mathbb{R}^{|\mathcal{S}\times\mathcal{A}|}, \|\cdot\|_\infty)$, i.e., $\|\widehat{\mathcal{T}}_{RCB}Q_1 - \widehat{\mathcal{T}}_{RCB}Q_2\|_\infty \leq \gamma\|Q_1 - Q_2\|_\infty$ for any Q_1 and Q_2.*

4.2 Dual Robustness against Dynamics Shifts

In this section, we conduct a comprehensive analysis of both train-time and test-time robustness against dynamics shifts when employing the practical RCB operator. We first make the Lipschitz continuity assumption about the learned Q function, which is widely used in prior theoretical studies of RL (Mao et al., 2024; Ran et al., 2023; Xiong et al., 2022; Liu et al., 2024b).

Assumption 4.1 (Lipschitz Q function)**.** *The learned Q function is K_Q-Lipschitz w.r.t. state s, i.e., $\forall a \in \mathcal{A}$, $\forall s_1, s_2 \in \mathcal{S}$, $|Q(s_1, a) - Q(s_2, a)| \leq K_Q \|s_1 - s_2\|$.*

We then analyze the train-time robustness against dynamics shifts from source domain data. Standard Bellman updates on source domain data might cause Q overestimation due to OOD dynamics issues (Liu et al., 2024a; Niu et al., 2022), necessitating a conservative Q estimation for robust performance. Proposition 4.4 shows that the learned $\hat{Q}_{\text{RCB}}$ maintains bounded by applying $\widehat{\mathcal{T}}_{\text{RCB}}$.

Proposition 4.4 (Train-time robustness against dynamics shifts)**.** *If a proper ϵ is selected such that* $\text{support}(P_{\text{tar}}(\cdot|s,a)) \subseteq U_\epsilon(s'_{\text{src}})$*, for any (s,a,s'_{src}) in the source domain dataset. Then under Assumption 4.1, the learned Q function by applying $\widehat{\mathcal{T}}_{RCB}$ satisfies:*

$$Q^\star_{\hat{\mu}}(s,a) - \frac{2\gamma\epsilon K_Q}{1-\gamma} \leq \hat{Q}_{RCB}(s,a) \leq Q^\star_{\hat{\mu}}(s,a), \quad \forall (s,a) \in \mathcal{D}_{\text{src}},$$

where $Q^\star_{\hat{\mu}}$ is the Q function of optimal $\hat{\mu}$-supported policy[1] *in the target domain.*

Proposition 4.4 suggests that, if a proper ϵ is chosen such that the uncertainty set $U_\epsilon(s'_{\text{src}})$ covers the support of $P_{\text{tar}}(\cdot|s,a)$, then the erroneous value overestimation will not occur, and the OOD dynamics issue is mitigated. Thus, the train-time robustness against dynamics shifts is guaranteed. We then analyze the test-time robustness against the environmental dynamics perturbations. Let π_{RCB} and $\widehat{V}_{\text{RCB}}$ be the policy and value function learned by applying the practical RCB operator, respectively. When the target environment undergoes dynamics perturbations ($P_{\text{tar}}(\cdot) \to P_{\text{per}}(\cdot)$), the value function of π_{RCB} within perturbed dynamics P_{per}, denoted as $V^{\pi_{\text{RCB}}}_{\text{per}}$, is bounded by Proposition 4.5.

Proposition 4.5 (Test-time robustness against dynamics shifts)**.** *If a proper ϵ is selected such that* $\text{support}(P_{\text{tar}}(\cdot|s,a)) \subseteq U_\epsilon(s'_{\text{src}})$*, for any (s,a,s'_{src}) in the source domain dataset. As long as $\mathcal{W}(P_{\text{per}}(\cdot|s,a), P_{\text{tar}}(\cdot|s,a)) \leq c$, then for $\forall s_0 \in \mathcal{D}_{\text{src}}$, we have*

$$V^{\pi_{\text{RCB}}}_{\text{per}}(s_0) \geq \widehat{V}_{\text{RCB}}(s_0), \tag{3}$$

where $c = \max\{c \,|U_c(s'_{\text{tar}}) \subseteq U_\epsilon(s'_{\text{src}}), s'_{\text{tar}} \sim P_{\text{tar}}(\cdot|s,a), (s,a,s'_{\text{src}}) \sim \mathcal{D}_{\text{src}}\}$.

Proposition 4.5 gives that, for any disturbance with intensity below the threshold c (measured in Wasserstein distance), the value of the learned policy π_{RCB} in P_{per} exceeds $\widehat{V}_{\text{RCB}}$. This implies that for any initial state $s_0 \in \mathcal{D}_{\text{src}}$, π_{RCB} achieves better performance under perturbed dynamics P_{per} than in the worst-case scenario, thereby improving test-time robustness against dynamics shifts. Furthermore, Proposition 4.4 and Proposition 4.5 reveal that, **(1)** dual robustness could be achieved by solely applying the RCB operator to the source domain data; **(2)** there is a trade-off between two robustness and is controlled by ϵ. More discussions can be found in Appendix A.

4.3 Practical Algorithm

In Section 4.1, we formalize the practical RCB operator. Its application presents two key challenges: (1) determining the uncertainty set $U_\epsilon(s'_{\text{src}})$; (2) computing the minimum Q value within this set. Although one can fix ϵ and adopt random sampling within $U_\epsilon(s'_{\text{src}})$, it lacks flexibility. In addition, if P_{src} deviates far from P_{tar}, then ϵ would be too large, leading to overconservatism, compromising the performance. To address these limitations, we propose our practical algorithm, DROCO.

Determining the uncertainty set via ensemble dynamics modeling. Instead of fixing ϵ and randomly sampling from $U_\epsilon(s'_{\text{src}})$, DROCO first trains an ensemble dynamics model (Janner et al., 2019; Yu et al., 2020; Liu et al., 2024c) $\widehat{P}_\psi(\cdot) = \{\widehat{P}_{\psi_i}(\cdot)\}_{i=1}^N$ on $\mathcal{D}_{\text{tar}}$ via maximum likelihood estimation (MLE) to simulate $P_{\text{tar}}(\cdot)$:

$$\mathcal{L}_{\psi_i} = \mathbb{E}_{(s,a,s')\in\mathcal{D}_{\text{tar}}}\left[\log \widehat{P}_{\psi_i}(s'|s,a)\right], \qquad i = 1,2,...,N \tag{4}$$

then we use the ensemble prediction set $\mathcal{X} = \left\{s'_1, \cdots, s'_N | s'_i \sim \widehat{P}_{\psi_i}(\cdot|s,a), (s,a) \in \mathcal{D}_{\text{src}}\right\}$ to approximate sampling from the uncertainty set. This replacement is motivated by two key insights: (1) dual-robustness only requires the uncertainty set around support of $P_{\text{tar}}(\cdot|s,a)$ rather than s'_{src}, thus alleviating the unnecessary conservatism; (2) each ensemble member's prediction naturally serves as a sample from this uncertainty set. In this way, the practical RCB operator for source domain data becomes:

$$\widehat{\mathcal{T}}_{\text{RCB}}Q = r + \gamma \inf_{\{s'_i\}^N \sim \widehat{P}_{\psi_i}} \left[\max_{a'_i \sim \hat{\mu}(\cdot|s'_i)} Q(s'_i, a'_i)\right], \qquad \text{if } \mathcal{M} = \mathcal{M}_{\text{src}}. \tag{5}$$

[1] $\pi(\cdot|s)$ is $\hat{\mu}$-supported if $\pi(a|s) = 0$ for any action a that $\hat{\mu}(a|s) = 0$.

However, the ensemble prediction set cannot cover the support of $P_{\text{tar}}(\cdot)$ as required in Proposition 4.4, such that the overestimation of Q value may still occur. Proposition 4.6 reveals that, only limited overestimation would occur when applying Equation 5 as the Bellman target.

Proposition 4.6 (Limited overestimation). *If* $\sup_{s,a} D_{TV}(\widehat{P}_\psi(\cdot|s,a), P_{\text{tar}}(\cdot|s,a)) \leq \epsilon < \frac{1}{2}$, *we have*

$$\inf_{\{s_i'\}^N \sim \widehat{P}_{\psi_i}(\cdot|s,a)} \left[\max_{a_i' \sim \hat{\mu}(\cdot|s_i')} Q(s_i', a_i') \right] \leq \mathbb{E}_{s' \sim P_{\text{tar}}(\cdot|s,a)} \left[\max_{a' \sim \hat{\mu}(\cdot|s')} Q(s', a') \right] + (1-(1-2\epsilon)^N) \frac{r_{\max}}{1-\gamma}.$$

Proposition 4.6 holds under the assumption that the prediction error of the dynamics model stays small, which is difficult to fulfill given that the target domain data is limited and the dynamics model tends to overfit. Moreover, value underestimation may also occur due to the infimum operator. Therefore, we introduce the following techniques for the underlying value estimation issue.

Tackling overestimation and underestimation. We adopt two techniques to address the value estimation issue: dynamic value penalty, and using Huber loss (Huber, 1973) for Bellman update.

Instead of directly using Equation 5 for Bellman backup, we introduce a value penalty term:

$$u(s,a,s') = \mathbb{I}\left(s' \sim P_{\text{src}}(\cdot|s,a)\right) \cdot \left(\max_{a' \sim \hat{\mu}(\cdot|s')} Q(s',a') - \inf_{\{s_i'\}^N \sim \widehat{P}_{\psi_i}(\cdot|s,a)} \left[\max_{a_i' \sim \hat{\mu}(\cdot|s_i')} Q(s_i', a_i') \right] \right). \tag{6}$$

We then unify the source and target dynamics in the practical RCB operator, reformulating it as

$$\widehat{\mathcal{T}}_{\text{RCB}} Q(s,a) = r(s,a) + \gamma \mathbb{E}_{s' \sim P_{\mathcal{M}}(\cdot|s,a)} \left[\max_{a' \sim \hat{\mu}(\cdot|s')} Q(s',a') - \beta \cdot u(s,a,s') \right], \tag{7}$$

where $\mathcal{M} = \mathcal{M}_{\text{tar}}$ or $\mathcal{M}_{\text{src}}$ and β serves as a dynamic penalty coefficient that provides flexible control over value estimation. Specifically, we recover the practical RCB operator by setting β to 1.0, $\beta > 1.0$ will increase the penalty to mitigate value overestimation, and $\beta < 1.0$ reduces the penalty to alleviate value underestimation. Although the dynamics model and value penalty are widely applied in offline RL (Yu et al., 2020; Sun et al., 2023; Liu et al., 2024c), our difference lies in the specific usage of the dynamics model and design of the penalty term.

Remark. If we use IQL (Kostrikov et al., 2021) for policy optimization, then $\max_{a' \sim \hat{\mu}(\cdot|s')} Q(s',a') \approx V(s')$, and Equation 6 can be re-written as:

$$u(s,a,s') = \mathbb{I}\left(s' \sim P_{\text{src}}(\cdot|s,a)\right) \cdot \left(V(s') - \inf_{\{s_i'\}^N \sim \widehat{P}_{\psi_i}(\cdot|s,a)} \left[V(s_i') \right] \right). \tag{8}$$

We note that Equation 8 resembles the value discrepancy term in VGDF (Xu et al., 2024), which is $V(s') - \mathbb{E}_{\{s_i'\}^N \sim \widehat{P}_{\psi_i}(\cdot|s,a)} [V(s_i')]$. However, we extend this term by incorporating additional penalties for test-time dynamics shifts, whereas VGDF only addresses train-time dynamics shifts.

The second technique we adopt is the Huber loss (Huber, 1973), a well-established technique for noise-resistant optimization (Yang et al., 2024b; Roy et al., 2021). We replace the regular ℓ_2 loss in the Bellman update with the Huber loss:

$$\mathcal{L}_Q = \mathbb{E}_{\mathcal{D}_{\text{src}}} \left[l_\delta \left(Q(s,a) - \widehat{\mathcal{T}}_{\text{RCB}} Q(s,a) \right) \right] + \frac{1}{2} \mathbb{E}_{\mathcal{D}_{\text{tar}}} \left[\left(Q(s,a) - \mathcal{T} Q(s,a) \right)^2 \right], \tag{9}$$

where $l_\delta(a) = \begin{cases} 0.5a^2, & |a| < \delta \\ \delta(|a| - 0.5\delta), & |a| \geq \delta \end{cases}$ with δ being the transition threshold and $\mathcal{T}$ being the standard Bellman operator for target domain data. Specifically, if severe value estimation error occurs such that $|Q(s,a) - \widehat{\mathcal{T}}_{\text{RCB}} Q(s,a)| > \delta$, the ℓ_2 loss would transition to ℓ_1 loss to improve robustness against outliers. This technique helps mitigate value estimation error. The last step is to utilize offline RL algorithms such as IQL to optimize the policy as in other works (Lyu et al., 2025; Wen et al., 2024). We present the detailed pseudo-code of DROCO in Appendix D.2.

Table 1: **Evaluation Results with train-time kinematic shifts.** half=halfcheetah, hopp=hopper, walk=walker2d, m=medium, me=medium-expert, mr=medium-replay, e=expert. We report the normalized score evaluated in the target domain and ± captures the standard deviation across 5 seeds.

Dataset	IQL*	CQL*	BOSA	DARA	IGDF	OTDF	DROCO (Ours)
half-m	**45.2**	37.7	39.6	44.1	**45.2±0.1**	42.2±0.1	**45.3±0.2**
half-mr	22.1	23.6	**26.3**	21.6	22.9±1.4	15.6±3.1	**26.9±3.2**
half-me	43.7	54.8	42.2	52.7	57.1±8.9	46.7±4.4	**60.1±7.1**
half-e	49.7	36.0	**84.3**	47.4	47.6±2.1	79.6±3.0	67.4±5.8
hopp-m	48.8	35.7	**71.4**	48.8	54.3±6.6	46.3±3.7	55.4±5.3
hopp-mr	40.2	43.2	29.5	41.6	30.0±5.2	26.2±4.4	**47.3±7.0**
hopp-me	12.5	7.8	49.6	17.0	11.6±0.6	**58.1±4.9**	54.0±6.4
hopp-e	62.6	47.9	94.8	59.1	70.1±3.2	**97.0±3.3**	89.3±9.6
walk-m	48.7	47.7	44.5	43.4	51.8±2.4	43.0±2.1	**70.8±3.3**
walk-mr	12.6	17.8	4.8	15.6	11.2±1.1	10.7±1.9	**27.7±3.0**
walk-me	**95.4**	61.4	35.1	85.3	90.6±3.4	63.1±6.6	78.5±6.7
walk-e	90.1	83.8	41.9	85.5	93.7±5.8	98.9±2.1	**106.0±0.8**
ant-m	89.9	58.2	28.4	**98.9**	88.0±4.6	86.1±3.7	92.7±6.3
ant-mr	46.8	39.4	22.0	42.1	**58.2±9.7**	39.6±8.1	44.8±4.5
ant-me	106.1	100.6	102.5	104.8	112.8±4.0	105.1±3.9	**119.0±3.6**
ant-e	111.0	94.3	57.6	115.1	**119.2±5.6**	111.6±2.9	**120.0±2.1**
Total	925.4	789.9	774.5	923.0	964.3	969.8	**1105.2**

5 EXPERIMENTS

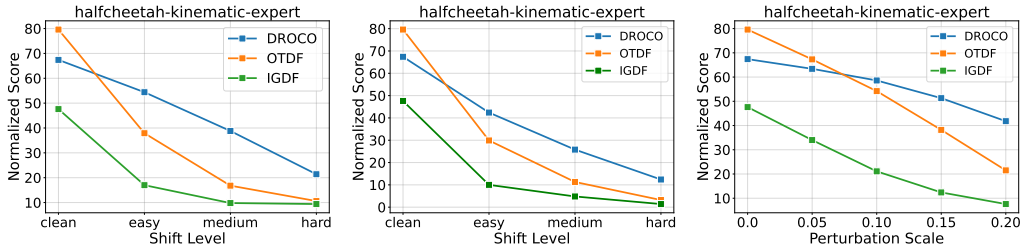
In this section, we conduct extensive experiments to examine our method. We aim to answer the following two questions: (1) Can DROCO outperform prior strong baselines across various train-time dynamics shifts and dataset qualities? (2) Can DROCO show enhanced robustness against test-time dynamics perturbations? We also test the parameter sensitivity of DROCO.

5.1 MAIN RESULTS

Experimental Settings. Following previous works (Lyu et al., 2025; Wen et al., 2024), We employ 4 MuJoCo (Todorov et al., 2012) tasks as source domains: `halfcheetah-v2`, `hopper-v2`, `walker2d-v2`, and `ant-v3`. For the target domain datasets, we utilize 4 data qualities from D4RL (Fu et al., 2020) for each task: `medium`, `medium-replay`, `medium-expert`, and `expert`, totaling 16 target domain datasets. For the source domain, we introduce kinematic shifts and morphology shifts on the target domain setup. We collect source domain datasets with 4 data qualities, resulting in a total of 32 ($4[\texttt{tasks}] \times 2[\texttt{shift types}] \times 4[\texttt{data qualities}]$) source domain datasets. Each pair of the source and target domain datasets shares the same task type (such as `hopper-v2`) and dataset quality (such as `expert`). More details about the tasks and datasets can be found in Appendix C.1.

