Towards Optimism-Pessimism Trade-off in Model-based Offline-to-Online Reinforcement Learning

Guochen Zhou[1,†] **Yijun Yang**[2,†] **Qiqi Duan**[3] **Qing Su**[4] **Weiming Ou**[5]
Li Shen[6] **Shihao Ji**[4] **Yuhui Shi**[7,*] **Zhisong Zhang**[1,*]

[1]Department of Computer Science, City University of Hong Kong [2]Tsinghua University
[3]Jiangxi University of Finance and Economics [4]University of Connecticut
[5]Shanghai University of Finance and Economics [6]Sun Yat-sen University
[7]Southern University of Science and Technology

Abstract

Model-based offline-to-online reinforcement learning (RL) enables sample-efficient adaptation by leveraging offline pre-training for online fine-tuning. However, the distribution shifts between offline and online stages often hinder fine-tuning performance. Many existing methods encode the optimism-pessimism trade-off through a single-objective formulation. Selecting its trade-off coefficient through online fine-tuning can require costly bi-level optimization. We identify the balance established during offline training as a key challenge: in our motivating experiments, optimistic policies adapt better to novel tasks, whereas pessimistic policies perform better on the same online tasks. To address this challenge, we propose a bi-objective formulation that captures this trade-off, yielding a Pareto policy pool during offline training. This policy pool enables adaptive selection during online fine-tuning across diverse online tasks. To generate the pool, we introduce Multi-Objective Soft Actor-critIC (MOSAIC), which solves bi-objective problems and constructs diverse Pareto policies. After offline training, a contextual bandit algorithm hierarchically selects the most suitable policy for fine-tuning at each online interaction step. Empirically, our pipeline, **Hi**erarchical **P**areto **P**olicy **P**ool (**HiP3**), achieves the strongest aggregate performance across both same and novel online tasks and maintains its advantage under stronger distribution shifts. Comprehensive ablations further clarify the roles of the policy pool, MOSAIC-based pool construction, and the online selection mechanism.

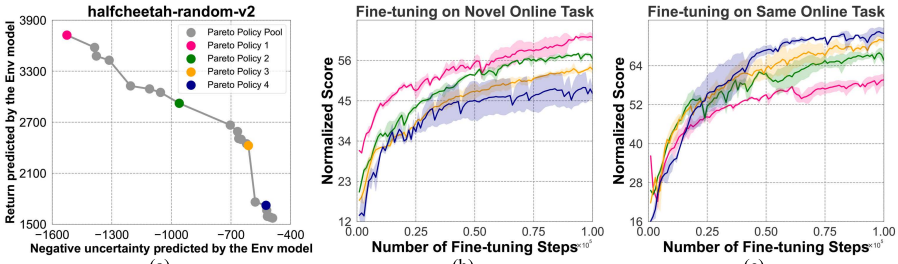
1 Introduction

A major obstacle to deploying reinforcement learning (RL) in real-world settings is its high sample complexity. Offline RL mitigates this issue by learning from pre-collected data [1]. However, distribution shifts between offline data and online interaction can hinder the deployment of offline-trained policies. This motivates offline-to-online RL (O2O RL), which adapts pretrained policies using limited online interaction [2, 3]. Model-based O2O RL further improves sample efficiency by learning environment models for offline pre-training and online fine-tuning [4, 5].

Despite this progress, distribution shifts continue to limit fine-tuning performance in model-based O2O RL. We argue that a key factor is the optimism-pessimism trade-off established during offline training. Following prior works [6–8], we use model-predicted return and uncertainty as proxies for optimism and pessimism, respectively. Figure 1(a) illustrates offline policies with different trade-offs

*Corresponding authors.
†Co-first authors. Contact email: guochzhou2-c@my.cityu.edu.hk.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).



(a) (b) (c)

Figure 1: (a) Pareto policy pool discovered by HiP3, containing policies with different optimism-pessimism trade-offs. P1 maximizes model-predicted return (most optimistic), P4 minimizes uncertainty (most pessimistic), while P2 and P3 represent balanced trade-offs. (b) On the novel online task (details in Appendix C.1), optimistic P1 achieves the highest return. (c) On the same online task, pessimistic P4 achieves the highest return. Together, these results motivate preserving multiple trade-off policies offline and deferring the choice among them to online adaptation, rather than committing to a single trade-off in advance.

between these objectives. On the novel online task with controlled dynamics and reward shifts, the optimistic policy P1 (higher model return) performs best in Figure 1(b). Conversely, the pessimistic policy P4 (lower model uncertainty) performs best on the same online task in Figure 1(c). This reversal is consistent with an optimistic bias that may support exploration beyond the offline data distribution and a pessimistic bias toward high-confidence regions. Policies P2 and P3 represent intermediate trade-offs. Appendix D.1 further reproduces this reversal across four controlled settings spanning different tasks and offline-data qualities. These observations suggest that the preferred optimism-pessimism balance depends on the task encountered during online fine-tuning.

Existing model-based approaches commonly encode this trade-off through a single scalarized objective, such as $\hat{r} - \lambda u$, thereby committing each offline policy to a particular balance. Adapting this balance to an online task by selecting λ according to fine-tuning performance requires an outer search over coefficients around the inner policy-optimization procedure, making task-specific tuning costly. Training multiple policies with different fixed coefficients is a natural alternative, but linear scalarization does not in general recover all Pareto-optimal trade-offs when the attainable objective set is non-convex. Appendix B.5 provides a detailed discussion and empirical comparison.

In this paper, we treat optimism and pessimism as two distinct objectives and introduce Hierarchical Pareto Policy Pool (HiP3), which constructs a diverse offline policy pool for adaptive online selection. During online adaptation, the agent encounters changing states and updates its candidate policies, making policy selection an ongoing decision. This design raises two algorithmic challenges. First, how can we generate policies that cover diverse optimism-pessimism trade-offs across the Pareto front? Second, how can we dynamically select a suitable policy for the current state during online fine-tuning, rather than committing to a single candidate throughout adaptation? To address the first challenge, we introduce Multi-Objective Soft Actor-critIC (MOSAIC), which solves constrained bi-objective policy optimization problems defined by reference vectors targeting different regions of the Pareto front. It then uses warm-started neighborhood search to discover additional trade-off policies around the obtained ones. To address the second challenge, HiP3 uses a contextual bandit as a high-level controller over a candidate subset of the offline policy pool. At each online interaction step, the controller selects a policy conditioned on the current state, and the selected policy is fine-tuned with model-based updates.

