H2O+: An Improved Framework for Hybrid Offline-and-Online RL with Dynamics Gaps

Haoyi Niu[♠†], **Tianying Ji**[♠†], **Bingqi Liu**[‡], **Haocheng Zhao**[†], **Xiangyu Zhu**[†], **Jianying Zheng**[‡], **Pengfei Huang**[†], **Guyue Zhou**[†], **Jianming Hu**[♣†], **Xianyuan Zhan**[♣†*]

Abstract

Solving real-world complex tasks using reinforcement learning (RL) without high-fidelity simulation environments or large amounts of offline data can be quite challenging. Online RL agents trained in imperfect simulation environments can suffer from severe sim-to-real issues. Offline RL approaches although bypass the need for simulators, often pose demanding requirements on the size and quality of the offline datasets. The recently emerged hybrid offline-and-online RL provides an attractive framework that enables joint use of limited offline data and imperfect simulator for transferable policy learning. In this paper, we develop a new algorithm, called **H2O+**, which offers great flexibility to bridge various choices of offline and online learning methods, while also accounting for dynamics gaps between the real and simulation environment. Through extensive simulation and real-world robotics experiments, we demonstrate superior performance and flexibility over advanced cross-domain online and offline RL algorithms. The real-world experiment videos are available at https://sites.google.com/view/h2oplusauthors/.

Keywords: Hybrid Offline-and-Online RL, Dynamics gaps

1 Introduction

The past successes of reinforcement learning (RL) are primarily restricted to single-domain tasks with the same environment dynamics during the training and testing phases (Silver et al., 2017; Mnih et al., 2015). However, it has been observed that most RL algorithms are highly vulnerable to changes in environment dynamics (Luo et al., 2022; Eysenbach et al., 2020; Niu et al., 2022), resulting in suboptimal policy performance and limiting the broader success of RL in real-world tasks. In robotics applications (Kober et al., 2013; Lee et al., 2020; O'Connell et al., 2022; Andrychowicz et al., 2020), for instance, we typically train control policies in simulators for the sake of training efficiency and safety considerations. However, the dynamics modeling within the simulator can be hard to strictly align with the diverse and complex real-world scenarios, leading to severe performance degradation due to dynamics mismatch (Peng et al., 2018; Sandha et al., 2021; Akkaya et al., 2019).

To address sim-to-real transfer issues, recent RL methods have adopted several design paradigms. System identification methods (Yu et al., 2017; Chebotar et al., 2019; Muratore et al., 2021; Du et al., 2021; Ramos et al., 2019) aim to calibrate and align the physical properties in simulation with those in the real world. Domain randomization techniques (Peng et al., 2018; Rajeswaran et al., 2016; Mehta et al., 2020; Akkaya et al., 2019) randomize simulation parameters to generalize policies across multiple environments. However, the

∗. [†]Tsinghua University. [‡]Beihang University. [♠]Equal Contribution. [♣]Equal Correspondence. {nhy22, jity20}@mails.tsinghua.edu.cn, {hujm@mail,zhanxianyuan@air}.tsinghua.edu.cn.

selection of parameters and the range of their randomization could require a great amount of human effort and domain expertise (Vuong et al., 2019; Andrychowicz et al., 2020), as well as sufficient configurability of the simulator (Chen et al., 2022). Thus, another avenue of works (Eysenbach et al., 2020; Liu et al., 2022) regards simulators as black boxes and turn to perform policy learning adaptation via modifying the reward to account for the sim-to-real dynamics gap. More recently, the rapid developments in offline RL (Levine et al., 2020; Fujimoto et al., 2019; Kumar et al., 2019; Fujimoto and Gu, 2021; Kostrikov et al., 2022; Xu et al., 2022, 2023; Garg et al., 2023) have brought renewed interest in learning policies directly from pre-collected real-world datasets to bypass the need for simulation environments. These methods adopt conservative principles to overcome the notorious distributional shift issue (Kumar et al., 2019) in offline learning, thus often requiring large, high state-action space coverage and high-quality datasets to achieve good performance (Li et al., 2023), which can be hard to satisfy in scenarios with high data collection costs.

All of the aforementioned approaches bear certain limitations, suggesting that solely relying on online simulation samples with imperfect dynamics or potentially limited, low-coverage real-world offline data may not be sufficient to achieve desirable policy transferability. To this end, dynamics-aware hybrid offline-and-online RL (H2O) (Niu et al., 2022) is the first study to combine offline and online policy learning using both limited offline real-world data and off-dynamics online simulated samples for cross-domain policy learning. It introduces a dynamics-aware value regularization scheme that punishes Q-values on simulation samples based on explicit dynamics gap quantification and boosts Q-values on offline real data. Although promising, H2O also bears several drawbacks. First, it is built upon the conservative Q-learning (CQL) (Kumar et al., 2020) offline RL framework, which is over-conservative and lacks flexibility for extension to stronger and less conservative offline RL paradigms. Its over-conservative design hinders sufficient exploration and state-action coverage improvement in the simulation environment. Lastly, explicit dynamics gap quantification in H2O also poses computation challenges.

In this paper, we follow the offline-and-online RL recipe in H2O, but develop a more flexible and powerful algorithm through a different lens, to enable sufficient utilization of both the offline dataset and imperfect simulator for transferable policy learning. We refer our algorithm as **H2O+**, which has two favorable design ingredients: 1) a *flexible and less conservative learning framework* that is compatible with various strong in-sample learning offline RL backbones and exploration designs; and 2) the *dynamics-aware mixed value update* that bridges offline and online value function learning, while also accounting for dynamics gaps between real and simulated samples. Through extensive simulation and real wheel-legged robot experiments, we demonstrate the superiority and flexibility of **H2O+** over competing online, offline and cross-domain RL baseline methods.

2 Related Work

High-fidelity simulators are crucial for online RL methods to learn deployable policies. However, as accurate simulators are hard to build, addressing the sim-to-real gaps become a pressing challenge. Various cross-domain online RL approaches have been proposed to tackle this challenge, such as using system identification methods (Ljung, 1998; Chebotar et al., 2019; Muratore et al., 2021; Du et al., 2021; Ramos et al., 2019) to align simulated dynamics with

real dynamics, or adding domain randomizations (Peng et al., 2018; Rajeswaran et al., 2016; Andrychowicz et al., 2020; Mehta et al., 2020; Akkaya et al., 2019) that trains RL policies in a randomized simulated dynamics setting. The former typically requires a considerable amount of offline or costly real-world interaction data (Yu et al., 2017), while the latter necessitates manually-specified randomized parameters (Vuong et al., 2019). Recently, another line of research leverages additional real-world data to mitigate the dynamics shift in simulation environments (Eysenbach et al., 2020; Liu et al., 2022; Niu et al., 2022). Specifically, DARC (Eysenbach et al., 2020) and DARA (Liu et al., 2022) add dynamics-gap-related penalization terms on rewards in online and offline RL settings, respectively. H2O (Niu et al., 2022) proposes a new setting that enables simultaneous offline-and-online policy learning on both real offline data and simulated samples, which shows promising results and advantages over prior methods.

3 Preliminaries

Reinforcement Learning We formulate RL problem as a Markov Decision Process (MDP) (Sutton et al., 1998), defined by a tuple $\mathcal{M} := (\mathcal{S}, \mathcal{A}, r, P_{\mathcal{M}}, \gamma)$. $\mathcal{S}$ and $\mathcal{A}$ denote the state and action space, r represents the reward function, $P_{\mathcal{M}}$ stands for the transition dynamics under $\mathcal{M}$. The goal of RL is to find the optimal policy π^* that maximizes cumulative discounted reward starting from an initial state distribution ρ, $\pi^* = \arg\max_\pi \mathbb{E}_{\mathbf{s}_0 \in \rho, \mathbf{a}_t \sim \pi, \mathbf{s}_{t+1} \sim P_{\mathcal{M}}} \left[\sum_{t=0}^{\infty} \gamma^t r\left(\mathbf{s}_t, \mathbf{a}_t\right)\right]$. RL methods based on approximated dynamic programming typically learn an action-value function $Q(s, a)$, and optionally, a state value function $V(s)$ to practically estimate the cumulative discounted reward for policy optimization.