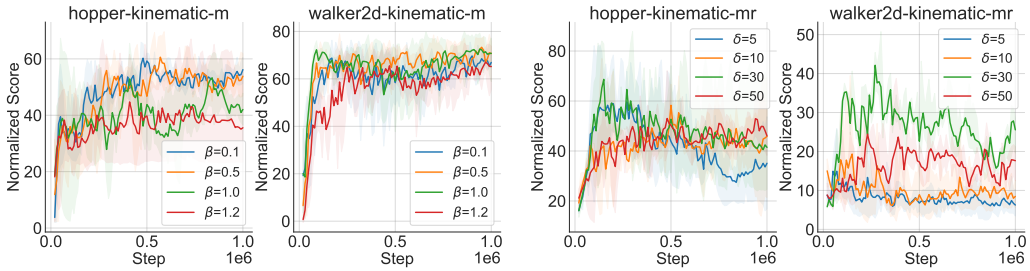
Baselines. We consider the following baselines: **IQL***, **CQL*** (which train IQL (Kostrikov et al., 2021) and CQL (Kumar et al., 2020) with the mixed dataset $\mathcal{D}_{\text{tar}} \cup \mathcal{D}_{\text{src}}$), **BOSA** (Liu et al., 2024a), **DARA** (Liu et al., 2022), **IGDF** (Wen et al., 2024) and **OTDF** (Lyu et al., 2025).

Results. We run each baseline and DROCO for 1M training steps over 5 random seeds, and report the results with train-time kinematic shifts in Table 1 (the results with morphology shifts are deferred to Appendix E.2). Note that our evaluation is under the clean target environment. Empirical results demonstrate that DROCO achieves superior performance in **9** out of 16 tasks, outperforming all 6 baselines. Furthermore, in terms of the total normalized score, DROCO achieves a remarkable **1105.2**, significantly surpassing the second-best method, OTDF (969.8), by **14.0%**. We attribute the suboptimal performance of DROCO on the remaining datasets to its trade-off between performance and robustness, whereas other methods only consider performance. However, DROCO still remains a competitive performance with other baselines on these datasets. These results indicate that DROCO exhibits superior train-time robustness against dynamics shifts.



(a) Kinematic shift (b) Morphology shift (c) min Q perturbation

Figure 2: Evaluation results under different types and levels of dynamics perturbations.



(a) Effect of β (b) Effect of δ

Figure 3: Parameter sensitivity experiments on β and δ.

5.2 Evaluation under Dynamics Perturbations

Experimental Settings. Test-time robustness can be measured by the degree of performance degradation in the face of dynamic perturbations, compared to the clean environment. To examine the test-time robustness of DROCO, we introduce three kinds of dynamics perturbations during evaluation in the target environment: kinematic perturbations, morphology perturbations, min Q perturbations (Yang et al., 2022; 2024b). The first two perturbation types mirror the source domain dynamics shifts, each implemented at three intensity levels following (Lyu et al., 2024b): *easy*, *medium*, and *hard*. The third perturbation type represents an adversarial attack strategy that modifies dynamics by finding the state $\bar{s}$ from $U_\epsilon(s')$ that minimizes the Q value, i.e., $\bar{s} = \arg\min_{\bar{s}\in U_\epsilon(s')} Q(\bar{s}, \pi(\bar{s}))$, where ϵ is related to the perturbation scale. We consider DROCO to exhibit better test-time robustness if it demonstrates less performance degradation than the baselines under the same shift severity.

Results. We evaluate our method against two baselines (IGDF and OTDF) under three perturbation types with varying intensity levels. Due to space constraints, we only report results when the source domain dataset is `halfcheetah-kinematic-expert`, as illustrated in Figure 2. A wider range of evaluation can be found in Appendix E.3.

Our experiments demonstrate that DROCO exhibits superior robustness to dynamic perturbations compared to baseline methods. Specifically, under easy-level kinematic shifts, DROCO shows only a **19.3%** performance degradation (from 67.4 to 54.4), whereas both IGDF and OTDF suffer over 50% performance deterioration. We notice that DROCO displays greater sensitivity to morphological perturbations than to kinematic perturbations, with a 42.1% performance decrease under easy-level morphological variations. We attribute this to the absence of morphology shifts in the source domain data, rendering the policy less adaptable to this unseen perturbation type. Nevertheless, DROCO still outperforms both baselines: under the same conditions, OTDF and IGDF exhibit performance declines of 62.4% and 78.9% respectively. Notably, DROCO maintains consistent robustness against min Q perturbations across all scales. At the highest perturbation scale of 0.2, DROCO's performance decreases by **37.9%**, compared to 73.6% and 84.0% for OTDF and IGDF.

5.3 Parameter Sensitivity

We examine the sensitivity of DROCO to the introduced hyperparameters. There are two main hyperparameters in DROCO: the penalty coefficient β and the transition threshold δ.

Penalty coefficient β. The parameter β controls the intensity of value penalty, a larger β leads to a stronger penalty and suppresses value overestimation, and vice versa. We sweep β across $\{0.1, 0.5, 1.0, 1.2\}$ and show the experimental results with `medium` datasets in Figure 3 (a). We observe different tasks prefer distinct β. For example, setting $\beta = 0.1$ achieves the best performance for `hopper-kinematic-medium`, while `walker2d-kinematic-medium` prefers $\beta = 1.0$.

Transition threshold δ. The parameter δ determines when the ℓ_2 loss turns to ℓ_1 loss for Bellman update. A larger δ corresponds to a more lenient transition condition. To test the effect of δ, we select δ among $\{5, 10, 30, 50\}$ and conduct experiments with `medium-replay` datasets. The results in Figure 3 (b) indicate that a too small δ (e.g., $\delta = 5$) leads to inferior performance, while setting $\delta = 30$ achieves a good performance. However, we find the optimal δ varies across different tasks through additional experiments, and more discussions are provided in Appendix E.5.

Remark. Although different values of β and δ are preferred for different tasks (as shown in Appendix E.5), we could still find some patterns across different tasks. We find that setting $\beta \leq 1.0$ works for most tasks, implying that value underestimation occurs more often due to the infimum operator. We also find a larger δ (30 and 50) is preferred for most tasks. We believe it is because ℓ_2 loss is beneficial for training stability. Therefore, for a new task, we could first try $\beta \leq 1.0$ and $\delta = 30$ (or $\delta = 50$). This could serve as a guideline for finding the best hyperparameter.

6 Related Work

Offline RL. Offline RL often suffers from OOD action issues (Kumar et al., 2020; Fujimoto et al., 2019). Existing solutions include incorporating policy constraints (Kumar et al., 2019; Fujimoto & Gu, 2021), learning a conservative value function (Kumar et al., 2020; Lyu et al., 2022), etc. However, these methods require that the offline dataset contains a large amount of data. In contrast, we focus on cross-domain offline RL, which relaxes the target data requirement.

Cross-domain RL. Cross-domain RL (Niu et al., 2024) faces the challenge of domain mismatch, including observation mismatch (Yang et al., 2023), viewpoints mismatch (Liu et al., 2018; Sadeghi et al., 2018), and dynamics mismatch (Wen et al., 2024; Lyu et al., 2025; Xu et al., 2024; Niu et al., 2022; 2023), etc. In this paper, we exclusively focus on dynamics mismatch. Previous studies handle this issue by adaptively penalizing Q value on source domain samples (Niu et al., 2022), capturing dynamics mismatch from a representation learning perspective (Lyu et al., 2024a) and value discrepancy perspective (Xu et al., 2024), modifying the reward function in the source domain (Liu et al., 2022; Eysenbach et al., 2020; Xue et al., 2023; Wang et al., 2024), etc. We focus on the offline setting, where the current works (Liu et al., 2022; 2024a; Wen et al., 2024; Lyu et al., 2025; Wang et al., 2024) primarily consider the dynamics shifts from the source domain data, while we further consider the dynamics shifts from environmental perturbations.

Robust RL. Robust RL (Iyengar, 2005; Xu & Mannor, 2010) aims to learn a policy resilient to environmental perturbations or data corruption. One line of research in robust RL focuses on train-time robustness against data corruption (Yang et al., 2024b; Zhang et al., 2021; 2022; Ye et al., 2023; Yang et al., 2024a), while another line addresses test-time robustness against environmental perturbations (Yang et al., 2022; Zhihe & Xu, 2023; Shi & Chi, 2024; Liu et al., 2024c). These works focus only on a single perspective of robustness (train-time or test-time) and the single-domain offline settings. For the cross-domain setting, (He et al., 2025; Liu & Xu, 2024a) study robust off-dynamics RL, but they are different from our work, since they still only consider one aspect of robustness and focus on the online setting. In contrast, our work addresses the cross-domain offline setting and jointly considers both train-time and test-time robustness.

7 Conclusion

In this paper, we investigate the dual (train-time and test-time) robustness against dynamics shifts in cross-domain offline RL. We propose a novel RCB operator and theoretically demonstrate its ability

of dual robustness. To further handle the potential value estimation error, we add a dynamic value penalty and use Huber loss for Bellman update, yielding our practical DROCO algorithm. Through extensive experiments across various dynamics shifts scenarios, we show that DROCO outperforms prior strong baselines and exhibits strong robustness to dynamics perturbations.

REFERENCES

Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

Benjamin Eysenbach, Swapnil Asawa, Shreyas Chaudhari, Sergey Levine, and Ruslan Salakhutdinov. Off-dynamics reinforcement learning: Training for transfer with domain classifiers. *arXiv preprint arXiv:2006.13916*, 2020.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*, pp. 2052–2062. PMLR, 2019.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. Pmlr, 2018.

Yiting He, Zhishuai Liu, Weixin Wang, and Pan Xu. Sample complexity of distributionally robust off-dynamics reinforcement learning with online interaction. In *Forty-second International Conference on Machine Learning*, 2025.

Peter J Huber. Robust regression: asymptotics, conjectures and monte carlo. *The annals of statistics*, pp. 799–821, 1973.

Garud N Iyengar. Robust dynamic programming. *Mathematics of Operations Research*, 30(2): 257–280, 2005.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in neural information processing systems*, 32, 2019.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Diederik P Kingma, Max Welling, et al. Auto-encoding variational bayes, 2013.

Lingkai Kong, Haichuan Wang, Tonghan Wang, Guojun Xiong, and Milind Tambe. Composite flow matching for reinforcement learning with shifted-dynamics data. *arXiv preprint arXiv:2505.23062*, 2025.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

Yufei Kuang, Miao Lu, Jie Wang, Qi Zhou, Bin Li, and Houqiang Li. Learning robust policy against disturbance in transition dynamics via state-conservative policy optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pp. 7247–7254, 2022.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. *Advances in neural information processing systems*, 32, 2019.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 33:1179–1191, 2020.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Anqi Li, Dipendra Misra, Andrey Kolobov, and Ching-An Cheng. Survival instinct in offline reinforcement learning. *Advances in neural information processing systems*, 36:62062–62120, 2023.

Jinxin Liu, Hongyin Zhang, and Donglin Wang. Dara: Dynamics-aware reward augmentation in offline reinforcement learning. *arXiv preprint arXiv:2203.06662*, 2022.

Jinxin Liu, Ziqi Zhang, Zhenyu Wei, Zifeng Zhuang, Yachen Kang, Sibo Gai, and Donglin Wang. Beyond ood state actions: Supported cross-domain offline reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 13945–13953, 2024a.

Tenglong Liu, Yang Li, Yixing Lan, Hao Gao, Wei Pan, and Xin Xu. Adaptive advantage-guided policy regularization for offline reinforcement learning. In *International Conference on Machine Learning*, pp. 31406–31424. PMLR, 2024b.

Xiao-Yin Liu, Xiao-Hu Zhou, Guotao Li, Hao Li, Mei-Jiang Gui, Tian-Yu Xiang, De-Xing Huang, and Zeng-Guang Hou. Micro: model-based offline reinforcement learning with a conservative bellman operator. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence*, pp. 4587–4595, 2024c.

YuXuan Liu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Imitation from observation: Learning to imitate behaviors from raw video via context translation. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 1118–1125. IEEE, 2018.

Zhishuai Liu and Pan Xu. Distributionally robust off-dynamics reinforcement learning: Provable efficiency with linear function approximation. In *International Conference on Artificial Intelligence and Statistics*, pp. 2719–2727. PMLR, 2024a.

Zhishuai Liu and Pan Xu. Minimax optimal and computationally efficient algorithms for distributionally robust offline reinforcement learning. *Advances in Neural Information Processing Systems*, 37:86602–86654, 2024b.

Jiafei Lyu, Xiaoteng Ma, Xiu Li, and Zongqing Lu. Mildly conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 35:1711–1724, 2022.

Jiafei Lyu, Chenjia Bai, Jing-Wen Yang, Zongqing Lu, and Xiu Li. Cross-domain policy adaptation by capturing representation mismatch. In *International Conference on Machine Learning*, pp. 33638–33663. PMLR, 2024a.

Jiafei Lyu, Kang Xu, Jiacheng Xu, Mengbei Yan, Jingwen Yang, Zongzhang Zhang, Chenjia Bai, Zongqing Lu, and Xiu Li. Odrl: A benchmark for off-dynamics reinforcement learning. In *The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2024b. URL https://openreview.net/forum?id=ap4x1kArGy.

Jiafei Lyu, Mengbei Yan, Zhongjian Qiao, Runze Liu, Xiaoteng Ma, Deheng Ye, Jing-Wen Yang, Zongqing Lu, and Xiu Li. Cross-domain offline policy adaptation with optimal transport and dataset constraint. In *The Thirteenth International Conference on Learning Representations*, 2025.

Yixiu Mao, Qi Wang, Yun Qu, Yuhang Jiang, and Xiangyang Ji. Doubly mild generalization for offline reinforcement learning. *arXiv preprint arXiv:2411.07934*, 2024.

Haoyi Niu, Yiwen Qiu, Ming Li, Guyue Zhou, Jianming Hu, Xianyuan Zhan, et al. When to trust your simulator: Dynamics-aware hybrid offline-and-online reinforcement learning. *Advances in Neural Information Processing Systems*, 35:36599–36612, 2022.

Haoyi Niu, Tianying Ji, Bingqi Liu, Haocheng Zhao, Xiangyu Zhu, Jianying Zheng, Pengfei Huang, Guyue Zhou, Jianming Hu, and Xianyuan Zhan. H2o+: an improved framework for hybrid offline-and-online rl with dynamics gaps. *arXiv preprint arXiv:2309.12716*, 2023.

Haoyi Niu, Jianming Hu, Guyue Zhou, and Xianyuan Zhan. A comprehensive survey of cross-domain policy transfer for embodied agents. *arXiv preprint arXiv:2402.04580*, 2024.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35: 27730–27744, 2022.

Kuan-Chen Pan, MingHong Chen, Xi Liu, and Ping-Chun Hsieh. Survive on planet pandora: Robust cross-domain rl under distinct state-action representations. In *ICML 2024 Workshop: Aligning Reinforcement Learning Experimentalists and Theorists*.

Gabriel Peyré, Marco Cuturi, et al. Computational optimal transport: With applications to data science. *Foundations and Trends® in Machine Learning*, 11(5-6):355–607, 2019.

Martin L Puterman. Markov decision processes. *Handbooks in operations research and management science*, 2:331–434, 1990.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.

Yuhang Ran, Yi-Chen Li, Fuxiang Zhang, Zongzhang Zhang, and Yang Yu. Policy regularization with dataset constraint for offline reinforcement learning. In *International Conference on Machine Learning*, pp. 28701–28717. PMLR, 2023.

Abhishek Roy, Krishnakumar Balasubramanian, and Murat A Erdogdu. On empirical risk minimization with dependent and heavy-tailed data. *Advances in Neural Information Processing Systems*, 34:8913–8926, 2021.

Fereshteh Sadeghi, Alexander Toshev, Eric Jang, and Sergey Levine. Sim2real viewpoint invariant visual servoing by recurrent control. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4691–4699, 2018.

Jiyuan Shi, Chenjia Bai, Haoran He, Lei Han, Dong Wang, Bin Zhao, Mingguo Zhao, Xiu Li, and Xuelong Li. Robust quadrupedal locomotion via risk-averse policy learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 11459–11466. IEEE, 2024.

Laixi Shi and Yuejie Chi. Distributionally robust model-based offline reinforcement learning with near-optimal sample complexity. *Journal of Machine Learning Research*, 25(200):1–91, 2024.

Yihao Sun, Jiaji Zhang, Chengxing Jia, Haoxin Lin, Junyin Ye, and Yang Yu. Model-bellman inconsistency for model-based offline reinforcement learning. In *International Conference on Machine Learning*, pp. 33177–33194. PMLR, 2023.

Richard S Sutton and Andrew G Barto. Reinforcement learning: An introduction. *Robotica*, 17(2): 229–235, 1999.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033. IEEE, 2012.

Cédric Villani et al. *Optimal transport: old and new*, volume 338. Springer, 2008.