We evaluate HiP3 on the standard D4RL continuous-control benchmark [9] against representative online and O2O RL baselines across both same and novel online tasks. HiP3 achieves the strongest average performance in both settings, with particularly clear advantages on novel tasks, and remains effective under stronger distribution shifts. Ablation studies examine the roles of the policy pool, MOSAIC-based pool construction, state-dependent policy selection, and candidate-set initialization. Additional analyses characterize policy-selection behavior during online adaptation. Our code is available at `https://github.com/GuochenZhou/HiP3`

2 Related Work

Offline RL and Model-based Variants. Offline RL trains policies on static datasets to avoid the high sample complexity of online methods [1]. A major challenge is poor extrapolation to out-of-distribution (OOD) actions, often causing overestimation. Most approaches rely on pessimism

to counter this, via policy constraints [10, 11], regularization [12], or uncertainty penalties [13, 14]. Model-based methods improve efficiency with learned dynamics for planning or simulated rollouts [15–18]. Notably, works like MOPO [6] and MOReL [7] incorporate uncertainty to penalize OOD regions (e.g., subtracting it from rewards). Critically, these fix the optimism-pessimism trade-off via a single penalty, limiting flexibility for online fine-tuning.

Offline-to-Online (O2O) RL. Distribution shifts hinder direct online deployment of offline policies [19], necessitating fine-tuning. However, fine-tuning often underperforms [20] due to (1) training instability from the shift [21], and (2) restricted exploration inherited from offline pessimism. Instability is mitigated via balanced replay [21], priority sampling [22], policy expansion [3], state-adaptive policy families [23], and more recently warm-start recalibration without retaining offline data [24]. Exploration or policy improvement is enhanced via critic guidance [25], guide policies [26], perturbed value updates and more frequent critic updates [27], optimistic critic reconstruction with constrained fine-tuning [28], energy-guided diffusion sampling that distills offline knowledge for online data generation [29], and expressive flow-based policies [30]. Beyond O2O-specific methods, scalable world-model agents such as TD-MPC2 [31] provide strong sample-efficient model-based baselines, but do not explicitly address the optimism-pessimism trade-off induced by offline pretraining. Most of these methods optimize a single policy or otherwise commit to a fixed adaptation strategy, rather than preserving multiple optimism-pessimism trade-offs for later selection. HiP3 instead treats the trade-off itself as an adaptation decision: it preserves diverse trade-off policies offline and performs state-dependent policy selection among them during online adaptation.

3 Preliminaries

Standard RL formulates agent-environment interaction as a Markov Decision Process (MDP) $(S, A, p, r, \rho_0, \gamma)$ [32], where the goal is to find a policy π_θ maximizing expected discounted return:

$$J(\pi_\theta) = \mathbb{E}_{\rho_0, \pi_\theta}\left[\sum_{t=0}^{+\infty} \gamma^t r(s_t, a_t)\right]. \tag{1}$$

Specifically, in model-based O2O RL, the agent first learns an environment model and a policy from pre-collected data, then fine-tunes them using limited online interaction [4, 5]. The environment model, typically parameterized by a feed-forward neural network, contains a learned transition probability model $\hat{p}(s'|s,a)$ and a reward model $\hat{r}(s,a)$. Given the current state s and action a selected by policy $\pi_\theta(s)$, it predicts the next state $s' \sim \hat{p}(s'|s,a)$ and reward value $\hat{r}$.

These model predictions naturally expose the optimism-pessimism trade-off studied in this work. Following prior works [6–8], we use the model-predicted reward $\hat{r}(s,a)$ and uncertainty estimate $u(s,a)$ as proxies for the optimism and pessimism objectives, respectively [1]. A common formulation combines the two terms through linear scalarization, e.g., $\overline{r}(s,a) = \hat{r}(s,a) - \lambda u(s,a)$, where λ specifies a global optimism-pessimism balance. For a new online task, selecting an appropriate λ through target-task rollouts and online fine-tuning requires an outer search over λ around the inner policy-optimization procedure, leading to costly bi-level tuning.

Rather than selecting a single trade-off coefficient online, we formulate optimism and pessimism as two distinct objectives and solve the following bi-objective optimization problem. Appendix B.5 provides a detailed comparison with linear scalarization.

$$\max_\theta J(\pi_\theta) = \max_\theta \big(J^{\hat{r}}(\pi_\theta),\ J^{\hat{u}}(\pi_\theta)\big) \tag{2}$$

where $\hat{u}(s_t, a_t) \triangleq \exp(-u(s_t, a_t)/\kappa)$ maps the raw uncertainty to a bounded score in $(0, 1]$, so that larger uncertainty yields a smaller objective value. We use $\kappa = 1.5$ by default, except when varying it in the sensitivity analysis. This bi-objective formulation replaces a single scalarized balance with explicit optimism and pessimism objectives, enabling offline construction of policies spanning different trade-off regions for subsequent state-dependent policy selection. Appendix B.4 provides a brief introduction to multi-objective optimization.

HiP3 instantiates this formulation with MOSAIC for offline policy-pool construction and a contextual bandit for state-dependent policy selection during online adaptation.

[1] Further information on the uncertainty estimation u is provided in Appendix B.2, and a description of the environment model architecture is given in Appendix B.3.

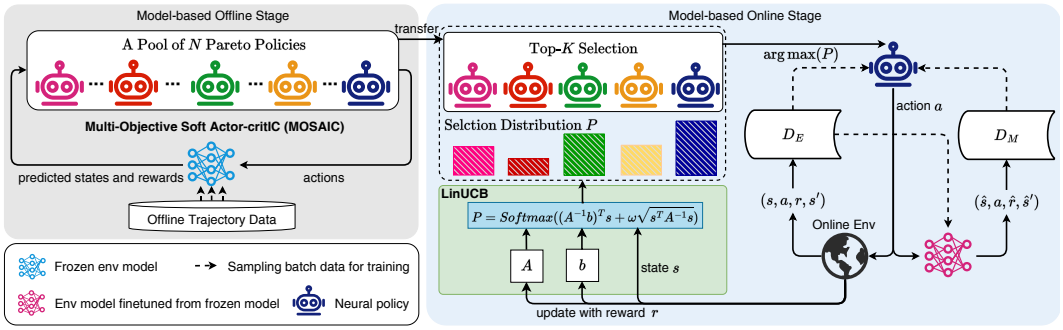


Figure 2: Overview of Hierarchical Pareto Policy Pool (HiP3). In the offline stage (gray box), our MOSAIC algorithm generates a diverse pool of Pareto policies exhibiting varying optimism-pessimism trade-offs via multi-objective optimization with reference vectors and neighborhood search. In the online stage (blue box), a contextual bandit (LinUCB [33]) selects a candidate policy conditioned on the current state from the top-K subset; the selected policy is then fine-tuned via a Dyna-style method [15].

4 HiP3: Hierarchical Pareto Policy Pool

Figure 2 provides an overview of HiP3 alongside the procedure summarized in Algorithm 1. Rather than committing offline training to a single optimism-pessimism balance, HiP3 preserves multiple trade-offs offline and performs state-dependent policy selection during online adaptation. In the offline stage, MOSAIC constructs a finite Pareto policy pool spanning different optimism-pessimism trade-offs through reference-vector-guided multi-objective optimization and neighborhood search. In the online stage, HiP3 first uses one-time ranking to select a candidate subset from the policy pool and then uses a contextual bandit to select a candidate policy conditioned on the current state. The selected policy is then fine-tuned online.