In many cases, RL training in the real environment is infeasible, so most online RL methods train the agents in simulation environments. However, building a high-fidelity simulator can be costly or even impossible in many real-world tasks. Learning with an imperfect simulator will lead to a MDP $\widehat{\mathcal{M}} := \left(\mathcal{S}, \mathcal{A}, r, P_{\widehat{\mathcal{M}}}, \gamma\right)$ with biased dynamics $P_{\widehat{\mathcal{M}}}$, which can cause serious sim-to-real transfer issues. When a large offline real-world dataset $\mathcal{D}$ generated by some behavior policy μ is given, one can also resort to offline RL (Fujimoto et al., 2019; Levine et al., 2020) to bypass the sim-to-real issue and directly learn a policy from the offline data. However, the performances of existing offline RL methods are heavily dependent on the size and quality of datasets, which restricts their practical application (Li et al., 2023).

Hybrid Offline-and-Online RL with Imperfect Simulator As both online and offline RL bear some practical challenges in solving real-world problems, there is a growing interest in merging online and offline RL for sample-efficient and high-performance policy learning (Song et al., 2023; Ball et al., 2023; Wagenmaker and Pacchiano, 2022; Niu et al., 2022). Many of these studies (Song et al., 2023; Ball et al., 2023; Wagenmaker and Pacchiano, 2022) assume identical online and offline system dynamics, thus are not applicable if an imperfect simulator is used as the online environment. Among them, H2O (Niu et al., 2022) is the first study that enables simultaneous offline and online policy learning with an imperfect simulator.

H2O is built upon the conservative Q-learning (CQL) (Kumar et al., 2020) framework, with its learning objective designed as follows:

$$\min_{Q} \underbrace{\alpha_c \cdot \left(\log \sum_{\mathbf{s},\mathbf{a}} \omega(\mathbf{s},\mathbf{a}) \exp\left(Q(\mathbf{s},\mathbf{a})\right) - \mathbb{E}_{\mathbf{s},\mathbf{a}\sim\mathcal{D}}\left[Q(\mathbf{s},\mathbf{a})\right] \right)}_{\textit{(i) Conservative value regularization}} + \underbrace{\mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}}\left[\left(Q - \hat{\mathcal{B}}^{\pi}\hat{Q}\right)(\mathbf{s},\mathbf{a})\right]^2 + \mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim B} \frac{P_{\mathcal{M}}(\mathbf{s}'|\mathbf{s},\mathbf{a})}{P_{\widehat{\mathcal{M}}}(\mathbf{s}'|\mathbf{s},\mathbf{a})} \left[\left(Q - \hat{\mathcal{B}}^{\pi}\hat{Q}\right)(\mathbf{s},\mathbf{a})\right]^2}_{\textit{(ii) Bellman error on offline and online data}} \tag{1}$$

H2O's learning objective is comprised of two parts: the first part pushes down dynamics-gap weighted Q-values and pulls up Q-values on trustworthy real offline data; the second part enables simultaneous offline and online learning on both offline dataset $\mathcal{D}$ and simulated replay buffer B while also correcting the problematic next state $\mathbf{s}'$ from the simulator dynamics $P_{\widehat{\mathcal{M}}}$ using the dynamics ratio as an importance weight. In H2O, the dynamics gap measure $\omega(\mathbf{s},\mathbf{a})$ is explicitly calculated as the normalized KL-divergence $D_{KL}(P_{\widehat{\mathcal{M}}}(\mathbf{s}'|\mathbf{s},\mathbf{a})\|P_{\mathcal{M}}(\mathbf{s}'|\mathbf{s},\mathbf{a}))$ over all $(\mathbf{s},\mathbf{a})$ pairs in the state-action space, which can only be approximated.

4 Method

Although H2O provides a successful attempt to tackle sim-to-real dynamics gaps by combining offline and online RL, it suffers from four notable drawbacks. First, the CQL backbone of H2O is over-conservative (Nakamoto et al., 2023; Li et al., 2023) and may cause conflict when incorporating online learning. For example, as shown in (Nakamoto et al., 2023), when performing online fine-tuning on a conservative value function initialization, policy learning has to first "unlearn" the underestimated values before making further progress. Second, the CQL framework lacks flexibility, which is not possible to be extended nor compatible with many recent strong offline RL frameworks (Kostrikov et al., 2022; Xu et al., 2023; Hansen-Estruch et al., 2023; Garg et al., 2023). H2O also has no exploration design, which hinders effective state-action coverage improvement through simulation interactions. Lastly, H2O has to approximate an explicit dynamics gap measure, which is costly and error-prone. These drawbacks motivate us to rethink what are the desirable properties in hybrid offline-and-online RL. In this paper, we propose **H2O+**, which offers a highly flexible and less conservative algorithm, enabling full utilization of the online samples from the imperfect simulator. The key of **H2O+** is the dynamics-aware mixed value update, which bridges various choices of offline and online learning methods, while also accounting for dynamics gaps between the real and simulation environment.

4.1 Separate Considerations for Offline and Online Learning

Before introducing our method, we first review two popular modeling frameworks in both offline and online RL: behavior-regularized RL (Eq.(2)) and maximum entropy RL (Eq.(3)):

$$\text{Offline: } \max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \left(r\left(\mathbf{s}_t,\mathbf{a}_t\right) - \alpha \cdot f\left(\frac{\pi\left(\mathbf{a}_t \mid \mathbf{s}_t\right)}{\mu\left(\mathbf{a}_t \mid \mathbf{s}_t\right)}\right)\right)\right] \tag{2}$$

$$\text{Online: } \max_{\pi} \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \left(r\left(\mathbf{s}_t,\mathbf{a}_t\right) + \beta \cdot \mathcal{H}\left(\pi\left(\mathbf{a}_t \mid \mathbf{s}_t\right)\right)\right)\right] \tag{3}$$

The behavior-regularized RL is formally studied in (Xu et al., 2023), which has been shown closely related to a class of recent state-of-the-art (SOTA) in-sample learning offline RL

methods (Xu et al., 2023; Hansen-Estruch et al., 2023). Depending on different choices of the f function, it can be shown that all these algorithms share the following general learning objectives for $V(\mathbf{s})$ and $Q(\mathbf{s}, \mathbf{a})$:

$$\min_V \mathbb{E}_{(\mathbf{s},\mathbf{a})\sim\mathcal{D}}\, \mathcal{L}_V^f(Q(\mathbf{s},\mathbf{a}) - V(\mathbf{s})) \tag{4}$$

$$\min_Q\ \mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}}\left[r(\mathbf{s},\mathbf{a}) + \gamma V(\mathbf{s}') - Q(\mathbf{s},\mathbf{a})\right]^2 \tag{5}$$

In particular, if $f = \log(x)$, it correspond to EQL (Xu et al., 2023) and XQL (Garg et al., 2023) with $\mathcal{L}_V^f(y) = \exp(y/\alpha) - y/\alpha$. If $f = x - 1$, it corresponds to SQL (Xu et al., 2023) (equivalent to an in-sample learning version of CQL) with $\mathcal{L}_V^f(y) = \mathbb{1}(1 + y/2\alpha > 0)(1 + y/2\alpha)^2 - y/\alpha$. The well-known offline RL algorithm IQL (Kostrikov et al., 2022) also belongs to this family of algorithms but does not have a closed-form f, with $\mathcal{L}_V^f(y) = |\tau - \mathbb{1}(y < 0)|y^2$, where $\tau \in (0, 1)$ is the expectile hyperparameter. These offline RL methods learn $V(\mathbf{s})$ and $Q(\mathbf{s}, \mathbf{a})$ completely using dataset samples, thus enjoying stable value function learning as compared to CQL-style algorithms that distort the value estimates. However, their in-sample learning nature also creates obstacles to incorporating online learning with imperfect simulators.