Ruhan Wang, Yu Yang, Zhishuai Liu, Dongruo Zhou, and Pan Xu. Return augmented decision transformer for off-dynamics reinforcement learning. *arXiv preprint arXiv:2410.23450*, 2024.

Xiaoyu Wen, Chenjia Bai, Kang Xu, Xudong Yu, Yang Zhang, Xuelong Li, and Zhen Wang. Contrastive representation for data filtering in cross-domain offline reinforcement learning. *arXiv preprint arXiv:2405.06192*, 2024.

Huaqing Xiong, Tengyu Xu, Lin Zhao, Yingbin Liang, and Wei Zhang. Deterministic policy gradient: Convergence analysis. In *Uncertainty in Artificial Intelligence*, pp. 2159–2169. PMLR, 2022.

Haoran Xu, Li Jiang, Jianxiong Li, Zhuoran Yang, Zhaoran Wang, Victor Wai Kin Chan, and Xianyuan Zhan. Offline rl with no ood actions: In-sample learning via implicit value regularization. *arXiv preprint arXiv:2303.15810*, 2023.

Huan Xu and Shie Mannor. Distributionally robust markov decision processes. *Advances in Neural Information Processing Systems*, 23, 2010.

Kang Xu, Chenjia Bai, Xiaoteng Ma, Dong Wang, Bin Zhao, Zhen Wang, Xuelong Li, and Wei Li. Cross-domain policy adaptation via value-guided data filtering. *Advances in Neural Information Processing Systems*, 36, 2024.

Zhenghai Xue, Qingpeng Cai, Shuchang Liu, Dong Zheng, Peng Jiang, Kun Gai, and Bo An. State regularized policy optimization on data with dynamics shift. *Advances in neural information processing systems*, 36:32926–32937, 2023.

Rui Yang, Chenjia Bai, Xiaoteng Ma, Zhaoran Wang, Chongjie Zhang, and Lei Han. Rorl: Robust offline reinforcement learning via conservative smoothing. *Advances in neural information processing systems*, 35:23851–23866, 2022.

Rui Yang, Lin Yong, Xiaoteng Ma, Hao Hu, Chongjie Zhang, and Tong Zhang. What is essential for unseen goal generalization of offline goal-conditioned rl? In *International Conference on Machine Learning*, pp. 39543–39571. PMLR, 2023.

Rui Yang, Jie Wang, Guoping Wu, and Bin Li. Uncertainty-based offline variational bayesian reinforcement learning for robustness under diverse data corruptions. *arXiv preprint arXiv:2411.00465*, 2024a.

Rui Yang, Han Zhong, Jiawei Xu, Amy Zhang, Chongjie Zhang, Lei Han, and Tong Zhang. Towards robust offline reinforcement learning under diverse data corruption. In *The Twelfth International Conference on Learning Representations*, 2024b.

Chenlu Ye, Rui Yang, Quanquan Gu, and Tong Zhang. Corruption-robust offline reinforcement learning with general function approximation. *Advances in Neural Information Processing Systems*, 36:36208–36221, 2023.

Heng You, Tianpei Yang, Yan Zheng, Jianye Hao, E Taylor, et al. Cross-domain adaptive transfer reinforcement learning based on state-action correspondence. In *Uncertainty in Artificial Intelligence*, pp. 2299–2309. PMLR, 2022.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.

Kevin Zakka, Philipp Wu, Laura Smith, Nimrod Gileadi, Taylor Howell, Xue Bin Peng, Sumeet Singh, Yuval Tassa, Pete Florence, Andy Zeng, et al. Robopianist: Dexterous piano playing with deep reinforcement learning. *arXiv preprint arXiv:2304.04150*, 2023.

Qiang Zhang, Tete Xiao, Alexei A Efros, Lerrel Pinto, and Xiaolong Wang. Learning cross-domain correspondence for control with dynamics cycle-consistency. *arXiv preprint arXiv:2012.09811*, 2020.

Xuezhou Zhang, Yiding Chen, Xiaojin Zhu, and Wen Sun. Robust policy gradient against strong data corruption. In *International Conference on Machine Learning*, pp. 12391–12401. PMLR, 2021.

Xuezhou Zhang, Yiding Chen, Xiaojin Zhu, and Wen Sun. Corruption-robust offline reinforcement learning. In *International Conference on Artificial Intelligence and Statistics*, pp. 5757–5773. PMLR, 2022.

YANG Zhihe and Yunjian Xu. Dmbp: Diffusion model-based predictor for robust offline reinforcement learning against state observation perturbations. In *The Twelfth International Conference on Learning Representations*, 2023.

A MORE DISCUSSIONS OF DROCO

In this section, we provide further clarifications on several questions regarding DROCO that readers might be concerned about.

1. Why not test DROCO on more tasks such as Antmaze and Adroit?

In our experiments, we evaluate our method DROCO and other baselines on MuJoCo-based tasks (e.g., `halfcheetah-v2` and `hopper-v2`). This experimental setting is standard and has been adopted by recent works such as VGDF (Xu et al., 2024), IGDF (Wen et al., 2024), OTDF (Lyu et al., 2025), and CompFlow (Kong et al., 2025). By following these established settings, we believe our experiments are sufficient for evaluating DROCO's effectiveness.

On the other hand, existing literature (Lyu et al., 2024b) indicates that Antmaze tasks with varying map structures are highly challenging for cross-domain RL (Observation 3, p.7), as adapting policies across structural barriers remains difficult. Similarly, cross-domain RL methods often fail on dexterous hand manipulation tasks (Adroit) with kinematic or morphology shifts (Observation 4, p.8). Empirically, we find that not only DROCO but all baseline methods (e.g., BOSA, DARA, IGDF, OTDF) struggle to achieve meaningful performance on Antmaze and Adroit tasks.

We emphasize that enabling cross-domain RL to succeed in such challenging settings (Antmaze and Adroit) remains an open problem (p.17 in (Lyu et al., 2024b)) and falls beyond the scope of this work. We believe that MuJoCo tasks with dynamics shift provide a sufficient and appropriate testbed for evaluating our method.

2. Can DROCO be extended into settings where the source and target domain have distinct state-action representations?

The answer is yes. Although this work follows the setting of recent studies (e.g., BOSA, DARA, IGDF, OTDF) and assumes identical state and action spaces across source and target domains, DROCO can be generalized to domains with distinct state-action representations. This can be achieved by incorporating techniques such as inter-domain mapping via dynamics cycle consistency (Zhang et al., 2020). Other mapping methods (You et al., 2022; Pan et al.) are also compatible with our framework and could be seamlessly integrated. Thus, DROCO remains applicable even under varying state-action spaces.

3. Why the Wasserstein uncertainty set is chosen instead of other uncertainty sets?

Although there are other possible choices of uncertainty set like the (s, a)-rectangularity (Iyengar, 2005) and d-rectangularity (Liu & Xu, 2024b), their dual reformulations often result in complex constraints or regularizations and are typically limited to simple linear MDP settings. In contrast, the Wasserstein uncertainty set admits an elegant closed-form dual reformulation (Proposition 4.2), which allows converting dynamics perturbations into a simple state uncertainty set—a property critical for practical implementation. Moreover, the Wasserstein metric inherently couples state transitions, enabling a natural mapping to state perturbations and providing geometric interpretability.

4. Why not apply the RCB operator on target domain data to improve test-time robustness?

On the one hand, Propositions 4.4 and 4.5 show that dual robustness is guaranteed as long as $\text{support}(P_{\text{tar}}(\cdot|s,a)) \subseteq U_\epsilon(s'_{\text{src}})$ holds—even when the RCB operator is applied only to source domain data. Even when this condition is not fully satisfied, our techniques (dynamic value penalty and Huber loss) still enhance robustness. Therefore, applying the RCB operator to target domain data is unnecessary.

On the other hand, target domain data is important for achieving high performance on the clean target environment. Applying the RCB operator to it would introduce conservatism and compromise performance. Moreover, since target data are scarce, any improvement in test-time robustness from using them would be limited. Thus, the optimal strategy is to apply the standard Bellman operator to target data to improve performance, and the RCB operator to source data to enhance robustness.

5. Is there a trade-off between train-time and test-time robustness?

There is a trade-off between train-time and test-time robustness, and it is controlled by ϵ. Specifically, when $\text{support}(P_{\text{tar}}(\cdot|s,a)) \subseteq U_\epsilon(s'_{\text{src}})$ is satisfied, further increasing ϵ might bring excessive conservatism. While this enhances test-time robustness against dynamics shifts (since c is mono-

tonically increasing with respect to ϵ), it sacrifices performance on the clean target domain, thereby reducing train-time robustness.

B Proofs of Propositions

B.1 Proof of Proposition 4.1

Proof. We recall the definition of the RCB operator below:

$$\mathcal{T}_{\text{RCB}}Q(s,a) = \begin{cases} r + \gamma \mathbb{E}_{s' \sim P_{\mathcal{M}}} \left[\max_{a' \sim \hat{\mu}(\cdot|s')} Q(s',a') \right], & \text{if } \mathcal{M} = \mathcal{M}_{\text{tar}} \\ r + \gamma \inf_{\mathcal{M} \in \mathcal{M}_\epsilon} \mathbb{E}_{s' \sim P_{\mathcal{M}}} \left[\max_{a' \sim \hat{\mu}(\cdot|s')} Q(s',a') \right], & \text{if } \mathcal{M} = \mathcal{M}_{\text{src}} \end{cases}.$$

Let Q_1 and Q_2 be two arbitrary Q functions. Then for any state-action pair (s,a), if the next state $s' \sim P_{\text{tar}}(\cdot|s,a)$, we have

$$\begin{aligned} \|\mathcal{T}_{\text{RCB}}Q_1 - \mathcal{T}_{\text{RCB}}Q_2\|_\infty &= \gamma \max_{s,a} \left| \mathbb{E}_{s'} \left[\max_{a' \sim \hat{\mu}(\cdot|s')} Q_1(s',a') - \max_{a' \sim \hat{\mu}(\cdot|s')} Q_2(s',a') \right] \right| \\ &\leq \gamma \max_{s,a} \mathbb{E}_{s'} \left| \max_{a' \sim \hat{\mu}(\cdot|s')} Q_1(s',a') - \max_{a' \sim \hat{\mu}(\cdot|s')} Q_2(s',a') \right| \\ &\leq \gamma \max_{s,a} \|Q_1 - Q_2\|_\infty \\ &= \gamma \|Q_1 - Q_2\|_\infty , \end{aligned}$$

where the second inequality holds from the fact that for any function f_1, f_2, any variant $x \sim \mathcal{X}$,

$$\left| \max_{x \sim \mathcal{X}} f_1(x) - \max_{x \sim \mathcal{X}} f_2(x) \right| \leq \max_{x \sim \mathcal{X}} |f_1(x) - f_2(x)| \tag{10}$$

If the next state $s' \sim P_{\text{src}}(\cdot|s,a)$, we have

$$\begin{aligned} &\|\mathcal{T}_{\text{RCB}}Q_1 - \mathcal{T}_{\text{RCB}}Q_2\|_\infty \\ &= \gamma \max_{s,a} \left| \inf_{\mathcal{M} \in \mathcal{M}_\epsilon} \mathbb{E}_{s'} \left[\max_{a' \sim \hat{\mu}(\cdot|s')} Q_1(s',a') \right] - \inf_{\mathcal{M} \in \mathcal{M}_\epsilon} \mathbb{E}_{s'} \left[\max_{a' \sim \hat{\mu}(\cdot|s')} Q_2(s',a') \right] \right| \\ &\leq \gamma \max_{s,a,s'} \left| \max_{a' \sim \hat{\mu}(\cdot|s')} Q_1(s',a') - \max_{a' \sim \hat{\mu}(\cdot|s')} Q_2(s',a') \right| \\ &\leq \gamma \max_{s,a} \|Q_1 - Q_2\|_\infty \\ &= \gamma \|Q_1 - Q_2\|_\infty . \end{aligned}$$

where the first inequality comes from the fact that for any function f_1, f_2, any variant $x \sim \mathcal{X}$,

$$\left| \min_{x \sim \mathcal{X}} f_1(x) - \min_{x \sim \mathcal{X}} f_2(x) \right| \leq \max_{x \sim \mathcal{X}} |f_1(x) - f_2(x)| . \tag{11}$$

Combining the results together, we conclude that the RCB operator is a γ-contraction operator in the complete state-action space, which naturally leads to the conclusion that any initial Q function would converge to a unique fixed point by repeatedly applying $\mathcal{T}_{\text{RCB}}$. This completes the proof. □

B.2 Proof of Proposition 4.2

We first introduce the following lemma before proving Proposition 4.2.

Lemma B.1. *Let $\mathcal{S}$ be a measure space, and P a probability measure on $\mathcal{S}$, and let $f : \mathcal{S} \to \mathbb{R}$ be any measure function, let $c : \mathcal{S} \times \mathcal{S} \to \mathbb{R}_{\geq 0}$ be a cost function. Then for any scalar $\lambda \geq 0$, the following equality holds:*

$$\inf_{\hat{P} \sim \mathcal{P}(\mathcal{S})} \left(\mathbb{E}_{\hat{s} \sim \hat{P}}[f(\hat{s})] + \lambda \mathcal{W}(\hat{P}, P) \right) = \mathbb{E}_{s \sim P} \left[\inf_{\hat{s} \sim \mathcal{S}} \left(f(\hat{s}) + \lambda c(s, \hat{s}) \right) \right] \tag{12}$$

where $\mathcal{P}(\mathcal{S})$ represents all probability measures on $\mathcal{S}$, and $\mathcal{W}$ is the Wasserstein distance w.r.t. the cost function c.

Proof. We prove this lemma by showing the left-hand-side (LHS) of Equation 12 is equivalent to its right-hand-side (RHS).

According to the definition of Wasserstein distance, the LHS could be written as:

$$\text{LHS} = \inf_{\hat{P}\in\mathcal{P}(\mathcal{S})} \left(\mathbb{E}_{\hat{s}\sim\hat{P}}[f(\hat{s})] + \lambda \inf_{\gamma\in\Gamma(P,\hat{P})} \mathbb{E}_{(s,\hat{s})\sim\gamma}[c(s,\hat{s})] \right) \tag{13}$$

The optimization of $\hat{P}$ and the inner optimization over $\gamma \in \Gamma(P, \hat{P})$ could be combined into a single optimization over all couplings γ whose first marginal is P, and the second marginal, $\hat{P}$, could be arbitrary in $\mathcal{P}(\mathcal{S})$. We then have:

$$\begin{aligned} \text{LHS} &= \inf_{\gamma\in\Gamma(P,\hat{P})} \left(\mathbb{E}_{\hat{s}\sim\hat{P}}[f(\hat{s})] + \lambda \mathbb{E}_{(s,\hat{s})\sim\gamma}[c(s,\hat{s})] \right) \\ &= \inf_{\gamma\in\Gamma(P,\hat{P})} \mathbb{E}_{(s,\hat{s})\sim\gamma} \left[f(\hat{s}) + \lambda c(s,\hat{s}) \right] \end{aligned} \tag{14}$$

where the second equality holds by the linearity of expectation.

By the disintegration theorem for measures, any coupling $\gamma \in \Gamma(P, \hat{P})$ could be represented as the product of its first marginal P and a stochastic kernel $K(d\hat{s}|s) : \mathcal{S} \to \mathcal{P}(\mathcal{S})$:

$$\gamma(ds, d\hat{s}) = K(d\hat{s}|s)P(ds) \tag{15}$$

This implies that optimizing over all couplings $\gamma \in \Gamma(P, \hat{P})$ is equivalent to optimizing over all possible stochastic kernels K. We substitute Equation 15 into Equation 14:

$$\begin{aligned} \text{LHS} &= \inf_{K} \int_{\mathcal{S}} \int_{\mathcal{S}} \left[f(\hat{s}) + \lambda c(s,\hat{s}) \right] K(d\hat{s}|s) P(ds) \\ &= \inf_{K} \mathbb{E}_{s\sim P} \left[\mathbb{E}_{\hat{s}\sim K(\cdot|s)} \left[f(\hat{s}) + \lambda c(s,\hat{s}) \right] \right] \end{aligned} \tag{16}$$

We change the position of the infimum operator and the inner expectation:

$$\text{LHS} = \mathbb{E}_{s\sim P} \left[\inf_{K(\cdot|s)\in\mathcal{P}(\mathcal{S})} \mathbb{E}_{\hat{s}\sim K(\cdot|s)} \left[f(\hat{s}) + \gamma c(s,\hat{s}) \right] \right] \tag{17}$$

We then solve the inner minimization problem for a fixed $s \in \mathcal{S}$:

$$\inf_{K(\cdot|s)\in\mathcal{P}(\mathcal{S})} \mathbb{E}_{\hat{s}\sim K(\cdot|s)} \left[f(\hat{s}) + \gamma c(s,\hat{s}) \right] \tag{18}$$

Let $g_s(\hat{s}) \triangleq f(\hat{s}) + \gamma c(s, \hat{s})$. The problem is to find a probability measure $K(\cdot|s)$ that minimizes the expectation of $g_s(\hat{s})$. It is obvious that this minimum is achieved by concentrating the entire probability mass on point $\hat{s}$ where $g_s(\hat{s})$ attains its infimum.