Algorithm 1 Hierarchical Pareto Policy Pool (HiP3)

Input: Offline dataset $\mathcal{D}$, #reference vectors n, #offline policies K_{off}, #online policies K_{on}, #offline steps $T_{\text{off}} = n(T_g + 2T_l)T_e$, #online steps T_{on}, model update interval S_M, bandit hyperparameter ω
Initialize: Pareto policy pool $\mathcal{P} = \emptyset$, env. model M, model data buffer $\mathcal{D}_M = \emptyset$, env. data buffer $\mathcal{D}_E = \emptyset$

// Offline learning via MOSAIC
Train M on dataset $\mathcal{D}$ (max. likelihood)
Generate reference vectors $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n\}$ (Eq. 3)
for $i = 1$ to n **do** //with $\mathcal{D}_M$ accumulated
  Initialize a policy π_θ
  Update π_θ for $T_g \times T_e$ steps with $\boldsymbol{v}_i$ (Alg. 2)
  $\boldsymbol{v}_i^+ \leftarrow \boldsymbol{v}_i + \epsilon$; $\boldsymbol{v}_i^- \leftarrow \boldsymbol{v}_i - \epsilon$
  for $\boldsymbol{v}' \in \{\boldsymbol{v}_i^+, \boldsymbol{v}_i^-\}$ **do**
    for $j = 1$ to T_l **do**
      Update π_θ for T_e steps with $\boldsymbol{v}'$ (Alg. 2)
      $\mathcal{P} \leftarrow \mathcal{P} \cup \{\pi_\theta\}$
      $\mathcal{P} \leftarrow$ top-$K_{\text{off}}(\mathcal{P})$ //non-dom. sort

// State-dependent policy selection via LinUCB
$\mathcal{D}_M = \emptyset$
$\mathcal{P}_{\text{on}} \leftarrow$ top-K_{on} policies by rollout ranking
Construct contextual bandit $B(A, b)$ with $\mathcal{P}_{on}$
for $t = 1$ to T_{on} **do**
  if $t \pmod{S_M} == 0$ **then**
    Train M on $\mathcal{D}_E$ (max. likelihood)
    $\mathcal{D}_M \leftarrow \mathcal{D}_M \cup \{\text{rollout}(M, B, \mathcal{P}_{on})\}$
  $P_t \leftarrow \text{Softmax}\left((A^{-1}b)^\top s_t + \omega\sqrt{s_t^\top A^{-1} s_t}\right)$
  $k_t \leftarrow \arg\max_k P_t(k)$
  Execute $\pi_\theta^{k_t} \in \mathcal{P}_{on}$ in env. E, observe r, τ
  $A_{k_t} \leftarrow A_{k_t} + \eta_B s_t s_t^T$; $b_{k_t} \leftarrow b_{k_t} + \eta_B s_t r_t$
  $\mathcal{D}_E \leftarrow \mathcal{D}_E \cup \{\tau\}$
  Update $\pi_\theta^{k_t}$ via SAC on $\mathcal{D}_E \cup \mathcal{D}_M$

4.1 MOSAIC: Generation of Pareto Policy Pool

The bi-objective policy optimization problem in Eq. 2 may admit many Pareto-optimal policies, making it infeasible in practice to enumerate the entire Pareto set. Reference vectors provide an explicit way to target different regions of the bi-objective space without prescribing a single global scalarization. Following prior works [34–36], we instantiate these regions using n evenly spaced reference vectors $\{\boldsymbol{v}_i\}_{i=1}^n$, each specifying a target trade-off between the two objectives. Specifically, we construct the reference vectors as follows:

$$\boldsymbol{v}_i = (v_i^1, v_i^2) = (\tau_b - (i-1)\tau_c,\ \tau_a + (i-1)\tau_c), \quad \text{with } \tau_c = \frac{\tau_b - \tau_a}{n-1},\ \ 0 < \tau_a < \tau_b < 1,\ \ i = 1, \cdots, n \tag{3}$$

where τ_a and τ_b control the range covered by the reference vectors. Each $\boldsymbol{v}_i = (v_i^1, v_i^2)$ defines a constrained bi-objective optimization problem whose solution lies in a targeted region of the Pareto front:

$$\begin{aligned} &\max_{\theta} \boldsymbol{J}(\pi_\theta) = \max_{\theta} \Big(J^{\hat{r}}(\pi_\theta), J^{\hat{u}}(\pi_\theta) \Big) \\ &\text{s.t. } \Psi(\pi_\theta, \boldsymbol{v}_i) = D_{\cos}\Big(\frac{\boldsymbol{J}(\pi_\theta)}{\|\boldsymbol{J}(\pi_\theta)\|_2}, \frac{\boldsymbol{v}_i}{\|\boldsymbol{v}_i\|_2} \Big) \geq \psi, \end{aligned} \tag{4}$$

where $\Psi(\pi_\theta, \boldsymbol{v}_i)$ defines the cosine similarity between the objective vector $\boldsymbol{J}(\pi_\theta)$ and the reference vector $\boldsymbol{v}_i$, and the hyper-parameter ψ controls the scope of the target region. To solve this bi-objective optimization problem with an inequality constraint, we introduce MOSAIC, a multi-objective extension of SAC, with the pseudocode provided in Algorithm 2. As an off-policy SAC-style algorithm, MOSAIC learns two soft critics associated with $\hat{r}$ and $\hat{u}$. Their Bellman targets follow the entropy-regularized form:

$$\begin{aligned} Q^{\hat{r}}_{\phi^{\hat{r}}}(s_t, a_t) &= \hat{r}(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1},\, a_{t+1} \sim \pi_\theta} \left[Q^{\hat{r}}_{\bar{\phi}^{\hat{r}}}(s_{t+1}, a_{t+1}) - \alpha \log \pi_\theta(a_{t+1} \mid s_{t+1}) \right], \\ Q^{\hat{u}}_{\phi^{\hat{u}}}(s_t, a_t) &= \hat{u}(s_t, a_t) + \gamma \mathbb{E}_{s_{t+1},\, a_{t+1} \sim \pi_\theta} \left[Q^{\hat{u}}_{\bar{\phi}^{\hat{u}}}(s_{t+1}, a_{t+1}) - \alpha \log \pi_\theta(a_{t+1} \mid s_{t+1}) \right]. \end{aligned} \tag{5}$$