On the other hand, the maximum entropy RL (Haarnoja et al., 2018) (Eq.(3)) also achieves great success in online RL studies, which maximizes the expected reward while also maximizing the entropy of the policy $\mathcal{H}(\pi)$ to promote exploration. If we consider an off-policy setting and denote B as the training replay buffer, its corresponding action-value function learning objective is given as:

$$\min_Q\ \mathbb{E}_{(\mathbf{s},\mathbf{a})\sim B}\left[r(\mathbf{s},\mathbf{a}) + \gamma \mathbb{E}_{\mathbf{s}'\sim P, \mathbf{a}'\sim\pi}\left[\hat{Q}(\mathbf{s}',\mathbf{a}') - \beta\cdot\log(\pi(\mathbf{a}'|\mathbf{s}'))\right] - Q(\mathbf{s},\mathbf{a})\right]^2 \tag{6}$$

4.2 Dynamics-Aware Mixed Value Update

Both the behavior-regularized RL and the maximum entropy RL frameworks bear some attractive features for the hybrid offline-and-online RL setting. Specifically, behavior-regularized RL ensures high-quality offline learned value functions without posing too much conservatism. While maximum entropy RL offers natural exploration capabilities to improve the state-action space coverage of the offline dataset. Now the question is: *how can we leverage the merits of both frameworks to build a strong hybrid RL algorithm while also being capable of tackling the sim-to-real dynamics gaps between real and simulation environments?*

In this paper, we provide a simple and elegant solution by proposing a dynamics-aware mixed value update that seamlessly mixes offline and online learning without introducing excessive conservatism. Our insight is by noting that we can use the more reliable state value function $V(\mathbf{s})$ learned solely with the real offline data in Eq.(4) as an anchor to mildly regulate $Q(\mathbf{s}, \mathbf{a})$ estimation on potentially biased simulation samples. We can achieve this by utilizing the following mixed Bellman operator, sharing a similar philosophy explored by previous work on balancing exploitation and exploration in online RL (Ji et al., 2023):

$$\mathcal{B}_\lambda^{\text{mix}}\hat{Q}(\mathbf{s},\mathbf{a}) = \lambda\left[r(\mathbf{s},\mathbf{a}) + \gamma V(\mathbf{s}')\right] + (1-\lambda)\left[r(\mathbf{s},\mathbf{a}) + \gamma\mathbb{E}_{\mathbf{a}'\sim\pi}\left[\hat{Q}(\mathbf{s}',\mathbf{a}') - \beta\cdot\log(\pi(\mathbf{a}'|\mathbf{s}'))\right]\right] \tag{7}$$

$$= r(\mathbf{s},\mathbf{a}) + \lambda\gamma V(\mathbf{s}') + (1-\lambda)\gamma\mathbb{E}_{\mathbf{a}'\sim\pi}\left[\hat{Q}(\mathbf{s}',\mathbf{a}') - \beta\cdot\log(\pi(\mathbf{a}'|\mathbf{s}'))\right] \tag{8}$$

where the state value function is learned only with real-world offline data $\mathcal{D}$ as in Eq.(4); $\lambda \in [0, 1]$ is a trade-off hyperparameter to control the level of influence between offline and

online learning. With the mixed Bellman operator, we can learn Q-function from both real dataset $\mathcal{D}$ and online simulation replay buffer B. Moreover, to correct potentially problematic next states $\mathbf{s}'$ from the simulator dynamics $P_{\widehat{\mathcal{M}}}$, we adopt the same dynamics ratio reweighting as in H2O (Niu et al., 2022):

$$\min_Q \mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}}\left[\left(Q-\mathcal{B}_\lambda^{\text{mix}}\hat{Q}\right)^2(\mathbf{s},\mathbf{a})\right]+\mathbb{E}_{(\mathbf{s},\mathbf{a})\sim B}\mathbb{E}_{\mathbf{s}'\sim P_{\mathcal{M}}}\left[\left(Q-\mathcal{B}_\lambda^{\text{mix}}\hat{Q}\right)^2(\mathbf{s},\mathbf{a})\right] \tag{9}$$

$$=\mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim\mathcal{D}}\left[\left(Q-\mathcal{B}_\lambda^{\text{mix}}\hat{Q}\right)^2(\mathbf{s},\mathbf{a})\right]+\mathbb{E}_{(\mathbf{s},\mathbf{a},\mathbf{s}')\sim B}\left[\frac{P_{\mathcal{M}}(\mathbf{s}'\mid\mathbf{s},\mathbf{a})}{P_{\widehat{\mathcal{M}}}(\mathbf{s}'\mid\mathbf{s},\mathbf{a})}\left(Q-\mathcal{B}_\lambda^{\text{mix}}\hat{Q}\right)^2(\mathbf{s},\mathbf{a})\right] \tag{10}$$

The dynamics ratio $P_{\widehat{\mathcal{M}}}/P_{\mathcal{M}}$ can be conveniently estimated by learning a pair of domain discriminators $p(\text{real}|\mathbf{s},\mathbf{a},\mathbf{s}')$ and $p(\text{real}|\mathbf{s},\mathbf{a})$ using the following formulation, which is also adopted in a number of previous studies (Eysenbach et al., 2020; Liu et al., 2022):

$$\frac{P_{\mathcal{M}}(\mathbf{s}'\mid\mathbf{s},\mathbf{a})}{P_{\widehat{\mathcal{M}}}(\mathbf{s}'\mid\mathbf{s},\mathbf{a})}=\frac{p(\mathbf{s}'\mid\mathbf{s},\mathbf{a},\text{real})}{p(\mathbf{s}'\mid\mathbf{s},\mathbf{a},\text{sim})}=\frac{p(\text{sim}\mid\mathbf{s},\mathbf{a})}{p(\text{real}\mid\mathbf{s},\mathbf{a})}\Big/\frac{p(\text{sim}\mid\mathbf{s},\mathbf{a},\mathbf{s}')}{p(\text{real}\mid\mathbf{s},\mathbf{a},\mathbf{s}')}=\frac{1-p(\text{real}\mid\mathbf{s},\mathbf{a})}{p(\text{real}\mid\mathbf{s},\mathbf{a})}\Big/\frac{1-p(\text{real}\mid\mathbf{s},\mathbf{a},\mathbf{s}')}{p(\text{real}\mid\mathbf{s},\mathbf{a},\mathbf{s}')} \tag{11}$$

Finally, with the learned action value function $Q(\mathbf{s},\mathbf{a})$, we can optimize the policy π by maximizing the following objective on both real and simulated samples:

$$\pi^*=\arg\max_\pi \mathbb{E}_{(\mathbf{s},\mathbf{a})\in\mathcal{D}\cup B}[Q(\mathbf{s},\mathbf{a})-\beta\cdot\log(\pi(\mathbf{s},\mathbf{a}))] \tag{12}$$

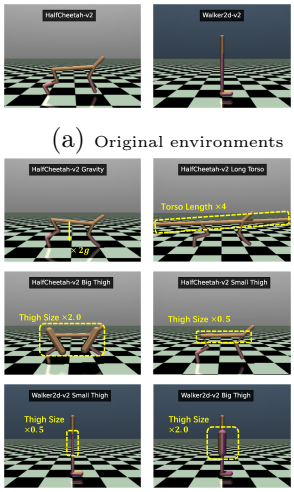
5 Experiments

In this section, we present empirical validations of our approach. We begin with our algorithmic implementation and experimental setups, followed by benchmark experiments with original and dynamics-modified MuJoCo simulation environments. Our baselines consist of the online RL method SAC (Haarnoja et al., 2018) for zero-shot transfer, offline RL algorithms CQL (Kumar et al., 2020) and IQL (Kostrikov et al., 2022), cross-domain online RL method DARC (Eysenbach et al., 2020), and H2O (Niu et al., 2022) that in a similar setting with **H2O+**. We run all experiments with 5 random seeds. Finally, we deploy **H2O+** and baselines on a wheel-legged robot to complete real-world tasks. Furthermore, we provide ablations on choices of dynamics-aware mixed value update designings, different levels of intensity of dynamics gap, and different offline RL backbones.

5.1 Experimental Setups

Algorithmic implementation of H2O+ In all our comparative experiments, we instantiate $\mathcal{L}_V^f(y)=|\tau-\mathbb{1}(y<0)|y^2$ as in IQL (Kostrikov et al., 2022), due to its simplicity. The scaling parameter β of the entropy term is automatically tuned following the treatment in SAC (Haarnoja et al., 2018). We follow the treatment in SAC (Haarnoja et al., 2018) to automatically tune the scaling parameter β of the entropy term. We set the trade-off hyperparameter λ to 0.1 in all our experiments. It might be preferable to select a larger λ for tasks that are carried out in more reliable simulators.