Let $\hat{s}^\star = \arg\inf_{\hat{s}\in\mathcal{S}} g_s(\hat{s})$. The optimal measure is a Dirac measure $\delta_{\hat{s}^\star}$ centered on $\hat{s}^\star$. Therefore:

$$\begin{aligned} &\inf_{K(\cdot|s)\in\mathcal{P}(\mathcal{S})} \mathbb{E}_{\hat{s}\sim K(\cdot|s)} \left[f(\hat{s}) + \gamma c(s,\hat{s}) \right] \\ &= \mathbb{E}_{\hat{s}\sim\delta_{\hat{s}^\star}} \left[g_s(\hat{s}) \right] \\ &= f(\hat{s}^\star) + \gamma c(s,\hat{s}^\star) \\ &= \inf_{\hat{s}\in\mathcal{S}} \left[f(\hat{s}) + \gamma c(s,\hat{s}) \right] \end{aligned} \tag{19}$$

Finally, substituting Equation 19 back into Equation 17, we obtain the RHS of the lemma:

$$\begin{aligned} \text{LHS} &= \mathbb{E}_{s\sim P} \left[\inf_{\hat{s}\in\mathcal{S}} \left(f(\hat{s}) + \gamma c(s,\hat{s}) \right) \right] \\ &= \text{RHS} \end{aligned} \tag{20}$$

This concludes the proof. □

Now we give our formal proof for Proposition 4.2. We restate it as follows.

Proposition B.1 (Proposition 4.2). *Let $\mathcal{M}_\epsilon$ be the Wasserstein uncertainty set defined by Equation 1, then the term* $\inf_{\widehat{\mathcal{M}}\in\mathcal{M}_\epsilon} \mathbb{E}_{s'\sim P_{\widehat{\mathcal{M}}}}\left[\max_{a'\sim\hat{\mu}(\cdot|s')} Q(s',a')\right]$ *in Equation 2 is equivalent to*

$$\mathbb{E}_{s'\sim P_{\mathcal{M}}}\left[\inf_{\bar{s}} \max_{a'\sim\hat{\mu}(\cdot|\bar{s})} Q(\bar{s},a')\right], \quad s.t. \quad d(s',\bar{s})\le\epsilon. \tag{21}$$

Proof. The original term (OT) is a constrained optimization problem which could be solved with Lagrange multiplier method. Let $\widehat{V}(s)=\max_{a\sim\hat{\mu}(\cdot|s)} Q(s,a)$, then we define the Lagrange function as:

$$\mathcal{L}(\widehat{\mathcal{M}},\lambda)=\mathbb{E}_{\hat{s}'\sim\widehat{\mathcal{M}}(\cdot|s,a)}\widehat{V}(\hat{s}')+\lambda\left(\mathcal{W}(\widehat{\mathcal{M}},\mathcal{M})-\epsilon\right) \tag{22}$$

The LHS is equivalent to solving the dual problem:

$$\mathrm{OT}=\sup_{\lambda\ge 0}\inf_{\widehat{\mathcal{M}}}\mathcal{L}(\widehat{\mathcal{M}},\lambda) \tag{23}$$

For a fixed λ, we solve the inner minimization problem $\inf_{\widehat{M}}\mathcal{L}(\widehat{M},\lambda)$:

$$\inf_{\widehat{\mathcal{M}}}\mathcal{L}(\widehat{\mathcal{M}},\lambda)=\inf_{\widehat{\mathcal{M}}}\left(\mathbb{E}_{\hat{s}'\sim\widehat{\mathcal{M}}(\cdot|s,a)}\widehat{V}(\hat{s}')+\lambda\mathcal{W}(\widehat{\mathcal{M}},\mathcal{M})\right)-\lambda\epsilon \tag{24}$$

According to Lemma B.1, we have:

$$\inf_{\widehat{\mathcal{M}}}\left(\mathbb{E}_{\hat{s}'\sim\widehat{\mathcal{M}}(\cdot|s,a)}\widehat{V}(\hat{s}')+\lambda\mathcal{W}(\widehat{\mathcal{M}},\mathcal{M})\right)=\mathbb{E}_{s'\sim\mathcal{M}(\cdot|s,a)}\left[\inf_{\hat{s}'}\left(\widehat{V}(\hat{s}')+\lambda c(s',\hat{s}')\right)\right] \tag{25}$$

Substituting Equation 25 into Equation 24 and Equation 23, we obtain the dual reformulation of the original term:

$$\mathrm{OT}=\sup_{\lambda\ge 0}\left\{\mathbb{E}_{s'\sim T(\cdot|s,a)}\left[\inf_{\hat{s}'}\left(\hat{V}(\hat{s}')+\lambda c(s',\hat{s}')\right)\right]-\lambda\epsilon\right\} \tag{26}$$

The RHS in Equation 26 is exactly the Lagrange dual reformulation of Equation 21. This implies Equation 21 holds, which concludes the proof. □

B.3 Proof of Proposition 4.3

Proof. We only discuss the case where $s'\sim P_{\text{src}}(\cdot|s,a)$, since for $s'\sim P_{\text{tar}}(\cdot|s,a)$, the proof is identical as Proposition 4.1. For $s'\sim P_{\text{src}}(\cdot|s,a)$, let Q_1 and Q_2 be two arbitrary Q functions, we have

$$\begin{aligned}
\left\|\mathcal{T}_{\text{RCB}}Q_1-\mathcal{T}_{\text{RCB}}Q_2\right\|_\infty &= \gamma\max_{s,a}\left|\mathbb{E}_{s'}\left[\inf_{\bar{s}\in U_\epsilon(s')}\max_{a'\sim\hat{\mu}(\cdot|\bar{s})}Q_1(\bar{s},a')-\inf_{\bar{s}\in U_\epsilon(s')}\max_{a'\sim\hat{\mu}(\cdot|\bar{s})}Q_2(\bar{s},a')\right]\right| \\
&\le \gamma\max_{s,a}\mathbb{E}_{s'}\left|\inf_{\bar{s}\in U_\epsilon(s')}\max_{a'\sim\hat{\mu}(\cdot|\bar{s})}Q_1(\bar{s},a')-\inf_{\bar{s}\in U_\epsilon(s')}\max_{a'\sim\hat{\mu}(\cdot|\bar{s})}Q_2(\bar{s},a')\right| \\
&\le \gamma\max_{s,a}\left\|Q_1-Q_2\right\|_\infty \\
&= \gamma\left\|Q_1-Q_2\right\|_\infty,
\end{aligned}$$

where the second inequality holds from Equation 10 and Equation 11. Then, we can conclude that the practical RCB operator is still a γ-contraction operator. □

B.4 Proof of Proposition 4.4

Proof. For any $(s,a)\in\mathcal{D}_{\text{src}}$,

$$\begin{aligned}
&\mathcal{T}_{\text{RCB}}Q(s,a)-\mathcal{T}Q(s,a) \\
&=\gamma\left(\mathbb{E}_{s'\sim P_{\mathcal{M}_{\text{src}}}}\left[\inf_{\bar{s}\in U_\epsilon(s')}\max_{a'\sim\hat{\mu}(\cdot|\bar{s})}Q(\bar{s},a')\right]-\mathbb{E}_{s'\sim P_{\mathcal{M}_{\text{tar}}}}\left[\max_{a'\sim\hat{\mu}(\cdot|s')}Q(s',a')\right]\right) \\
&\le\gamma\left(\inf_{s'\sim P_{\mathcal{M}_{\text{tar}}}}\max_{a'\sim\hat{\mu}(\cdot|s')}Q(s',a')-\mathbb{E}_{s'\sim P_{\mathcal{M}_{\text{tar}}}}\left[\max_{a'\sim\hat{\mu}(\cdot|s')}Q(s',a')\right]\right) \\
&\le 0,
\end{aligned} \tag{27}$$

where the first inequality holds by support$(P_{\text{tar}}(\cdot|s,a)) \subseteq U_\epsilon(s'_{\text{src}})$. In the mean time, for any $(s,a) \in \mathcal{D}_{\text{src}}$,

$$\begin{aligned}
&\mathcal{T}_{\text{RCB}}Q(s,a) - \mathcal{T}Q(s,a) \\
&= \gamma\left(\mathbb{E}_{s'\sim P_{\mathcal{M}_{\text{src}}}}\left[\inf_{\bar{s}\in U_\epsilon(s')}\max_{a'\sim\hat{\mu}(\cdot|\bar{s})} Q(\bar{s},a')\right] - \mathbb{E}_{s'\sim P_{\mathcal{M}_{\text{tar}}}}\left[\max_{a'\sim\hat{\mu}(\cdot|s')} Q(s',a')\right]\right) \\
&= \gamma\left(\mathbb{E}_{\bar{s}}\left[\max_{a'\sim\hat{\mu}(\cdot|\bar{s})} Q(\bar{s},a')\right] - \mathbb{E}_{s'\sim P_{\mathcal{M}_{\text{tar}}}}\left[\max_{a'\sim\hat{\mu}(\cdot|s')} Q(s',a')\right]\right) \\
&\geq \gamma\left(\min_{\bar{s}}\left[\max_{a'\sim\hat{\mu}(\cdot|\bar{s})} Q(\bar{s},a')\right] - \max_{s'\sim P_{\mathcal{M}_{\text{tar}}}}\left[\max_{a'\sim\hat{\mu}(\cdot|s')} Q(s',a')\right]\right).
\end{aligned} \tag{28}$$

Let $\bar{s}^\star = \arg\min_{\bar{s}}\left[\max_{a'\sim\hat{\mu}(\cdot|\bar{s})} Q(\bar{s},a')\right]$ and $s^\star = \arg\max_{s'\sim P_{\mathcal{M}_{\text{tar}}}}\left[\max_{a'\sim\hat{\mu}(\cdot|s')} Q(s',a')\right]$, then we have

$$\begin{aligned}
&\mathcal{T}_{\text{RCB}}Q(s,a) - \mathcal{T}Q(s,a) \\
&\geq \gamma\left(\max_{a'\sim\hat{\mu}(\cdot|\bar{s}^\star)} Q(\bar{s},a') - \max_{a'\sim\hat{\mu}(\cdot|s^\star)} Q(s^\star,a')\right) \\
&\geq \gamma\left(Q(\bar{s},a^\star) - Q(s^\star,a^\star)\right) \\
&\geq -2\gamma\epsilon K_Q,
\end{aligned}$$

where $a^\star = \arg\max_{a'\sim\hat{\mu}(\cdot|s^\star)} Q(s^\star,a')$, and the last inequality holds by the Lipschitz continuity assumption and triangle inequality.

Combining the above results, we have

$$\mathcal{T}Q(s,a) - 2\gamma\epsilon K_Q \leq \mathcal{T}_{\text{RCB}}Q(s,a) \leq \mathcal{T}Q(s,a). \tag{29}$$

Let Q^k denote the Q value at iteration k, and the initial Q value is Q^0. After one iteration using the RCB operator and the oracle optimal Bellman operator, according to Equation 29,

$$Q^1(s,a) - \frac{2\gamma\epsilon K_Q}{1-\gamma}(1-\gamma) \leq \hat{Q}^1_{\text{RCB}}(s,a) \leq Q^1(s,a). \tag{30}$$

Suppose when $k=i$, we have

$$Q^i(s,a) - \frac{2\gamma\epsilon K_Q}{1-\gamma}(1-\gamma^i) \leq \hat{Q}^i_{\text{RCB}}(s,a) \leq Q^i(s,a), \quad i \in \mathbb{Z}^+ \tag{31}$$

For $k = i+1$, we have

$$\mathcal{T}\hat{Q}^i_{\text{RCB}}(s,a) - 2\gamma\epsilon K_Q \leq \hat{Q}^{i+1}_{\text{RCB}}(s,a) = \mathcal{T}_{\text{RCB}}\hat{Q}^i_{\text{RBC}}(s,a) \leq \mathcal{T}\hat{Q}^i_{\text{RCB}}(s,a).$$

On the one hand, we have

$$\begin{aligned}
&\mathcal{T}\hat{Q}^i_{\text{RCB}}(s,a) \\
&\geq \mathcal{T}\left(Q^i(s,a) - \frac{2\gamma\epsilon K_Q}{1-\gamma}(1-\gamma^i)\right) \\
&= r(s,a) + \gamma\mathbb{E}_{s'\sim P_{\text{tar}}}\left[\max_{a'\sim\hat{\mu}(\cdot|s')}\left(Q^i(s',a') - \frac{2\gamma\epsilon K_Q}{1-\gamma}(1-\gamma^i)\right)\right] \\
&= r(s,a) + \gamma\mathbb{E}_{s'\sim P_{\text{tar}}}\left[\max_{a'\sim\hat{\mu}(\cdot|s')} Q^i(s',a')\right] - \gamma\frac{2\gamma\epsilon K_Q}{1-\gamma}(1-\gamma^i) \\
&= \mathcal{T}Q^i(s,a) - \gamma\frac{2\gamma\epsilon K_Q}{1-\gamma}(1-\gamma^i) \\
&= Q^{i+1}(s,a) - \gamma\frac{2\gamma\epsilon K_Q}{1-\gamma}(1-\gamma^i).
\end{aligned}$$

Therefore, we have

$$\begin{aligned}
&\hat{Q}^{i+1}_{\text{RCB}}(s,a) \\
&\geq Q^{i+1}(s,a) - \gamma\frac{2\gamma\epsilon K_Q}{1-\gamma}(1-\gamma^i) - 2\gamma\epsilon K_Q \\
&= Q^{i+1}(s,a) - \frac{2\gamma\epsilon K_Q}{1-\gamma}(1-\gamma^{i+1}).
\end{aligned} \tag{32}$$

On the other hand,

$$\mathcal{T}\hat{Q}^i_{\text{RCB}}(s,a) \leq \mathcal{T}Q^i(s,a) = Q^{i+1}(s,a).$$

Therefore, we have

$$\hat{Q}^{i+1}_{\text{RCB}}(s,a) \leq Q^{i+1}(s,a). \tag{33}$$

Combining the results of Equation 32 and Equation 33, we have

$$Q^{i+1}(s,a) - \frac{2\gamma\epsilon K_Q}{1-\gamma}(1-\gamma^{i+1}) \leq \hat{Q}^{i+1}_{\text{RCB}}(s,a) \leq Q^{i+1}(s,a).$$

Hence, Equation 31 still holds for $k = i+1$. Therefore, Equation 31 holds for all $k \in \mathbb{Z}^+$. If k is large enough, such that $\hat{Q}_{\text{RCB}}$ and $Q(s,a)$ converge to the fixed point, then we have

$$Q^\star_{\hat{\mu}}(s,a) - \frac{2\gamma\epsilon K_Q}{1-\gamma} \leq \hat{Q}_{\text{RCB}}(s,a) \leq Q^\star_{\hat{\mu}}(s,a),$$

which concludes the proof. □

B.5 Proof of Proposition 4.5

Proof. The learned value function $\widehat{V}_{\text{RCB}}(s)$ by repeatedly applying $\mathcal{T}_{\text{RCB}}$ satisfies:

$$\widehat{V}_{\text{RCB}}(s) = r(s,a^\star) + \gamma\mathbb{E}_{s'\sim P_{\text{src}}}\left[\inf_{\bar{s}\in U_\epsilon(s')} \widehat{V}_{\text{RCB}}(\bar{s})\right]$$

where $a^\star = \pi_{\text{RCB}}(\cdot|s) = \arg\max_{a\sim\hat{\mu}(\cdot|s)}\left[r(s,a) + \gamma\mathbb{E}_{s'\sim P_{\text{src}}}\left[\inf_{\bar{s}\in U_\epsilon(s')} \widehat{V}_{\text{RCB}}(\bar{s})\right]\right]$.