During each update iteration, the two soft critics and their target networks are updated by minimizing the corresponding entropy-regularized Bellman residuals on batches sampled from the model buffer $\mathcal{D}_M$, as outlined in Lines 6-9 of Algorithm 2. MOSAIC then checks whether the current policy lies in the target region induced by the reference vector. If not, the *Correction stage* prioritizes the objective that is currently under-emphasized relative to the trade-off prescribed by $\boldsymbol{v}_i$. Once the alignment criterion is met, MOSAIC enters the *Ascent stage* and treats the alignment term as the third objective:

$$\max_{\theta} \boldsymbol{F}(\pi_\theta) \triangleq \max_{\theta} \left(J^{\hat{r}}(\pi_\theta), J^{\hat{u}}(\pi_\theta), \Psi(\pi_\theta, \boldsymbol{v}_i) \right). \tag{6}$$

In Algorithm 2, $L^{Q^{\hat{r}}}$, $L^{Q^{\hat{u}}}$ and L^{α} denote the standard SAC soft-critic losses for $\hat{r}$ and $\hat{u}$, and the temperature loss, respectively. In practice, $J^{\hat{r}}$ and $J^{\hat{u}}$ denote the SAC-style actor surrogates induced by the two critics, and the ascent stage combines $\nabla_\theta J^{\hat{r}}(\pi_\theta)$, $\nabla_\theta J^{\hat{u}}(\pi_\theta)$ and $\nabla_\theta \Psi$ through MGDA. For analysis, we use the equivalent objective-space view, where minimizing the actor-side surrogates corresponds to ascending the paired objectives. We then solve the resulting tri-objective problem with MGDA [37], which finds coefficients β_t for a convex combination of the three objective gradients, yielding a common ascent direction that does not decrease any objective. This is equivalent to the following min-norm problem:

$$\min_{\boldsymbol{\beta}_t} \|\boldsymbol{\beta}_t \nabla_{\theta_t} \boldsymbol{F}(\pi_{\theta_t})\|_2 \quad s.t. \quad \|\boldsymbol{\beta}_t\|_1 = 1, \boldsymbol{\beta}_t \geq 0, \tag{7}$$

which can be effectively tackled via the Frank-Wolfe algorithm [38]. Subsequently, we perform the policy update as specified in Line 17 of Algorithm 2. The resulting ascent stage optimizes the three-objective formulation above and admits a first-order convergence guarantee toward Pareto stationarity in expectation. A complete proof is provided in Appendix B.1, where Theorem B.1 establishes the convergence rate of the ascent stage of Algorithm 2. For the correction stage, which reduces algorithmically to a standard SAC-style single-objective surrogate update, we focus the new analysis on the ascent stage and refer to prior SAC [39] work for the underlying update form.

MOSAIC uses multiple reference vectors to generate Pareto policies with diverse optimism-pessimism trade-offs. However, using too few reference vectors may miss useful trade-offs and lead to suboptimal policy selection in the online stage. To address this problem, MOSAIC expands trade-off coverage through warm-started neighborhood search.

For each reference vector v_i, we perturb it to create augmented reference vectors v_i^+ and v_i^- in Line 6 of Algorithm 1, thereby constructing additional constrained bi-objective optimization problems based on Eq. 4. By solving these bi-objective problems in the neighborhood of the original ones, additional trade-off candidates can be obtained. The resulting candidates are added to the policy pool, followed by non-dominated sorting. This procedure reuses previously optimized policies to populate additional trade-off regions within the allocated offline budget.

4.2 State-Dependent Policy Selection and Online Optimization

The offline policy pool preserves multiple optimism-pessimism trade-offs, but online adaptation still requires deciding how to use this diversity. Committing to a single policy for the entire trajectory

Algorithm 2 Multi-Objective Soft Actor-critIC (MOSAIC)

Input: ref. vector $\boldsymbol{v}$, threshold ψ, #iterations T, env. model M, data buffer $\mathcal{D}_M$ and its update interval S_M

Initialize: $\phi_i^{\hat{r}}$, $\phi_i^{\hat{u}}$, $\overline{\phi_i^{\hat{r}}}$, $\overline{\phi_i^{\hat{u}}}$ for $i \in \{1, 2\}$

1: **for** $t = 1$ to T **do**
2: **if** $t \pmod{S_M} == 0$ **then**
3: Obtain τ_M with k-step rollout on M by π_θ
4: $\mathcal{D}_M \leftarrow \mathcal{D}_M \cup \{\tau_M\}$
5: Sample a batch from $\mathcal{D}_M$
6: $\phi_i^{\hat{r}} \leftarrow \phi_i^{\hat{r}} - \eta \nabla_{\phi_i^{\hat{r}}} L^{Q^{\hat{r}}}(\phi_i^{\hat{r}})$ for $i \in \{1, 2\}$
7: $\phi_i^{\hat{u}} \leftarrow \phi_i^{\hat{u}} - \eta \nabla_{\phi_i^{\hat{u}}} L^{Q^{\hat{u}}}(\phi_i^{\hat{u}})$ for $i \in \{1, 2\}$
8: $\overline{\phi_i^{\hat{r}}} \leftarrow \tau \phi_i^{\hat{r}} + (1-\tau)\overline{\phi_i^{\hat{r}}}$ for $i \in \{1, 2\}$
9: $\overline{\phi_i^{\hat{u}}} \leftarrow \tau \phi_i^{\hat{u}} + (1-\tau)\overline{\phi_i^{\hat{u}}}$ for $i \in \{1, 2\}$
// Correction stage
10: **if** $\Psi(\pi_\theta, \boldsymbol{v}) < \psi$ **then**
11: **if** $\frac{J^{\hat{r}}(\pi_\theta)}{J^{\hat{u}}(\pi_\theta)} < \frac{\boldsymbol{v}^1}{\boldsymbol{v}^2}$ **then**
12: $\pi_\theta \leftarrow \pi_\theta + \eta \nabla_\theta J^{\hat{r}}(\pi_\theta)$
13: **else**
14: $\pi_\theta \leftarrow \pi_\theta + \eta \nabla_\theta J^{\hat{u}}(\pi_\theta)$
// Ascent stage
15: **else**
16: Find $\boldsymbol{\beta}^*$ according to Eq. 7
17: $\pi_\theta \leftarrow \pi_\theta + \eta \boldsymbol{\beta}^* \nabla_\theta \boldsymbol{F}(\pi_\theta)$
18: $\alpha \leftarrow \alpha - \lambda \nabla_\alpha L^\alpha(\alpha)$ //SAC temperature α update

Output: policy π_θ

discards the possibility that different states favor different candidates. To use this offline diversity during online adaptation, HiP3 first narrows the pool to a compact candidate set and then performs state-dependent policy selection within this set.

Before fine-tuning, HiP3 performs a one-time rollout ranking on the target task and selects the top-K policies as the online candidate set $\mathcal{P}_{on}$, without reusing the ranking trajectories for policy or model updates. A contextual bandit then serves as the high-level controller over these low-level candidate policies throughout online adaptation (Lines 12–24 of Algorithm 1 and the blue box in Figure 2).