Simulation experiments We treat the original MuJoCo task environment as the "real-world" scenario, and create ten imperfect simulation environments by deliberately introducing various types of dynamics gaps (illustrated in Figure 1). These dynamics gaps are introduced by adjusting either the dynamics parameters of the robot or the environmental physical



(a) Original environments

(b) Example modified environments

Figure 1: Original environments and some illustrations of the modified dynamics

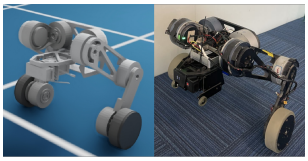
Table 1: Average returns for MuJoCo HalfCheetah and Walker2d tasks

Data	Dynamics Gap	SAC	CQL	IQL	DARC	H2O	**H2O+**
HalfCheetah-mr	Gravity	4513±513	5774±214	5207±149	5105±460	**6813±289**	**6861±268**
	Friction	2684±2646	5774±214	5207±149	5503±263	5928±896	**6278±1336**
	Joint Noise	4137±805	5774±214	5207±149	5137±225	6747±427	**6985±328**
	Big Thigh	4509±877	5774±214	5207±149	5336±389	6278±305	**6675±231**
	Small Thigh	6632±1027	5774±214	5207±149	**8331±454**	6751±231	7425±148
	Broken Thigh	6517±1076	5774±214	5207±149	**8704±1726**	6717±226	7018±147
	Flexible Thigh	5623±2862	5774±214	5207±149	5554±88	6976±234	**7497±196**
	Long Torso	1047±3089	5774±214	5207±149	45±322	6225±100	**6718±245**
	Soft Feet	5684±587	5774±214	5207±149	**9058±374**	6731±319	7068±244
	Mean Return	4594	5774	5207	5863	6573	**6947**
HalfCheetah-m	Gravity	4513±513	6066±73	5605±25	5011±456	**7085±416**	**6965±659**
	Friction	2684±2646	6066±73	5605±25	6113±104	6848±445	**7186±859**
	Joint Noise	4137±805	6066±73	5605±25	5484±171	7212±236	**7503±237**
	Big Thigh	4509±877	6066±73	5605±25	6302±1832	6625±579	**7094±371**
	Small Thigh	6632±1027	6066±73	5605±25	**9127±907**	7020±337	7706±185
	Broken Thigh	6517±1076	6066±73	5605±25	**7509±707**	6800±378	7321±213
	Flexible Thigh	5623±2862	6066±73	5605±25	7266±1771	7005±757	**7805±139**
	Long Torso	1047±3089	6066±73	5605±25	724±921	**6327±602**	5484±1382
	Soft Feet	5684±587	6066±73	5605±25	6952±3330	7138±326	**7622±53**
	Mean Return	4594	6066	5605	6054	6896	**7187**
Walker2d-mr	Gravity	1698±1611	3261±802	3390±326	2969±1043	3366±740	**3518±605**
	Friction	2779±870	3261±802	3390±326	3644±213	**3916±549**	**3866±840**
	Joint Noise	173±727	3261±802	3390±326	-3±0	3045±911	**3446±862**
	Big Thigh	1151±716	3261±802	**3390±326**	57±126	1789±1781	2977±771
	Small Thigh	894±519	3261±802	3390±326	1294±905	2455±1301	**3920±417**
	Broken Thigh	**3845±607**	3261±802	3390±326	893±180	2702±1054	**3911±405**
	Flexible Thigh	2518±1627	3261±802	3390±326	2511±1048	1891±1001	**3535±493**
	Mean Return	1865	3261	3390	1624	2738	**3596**

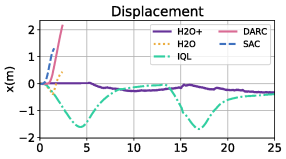
properties. For example, in the HalfCheetah and Walker2d task environment, the modifications include modifying the gravitational gravity ($\times 2$, **Gravity**), friction coefficient ($\times 0.3$, **Friction**), thigh size ($\times 0.5$ and $\times 2$, **Small/Big Thigh**), the motion range of the joint connections of thighs ($\times 0.5$ and $\times 2$, **Broken/Flexible Thigh**), stretching the torso length ($\times 4$, **Long Torso**, only for HalfCheetah), lowering the foot stiffness ($\times 0$, **Soft Feet**, only for HalfCheetah) and adding joint noise ($N(0, \mathbf{I})$, **Joint Noise**). In terms of the "real-world" offline dataset (original MuJoCo environment), we utilize the corresponding task datasets from the widely-used offline RL benchmark D4RL (Fu et al., 2020). Specifically, the Medium (-m) and Medium Replay (-mr) datasets are considered as they are closer to real-world settings, where we are more likely to obtain medium-level or highly mixed offline datasets from real systems. The online training of algorithms is performed in the created imperfect simulation environment, and we evaluate the learned policy in the original MuJoCo environment.

Real-robot transfer experiments We also perform real robot transfer experiments on a wheel-legged robot with a main body and a pair of legs with wheels attached to the end. We also construct the simulation environment based on Isaac Gym (Makoviychuk et al., 2021). Both the real robot and its simulation are shown in Figure 2a. Our wheel-legged robot bears a substantially large weight (about 12 kg) and possesses an intricate mechanical structure. Moreover, our testing environment features a furry carpet on the ground, which introduces the possibility of sagging and unloading as the robot traverses the surface. These distinctive factors collectively contribute to a very challenging real-world transfer procedure.

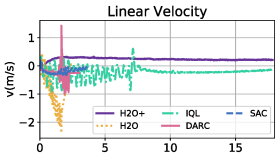
According to its sensors and actuators, the state space of the robot control tasks is designed as a quadratic-tuple $(\theta, \dot{\theta}, x, v)$ where θ denotes the forward pitch angle of the body, x is the displacement of the robot, $\dot{\theta}$ and v are the angular and linear velocity respectively. The execution action is the torque τ of the motors at the two wheels. We construct two tasks for real-world validation: (1) **standing still**: the robot needs to keep balanced at the initial location and maintain stability as much as possible. (2) **moving forward**: the robot needs



(a) Simulation/Real robot



(b) Displacement in standing still



(c) Linear velocity in moving forward

Figure 2: Real-robot experiment results of the "(b) standing still" and "(c) moving forward" tasks.

to move at a fixed forward speed $v_{tgt} = 0.2m/s$ and maintain balance as long as possible. Detailed setups are elaborated in Appendix D.

5.2 Comparative Results

Simulation experiments. The results in Table 1 highlight the superiority of **H2O+** compared to all the baselines in terms of the mean return across all tasks in the HalfCheetah and Walker2d environments. Note that for offline RL baselines (CQL/IQL), we train the policies using the "real" offline datasets and evaluate them in the "real" environments, so their scores remain the same across modified environments with dynamics gaps.

It is found that online cross-domain baseline DARC performs strongly when the dynamics gap is small (i.e. when online SAC achieves better performance than offline RL baselines). However, DARC fails miserably when the dynamics gap is large (e.g. HalfCheetah long torso and Walker2d joint noise tasks), which underscores its limitations of the dynamics-gap-related reward penalization scheme. On the other hand, offline RL baselines are not impacted by the sim-to-real issue, but their performances are heavily impacted by the dataset quality. Our **H2O+** not only outperforms H2O in most tasks but also consistently achieves comparable or better performance than online, offline, and cross-domain RL methods in both small and large dynamics gap settings. These results showcase the effectiveness of **H2O+** in leveraging both offline data and imperfect simulation for improved and transferable policy learning. In-depth ablation studies in terms of hyperparameter λ and dynamics ratio, generalizability of offline RL backbones, and ability to face different levels of dynamics gaps are in Appendix C.

Real-robot experiments In the real-robot experiments, **H2O+** demonstrated a strong transfer ability compared to other benchmarks. As shown in Figure 2b and 2c, the control performance of this method far exceeds that of others in both tasks. In the **standing still** task, only **H2O+** and IQL policies successfully maintain the balance of the robot for over 30 seconds (s), while the robot deployed with SAC, H2O, and DARC policies cannot even keep the balance for over 3s before it hit the ground. Moreover, as illustrated in Figure 2b, **H2O+** regulated the displacement of the robot within 0.2m, whereas IQL only barely maintains balance, yet with a large range swinging even reaches 1.6m. The advantage of **H2O+** is more significant in the **moving forward** task. As illustrated in Figure 2c, only the **H2O+** policy achieves the goal of moving forward and even follows the target velocity precisely. During the moving process, the speed of robot changes smoothly and the pitch angle remains steady. In comparison, IQL policy is capable to keep balance, but the robot moves backward and also spends more time and effort on keeping balance, resulting in a shaking period of over 7s. In addition, H2O, SAC and DARC fail to maintain balance and fall down in 4s. Additionally, as in Figure 3 in the Appendix, we observe that H2O explores a more focused high-value area, whereas **H2O+** spans a broader high-value region, thus indicating superior diversity characteristics in simulated data, which would benefit the overall performance.