Let $c = \max\{c \,|U_c(s'_{\text{tar}}) \subseteq U_\epsilon(s'_{\text{src}}), s'_{\text{tar}} \sim P_{\text{tar}}(\cdot|s,a), (s,a,s'_{\text{src}}) \sim \mathcal{D}_{\text{src}}\}$, then we have

$$\mathbb{E}_{s'\sim P_{\text{tar}}}\left[\inf_{\bar{s}\in U_c(s')} \widehat{V}_{\text{RCB}}(\bar{s})\right] \geq \mathbb{E}_{s'\sim P_{\text{src}}}\left[\inf_{\bar{s}\in U_\epsilon(s')} \widehat{V}_{\text{RCB}}(\bar{s})\right],$$

since $U_\epsilon(s'_{\text{src}})$ has a broader region than $U_c(s'_{\text{tar}})$. Given any dynamics P which satisfies $\mathcal{W}(P(\cdot|s,a), P_{\text{tar}}(\cdot|s,a)) \leq c$, we can iteratively evaluate π_{RCB} within P:

$$\begin{aligned}
V^{k+1}(s) &= \mathcal{T}_P^{\pi_{\text{RCB}}}(V^k(s)) \\
&= r(s, a\sim\pi_{\text{RCB}}(\cdot|s)) + \gamma\mathbb{E}_{s'\sim P(\cdot|s,a)}\left(V^k(s')\right) \\
&\geq r(s, a\sim\pi_{\text{RCB}}(\cdot|s)) + \gamma\mathbb{E}_{s'\sim P_{\text{tar}}}\left[\inf_{\bar{s}\in U_c(s')} V^k(\bar{s})\right].
\end{aligned}$$

if we initialize $V^0(s)$ as $\widehat{V}_{\text{RCB}}(s)$, we have

$$\begin{aligned}
V^1(s) &= \mathcal{T}_P^{\pi_{\text{RCB}}}(V^0(s)) \\
&\geq r(s, a\sim\pi_{\text{RCB}}(\cdot|s)) + \gamma\mathbb{E}_{s'\sim P_{\text{tar}}}\left[\inf_{\bar{s}\in U_c(s')} \widehat{V}_{\text{RCB}}(\bar{s})\right] \\
&\geq r(s, a\sim\pi_{\text{RCB}}(\cdot|s)) + \gamma\mathbb{E}_{s'\sim P_{\text{src}}}\left[\inf_{\bar{s}\in U_\epsilon(s')} \widehat{V}_{\text{RCB}}(\bar{s})\right] \\
&= \widehat{V}_{\text{RCB}}(s) \\
&= V^0(s).
\end{aligned}$$

According to the monotonicity of the Bellman operator, we have $V^2(s) = \mathcal{T}_P^{\pi_{\text{RCB}}}(V^1(s)) \geq \mathcal{T}_P^{\pi_{\text{RCB}}}(V^0(s)) = V^1(s)$. Similarly, we can get $V^k(s) \geq V^{k-1}(s) \geq \cdots \geq V^0(s) = \hat{V}_{\text{RCB}}(s)$. Given that $\mathcal{T}_P^{\pi_{\text{RCB}}}$ is a γ-contraction, $V_P^{\pi_{\text{RCB}}}(s) = \lim_{k\to\infty} V^k(s) \geq \widehat{V}_{\text{RCB}}(s)$, which proves Equation 3 and conclude the proof. □

B.6 Proof of Proposition 4.6

Proof. We draw the proof inspiration from (Lyu et al., 2022). Given that $\sup_{s,a} D_{TV}(\widehat{P}_{\text{tar}}(\cdot|s,a), P_{\text{tar}}(\cdot|s,a)) \leq \epsilon < \frac{1}{2}$, we have

$$
\begin{aligned}
1 &> 2\epsilon \\
&\geq 2 \sup_{s,a} D_{TV}(\widehat{P}_{\text{tar}}(\cdot|s,a), P_{\text{tar}}(\cdot|s,a)) \\
&\geq \sum_{s'} \left| \widehat{P}_{\text{tar}}(s'|s,a) - P_{\text{tar}}(s'|s,a) \right| \\
&= \sum_{s' \in \text{support}(P_{\text{tar}}(\cdot|s,a))} \left| \widehat{P}_{\text{tar}}(s'|s,a) - P_{\text{tar}}(s'|s,a) \right| + \sum_{s' \notin \text{support}(P_{\text{tar}}(\cdot|s,a))} \left| \widehat{P}_{\text{tar}}(s'|s,a) - P_{\text{tar}}(s'|s,a) \right| \\
&\geq \sum_{s' \notin \text{support}(P_{\text{tar}}(\cdot|s,a))} \widehat{P}_{\text{tar}}(s'|s,a).
\end{aligned}
$$

Note that the maximum Q value $Q_{\max} \leq \frac{r_{\max}}{1-\gamma}$. Thus, we have

$$
\begin{aligned}
&\inf_{\{s'_i\}^N \sim \widehat{P}_{\text{tar}}(\cdot|s,a)} \left[\max_{a'_i \sim \hat{\mu}(\cdot|s'_i)} Q(s'_i, a'_i) \right] \\
&\leq \mathbb{E}_{\{s'_i\}^N \sim \widehat{P}_{\text{tar}}(\cdot|s,a)} \left[\max_{a'_i \sim \hat{\mu}(\cdot|s'_i)} Q(s'_i, a'_i) \right] \\
&\leq \mathbb{P}\left(\bigcap_i \{s' \in \text{support}(P_{\text{tar}}(\cdot|s,a))\} \right) \cdot \mathbb{E}_{s' \sim P_{\text{tar}}(\cdot|s,a)} \left[\max_{a' \sim \hat{\mu}(\cdot|s')} Q(s', a') \right] \\
&\quad + \mathbb{P}\left(\bigcup_i \{s' \notin \text{support}(P_{\text{tar}}(\cdot|s,a))\} \right) \cdot Q_{\max} \\
&\leq \mathbb{E}_{s' \sim P_{\text{tar}}(\cdot|s,a)} \left[\max_{a' \sim \hat{\mu}(\cdot|s')} Q(s', a') \right] + \left(1 - (\mathbb{P}(s'_1 \in \text{support}(P_{\text{tar}}(\cdot|s,a))))^N \right) \frac{r_{\max}}{1-\gamma} \\
&\leq \mathbb{E}_{s' \sim P_{\text{tar}}(\cdot|s,a)} \left[\max_{a' \sim \hat{\mu}(\cdot|s')} Q(s', a') \right] + \left(1 - (1-2\epsilon)^N \right) \frac{r_{\max}}{1-\gamma},
\end{aligned}
$$

where the first inequality uses the law of total expectation. Thus, we conclude the proof. □

C Experimental Settings

In this section, we introduce the detailed environmental settings missing in the main text.

C.1 Tasks and Datasets

Target domain and datasets. We directly adopt the four locomotion tasks from MuJoCo Engine (Todorov et al., 2012) as the target domain tasks: `halfcheetah-v2`, `hopper-v2`, `walker2d-v2`, `ant-v3`. For the target domain datasets, we reuse the datasets in D4RL (Fu et al., 2020) for each task. Since cross-domain offline RL only allows a small quantity of target domain data, we sample 10% data from the original D4RL datasets as the target domain datasets. The target domain datasets consist of four data qualities for each task: the **medium** datasets that contain samples collected by an early-stopped SAC policy; the **medium-replay** datasets that represent the replay buffer of the medium-level SAC agent; the **medium-expert** datasets that mix the medium data and expert data at a 50-50 ratio; the **expert** datasets that are collected by an SAC policy trained to the expert level. The trained policy is evaluated in the target domain, and the evaluation metric we use is `Normalized Score` in D4RL:

$$
\text{Normalized Score} = \frac{J_\pi - J_{\text{random}}}{J_{\text{expert}} - J_{\text{random}}} \times 100\%, \tag{34}
$$

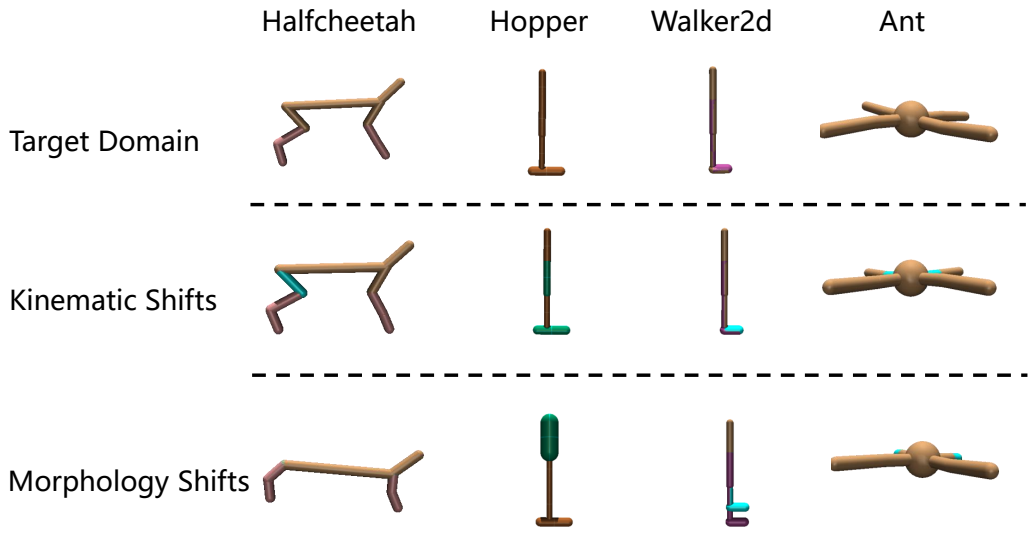


Figure 4: Visualization of the target domains and source domains with kinematic shifts and morphology shifts, across four tasks (`halfcheetah`, `hopper`, `walker2d`, `ant`).

where J_π is the return acquired by the trained policy in the target domain, and J_{expert} and J_{random} are the returns acquired by the expert policy and the random policy in the target domain, respectively.

Source domain and datasets. To simulate the source domain with dynamics shifts, we consider the four MuJoCo tasks (`halfcheetah-v2`, `hopper-v2`, `walker2d-v2`, `ant-v3`) with kinematic shifts and morphology shifts introduced as the source domain. The kinematic shifts refer to some joints of the robot being broken and unable to rotate, while the morphology shifts indicate that the robot's morphology is modified, differing from the target domain. To illustrate this more clearly, we visualize the robots in both the target and source domains for all four tasks in Figure 4. We also provide detailed code-level modifications for implementing the dynamics shifts in the following section.

For the source domain datasets, we follow a data collection process similar to D4RL. Specifically, we train an SAC policy in the source domain for 1M environmental steps and log policy checkpoints at different steps for trajectory rollouts. The **medium** datasets are collected using a logged policy that achieves approximately half the performance of the expert policy. The **medium-replay** datasets consist of the logged replay buffer from the medium-level agent. The **expert** datasets are collected using the final policy checkpoint, while the **medium-expert** datasets are a 50-50 mixture of medium-level and expert-level data. Note that all the source domain datasets contain about 1M samples, whereas the target domain datasets contain much fewer samples.

C.2 Kinematic Shifts Realization

To simulate the kinematic shifts in the source domain, we modify the `xml` files of the original environments. Specifically, we change the rotation angle of some joints of the simulated robot for different tasks:

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

```
# broken back thigh joint
<joint axis="0 1 0" damping="6" name="bthigh" pos="0 0 0" range="
    -.0052 .0105" stiffness="240" type="hinge"/>
```

***hopper-kinematic*:** The rotation angle of the head joint is modified from $[-150, 0]$ to $[-0.15, 0]$ and the rotation angle of the foot joint is modified from $[-45, 45]$ to $[-18, 18]$.

```
# broken head joint
<joint axis="0 -1 0" name="thigh_joint" pos="0 0 1.05" range="
    -0.15 0" type="hinge"/>
# broken foot joint
<joint axis="0 -1 0" name="foot_joint" pos="0 0 0.1" range="-18 18
    " type="hinge"/>
```

***walker2d-kinematic*:** The rotation angle of the right foot joint is modified from $[-45, 45]$ to $[-0.45, 0.45]$.

```
# broken right foot joint
<joint axis="0 -1 0" name="foot_joint" pos="0 0 0.1" range="-0.45
    0.45" type="hinge"/>
```

***ant-kinematic*:** The rotation angles of the joints on the hip of two front legs are modified from $[-30, 30]$ to $[-0.3, 0.3]$.

```
# broken hip joints of front legs
<joint axis="0 0 1" name="hip_1" pos="0.0 0.0 0.0" range="-0.3 0.3
    " type="hinge"/>
<joint axis="0 0 1" name="hip_2" pos="0.0 0.0 0.0" range="-0.3 0.3
    " type="hinge"/>
```

C.3 Morphology Shifts Realization

Akin to the kinematic shifts, we modify the `xml` files to simulate morphology shifts:

***halfcheetah-morph*:** The sizes of the back thigh and the forward thigh are modified.

```
# 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">
# front 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-morph*:** The head size of the robot is modified.

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

***walker2d-morph*:** The thigh on the right leg of the robot is modified.

```
# 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"/>
<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-morph*:** The size of the robot's two front legs is reduced.

```
# front 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"/>
# front 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"/>
```

D Implementation Details

In this section, we provide the implementation details for the baselines we use in our experiments and our method, DROCO.

D.1 Baselines

IQL*: IQL* is the cross-domain adaptation of IQL (Kostrikov et al., 2021). IQL* follows the same algorithmic procedure except being trained on both target and source domain datasets. The state value function is trained by expectile regression:

$$\mathcal{L}_V = \mathbb{E}_{(s,a)\sim\mathcal{D}_{\text{src}}\cup\mathcal{D}_{\text{tar}}}\left[L_2^\tau(Q_{\theta'}(s,a) - V_\psi(s))\right],$$

where $L_2^\tau(u) = |\tau - \mathbb{I}(u<0)|\,u^2$, $\mathbb{I}(\cdot)$ is the indicator function, and θ' is the parameter of the target network. This expectile regression enables learning an in-sample optimal value function. Subsequently, the state-action value function is updated by:

$$\mathcal{L}_Q = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}_{\text{src}}\cup\mathcal{D}_{\text{tar}}}\left[(r(s,a) + \gamma V_\psi(s') - Q_\theta(s,a))^2\right].$$

Then the advantage value is computed as $A(s,a) = Q(s,a) - V(s,a)$. Based on this, the policy is obtained through exponential advantage-weighted behavior cloning:

$$\mathcal{L}_\pi = -\mathbb{E}_{(s,a)\sim\mathcal{D}_{\text{src}}\cup\mathcal{D}_{\text{tar}}}\left[\exp(\beta\times A(s,a))\log\pi_\phi(a|s)\right],$$

where β is the inverse temperature coefficient. We implement IQL* based on the official codebase[2] of IQL.

CQL*: the cross-domain version of CQL (Kumar et al., 2020) similar to IQL*. CQL learns a conservative value function that lower bounds the true value function:

$$\mathcal{L}_Q = \alpha\mathbb{E}_{s\in\mathcal{D}}\left[\log\sum_a\exp(Q(s,a)) - \mathbb{E}_{a\sim\mu}[Q(s,a)]\right] + \frac{1}{2}\mathbb{E}_{(s,a,s')\in\mathcal{D}}\left[(Q(s,a) - \mathcal{T}Q(s,a))^2\right]$$

The policy π is then optimized with SAC (Haarnoja et al., 2018). We implement CQL* based on the implementation of CORL[3].

BOSA: BOSA (Liu et al., 2024a) identifies two key challenges in cross-domain offline RL: the state-action OOD problem and the dynamics OOD problem. To address these, BOSA proposes two support constraints. Specifically, BOSA handles the OOD state-action problem by supported policy

[2] https://github.com/ikostrikov/implicit_q_learning.git
[3] https://github.com/tinkoff-ai/CORL.git

optimization, and mitigates the OOD dynamics problem by supported value optimization. The critic is updated through supported value optimization:

$$\mathcal{L}_Q = \mathbb{E}_{(s,a)\sim\mathcal{D}_{\text{src}}}\left[Q_{\theta_i}(s,a)\right] + \mathbb{E}_{\substack{(s,a,r,s')\sim\mathcal{D}_{\text{src}}\cup\mathcal{D}_{\text{tar}},\\ a'\sim\pi_\phi(s')}}\left[\mathbb{I}(\hat{P}_{tar}(s'|s,a)>\epsilon)(Q_{\theta_i}(s,a)-y)^2\right],$$

where $\mathbb{I}(\cdot)$ is the indicator function, and $\hat{P}_{\text{tar}}(s'|s,a)$ is the estimated target domain dynamics, and ϵ is the threshold coefficient. The policy in BOSA is updated by supported policy optimization to mitigate the OOD action issue:

$$\mathcal{L}_\pi = \mathbb{E}_{s\sim\mathcal{D}_{\text{src}}\cup\mathcal{D}_{\text{tar}},\ a\sim\pi_\phi(s)}\left[Q_{\theta_i}(s,a)\right], \quad \text{s.t. } \mathbb{E}_{s\sim\mathcal{D}_{\text{src}}\cup\mathcal{D}_{\text{tar}}}\left[\hat{\pi}_{\text{mix}}(\pi_\phi(s)\mid s)\right] > \epsilon',$$

where ϵ' is the threshold coefficient, and $\hat{\pi}_{\phi_{\text{mix}}}(\cdot|s)$ is the behavior policy of the mixed datasets $\mathcal{D}_{\text{src}}\cup\mathcal{D}_{\text{tar}}$ learned with CVAE (Kingma et al., 2013). We adopt the BOSA implementation from ODRL[4] benchmark (Lyu et al., 2024b), which provides reliable implementations for various off-dynamics RL algorithms.