At each online interaction step t, LinUCB computes a confidence-adjusted score for each candidate policy from the current state s_t. For notational compactness, we write $\mathcal{P}_{on} = \{\pi_\theta^k\}_{k=1}^{K_{on}}$ and the LinUCB statistics as $B(A, b)$, where $A = \{A_k\}_{k=1}^{K_{on}}$ and $b = \{b_k\}_{k=1}^{K_{on}}$. We softmax-normalize the LinUCB scores into a policy-preference vector P_t for visualization. Since softmax preserves their ordering, execution is equivalent to selecting the candidate with the highest raw LinUCB score. LinUCB then uses the observed environment reward as direct target-task feedback to update only the statistics (A_{k_t}, b_{k_t}) of the executed policy. Additional details are provided in Appendix B.6.

After selection, only the executed policy is fine-tuned using a Dyna-style SAC update with real online transitions in $\mathcal{D}_E$ and short model-generated rollouts in $\mathcal{D}_M$. The original offline dataset $\mathcal{D}$ is not retained during online fine-tuning.

5 Experimental Results

5.1 Experimental Setup

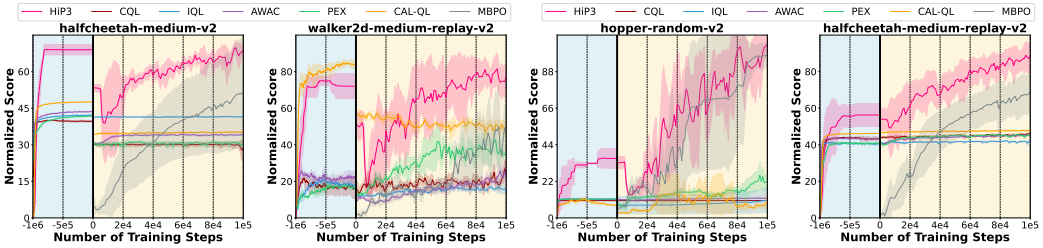
We evaluate HiP3 against O2O RL and online RL baselines on two task categories derived from the D4RL benchmark [9]: same online tasks and novel online tasks (see Appendix C.1 for a brief overview). Our primary evaluation target is adaptation under changes in dynamics and rewards. Fine-tuning on the same online tasks provides a complementary evaluation of the same framework.

We compare HiP3 with representative O2O RL baselines, including Cal-QL [2], PEX [3], AWAC [20], IQL [40], CQL [11], and FamO2O [23], together with the online RL baseline MBPO [15]. CQL, Cal-QL, AWAC, and IQL are run using the publicly available CORL [41] codebase with default hyperparameters unless otherwise specified, while PEX and MBPO use their official implementations. After 1M offline steps, all O2O methods except FamO2O are fine-tuned for 100K online steps, and MBPO is trained from scratch for 100K steps. For HiP3, the online candidate set is initialized by the one-time ranking described in Section 4 before the 100K-step fine-tuning stage. Implementation details are provided in Appendix C.2.

The offline policy-optimization budget is distributed across the complete policy pool rather than assigned independently to each policy, while each online update optimizes only the selected policy. Appendix C.4 reports matched end-to-end computational costs together with additional stage-wise runtime measurements.

		On the Same Online Tasks					On Novel Online Tasks				
		AWAC	PEX	Cal-QL	MBPO	HiP3 (Ours)	AWAC	PEX	Cal-QL	MBPO	HiP3 (Ours)
Random	HalfCheetah	35.6 ±0.5	56.9 ±1.2	15.4 ±5.6	66.0 ±15.4	82.6 ±2.3	33.2 ±0.4	42.2 ±2.6	11.1 ±2.5	47.1 ±22.1	63.5 ±2.2
	Hopper	11.7 ±0.2	22.2 ±4.1	12.7 ±8.5	98.2 ±6.1	105.7 ±4.1	9.2 ±0.1	23.5 ±17.2	11.7 ±4.7	86.4 ±4.6	95.2 ±5.5
	Walker2d	12.2 ±6.7	9.8 ±2.2	1.6 ±1.3	51.4 ±35.9	35.7 ±26.0	7.0 ±1.3	8.5 ±0.7	-0.6 ±1.1	44.6 ±11.4	49.2 ±26.7
Medium	HalfCheetah	43.6 ±0.8	44.1 ±0.3	49.7 ±0.3	66.0 ±15.4	90.1 ±4.1	33.9 ±0.4	30.4 ±0.9	35.4 ±0.1	47.1 ±22.1	69.3 ±2.1
	Hopper	101.5 ±0.4	99.4 ±1.2	83.3 ±12.3	98.2 ±6.1	106.7 ±3.0	88.5 ±0.1	74.8 ±15.9	37.6 ±4.6	86.4 ±4.6	95.4 ±3.8
	Walker2d	79.0 ±1.7	72.0 ±1.2	83.1 ±2.2	51.4 ±35.9	76.4 ±16.0	70.4 ±0.3	61.5 ±1.0	59.6 ±0.4	44.6 ±11.4	59.2 ±6.4
Medium -replay	HalfCheetah	45.3 ±0.1	45.9 ±0.1	47.8 ±0.1	66.0 ±15.4	89.6 ±2.7	29.9 ±3.8	30.4 ±3.5	32.6 ±0.2	47.1 ±22.1	69.4 ±3.6
	Hopper	29.6 ±2.8	36.0 ±7.0	95.1 ±1.4	98.2 ±6.1	103.4 ±3.7	29.6 ±2.8	39.4 ±14.7	65.9 ±14.0	86.4 ±4.6	100.1 ±11.0
	Walker2d	34.5 ±8.4	40.6 ±7.3	85.5 ±4.9	51.4 ±35.9	94.1 ±12.1	20.0 ±3.2	38.0 ±4.3	46.4 ±3.7	44.6 ±11.4	77.3 ±10.4
Medium -expert	HalfCheetah	103.9 ±1.5	48.3 ±5.6	92.5 ±1.6	66.0 ±15.4	92.8 ±3.4	77.6 ±0.7	29.9 ±1.0	64.4 ±1.9	47.1 ±22.1	74.7 ±2.0
	Hopper	112.3 ±0.4	89.0 ±31.6	112.2 ±0.2	98.2 ±6.1	106.5 ±4.7	88.8 ±4.4	50.9 ±9.5	88.0 ±5.5	86.4 ±4.6	99.1 ±7.6
	Walker2d	105.8 ±1.8	89.0 ±18.0	109.6 ±0.7	51.4 ±35.9	79.3 ±10.6	50.6 ±8.5	54.3 ±2.8	79.2 ±3.4	44.6 ±11.4	74.6 ±3.9
Total Mean		59.6 ±2.2	54.4 ±6.6	65.7 ±3.3	71.9 ±19.1	88.6 ±7.7	44.9 ±4.9	40.3 ±6.2	44.3 ±3.6	59.4 ±12.7	77.3 ±7.1

Table 1: Performance comparison on the same and novel online tasks. Novel online tasks are created by modifying the environments' gravity, friction, noise, and reward terms. Scores for each dataset are normalized according to the D4RL benchmark [9]: **Normalized Score** $= 100 \times \frac{\text{return} - \text{random return}}{\text{expert return} - \text{random return}}$. Full results can be found in Table 21 in the Appendix.