6 Conclusion

In this paper, we propose an improved hybrid offline-and-online RL framework (**H2O+**) to enable full utilization of real-world offline datasets and imperfect simulators for cross-domain policy learning. Our method addresses several key weaknesses in the previous method H2O, and offers flexibility to bridge various choices of strong offline RL backbones without introducing excessive conservatism. Through extensive simulation and real-world experiments, we show that our method outperforms the SOTA cross-domain RL methods in a wide range of dynamics gap settings. This makes **H2O+** an ideal candidate for many real-world tasks without high-fidelity simulators and sufficient offline data.

Acknowledgement

This work is supported by National Natural Science Foundation of China under Grant No. 62333015, No. 62133002 and Beijing Natural Science Foundation L231014.

References

I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

P. J. Ball, L. Smith, I. Kostrikov, and S. Levine. Efficient online reinforcement learning with offline data. *arXiv preprint arXiv:2302.02948*, 2023.

Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. Ratliff, and D. Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8973–8979. IEEE, 2019.

S. Chen, K. Werling, A. Wu, and C. K. Liu. Real-time model predictive control and system identification using differentiable simulation. *IEEE Robotics and Automation Letters*, 8(1): 312–319, 2022.

Y. Du, O. Watkins, T. Darrell, P. Abbeel, and D. Pathak. Auto-tuned sim-to-real transfer. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1290–1296. IEEE, 2021.

B. Eysenbach, S. Chaudhari, S. Asawa, S. Levine, and R. Salakhutdinov. Off-dynamics reinforcement learning: Training for transfer with domain classifiers. In *International Conference on Learning Representations*, 2020.

J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

S. Fujimoto and S. S. Gu. A minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 34, 2021.

S. Fujimoto, D. Meger, and D. Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*, pages 2052–2062. PMLR, 2019.

D. Garg, J. Hejna, M. Geist, and S. Ermon. Extreme q-learning: Maxent RL without entropy. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=SJ0Lde3tRL.

T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning*, pages 1861–1870, 2018.

P. Hansen-Estruch, I. Kostrikov, M. Janner, J. G. Kuba, and S. Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.

T. Ji, Y. Luo, F. Sun, X. Zhan, J. Zhang, and H. Xu. Seizing serendipity: Exploiting the value of past success in off-policy actor-critic. *arXiv preprint arXiv:2306.02865*, 2023.

J. Kober, J. A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

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

A. Kumar, J. Fu, M. Soh, G. Tucker, and S. Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, pages 11761–11771, 2019.

A. Kumar, A. Zhou, G. Tucker, and S. Levine. Conservative q-learning for offline reinforcement learning. In *Neural Information Processing Systems (NeurIPS)*, 2020.

J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning quadrupedal locomotion over challenging terrain. *Science robotics*, 5(47):eabc5986, 2020.

S. Lee, Y. Seo, K. Lee, P. Abbeel, and J. Shin. Offline-to-online reinforcement learning via balanced replay and pessimistic q-ensemble. In *Conference on Robot Learning*, pages 1702–1712. PMLR, 2022.

S. Levine, A. Kumar, G. Tucker, and J. Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

J. Li, X. Zhan, H. Xu, X. Zhu, J. Liu, and Y.-Q. Zhang. When data geometry meets deep function: Generalizing offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023.

J. Liu, Z. Hongyin, and D. Wang. Dara: Dynamics-aware reward augmentation in offline reinforcement learning. In *International Conference on Learning Representations*, 2022.

L. Ljung. System identification. In *Signal analysis and prediction*, pages 163–173. Springer, 1998.

F.-M. Luo, S. Jiang, Y. Yu, Z. Zhang, and Y.-F. Zhang. Adapt to environment sudden changes by learning a context sensitive policy. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 7637–7646, 2022.

V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021.

B. Mehta, M. Diaz, F. Golemo, C. J. Pal, and L. Paull. Active domain randomization. In *Conference on Robot Learning*, pages 1162–1176. PMLR, 2020.

V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

F. Muratore, C. Eilers, M. Gienger, and J. Peters. Data-efficient domain randomization with bayesian optimization. *IEEE Robotics and Automation Letters*, 6(2):911–918, 2021.

A. Nair, A. Gupta, M. Dalal, and S. Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

M. Nakamoto, Y. Zhai, A. Singh, Y. Ma, C. Finn, A. Kumar, and S. Levine. Cal-QL: Calibrated offline RL pre-training for efficient online fine-tuning. In *Workshop on Reincarnating Reinforcement Learning at ICLR 2023*, 2023. URL https://openreview.net/forum?id=PhCWNmatOX.

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

M. O'Connell, G. Shi, X. Shi, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung. Neural-fly enables rapid learning for agile flight in strong winds. *Science Robotics*, 7(66): eabm6597, 2022.

X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3803–3810. IEEE, 2018.

A. Rajeswaran, S. Ghotra, B. Ravindran, and S. Levine. Epopt: Learning robust neural network policies using model ensembles. *arXiv preprint arXiv:1610.01283*, 2016.

F. Ramos, R. Possas, and D. Fox. Bayessim: adaptive domain randomization via probabilistic inference for robotics simulators. In *Robotics: Science and Systems (RSS)*, 2019.

S. S. Sandha, L. Garcia, B. Balaji, F. Anwar, and M. Srivastava. Sim2real transfer for deep reinforcement learning with stochastic state transition delays. In *Conference on Robot Learning*, pages 1066–1083. PMLR, 2021.

D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, et al. Mastering the game of go without human knowledge. *nature*, 550(7676):354–359, 2017.

Y. Song, Y. Zhou, A. Sekhari, D. Bagnell, A. Krishnamurthy, and W. Sun. Hybrid RL: Using both offline and online data can make RL efficient. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=yyBis80iUuU.

R. S. Sutton, A. G. Barto, et al. Introduction to reinforcement learning. 1998.

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

A. Wagenmaker and A. Pacchiano. Leveraging offline data in online reinforcement learning. *arXiv preprint arXiv:2211.04974*, 2022.

H. Xu, J. Li, J. Li, and X. Zhan. A policy-guided imitation approach for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2022.

H. Xu, L. Jiang, J. Li, Z. Yang, Z. Wang, V. W. K. Chan, and X. Zhan. Offline rl with no ood actions: In-sample learning via implicit value regularization. In *The Eleventh International Conference on Learning Representations*, 2023.

W. Yu, J. Tan, C. K. Liu, and G. Turk. Preparing for the unknown: Learning a universal policy with online system identification. *arXiv preprint arXiv:1702.02453*, 2017.

H. Zhang, W. Xu, and H. Yu. Policy expansion for bridging offline-to-online reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=-Y34L45JR6z.

Appendix A. Discussion and Comparison with H2O

Using the above dynamics-aware mixed value update, **H2O+** effectively addresses all the aforementioned drawbacks of H2O. First, **H2O+** uses in-sample learning state-value function $V(s)$ and the dynamics ratio to mildly regulate the value function learning on potentially problematic online simulated samples. There is no distortion nor extra conservative penalty on the Q-values, thus removing excessive conservatism during policy learning. Second, **H2O+** is compatible with a series of recent strong in-sample learning offline RL methods (Kostrikov et al., 2022; Xu et al., 2023; Hansen-Estruch et al., 2023; Garg et al., 2023), and the policy entropy in Eq.(7) is also possible to be replaced with other terms to promote exploration, thus offering great flexibility. Moreover, **H2O+** removes the need to estimate explicit dynamics gap measures, thus providing a simpler and more efficient algorithmic implementation. In the next section, we will show in empirical experiments, that although it removes much conservatism to regulate off-dynamics samples, **H2O+** consistently outperforms H2O and other cross-domain RL baselines.