DARA. DARA (Liu et al., 2022) employs dynamics-aware reward modification to achieve dynamics adaptation, extending DARC (Eysenbach et al., 2020) to the offline setting. Specifically, DARA trains two domain classifiers $q_{\theta_{SAS}}(\text{target}|s_t,a_t,s_{t+1})$ and $q_{\theta_{SA}}(\text{target}|s_t,a_t)$ as follows:

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

The domain classifiers are used to quantify the dynamics gap $\log\frac{P_{\mathcal{M}_{\text{tar}}}(s_{t+1}|s_t,a_t)}{P_{\mathcal{M}_{\text{src}}}(s_{t+1}|s_t,a_t)}$ between the source domain and the target domain according to Bayes' rule. Then the estimated dynamics gap serves as a penalty to the source domain rewards:

$$\hat{r}_{\text{DARA}} = r - \lambda\times\delta_r, \quad \delta_r(s_t,a_t) = -\log\frac{q_{\theta_{\text{SAS}}}(\text{target}|s_t,a_t,s_{t+1})q_{\theta_{\text{SA}}}(\text{source}|s_t,a_t)}{q_{\theta_{\text{SAS}}}(\text{source}|s_t,a_t,s_{t+1})q_{\theta_{\text{SA}}}(\text{target}|s_t,a_t)}, \tag{35}$$

where λ controls the intensity of the reward penalty. We use the DARA implementation from ODRL and follow the hyperparameter setting in the original paper: λ is set to 0.1, and the reward penalty is clipped within $[-10,10]$ for training stability.

IGDF. IGDF (Wen et al., 2024) quantifies the domain discrepancy between the source domain and the target domain with contrastive representation learning. To facilitate effective knowledge transfer, IGDF implements data filtering to selectively share source domain samples exhibiting smaller dynamics gaps. Specifically, IGDF trains a score function $h(\cdot)$ using $(s,a,s'_{\text{tar}})\sim\mathcal{D}_{\text{tar}}$ as the positive samples, and transitions (s,a,s'_{src}) as the negative samples, where $(s,a)\sim\mathcal{D}_{\text{tar}}$ and $s'_{\text{src}}\sim\mathcal{D}_{\text{src}}$. $h(\cdot)$ is optimized via the following contrastive learning objective:

$$\mathcal{L} = -\mathbb{E}_{(s,a,s'_{\text{tar}})}\mathbb{E}_{s'_{\text{src}}}\left[\log\frac{h(s,a,s'_{\text{tar}})}{\sum_{s'\in s'_{\text{tar}}\cup s'_{\text{src}}}h(s,a,s')}\right].$$

Based on the learned score function, IGDF proposes to selectively share source domain data for training value functions:

$$\mathcal{L}_Q = \frac{1}{2}\mathbb{E}_{\mathcal{D}_{\text{tar}}}\left[(Q_\theta-\mathcal{T}Q_\theta)^2\right] + \frac{1}{2}\alpha\cdot h(s,a,s')\mathbb{E}_{(s,a,s')\sim\mathcal{D}_{\text{src}}}\left[\mathbb{I}(h(s,a,s')>h_{\xi\%})(Q_\theta-\mathcal{T}Q_\theta)^2\right],$$

where $\mathbb{I}(\cdot)$ is the indicator function, α is the weighting coefficient, ξ is the data selection ratio. We implement IGDF based on its official codebase[5].

OTDF. OTDF (Lyu et al., 2025) estimates the discrepancy between the source domain and target domain by computing the Wasserstein distance (Peyré et al., 2019):

$$\mathcal{W}(u,u') = \min_{\mu\in M}\sum_{t=1}^{|\mathcal{D}_{\text{src}}|}\sum_{t'=1}^{|\mathcal{D}_{\text{tar}}|}C(u_t,u'_{t'})\cdot\mu_{t,t'}, \tag{36}$$

[4] https://github.com/OffDynamicsRL/off-dynamics-rl.git
[5] https://github.com/BattleWen/IGDF.git

where $u = s_{\text{src}} \oplus a_{\text{src}} \oplus s'_{\text{src}}$, $u' = s_{\text{tar}} \oplus a_{\text{tar}} \oplus s'_{\text{tar}}$ are the concatenated vectors, C is the cost function and M is the coupling matrices. After solving Equation 36 for the optimal coupling matrix $\mu^\star$, the OTDF computes the distance between a source domain sample and the target domain dataset via

$$d(u_t) = -\sum_{t'=1}^{|\mathcal{D}_{\text{tar}}|} C(u_t, u_{t'})\mu^\star_{t,t'}, \quad u_t = (s^t_{\text{src}}, a^t_{\text{src}}, (s'_{\text{src}})^t) \sim \mathcal{D}_{\text{src}}.$$

Then the critic is updated by

$$\mathcal{L}_Q = \mathbb{E}_{\mathcal{D}_{\text{tar}}}\left[(Q_\theta - \mathcal{T}Q_\theta)^2\right] + \mathbb{E}_{(s,a,s')\sim\mathcal{D}_{\text{src}}}\left[\exp(\alpha \times d)\mathbb{I}(d > d_{\%})(Q_\theta - \mathcal{T}Q_\theta)^2\right].$$

Furthermore, OTDF incorporates a policy regularization term that forces the policy to stay close to the support of the target dataset:

$$\widehat{\mathcal{L}_\pi} = \mathcal{L}_\pi - \beta \times \mathbb{E}_{s\sim\mathcal{D}_{\text{src}}\cup\mathcal{D}_{\text{tar}}} \log \pi^b_{\text{tar}}(\pi(\cdot|s)|s),$$

where $\mathcal{L}_\pi$ is the original policy optimization objective and β is the weight coefficient, π^b_{tar} is the behavior policy of the target domain dataset learned with a CVAE. We run the official code[6] for OTDF in our experiments.

D.2 Implementation details of DROCO

In this part, we provide more implementation details of DROCO omitted in the main text.

First, we model the target domain dynamics using a neural network that outputs a Gaussian distribution over the next state: $\widehat{P}_\psi(s'|s,a) = \mathcal{N}(\mu_\psi(s,a), \Sigma_\psi(s,a))$. We learn an ensemble of N dynamics models $\{\widehat{P}_{\psi_i} = \mathcal{N}(\mu_{\psi_i}, \Sigma_{\psi_i})\}_{i=1}^N$, with each model trained independently with maximum likelihood estimation (MLE) on the target domain dataset:

$$\mathcal{L}_{\psi_i} = \mathbb{E}_{(s,a,s')\in\mathcal{D}_{\text{tar}}}\left[\log \widehat{P}_{\psi_i}(s'|s,a)\right]. \tag{37}$$

When we sample from the uncertainty set, we can directly sample from each dynamics model $\mathcal{N}(\mu_{\psi_i}, \Sigma_{\psi_i})$ as the sampling points. We can then compute the value penalty and penalize the Q value of source domain data when leveraging IQL for policy optimization. Specifically, we perform expectile regression to train the V function:

$$\mathcal{L}_V(\eta) = \mathbb{E}_{(s,a)\sim\mathcal{D}_{\text{src}}\cup\mathcal{D}_{\text{tar}}}\left[\mathcal{L}_2^\tau(Q_\theta(s,a) - V_\eta(s))\right],$$

where $\mathcal{L}_2^\tau(u) = |\tau - \mathbb{I}(u<0)|u^2$ and $\tau \in (0,1)$. For $\tau \approx 1$, V_η can capture the in-sample maximal Q (Kostrikov et al., 2021): $V_\eta(s) \approx \max_{a\sim\hat{\mu}(\cdot|s)} Q(s,a)$. We can then practically compute the value penalty as:

$$u(s,a,s') = \mathbb{I}\left(s' \sim P_{\text{src}}(\cdot|s,a)\right) \cdot \left(V(s') - \inf_{\{s'_i\}^N\sim\widehat{P}_{\psi_i}(\cdot|s,a)} [V(s'_i)]\right), \tag{38}$$

and the practical Bellman target can be written as

$$\widehat{\mathcal{T}}_{\text{RCB}}Q(s,a) = r(s,a) + \gamma\mathbb{E}_{s'\sim P_{\mathcal{M}}(\cdot|s,a)}\left[V(s') - \beta \cdot u(s,a,s')\right]. \tag{39}$$

Then, we incorporate Huber loss and have the following Q training loss:

$$\mathcal{L}_Q(\theta) = \mathbb{E}_{\mathcal{D}_{\text{src}}}\left[l_\delta\left(Q_\theta(s,a) - \widehat{\mathcal{T}}_{\text{RCB}}Q_\theta(s,a)\right)\right] + \frac{1}{2}\mathbb{E}_{\mathcal{D}_{\text{tar}}}\left[\left(Q_\theta(s,a) - \mathcal{T}Q_\theta(s,a)\right)^2\right], \tag{40}$$

where l_δ is the Huber loss. The final step is policy learning. We follow IQL and utilize exponential advantage-weighted imitation learning to extract the policy:

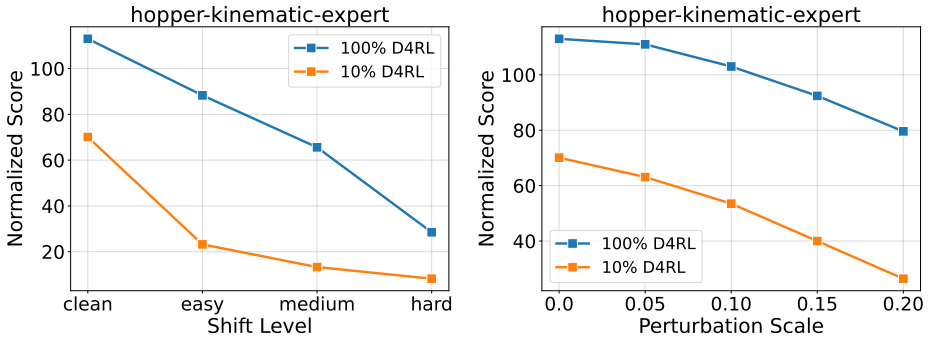
$$\mathcal{L}_\pi(\phi) = -\mathbb{E}_{\mathcal{D}_{\text{src}}\cup\mathcal{D}_{\text{tar}}}\left[\exp(Q(s,a) - V(s)) \log \pi_\phi(a|s)\right].$$

We show the detailed pseudocode of DROCO in Algorithm 1.

[6] https://github.com/dmksjfl/OTDF.git

Algorithm 1 Dual-Robust Cross-domain Offline RL (DROCO)

Require: Source domain offline dataset $\mathcal{D}_{\text{src}}$, target domain offline dataset $\mathcal{D}_{\text{tar}}$, mixed offline dataset $\mathcal{D}_{\text{mix}}$
Initialization: Policy network π_ϕ, value network V_η, Q network Q_θ, ensemble dynamics model $\widehat{P}_\psi = \{\widehat{P}_{\psi_i}\}_{i=1}^N$, penalty coefficient β, transition threshold δ for Huber loss
// Train the ensemble dynamics model
for each model gradient step **do**
  for each ensemble member $\hat{P}_{\psi_i}$ **do**
    Compute loss $\mathcal{L}_{\psi_i} = \mathbb{E}_{(s,a,s')\in\mathcal{D}_{\text{tar}}}\left[\log \widehat{P}_{\psi_i}(s'|s,a)\right]$
    Update $\widehat{P}_{\psi_i}$ using $\mathcal{L}_{\psi_i}$
  end for
end for
// TD Learning
for each gradient step **do**
  Sample $b_{\text{src}} := \{(s,a,r,s')\}$ from $\mathcal{D}_{\text{src}}$
  Sample $b_{\text{tar}} := \{(s,a,r,s')\}$ from $\mathcal{D}_{\text{tar}}$
  // Optimize the V_β function
  Compute loss $\mathcal{L}_V$:
    $\mathcal{L}_V = \mathbb{E}_{(s,a)\sim\mathcal{D}_{\text{src}}\cup\mathcal{D}_{\text{tar}}}\left[\mathcal{L}_2^\tau\left(Q_\theta(s,a) - V_\eta(s)\right)\right]$
  Update V_η using $\mathcal{L}_V$
  // Compute the value penalty
  compute $u(s,a,s') = \mathbb{I}\left(s' \sim P_{\text{src}}(\cdot|s,a)\right) \cdot \left(V(s') - \inf_{\{s_i'\}^N\sim\widehat{P}_{\psi_i}(\cdot|s,a)}\left[V(s_i')\right]\right)$
  // Optimize the Q_θ function
  Compute loss $\mathcal{L}_Q$:
    $\mathcal{L}_Q = \frac{1}{2}\cdot\mathbb{E}_{(s,a,r,s')\sim\mathcal{D}_{\text{tar}}}\left[\left(Q_\theta(s,a) - (r + \gamma V_\eta(s'))\right)^2\right]$
    $+\frac{1}{2}\cdot\mathbb{E}_{(s,a,r,s')\sim\mathcal{D}_{\text{src}}}\left[l_\delta\left(Q_\theta(s,a) - (r + \gamma V_\eta(s') - \beta u(s,a,s'))\right)\right]$
  Update Q_θ using $\mathcal{L}_Q$
  // Update target network
  Update target network parameters: $\theta' \leftarrow (1-\mu)\theta + \mu\theta'$
  // Policy Extraction (AWR)
  Compute advantage $A(s,a) = Q_\theta(s,a) - V_\eta(s)$
  Optimize policy network π_η using advantage-weighted regression (AWR):
    $\mathcal{L}_\pi = \mathbb{E}_{(s,a)\sim\mathcal{D}_{\text{src}}\cup\mathcal{D}_{\text{tar}}}\left[\exp(\alpha A(s,a))\log\pi_\phi(a|s)\right]$
end for



(a) Morphology Perturbation (b) min Q perturbation

Figure 5: Evaluation results of IGDF under morphology and min Q perturbations with different sizes of target domain data.

E Extended Experimental Results

E.1 Extended Results of Motivation Example

In Section 3, we demonstrate our motivation with a simple example. In this part, we provide more details and results for the motivation example.

The source and target domains are `hopper-kinematic-v2` and `hopper-v2` respectively, with their corresponding datasets being `hopper-kinematic-expert` and `hopper-expert`. Figure 1 in Section 3 demonstrates performance across different target domain data sizes under three test-time kinematic perturbation levels (easy, medium, hard), implemented as in (Lyu et al., 2024b). We further evaluate the trained IGDF under morphology perturbations and min-Q perturbations (with other settings unchanged), presenting results in Figure 5.

The results clearly show that with only 10% D4RL data, IGDF's robustness to dynamics perturbations is significantly weaker compared to using 100% D4RL data. Notably, under easy-level morphology perturbations, IGDF with 10% D4RL data exhibits a 66.9% performance drop, versus only 21.8% degradation with 100% data. These findings, combined with the results in Section 3, validate our motivation that cross-domain offline RL is particularly sensitive to dynamics perturbations when limited target domain data is available, underscoring the need for enhanced test-time robustness.

E.2 Evaluation under Morphology Shifts

In the main text, we present DROCO's evaluation results under kinematic shifts. In this section, we supplement with additional results under morphological shifts, providing a comprehensive assessment of DROCO's train-time robustness against diverse dynamics shifts.

Experimental Settings. The target domain tasks and datasets remain consistent with Section 5.1: the target domain tasks include `halfcheetah-v2`, `hopper-v2`, `walker2d-v2` and `ant-v3`, and the target domain datasets comprise four data qualities (`medium`, `medium-replay`, `medium-expert`, `expert`) for each task. The difference lies in the dynamics shift type in the source domain. We implement morphology shifts as described in Appendix C.3 and collect the corresponding source domain datasets.

Baselines. We adopt the same baselines as in Section 5.1: IQL*, CQL*, BOSA, DARA, IGDF and OTDF.

Results. We run each baseline and DROCO for 1M steps over 5 random seeds, and present the results with train-time morphology shifts in Table 2. It is clear that DROCO delivers superior performance to baselines. Specifically, DROCO achieves the highest performance in **9** out of 16 tasks. In terms of the total normalized score across all 16 tasks, DROCO attains a remarkable **1166.4**, significantly outperforming the second-best baseline OTDF (1025.1). Combined with the results in

Table 2: **Evaluation results with train-time morphology shifts.** half=halfcheetah, hopp=hopper, walk=walker2d, m=medium, me=medium-expert, mr=medium-replay, e=expert. We report the normalized score evaluated in the target domain and ± captures the standard deviation across 5 seeds. We **bold** the highest scores for each task.

Dataset	IQL*	CQL*	BOSA	DARA	IGDF	OTDF	DROCO (Ours)
half-m	**45.8**	40.2	41.3	**45.6**	**45.5±0.1**	44.3±0.2	**45.8±0.2**
half-mr	26.1	21.3	27.8	**28.9**	24.2±3.3	19.7±2.5	27.9±4.4
half-me	63.0	54,6	44.4	59.2	61.9± 4.9	42.9±3.6	**70.1±5.6**
half-e	65.2	66.7	**78.6**	55.4	56.0±6.2	74.2±5.0	**79.2±3.9**
hopp-m	**56.4**	32.8	28.7	49.5	55.5±2.9	49.1±2.2	**56.3±1.6**
hopp-mr	51.3	37.6	40.6	53.5	**54.9±5.8**	24.9±3.4	51.6±8.7
hopp-me	35.8	36.6	20.2	38.2	43.3±3.6	51.8±3.9	**82.3±4.1**
hopp-e	87.2	67.9	64.3	77.1	51.5±2.9	**113.2±5.9**	92.5±1.2
walk-m	32.6	43.1	40.3	25.0	33.0±2.3	40.3±7.1	**60.1±3.4**
walk-mr	9.0	2.0	2.9	6.9	9.5±0.4	14.1±1.8	**15.5±4.7**
walk-me	27.6	22.4	46.7	42.2	**75.7±11.8**	66.7±5.3	**78.9±9.4**
walk-e	103.4	79.0	30.2	102.7	**108.3±6.7**	103.5±1.9	104.5±1.7
ant-m	89.1	57.3	36.1	**96.4**	91.6±4.4	92.5±2.7	94.5±2.8
ant-mr	59.7	39.5	24.0	64.1	58.2±7.1	**69.6±8.1**	66.9±4.9
ant-me	113.1	107.3	100.5	111.9	116.8±3.5	107.3±4.4	**120.3±1.5**
ant-e	116.3	94.4	76.3	124.5	**126.8±1.7**	111.0±2.4	120.0±1.3
Total	981.6	802.7	702.9	981.1	1012.7	1025.1	**1166.4**

Section 5.1, these findings conclusively demonstrate DROCO's superiority across different types of dynamics shifts, highlighting its strong train-time robustness against dynamics shifts.