(a) Learning curves on novel online tasks (b) Learning curves on the same online tasks

Figure 3: Offline-to-online learning curves on the same and novel online tasks. The blue and yellow shaded regions denote offline pre-training and online fine-tuning, respectively. Curves are averaged over 5 random seeds with 10 evaluations per seed, and shaded areas show standard deviation. Full results across all datasets are available in Figure 8 and Figure 9 in the Appendix.

5.2 Main Results

As shown in Table 1, HiP3 achieves the strongest average performance across both the same-task and novel-task settings, with the best final performance on 7 of 12 same-task datasets and 9 of 12 novel-task datasets. Consistent with these aggregate results, Figure 3 shows that several representative O2O baselines struggle on novel tasks and, in some cases, also underperform on same-task settings initialized from lower-quality offline data such as random or medium-replay datasets. For readability, the main table reports representative baselines, while complete per-task comparisons, including IQL, CQL, and the available FamO2O results, are given in Appendix Table 21.

Together, these results indicate that HiP3 improves adaptation across varied online tasks by combining a diverse offline policy pool with state-dependent online selection. Because different states favor different optimism-pessimism trade-offs, HiP3 can switch among candidate policies during fine-tuning, whereas fixed-trade-off baselines remain tied to the bias of a single offline policy.

To test whether HiP3 generalizes beyond MuJoCo locomotion, we also evaluate it on Adroit and Maze2D. HiP3 remains competitive on the Adroit tasks and achieves the best performance on Maze2D among the compared baselines, suggesting that the benefits of HiP3 may extend to additional continuous-control and navigation settings. Detailed results are provided in Appendix D.2.

5.3 Adaptation under Stronger Distribution Shifts

We next evaluate HiP3 on a harder benchmark designed to stress adaptation under larger mismatches between the offline and online stages. Starting from the novel online tasks, we further modify both the environment dynamics and the reward design: we increase gravity from 11.5 to 13.5, scale the friction coefficients from $1.2\times$ to $1.25\times$, decrease the forward-reward weight from 0.85 to 0.8, and increase the control-cost weight from 0.2 to 0.25.

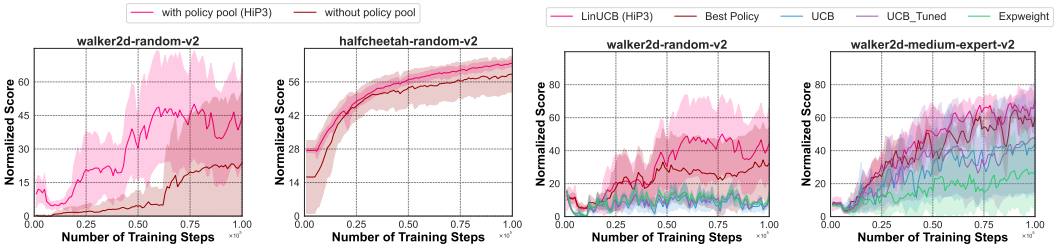


Figure 4: Ablations on novel online tasks: (a) removing the policy pool; (b) policy-selection algorithms. Full curves are provided in Figure 10 and Figure 11 in the Appendix, respectively.

These modifications create substantially harder adaptation settings than the standard novel online tasks. We compare HiP3 with several recent O2O RL baselines, including SO2 [27], WSRL [24], FQL [30] and EDIS instantiated with IQL and Cal-QL backbones [29], under the same 100K-step online fine-tuning budget using their official implementations or default settings. As shown in Table 2, HiP3 achieves the strongest average performance and remains competitive or superior on most tasks under these harder shifts. These results indicate that the combination of a diverse policy pool and state-dependent policy selection remains effective when the online task departs more substantially from the offline data distribution.

Method	HiP3	SO2	WSRL	FQL	IQL-EDIS	Cal-QL-EDIS
hc-r-v2	42.9±3.9	29.8±0.7	38.7±1.5	22.2±1.8	15.7±0.6	**48.9±4.6**
hc-mr-v2	**55.8±4.6**	35.9±0.5	52.4±0.5	44.1±0.3	27.2±0.3	43.0±2.1
hop-r-v2	**65.4±6.3**	14.0±11.3	9.5±0.6	12.9±0.4	6.0±0.1	8.7±1.5
hop-mr-v2	**102.1±4.1**	17.1±1.1	87.3±0.3	50.8±14.8	12.8±0.7	75.2±0.9
wk-r-v2	**55.2±7.2**	39.5±2.8	53.6±7.6	5.7±1.8	3.5±0.3	27.0±11.9
wk-mr-v2	74.9±5.2	45.3±10.6	**75.9±5.9**	29.3±12.0	54.5±0.8	67.6±10.1
mean	**66.1±5.2**	30.3±4.5	52.9±2.7	27.5±5.2	20.0±0.5	45.1±5.2

Table 2: Performance under more challenging distribution shifts. We modify environment dynamics and reward functions to create harder adaptation scenarios. (Notations: hc = halfcheetah, hop = hopper, wk = walker2d; r = random, mr = medium-replay.)

5.4 Ablation Studies

The preceding results establish HiP3's adaptation advantage under both standard and stronger task shifts. We next isolate four design choices behind this performance: preserving multiple offline policies, state-dependent policy selection, MOSAIC-based pool construction, and candidate-set initialization. Figure 4 examines the first two components, while Table 3 examines the latter two. Additional hyperparameter sensitivity analyses are reported in Appendix D.4, while model and training sensitivity analyses are provided in Appendix D.5.1.

Figure 4a compares HiP3 with a single-policy variant that keeps the same Dyna-style online fine-tuning procedure but removes the offline policy pool. The multi-policy HiP3 framework achieves a higher mean score in all 12 reported novel online task settings. The complete 12-task comparison in Appendix Table 20 shows that preserving multiple offline trade-offs provides a stronger starting set for online adaptation.

Figure 4b shows the performance of HiP3 with various policy selection algorithms, including Best Policy, UCB [42], UCB-Tuned [42], Expweight [43] and LinUCB [33], which is the policy-selection algorithm used in HiP3. Holding the candidate pool fixed, LinUCB achieves the strongest performance on most evaluated tasks relative to fixed-policy and non-contextual selection alternatives. This pattern suggests that a single fixed policy is often insufficient during deployment because different states may favor different optimism-pessimism trade-offs. For instance, a pessimistic policy may be preferable in in-distribution states, whereas an optimistic policy may be more effective in under-explored states. These results support the effectiveness of state-dependent policy selection with LinUCB during online fine-tuning.