Appendix B. Additional Related Work on Combining Offline/Online RL

The recently emerged offline RL methods (Fujimoto et al., 2019; Kumar et al., 2019; Fujimoto and Gu, 2021; Kostrikov et al., 2022; Xu et al., 2022, 2023; Garg et al., 2023) has provided an attractive solution to learn policies directly from offline data without online interactions. However, the performances of existing offline RL methods are heavily limited by the quality and state-action space coverage of offline datasets (Kumar et al., 2019; Li et al., 2023). To mitigate this issue, offline-to-online RL methods (Nair et al., 2020; Lee et al., 2022; Zhang et al., 2023) are developed to separate RL policy learning into a two-stage training process: first pretrain a policy using offline RL and then finetune with online RL. It can improve sample efficiency with favorable initialization for the online learning stage. More recently, some RL studies (Song et al., 2023; Ball et al., 2023; Wagenmaker and Pacchiano, 2022; Ji et al., 2023) directly merge offline RL ingredients into online RL algorithms as a single-stage learning process, which have been shown to greatly improve sample efficiency and policy performance. However, all these methods are only applicable to a single domain, with no dynamics gaps between the online and offline data. H2O (Niu et al., 2022) also adopts simultaneous offline and online learning, but is specifically designed to tackle off-dynamics online samples from an imperfect simulator. Our proposed **H2O+** follows the same hybrid offline-and-online RL setting, but uses a different methodological framework to address several key drawbacks of H2O.

Appendix C. Ablation Studies

Ablation on the hyperparameter λ and dynamics ratio Table 2 presents the results of different choices of the trade-off parameter λ and the dynamics ratio on Bellman error in dynamics-aware mixed value update, as used in Eq. 10. Specifically, we investigate the cases where $\lambda = 0$ corresponds to Q-value update on both simulation and real data using Eq. 6, and $\lambda = 1$ formulates Q-value update with Eq. 5. Among the different parameter choices, the original implementation of **H2O+** with a mixed Q-value update at $\lambda = 0.1$ achieves

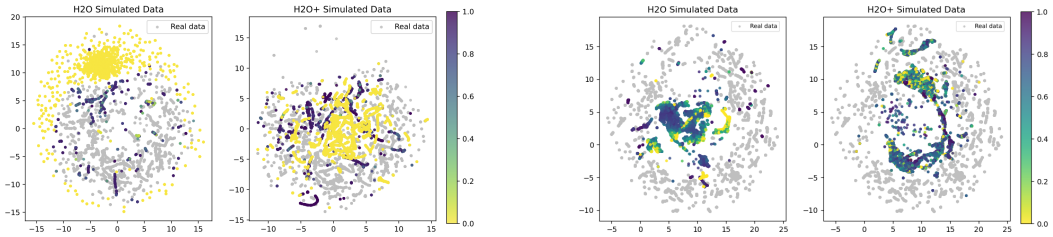


Figure 3: Comparison of H2O / **H2O+** simulation data quality in real-world tasks. (Top: standing still; Down: moving forward)

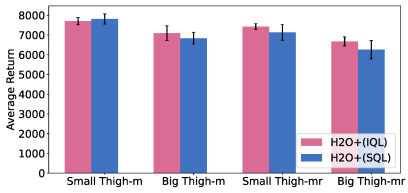


Figure 4: Different choices of offline RL backbone for state-value function learning

the highest performance compared to other selections. It reveals that there is no necessity to heavily regulate the Q target by incorporating too much information from the value function learned as an anchor from offline data. Moreover, it is evident that **H2O+** exhibits a remarkable level of hyper-parameter insensitivity, as indicated by its minor performance discrepancies across the λ range of 0.0 to 0.5. Essentially, our analysis also reveals that the absence of dynamics ratio reweighting in the Bellman error results in significant performance degradation.

Ablation on offline RL backbones Furthermore, we plug in other offline RL backbones like SQL (Xu et al., 2023) into the **H2O+** paradigm for state-value learning, by replacing the value loss in Eq.(2) with $\mathcal{L}_V^f(y) = \mathbb{1}(1 + y/2\alpha > 0)(1 + y/2\alpha)^2 - y/\alpha$. We demonstrate that **H2O+** offers flexibility to bridge other offline RL algorithms in Small and Big Thigh tasks on HalfCheetah Medium and Medium Replay datasets, producing comparable performance as shown in Figure 4.

Investigations on different levels of dynamics gaps We further compare H2O and **H2O+** under different levels of dynamics gaps (HalfCheetah-mr with 1.25 to 3 times gravity). The results are presented in Table 3. It is observed that **H2O+** beats H2O in all different dynamics discrepancy levels, despite using a simpler approach to handle dynamics gaps. In addition, **H2O+** consistently demonstrates a lower variance across all tasks, underscoring its heightened stability compared to H2O. Furthermore, **H2O+** performs much better in tasks with low dynamics gaps and still maintains competitive performance in high dynamics scenarios. This alignment with the underlying design philosophy of leveraging the full potential of online learning with less conservatism further accentuates the superiority of **H2O+**.

Table 2: Ablations on choices in dynamics-aware mixed value update designings (λ and dynamics ratio)

Trade-off λ	0.0	0.1	0.2	0.5	1.0	0.1
Dynamics ratio	✓	✓	✓	✓	✓	✗
Average return	6738±444	**6861±268**	6677±252	6563±752	6242±68	5579±530

Table 3: Ablations on different levels of dynamics gap

Gravity	@1.25	@1.5	@2.0	@3.0
H2O	6846±572	6483±529	6813±289	6171±1209
H2O+	**7165±134**	**6948±258**	**6861±268**	6135±811

Appendix D. Experimental Details

Simulation experiments setups. We create nine simulation environments upon the MuJoCo physics simulator with deliberately introduced dynamics gaps. These imperfect simulators are derived from the original MuJoCo-HalfCheetah task environments, which act as our real-world scenarios. These alterations are achieved by adjusting either the dynamics parameters of the robot or the environmental physical properties, as detailed below: (1) **Gravity**: we apply 2 times the gravitational acceleration in the simulation dynamics; (2) **Friction**: we use 0.3 times the friction coefficient to make the agent harder to maintain balance; (3) **Joint Noise**: we use a random noise, drawn from a standard normal distribution $\mathcal{N}(0, 1)$, to disturb each action dimension, which mimics systems with control noise; (4) **Thigh**: we construct four variants corresponding to both the back and front thighs, including "Big Thigh" (doubling the thigh size), "Small Thigh" (halving the thigh size), "Broken Thigh" (halving the motion range of the joint connections of thighs) and "Flexible Thigh" (doubling the motion range of the joint connections of thighs); (5) **Torso**: we construct a half-cheetah with a longer torso with four times length; (6) **Feet**: we reduce the stiffness of both feet to 0 to make them soft-body feet.

In terms of the real-world offline dataset (original HalfCheetah environment), we utilize the corresponding task datasets from the widely-used offline RL benchmark D4RL (Fu et al., 2020). Specifically, the Medium and Medium Replay datasets are considered since they reflect typical real-world settings where it is considerably impractical to acquire datasets of both high quality and broad coverage. Online training with offline data takes place in our created simulation environments, while the evaluation of the learned policy performance is conducted in the original untouched HalfCheetah environment. We visualize some of our constructed simulators and the corresponding real environment in Figure 1.