E.3 Extended Evaluation under Dynamics Perturbations

In this section, we supplement with more experimental results for evaluating the test-time robustness of DROCO.

We first extend the results in Section 5.2 by incorporating a broader range of datasets. We evaluate the robustness of DROCO against two baselines (IGDF, OTDF) under varying levels of three perturbation types: kinematic, morphology, and min Q perturbations, following the methodology in Section 5.2. Additional experiments are conducted using `hopper-morph-expert`, `walker2d-kinematic-expert`, and `ant-morph-expert` as source domain datasets, with results presented in Figure 6. We can see that DROCO demonstrates superior robustness to all three perturbation types compared to the baselines. For instance, on the `walker2d-kinematic-expert` dataset under min Q perturbations, DROCO exhibits only 23.4% performance degradation (from 106.0 to 81.2) at the highest perturbation level (0.2), substantially lower than IGDF (75.3%) and OTDF (55.9%). This enhanced robustness is consistently observed across all datasets and perturbation types, confirming DROCO's improved test-time robustness against dynamics perturbations.

We further evaluate DROCO's test-time robustness using varying target domain dataset sizes. Experiments are conducted under different levels of min Q perturbations, with target domain sizes set to 100%, 50%, and 10% of the original D4RL datasets. The source domain datasets comprise `hopper-morph-expert`, `walker2d-kinematic-expert`, and `ant-morph-expert`. As shown in Figure 7, all methods demonstrate improved robustness against dynamics perturbations with increasing target domain data size, consistent with our claim in Section 3. Notably, DROCO maintains superior robustness across varying data sizes and perturbation scales compared to IGDF and OTDF, further validating its effectiveness in enhancing test-time robustness.

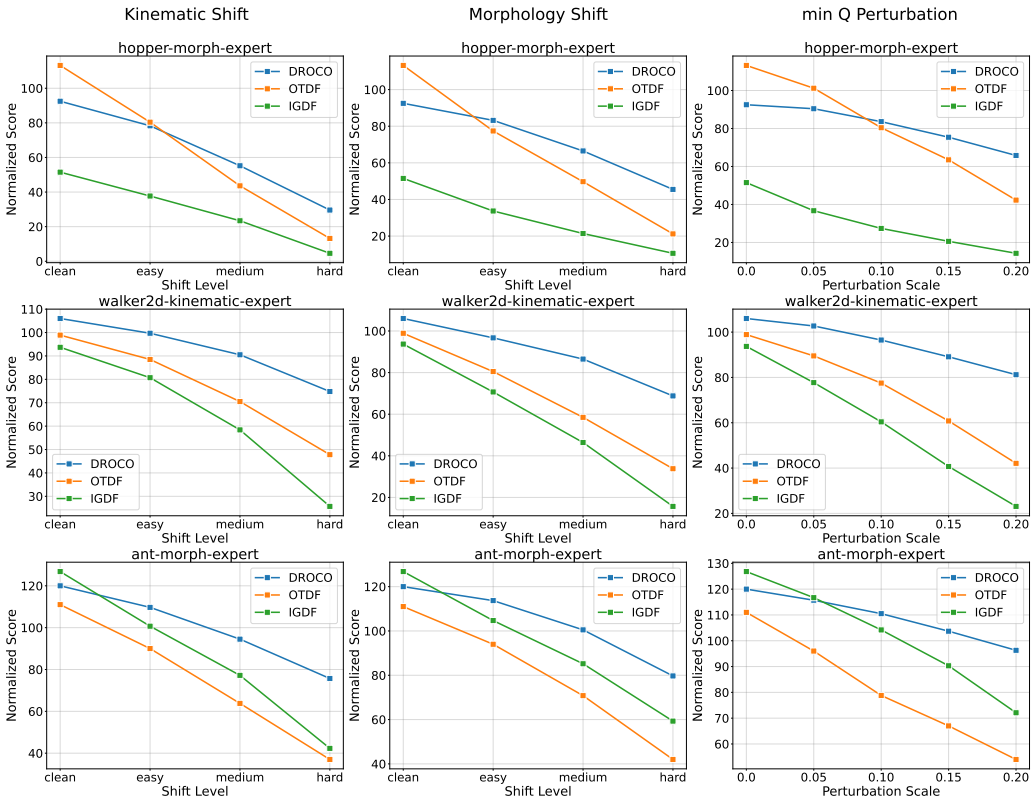


Figure 6: Evaluation results under different types and levels of dynamics perturbations.

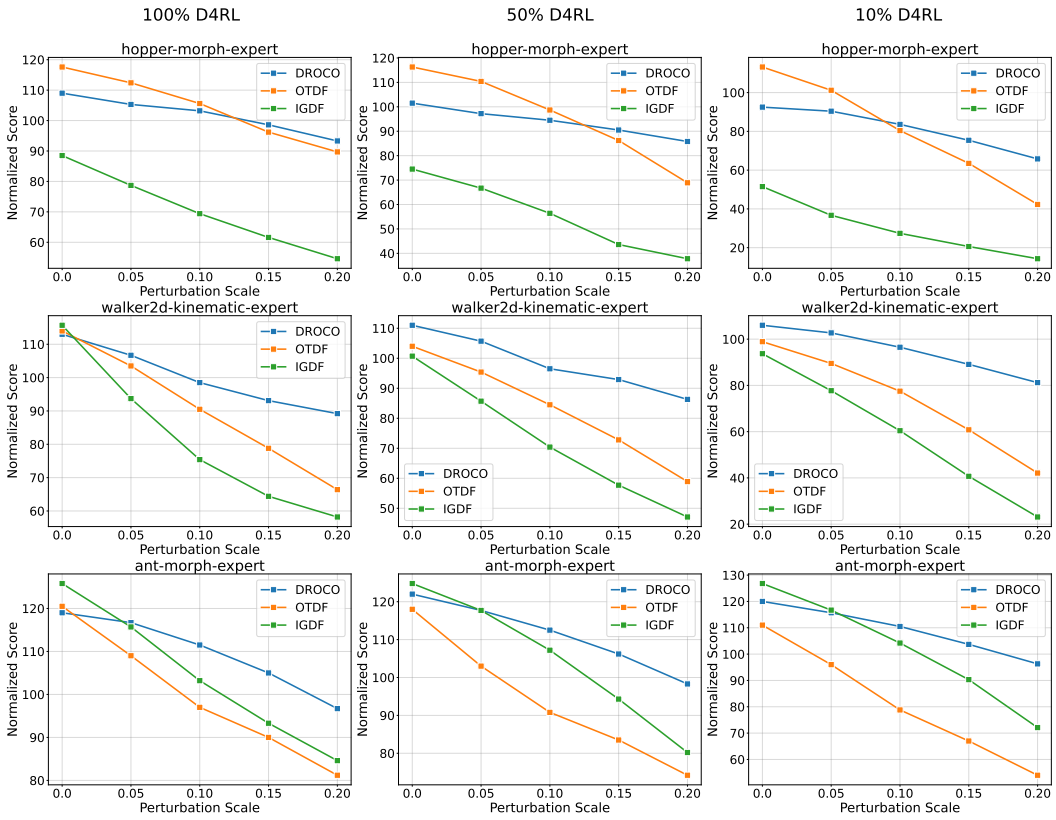


Figure 7: Evaluation results under different perturbation levels and different data size.

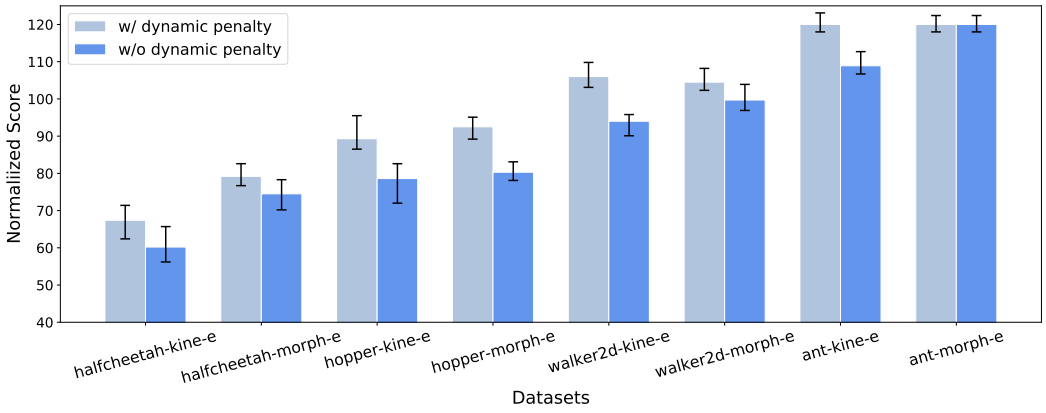


Figure 8: Ablation study on value penalty

E.4 Ablation Study

We provide supplementary ablation study results that are omitted from the main text. Specifically, we examine the effects of replacing the adaptive value penalty with a fixed value penalty and substituting the Huber loss with the regular ℓ_2 loss.

Fixed Value Penalty. A fixed value penalty corresponds to setting $\beta = 1.0$ across all tasks. Figure 8 compares the performance of DROCO with dynamic versus fixed penalties across eight datasets. The results demonstrate that the dynamic value penalty generally outperforms the fixed penalty ($\beta = 1.0$), with the exception of the `ant-morph-expert` dataset where the fixed penalty achieves the highest performance.

We further evaluate the test-time robustness of DROCO under diverse dynamic shifts using both penalty schemes. Following the experimental setup in Appendix E.3, our results in Figure 9 reveal an interesting trade-off: while the fixed value penalty leads to slightly degraded performance, it provides marginally improved robustness against dynamic perturbations. This suggests that setting β to a larger value induces a more conservative policy that is less sensitive to dynamic perturbations, albeit at the cost of policy performance.

Regular ℓ_2 Loss. The standard ℓ_2 loss implements conventional Bellman updates for source domain data without special outlier handling. We evaluate DROCO's performance on 8 `medium-expert` datasets comparing the Huber loss versus the ℓ_2 loss and present the results in Figure 10. The results show that Huber loss generally produces superior performance, while ℓ_2 loss achieves marginally better results on `halfcheetah-morph-me` and `walker2d-kine-me` datasets.

We further examine the test-time robustness against dynamic perturbations using both loss functions. Figure 11 reveals that using Huber loss consistently provides stronger robustness across perturbation types, underscoring its critical role in enhancing robustness.

E.5 Extended Parameter Study

In the main text, we test the sensitivity of DROCO to the penalty coefficient β and the transition threshold δ on certain datasets. In this section, we present extended results for a more comprehensive analysis.

Penalty coefficient β. β controls the intensity of the value penalty. We sweep β across $\{0.1, 0.5, 1.0, 1.2\}$ and further conduct experiments on `walker2d-morph-expert` and `ant-morph-expert` datasets, we present the learning curves of the performance and the Q value in Figure 12. We find that $\beta \leq 1.0$ is generally preferred, yielding better performance and Q value convergence, while setting $\beta = 1.2$ would cause value underestimation and inferior performance.

Transition threshold δ. δ determines the transition point from ℓ_2 loss to ℓ_1 loss. We vary δ among $\{5, 10, 30, 50\}$ and conduct experiments on `walker2d-morph-me` and `ant-morph-me`