Method	HiP3	HiP3-LS
halfcheetah-r-v2	63.5±2.2	47.5±17.1
halfcheetah-mr-v2	69.4±3.6	57.8±15.4
hopper-r-v2	95.2±5.5	62.3±21.9
hopper-mr-v2	100.1±11.0	82.9±6.0
mean	82.1±5.5	62.6±15.1

(a)

Method	HiP3 (Rollout)	HiP3 (FQE)	PEX	Cal-QL
hc-r-v2	63.5±2.2	58.7±1.7	42.2±2.6	11.1±2.5
hc-mr-v2	69.4±3.6	67.8±3.3	30.4±3.5	32.6±0.2
hop-r-v2	95.2±5.5	87.1±9.5	23.5±17.2	11.7±4.7
hop-mr-v2	100.1±11.0	100.1±6.8	39.4±14.7	65.9±14.0
mean	82.1±5.5	78.4±5.3	33.9±9.5	30.3±5.4

(b)

Table 3: Ablation and controlled comparison studies. (a) Comparison with a matched linear-scalarization baseline. (b) Comparison of candidate-set initialization strategies, including target-interaction-free candidate ranking via FQE.

The multi-policy ablation establishes the value of preserving multiple candidates. Table 3a compares MOSAIC-based and fixed-scalarization pool-construction pipelines under matched policy-optimization budgets. HiP3-LS uses the same model-based backbone, offline data, total policy-optimization budget, and online selection framework, while training five policies with $wJ^{\hat{r}}(\pi_\theta) + (1-w)J^{\hat{u}}(\pi_\theta)$, where $w \in \{0.9, 0.7, 0.5, 0.3, 0.1\}$. Under this controlled protocol, HiP3 consistently outperforms HiP3-LS, supporting the advantage of MOSAIC-based policy-pool construction.

Table 3b examines whether HiP3 remains effective when candidate initialization avoids target-task interaction. Replacing rollout-based ranking with Fitted Q Evaluation (FQE) [44] avoids target-environment interactions during candidate ranking, with only a modest reduction in subsequent adaptation performance. HiP3+FQE still substantially outperforms PEX and Cal-QL in these settings, demonstrating that HiP3 retains strong adaptation performance even with interaction-free candidate initialization.

5.5 Online Policy-Selection Dynamics

The preceding ablations establish the value of both policy diversity and state-dependent policy selection. We therefore inspect how LinUCB's preferences over the candidate pool evolve during online adaptation. Figure 5 illustrates two representative patterns. On walker2d-random-v2, LinUCB develops a stronger preference for one candidate, and HiP3 closely tracks or slightly exceeds that candidate's performance. On halfcheetah-medium-replay-v2, several candidates retain comparable preferences, while HiP3 outperforms the fixed-candidate baselines. These examples illustrate how HiP3 uses the policy pool during adaptation and complement the policy-selection ablation in Figure 4b.

Table 4 complements these preference visualizations by summarizing the policies actually executed across all 12 novel online task settings. The dominant-policy selection frequency varies across domains and tasks, and no single candidate accounts for the entire online run. Thus, a pronounced preference for one candidate should not be interpreted as exclusive execution of that policy. Additional policy-preference visualizations and a comparison of bandit model capacity are provided in Appendix D.5.2.

Domain	Mean Dominant-policy Selection Frequency	Range Across Settings
HalfCheetah	37.6%	28.6%-62.0%
Hopper	35.1%	29.0%-44.4%
Walker2d	54.4%	46.1%-58.8%
Overall	42.3%	28.6%-62.0%

Table 4: Quantitative analysis of LinUCB policy selection behavior on 12 novel online task settings. The dominant policy is defined as the candidate selected most frequently over the full online run. Its selection frequency is the fraction of online steps assigned to that candidate.

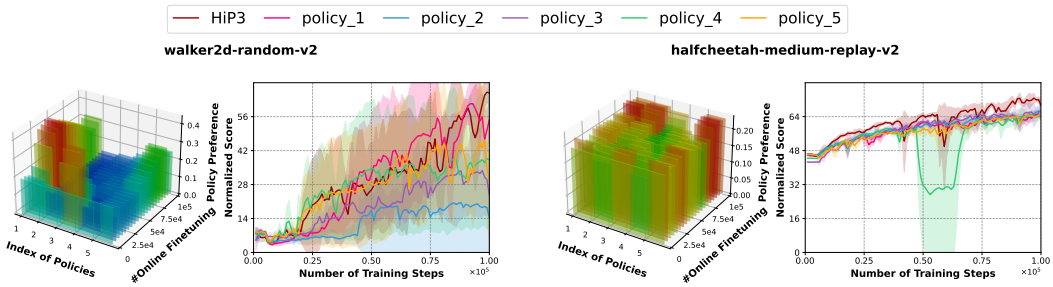


Figure 5: Heat maps visualize the policy preferences P_t, obtained by softmax-normalizing the LinUCB scores. HiP3 executes $k_t = \arg\max_k P_t(k)$ at each interaction step. A sharply peaked preference profile indicates a clear preference for one candidate, whereas a diffuse profile indicates smaller preference margins among candidates. The accompanying curves compare HiP3 with the five candidate policies on two representative novel online tasks. Additional quantitative and qualitative analyses are provided in Appendix D.5.2.

6 Conclusion

This paper studies the optimism-pessimism trade-off in model-based O2O RL and introduces HiP3, which preserves diverse trade-offs offline and performs state-dependent policy selection online rather than committing to a single balance in advance. HiP3 formulates optimism and pessimism as two explicit objectives, uses MOSAIC to construct a Pareto policy pool, and uses a contextual bandit to select candidate policies conditioned on the current state throughout model-based fine-tuning. Across both same and novel online tasks, HiP3 achieves strong aggregate performance and maintains its advantage under stronger distribution shifts. These results support a simple design principle for model-based O2O RL: preserve trade-off diversity offline and defer the choice among candidate policies to online adaptation.

Limitation: While HiP3 performs strongly on the evaluated O2O RL benchmarks, we identify three key areas for further investigation. First, similar to prior methods, HiP3 still exhibits an initial performance degradation at the early stage of online fine-tuning. We provide an empirical analysis of this behavior in Appendix D.3 and believe that mitigating such transient instability remains a promising avenue for future work. Second, while our evaluation primarily focuses on the widely-adopted D4RL MuJoCo suite to ensure reproducibility and fair comparison, extending the proposed framework to higher-dimensional environments with more pronounced distribution shifts represents a valuable future direction. Third, because reward and uncertainty are model-derived surrogates, offline Pareto trade-offs may not exactly match real-environment trade-offs under severe model bias.

Acknowledgments and Disclosure of Funding

Funding. This work is partially supported by the CityUHK Startup Grant (Grant No. 9382014) and the Jiangxi Provincial Natural Science Foundation (Grant No. 20262BAC200479).

Competing Interests. The authors declare no competing interests.

References

[1] 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.