Real-robot transfer experiment setups. To estimate the performance of H2O+ in real-life application scenarios, we use a real wheel-legged robot as a validation. During the control process, the pitch angle θ and angular velocity $\dot{\theta}$ are measured by the on-board IMU and Gyroscope, while the linear velocity v and displacement x are measured by the wheel motor encoders. For the actuator, we lock the four joints on its legs and simply control

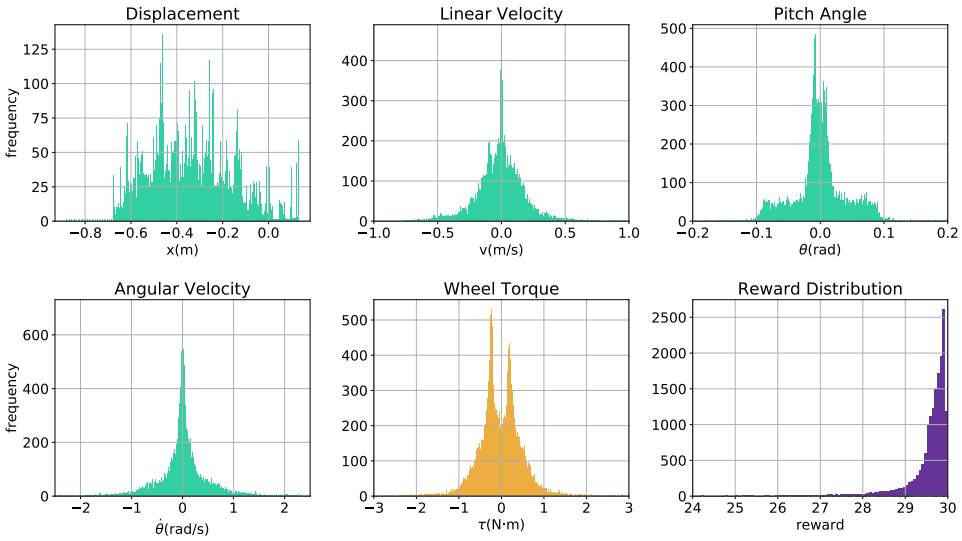


Figure 5: State, action and reward distribution of the standing still dataset

the robot by regulating the wheels' torque, which is limited to the range of $(-3, 3)$. The real-world control frequency is 200Hz. In each process of deployment, we place the robot to the origin and initialize it to an equilibrium position as shown in Figure 2a. In the following, we further describe the task details.

Standing still: In this task, we want the robot to keep the balance at all times and to maintain a stable state, which means that the displacement, linear velocity, pitch angle and angular velocity of the robot should be close to zero. Specifically, the state space of the robot is represented by $\mathbf{s} = (\theta, \dot{\theta}, x, v)$, where θ denotes the forward pitch angle of the body, x is the displacement of the robot, $\dot{\theta}$ is the angular velocity and v is the linear velocity. The reward r is calculated by the following formulation:

$$r = 30.0 - x^2 - v^2 - \theta^2 - \dot{\theta}^2 - \tau^2 \tag{13}$$

Ideally, when the robot standing at the original space without any swinging, the penalty item $x^2 + v^2 + \theta^2 + \dot{\theta}^2$ will be minimized. To further avoid shaking and to protect the wheel motor, we limit the wheel torque by adding a penalty τ^2 into the reward.

As for the offline dataset of the task, we collect 16588 transitions of data (about 90s of real-time control) based on the real robot and filter out the data with torque greater than $3N \cdot m$. The visualized results of the state, action and reward distribution of the dataset are illustrated in Figure 5.

Moving forward: In this task, our target is to control the robot to move forward at a constant speed. While it is moving, we want its speed to be close to 0.2m/s and maintain stability. The state space of the task $\mathbf{s} = (v, \theta, \dot{\theta})$, where the state x is no longer included as we hope that the robot could move forward in any displacement. The reward is calculated

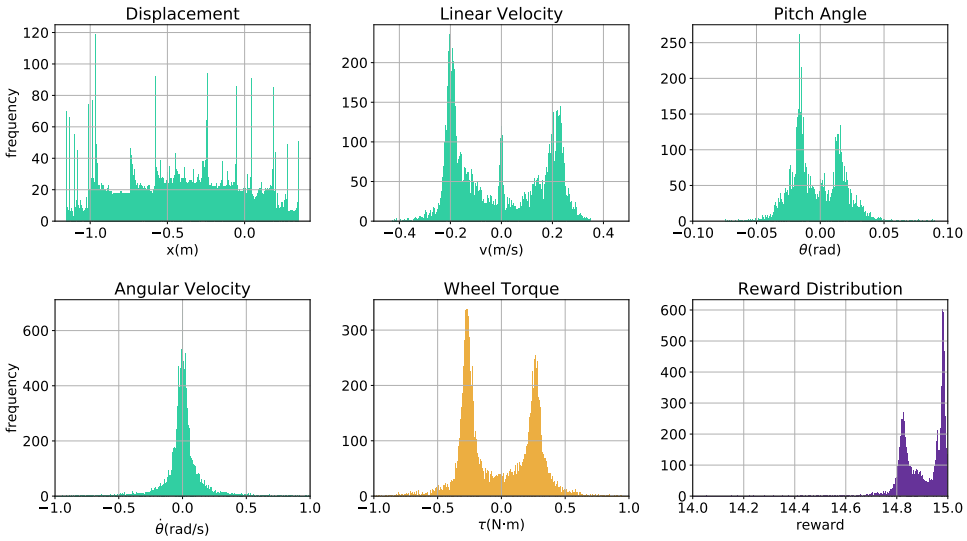


Figure 6: State, action and reward distribution of the moving forward dataset

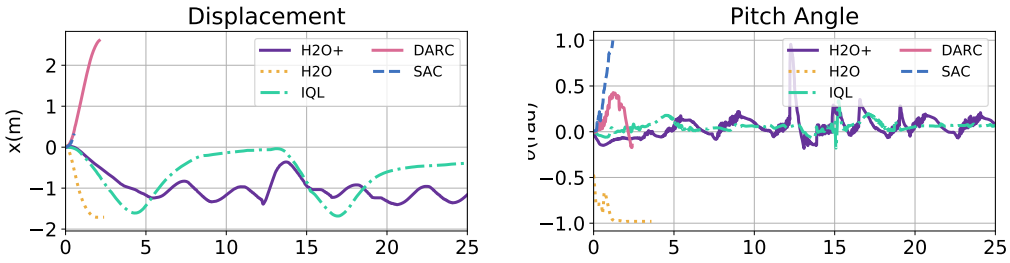
according to the following formula:

$$r = 15.0 - (v - 0.2)^2 - \tau^2 \tag{14}$$

in which we use the penalty item $(v - 0.2)^2$ to regulate the moving speed of the robot while adding 15 to encourage it to keep balance and $-\tau^2$ to avoid shaking. For the offline dataset of moving forward, we collect 16588 transitions of data from the moving process of the real robot and also exclude the data with torque over $3N \cdot m$. The state, action and reward distribution of the dataset is shown in Figure 6.

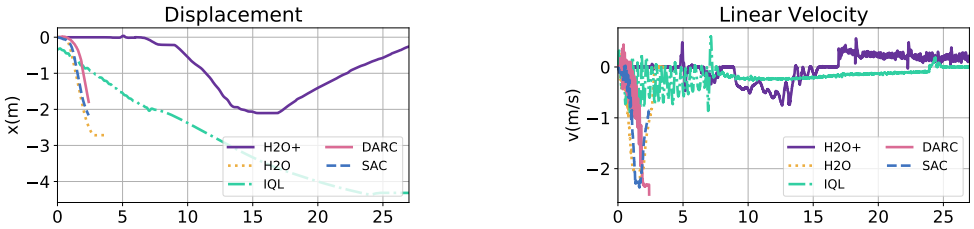
Through the training process, we run all the algorithms for 100 epochs based on the same environment and offline dataset and also make sure all the policies are derived from convergent models.

Baselines. There are few works under the hybrid offline-and-online setting, thus our comparative analysis includes a mix of purely online or offline RL algorithms, along with some representative methods that incorporate offline data into online or offline policy learning, albeit in a less integrated manner. These methods include: (1) *SAC*: We train the SAC (Haarnoja et al., 2018) agent in the imperfect simulator and evaluate its zero-shot transfer performance in real environments. (2) *CQL* (Kumar et al., 2020): an representative offline RL algorithm that uses value regularization. (3) *IQL*: the SOTA offline RL algorithm, which uses in-sample learning to tackle the OOD problem. We train CQL and IQL on the D4RL datasets or the datasets collected during actual robot control, which remain unaffected by dynamics gaps, yet are hindered by the limited state-action coverage or poor performance of the offline data. (4) *DARC* (Eysenbach et al., 2020): adds a correction term on the original reward to compensate for dynamics discrepancy in simulation trajectories. To allow fair comparison with H2O+,



(a) Displacement (b) Pitch angle

Figure 7: Additional real-robot experiment results on the "standing still" task.