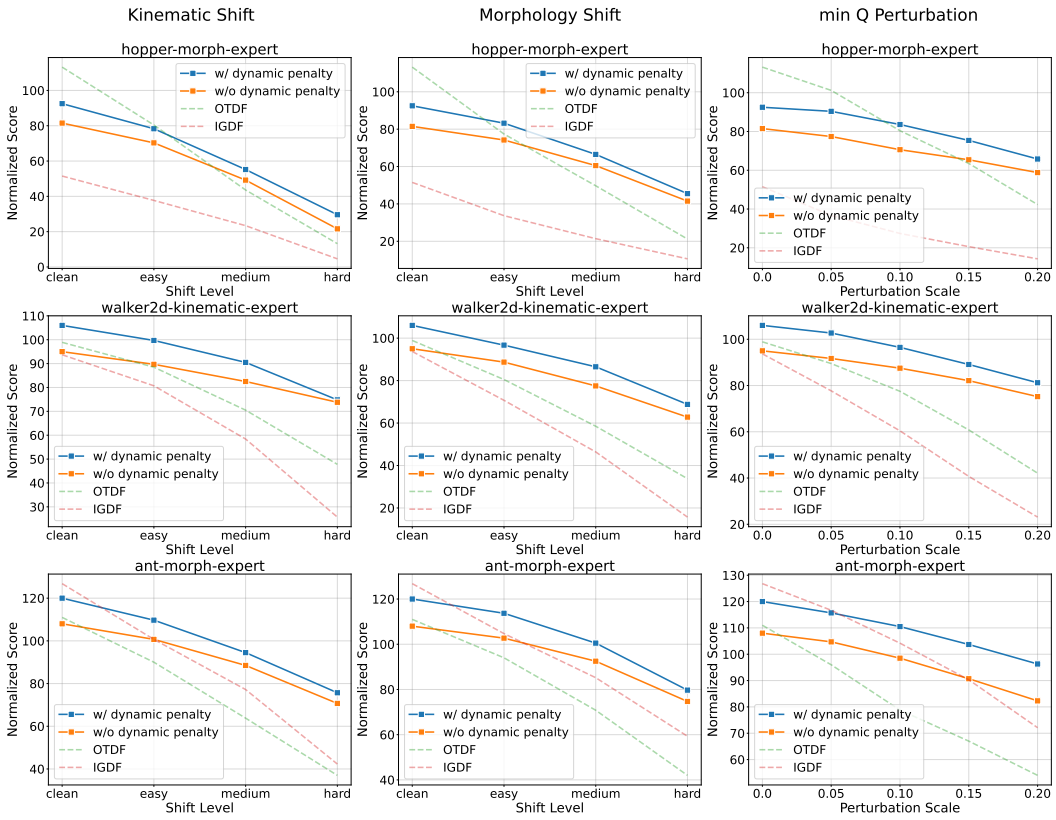


Figure 9: Ablation study on value penalty

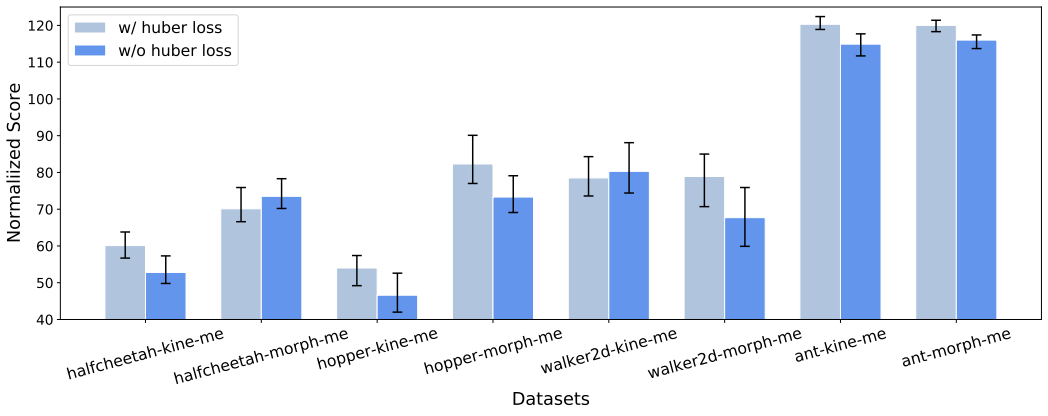


Figure 10: Ablation study on Huber loss

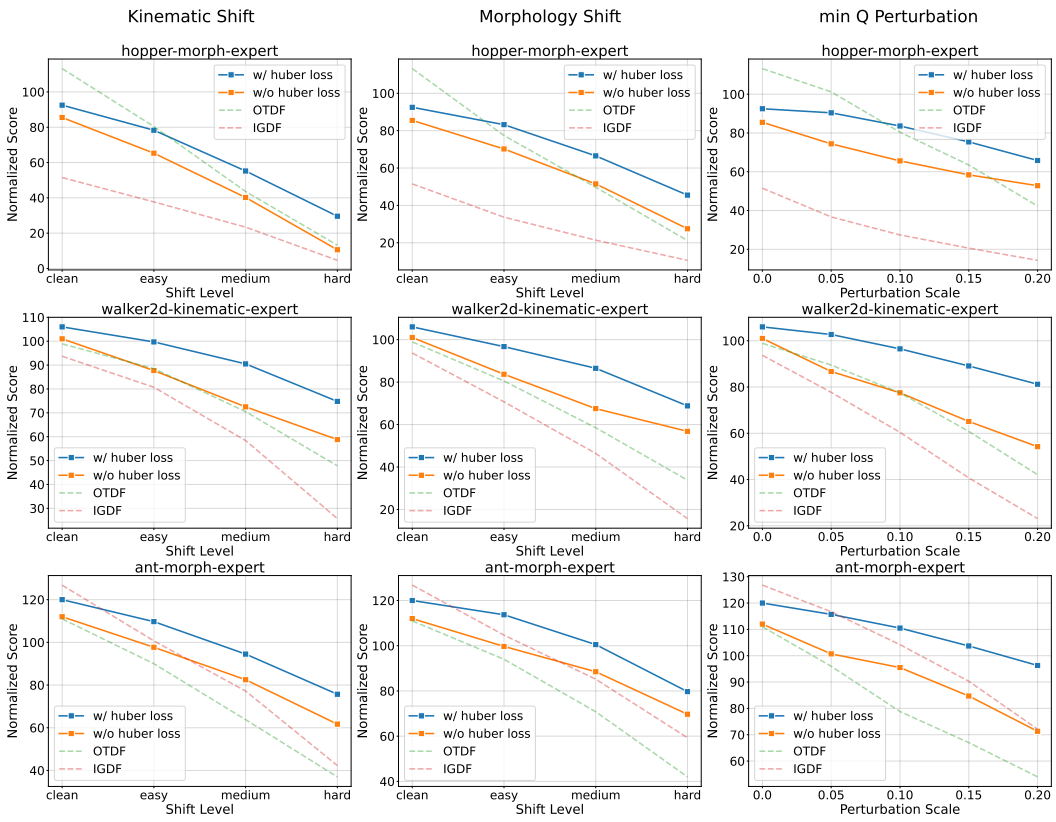


Figure 11: Ablation study on Huber loss

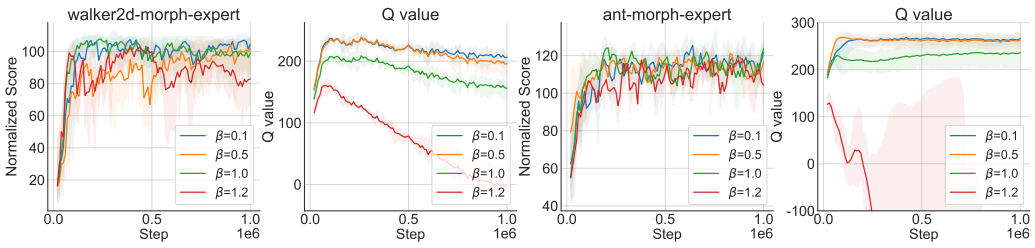


Figure 12: Effect of β

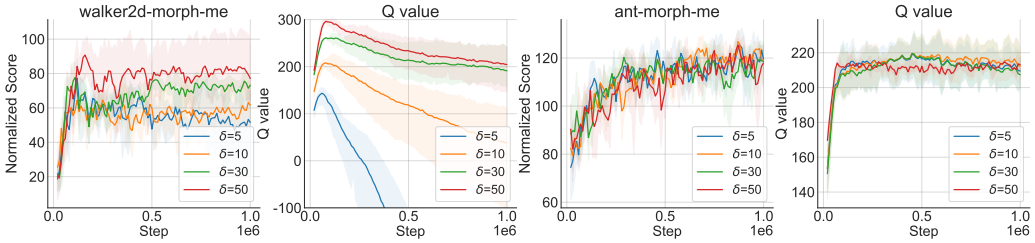


Figure 13: Effect of δ

datasets, with Figure 13 showing the performance and Q value learning curves. Our results demonstrate dataset-dependent sensitivity to δ: while $\delta = 50$ exhibits a satisfying performance and $\delta = 5$ yields suboptimal performance on `walker2d-morph-me`, DROCO is not sensitive to δ on `ant-morph-me`. Since no single δ value universally outperforms across all tasks, we specify the δ values used for each dataset in Appendix F.

Ensemble size N. We also examine the effect of dynamics model ensemble size N on training. In DROCO, N represents the sampling number within the uncertainty set. Typically, a larger N corresponds to a smaller sampling error when computing $\widehat{\mathcal{T}}_{\mathrm{RCB}}$. We conduct experiments on various datasets with N across $\{3, 5, 7, 9\}$. The results are presented in Table 3, where we find no distinct difference across different N, which means the ensemble size is not a sensitive hyperparameter. Thus, we could use the default value of 7.

Table 3: Effect of N

Dataset	$N = 3$	$N = 5$	$N = 7$	$N = 9$
half-me-kinematic	58.4 ± 4.4	62.2 ± 8.6	60.1 ± 7.1	57.9 ± 9.6
half-me-morph	65.4 ± 8.4	71.7 ± 5.9	70.1 ± 5.6	74.9 ± 3.3
half-e-kinematic	68.7 ± 6.8	67.0 ± 4.7	67.4 ± 5.8	66.3 ± 7.2
half-e-morph	76.0 ± 4.1	75.6 ± 5.1	79.2 ± 3.9	78.4 ± 3.0
hopper-me-kinematic	51.7 ± 3.4	54.4 ± 5.7	54.0 ± 6.4	52.5 ± 6.9
hopper-me-morph	85.6 ± 6.7	84.9 ± 5.5	82.3 ± 4.1	83.2 ± 4.0
hopper-e-kinematic	88.3 ± 10.2	87.0 ± 8.1	89.3 ± 9.6	86.9 ± 7.2
hopper-e-morph	91.1 ± 1.0	94.9 ± 2.2	92.5 ± 1.2	90.7 ± 0.8
Average	73.2	74.7	74.4	73.9

E.6 Performance Comparison under Observation and Reward Shifts

In this part, we further examine the generality of DROCO under observation and reward shifts, in addition to dynamics shifts.

Observation shift. To simulate the observation shift, we follow the observation corruption setting in (Yang et al., 2024b), and corrupt 30% source domain data by modifying the state of transitions (s, a, r, s') to $\hat{s} = s + \lambda \cdot \mathrm{std}(s), \lambda \sim \mathrm{Uniform}[-1, 1]^{d_s}$. d_s represents the state dimension, and

Table 4: Performance comparison under observation shifts without observation normalization.

Dataset	IQL	IGDF	OTDF	DROCO
half-e-kinematic	21.7±6.2	13.4±2.0	28.6±3.4	**34.7**±5.8
half-e-morph	43.3±9.5	33.8±5.8	**46.4**±8.3	40.8±5.5
half-me-kinematic	38.9±4.4	40.0±4.6	37.5±6.2	**46.3**±9.3
half-me-morph	37.2±4.1	**45.3**±5.4	33.7±4.0	43.4±6.8
hopper-e-kinematic	34.6±6.4	43.5±4.8	36.2±7.9	**48.6**±7.4
hopper-e-morph	60.6±4.5	32.3±4.9	53.9±11.5	**62.9**±8.7
hopper-me-kinematic	1.4±0.1	0.0±0.0	16.9±3.1	**20.3**±7.4
hopper-me-morph	16.7±2.3	22.7±4.0	31.4±5.7	**36.5**±6.8
Average	31.8	28.9	35.6	**41.7**

Table 5: Performance comparison under observation shifts with observation normalization.

Dataset	IQL	IGDF	OTDF	DROCO
half-e-kinematic	26.5±4.7	21.4±5.0	33.8±6.1	**42.6**±7.1
half-e-morph	42.7±7.0	42.5±5.7	**51.9**±4.4	44.6±7.4
half-me-kinematic	41.2±5.2	35.5±2.9	44.3±2.9	**51.3**±3.6
half-me-morph	46.4±3.6	**49.2**±6.0	45.6±3.3	43.0±2.1
hopper-e-kinematic	39.6±7.3	**57.8**±6.2	49.3±5.7	54.4±9.3
hopper-e-morph	66.3±6.9	38.5±3.7	57.7±4.0	**73.2**±6.2
hopper-me-kinematic	9.0±1.3	2.4±0.1	22.2±3.4	**34.2**±5.6
hopper-me-morph	16.4±2.4	29.7±3.0	26.7±1.1	**46.6**±8.2
Average	36.0	34.6	41.4	**48.7**

$\mathrm{std}(s)$ is the d_s-dimensional standard deviation of all states in the source dataset. Our experiments consist of two parts: **(1)** we directly employ several baselines (IQL, IGDF, OTDF) and DROCO in this observation shift setting without introducing other techniques; **(2)** we introduce the observation normalization technique (Yang et al., 2024b) to baselines and DROCO. Both parts of the experiments are conducted on multiple datasets, with results presented in Table 4 and Table 5. We find that introducing observation shifts would degrade the algorithm's performance, and the observation normalization technique can mitigate performance degradation. In both experimental settings, DROCO demonstrates better performance than baselines on most datasets.

Reward shift. To examine the generality of DROCO to reward shifts, we further design a reward shift setting: we randomly select 30% of source transitions (s, a, r, s') and modify the reward r to $\hat{r} \sim \mathrm{Uniform}[-1, 1]$. That is, we completely abandon the reward information and switch to random rewards.

Under this reward shift setting, we conduct experiments on multiple datasets to compare the performance of DROCO with baseline methods (IQL, IGDF, OTDF). The experimental results are reported in Table 6. Surprisingly, we find that reward shift does not significantly affect performance. This observation may be explained by the survival instinct of offline RL (Li et al., 2023), which suggests that offline RL naturally exhibits robustness to misspecified reward.

The results show that DROCO still outperforms other baselines under both reward shift and dynamics shift settings. We attribute the enhanced performance of DROCO under observation and reward shifts to the components of dynamic value penalty and Huber loss which mitigate value estimation error caused by observation and reward shifts. We believe this finding, along with our above results under the observation shift setting, demonstrates the generality of DROCO across observation, reward, and dynamics shift.

Table 6: Performance comparison under reward shifts.

Dataset	IQL	IGDF	OTDF	DROCO
half-e-kinematic	47.5±4.2	45.8±3.0	**72.2**±3.8	66.0±6.3
half-e-morph	60.7±5.3	52.2±3.6	70.3±5.8	**76.4**±4.2
half-me-kinematic	41.1±3.7	55.2±4.4	43.6±4.9	**57.4**±3.6
half-me-morph	61.7±2.1	55.8±4.9	39.0±3.3	**63.9**±2.4
hopper-e-kinematic	58.8±6.1	67.0±5.7	**95.5**±11.3	85.0±8.2
hopper-e-morph	84.7±5.1	46.2±4.8	**100.3**±6.8	91.4±4.5
hopper-me-kinematic	10.1±1.3	8.3±0.7	42.6±6.3	**48.1**±6.4
hopper-me-morph	34.8±4.3	41.1±4.7	47.3±5.6	**78.6**±8.3
Average	49.9	46.5	63.9	**70.9**

Table 7: Performance comparison under distinct behavior policies between source and target domain datasets.

Source	Target	IQL	IGDF	OTDF	DROCO
half-medium	medium	**45.2**±0.1	**45.2**±0.1	42.2±0.1	**45.3**±0.2
half-medium	expert	47.5±1.1	45.4±1.3	**58.3**±2.8	52.6±4.2
half-expert	medium	47.1±1.5	46.8±2.4	51.7±0.4	**58.5**±0.3
half-expert	expert	49.7±3.6	47.6±2.1	**79.6**±3.0	67.4±5.8
hopper-medium	medium	48.8±2.1	54.3±6.6	46.3±3.7	**55.4**±5.3
hopper-medium	expert	56.1±4.4	61.8±4.4	69.3±3.9	**80.8**±6.2
hopper-expert	medium	53.6±2.4	**61.3**±4.7	51.4±2.1	**62.2**±4.6
hopper-expert	expert	62.6±6.9	70.1±3.2	**97.0**±3.3	89.3±9.6
walker2d-medium	medium	48.7±1.9	51.8±2.4	43.0±2.1	**70.8**±3.3
walker2d-medium	expert	71.4±3.7	82.5±5.3	76.8±4.1	**94.6**±5.8
walker2d-expert	medium	55.4±3.1	58.6±5.5	57.9±2.0	**83.0**±4.8
walker2d-expert	expert	90.1±3.2	93.7±5.8	98.9±2.1	**106.0**±0.8
ant-medium	medium	89.9±5.1	88.0±4.6	86.1±3.7	**92.7**±6.3
ant-medium	expert	107.6±1.8	**112.4**±3.3	105.9±2.3	110.3±2.0
ant-expert	medium	93.7±3.5	90.2±2.8	98.6±4.5	**100.4**±2.3
ant-expert	expert	111.0±3.3	**119.2**±5.6	111.6±2.9	**120.0**±2.1
Average		67.4	70.6	73.4	**80.6**

E.7 Performance Comparison under Distinct Source and Target Behavior Policies

In practice, the behavior policies between the source and target domain datasets could be different. To address this concern, We consider four tasks (`halfcheetah`, `hopper`, `walker2d`, `ant`) with kinematic shifts. We relax the constraint of identical behavior policies, allowing the source and target datasets to have different qualities (`medium` or `expert`). For instance, a medium-quality source dataset may be paired with either a medium- or expert-quality target dataset. All other experimental settings follow Section 5.1, with IQL, IGDF, and OTDF as baselines. The results are presented in Table 7. The results indicate that DROCO maintains its superiority over the baselines even when the source and target behavior policies differ. It achieves the highest average score (80.6) and best performance on 12 out of 16 datasets. These findings demonstrate the effectiveness of DROCO in scenarios with differing behavior policies.

F Hyperparameter Setup

In this section, we provide the detailed hyperparameter setup for DROCO in our experiments. In Table 8, we list the network architecture and the training setup of DROCO, as well as the main hyperparameters of IQL, since we utilize IQL for policy optimization. The distinct value of β and δ for each dataset under kinematic shifts and morphology shifts are presented in Table 9 and Table 10.

Table 8: Hyperparameter setup for DROCO

Hyperparameter	Value
Network	
Actor network	(256, 256)
Critic network	(256, 256)
Ensemble model network	(400,400,400,400)
Ensemble size	7
Activation function	ReLU (Agarap, 2018)
Training	
Learning rate	3×10^{-4}
Optimizer	Adam (Kingma & Ba, 2014)
Discount factor	0.99
Target update rate	5×10^{-3}
Source domain batch size	128
Target domain batch size	128
Dynamics model batch size	256
Dynamics model training steps	1×10^{5}
Policy training steps	1×10^{6}
IQL	
Temperature coefficient	0.2
Maximum log std	2
Minimum log std	-20
Inverse temperature parameter β	3.0
Expectile parameter τ	0.7

Table 9: Detailed hyperparameter setup for DROCO, where the source domain datasets are under **kinematic shifts**.

Dataset	Value of β	Value of δ
half-m	0.1	30
half-mr	0.5	50
half-me	0.5	30
half-e	0.1	30
hopp-m	0.1	50
hopp-mr	0.5	50
hopp-me	1.0	30
hopp-e	0.5	30
walk-m	1.0	50
walk-mr	0.5	30
walk-me	0.5	50
walk-e	0.1	10
ant-m	0.1	30
ant-mr	1.0	30
ant-me	0.1	30
ant-e	1.0	30

Table 10: Detailed hyperparameter setup for DROCO, where the source domain datasets are under **morphology shifts**.

Dataset	Value of β	Value of δ
half-m	0.1	10
half-mr	0.5	50
half-me	1.2	30
half-e	1.2	30
hopp-m	0.5	50
hopp-mr	0.1	50
hopp-me	0.1	10
hopp-e	0.1	10
walk-m	0.1	50
walk-mr	0.5	50
walk-me	0.1	10
walk-e	0.1	10
ant-m	0.1	30
ant-mr	0.1	30
ant-me	0.1	10
ant-e	1.0	30

G Compute Infrastructure

The compute infrastructure we use for all experiments is listed in Table 11.

Table 11: Compute Infrastructure

CPU	**GPU**	**Memory**
AMD EPYC 7452	RTX3090×8	288GB

Table 12: Training time comparison between various methods. h=hour(s), m=minute(s).

IQL*	CQL*	BOSA	DARA	IGDF	OTDF	DROCO
5h24m	10h22m	5h49m	6h13m	6h56m	9h17m	7h26m

H TIME COST

We list the training time of DROCO and all baselines (IQL*, CQL*, BOSA, DARA, IGDF, OTDF) for 1M training steps in Table 12. We note that the additional time cost for DROCO mainly comes from the training of the ensemble dynamics model. However, since we can save the trained dynamics model weights, no retraining is required for subsequent experiments.

I BROADER IMPACTS

This paper presents a method aimed at enhancing dual robustness against dynamic shifts in cross-domain offline RL. Our work has potential positive social impacts; for example, it could inspire the development of humanoid robots capable of robust performance in non-stationary environments. Currently, we have not identified any negative impacts of our research.

J DECLARATION ON LLM USE

In this work, LLMs are used solely for grammar polishing of an early draft and are excluded from core aspects of the research, such as method conception, theoretical proof, and experimental work.