(a) Displacement (b) Linear velocity

Figure 8: Additional real-robot experiment results on the "moving forward" task.

we re-implement it in our setting that restricts any further interactions in reality beyond the offline data. (5) *H2O* (Niu et al., 2022): the first RL paradigm to address dynamics mismatch under the hybrid offline-and-online setting, which might introduce over-conservatism and restrict the use of simulation in principle.

Appendix E. Additional Experiment Results

E.1 Additional Real-World Experiment

To further corroborate the efficacy of H2O+, we intentionally amplify the dynamics gap within this task. Specifically, we undertake modifications by adjusting both the mass of the robot model and the mass distribution within the simulation scenario. The initial robot model bears a total weight of 12.1 kilograms (kg), which closely approximates the actual robot mass of 12.0kg. Conversely, the adapted new simulation model we experiment with features a total weight of 4.6kg. This revised simulation model stands at approximately one-third of the magnitude of the original one, thereby intensifying the challenges associated with controlling the robot within a setting characterized by significantly flawed simulation dynamics.

Building on this foundation, we adopt an identical setup as we describe in Appendix D and train the control policies for standing still and moving forward tasks using SAC, DARC, H2O and H2O+ and deploy the control models to the real robot to compare the control performance.

Standing Still: As shown in Figure 7, only IQL and H2O+ successfully avoid the robot from falling to the ground and maintain steady, while SAC, DARC and H2O fail at the beginning. Furthermore, it is noteworthy that H2O+ exhibits a distinct advantage over IQL in terms of the robot's capacity to maintain a stationary stance. H2O+ effectively confines the robot's displacement within approximately -1 meter, whereas IQL leads the robot to oscillate between its original position and a broader range of approximately -1.5 meters.

Moving Forward: In this task, H2O+ still exhibits the best performance among all the methods. As shown in Figure 8, none of the methods except H2O+ are able to control the robot to move forward, and only IQL maintains equilibrium for a long period of time. H2O+, despite its moving backward at first, moves forward at a steady speed close to 0.2 m/s for a long period of time, being the only one of all the methods that achieves the goal of moving forward.

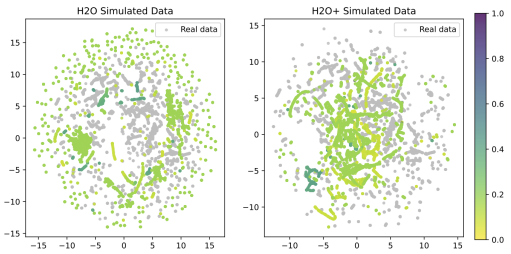
Overall, our results show that H2O+ consistently emerges as the most effective approach among the various baselines, reaffirming its superior performance across a range of challenging real-world contexts.

E.2 Further Investigation on Data Quality

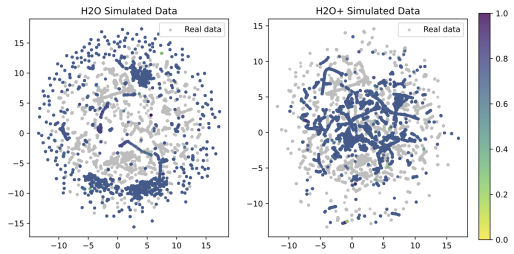
We further investigate on the quality of online interactions explored by H2O+ compared to H2O, from the data coverage and the value. In Figure 9 and Figure 10, we visualize the coverage and the normalized value of displacement, velocity, angle, angular velocity, and action in the real-world robot task, "standing still" and "moving forward" respectively. Specifically, we also visualize the real data (collected offline in the real environment) with gray points in these two figures, and the colorful points reveal the data collected in the simulator.

Comparison of data quality on "standing still" task. In the "standing still" task, we observe that H2O explores a more focused high-value area, whereas H2O+ spans a broader high-value area, thus demonstrating superior diversity characteristics in simulated data, which would benefit the overall performance.

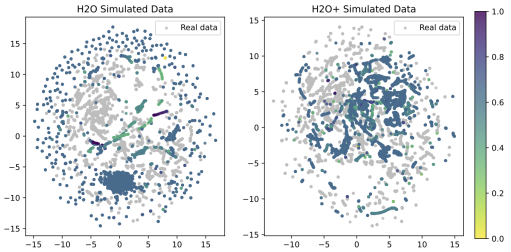
Comparison of data quality on "moving forward" task. For the task of "moving forward", a clear distinction can be observed between the quality of the simulated data gathered by H2O and H2O+. Notably, the H2O+ approach excels in terms of data coverage, meaning it successfully spans a broader region of the state-action space. Additionally, the data collected by H2O+ exhibits better dispersion compared to H2O's data, indicating a higher degree of diversity. This superior diversity, in turn, contributes to a more comprehensive exploration of the state-action space and enhances the robustness of the learned policy.



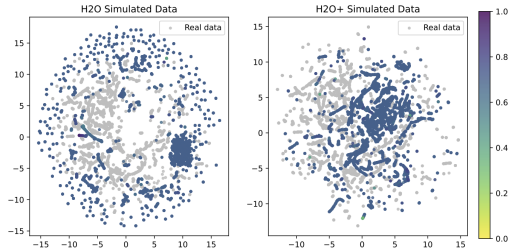
(a) displacement



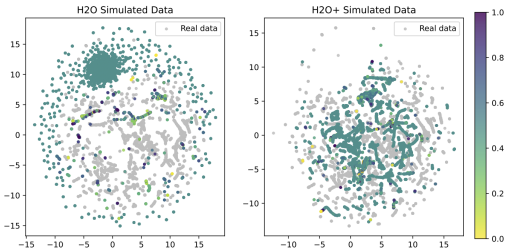
(b) velocity



(c) angle

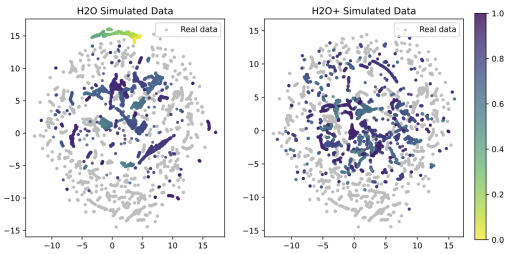


(d) angular velocity

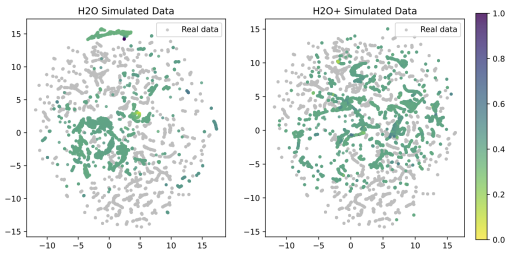


(e) action

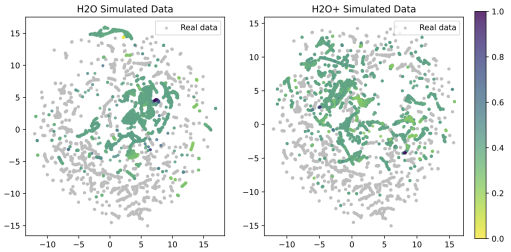
Figure 9: Comparison of H2O+ and H2O simulated data quality on the real-world robot "standing still" task. We visualize the coverage and the normalized value of displacement, velocity, angle, angular velocity, and action.



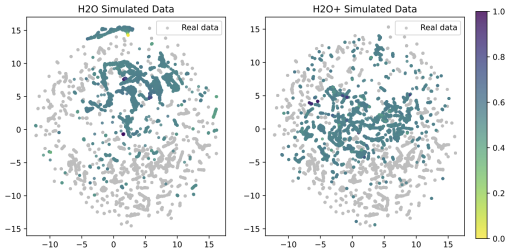
(a) displacement



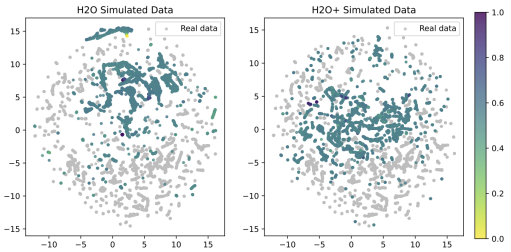
(b) velocity



(c) angle



(d) angular velocity



(e) action

Figure 10: Comparison of H2O+ and H2O simulated data quality on the real-world robot "moving forward" task. We visualize the coverage and the normalized value of displacement, velocity, angle, angular velocity, and action.