[2] Mitsuhiko Nakamoto, Simon Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-QL: Calibrated offline RL pre-training for efficient online fine-tuning. In *Advances in Neural Information Processing Systems*, 2023.

[3] Haichao Zhang, Wei Xu, and Haonan Yu. Policy expansion for bridging offline-to-online reinforcement learning. In *International Conference on Learning Representations*, 2023.

[4] Yihuan Mao, Chao Wang, Bin Wang, and Chongjie Zhang. MOORe: Model-based offline-to-online reinforcement learning. *arXiv preprint arXiv:2201.10070*, 2022.

[5] Rafael Rafailov, Kyle Beltran Hatch, Victor Kolev, John D Martin, Mariano Phielipp, and Chelsea Finn. MOTO: Offline to online fine-tuning for model-based reinforcement learning. In *Workshop on Reincarnating Reinforcement Learning at ICLR 2023*, 2023.

[6] Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. In *Advances in Neural Information Processing Systems*, 2020.

[7] Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. MOReL: Model-based offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2020.

[8] Yijun Yang, Jing Jiang, Tianyi Zhou, Jie Ma, and Yuhui Shi. Pareto policy pool for model-based offline reinforcement learning. In *International Conference on Learning Representations*, 2021.

[9] 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.

[10] Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *International Conference on Machine Learning*. PMLR, 2019.

[11] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative Q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems*, 2020.

[12] Ziyu Wang, Alexander Novikov, Konrad Zolna, Josh S Merel, Jost Tobias Springenberg, Scott E Reed, Bobak Shahriari, Noah Siegel, Caglar Gulcehre, Nicolas Heess, et al. Critic regularized regression. In *Advances in Neural Information Processing Systems*, 2020.

[13] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[14] Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy Q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, 2019.

[15] Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*, 2019.

[16] Arthur Argenson and Gabriel Dulac-Arnold. Model-based offline planning. In *International Conference on Learning Representations*, 2021.

[17] Tatsuya Matsushima, Hiroki Furuta, Yutaka Matsuo, Ofir Nachum, and Shixiang Gu. Deployment-efficient reinforcement learning via model-based offline optimization. In *International Conference on Learning Representations*, 2021.

[18] Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. COMBO: Conservative offline model-based policy optimization. In *Advances in Neural Information Processing Systems*, 2021.

[19] Pang Wei Koh, Shiori Sagawa, Henrik Marklund, Sang Michael Xie, Marvin Zhang, Akshay Balsubramani, Weihua Hu, Michihiro Yasunaga, Richard Lanas Phillips, Irena Gao, et al. WILDS: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*. PMLR, 2021.

[20] Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. AWAC: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

[21] Seunghyun Lee, Younggyo Seo, Kimin Lee, Pieter Abbeel, and Jinwoo Shin. Offline-to-online reinforcement learning via balanced replay and pessimistic Q-ensemble. In *Conference on Robot Learning*. PMLR, 2022.

[22] Han Zheng, Xufang Luo, Pengfei Wei, Xuan Song, Dongsheng Li, and Jing Jiang. Adaptive policy learning for offline-to-online reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[23] Shenzhi Wang, Qisen Yang, Jiawei Gao, Matthieu Lin, Hao Chen, Liwei Wu, Ning Jia, Shiji Song, and Gao Huang. Train once, get a family: State-adaptive balances for offline-to-online reinforcement learning. In *Advances in Neural Information Processing Systems*, 2023.

[24] Zhiyuan Zhou, Andy Peng, Qiyang Li, Sergey Levine, and Aviral Kumar. Efficient online reinforcement learning fine-tuning need not retain offline data. In *International Conference on Learning Representations*, 2025.

[25] Max Sobol Mark, Ali Ghadirzadeh, Xi Chen, and Chelsea Finn. Fine-tuning offline policies with optimistic action selection. In *Deep Reinforcement Learning Workshop NeurIPS 2022*, 2022.

[26] Ikechukwu Uchendu, Ted Xiao, Yao Lu, Banghua Zhu, Mengyuan Yan, Joséphine Simon, Matthew Bennice, Chuyuan Fu, Cong Ma, Jiantao Jiao, et al. Jump-start reinforcement learning. In *International Conference on Machine Learning*. PMLR, 2023.

[27] Yinmin Zhang, Jie Liu, Chuming Li, Yazhe Niu, Yaodong Yang, Yu Liu, and Wanli Ouyang. A perspective of Q-value estimation on offline-to-online reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

[28] Qin-Wen Luo, Ming-Kun Xie, Yewen Wang, and Sheng-Jun Huang. Optimistic critic reconstruction and constrained fine-tuning for general offline-to-online RL. In *Advances in Neural Information Processing Systems*, 2024.

[29] Xu-Hui Liu, Tian-Shuo Liu, Shengyi Jiang, Ruifeng Chen, Zhilong Zhang, Xinwei Chen, and Yang Yu. Energy-guided diffusion sampling for offline-to-online reinforcement learning. In *International Conference on Machine Learning*. PMLR, 2024.

[30] Seohong Park, Qiyang Li, and Sergey Levine. Flow Q-learning. In *International Conference on Machine Learning*. PMLR, 2025.

[31] Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024.

[32] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT Press, 2018.

[33] Wei Chu, Lihong Li, Lev Reyzin, and Robert Schapire. Contextual bandits with linear payoff functions. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, 2011.

[34] Xi Lin, Hui-Ling Zhen, Zhenhua Li, Qing-Fu Zhang, and Sam Kwong. Pareto multi-task learning. In *Advances in Neural Information Processing Systems*, 2019.

[35] Debabrata Mahapatra and Vaibhav Rajan. Multi-task learning with user preferences: Gradient descent with controlled ascent in Pareto optimization. In *International Conference on Machine Learning*. PMLR, 2020.

[36] Weiyu Chen and James Kwok. Multi-objective deep learning with adaptive reference vectors. In *Advances in Neural Information Processing Systems*, 2022.

[37] Jean-Antoine Désidéri. Multiple-gradient descent algorithm (MGDA) for multiobjective optimization. *Comptes Rendus Mathematique*, 350(5-6):313–318, 2012.

[38] Martin Jaggi. Revisiting Frank-Wolfe: Projection-free sparse convex optimization. In *International Conference on Machine Learning*. PMLR, 2013.

[39] 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*. PMLR, 2018.

[40] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit Q-learning. In *International Conference on Learning Representations*, 2022.

[41] Denis Tarasov, Alexander Nikulin, Dmitry Akimov, Vladislav Kurenkov, and Sergey Kolesnikov. CORL: Research-oriented deep offline reinforcement learning library. In *Advances in Neural Information Processing Systems*, 2023.

[42] Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002.

[43] Samuel Kessler, Jack Parker-Holder, Philip Ball, Stefan Zohren, and Stephen J Roberts. Same state, different task: Continual reinforcement learning without interference. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2022.

[44] Hoang Le, Cameron Voloshin, and Yisong Yue. Batch policy learning under constraints. In *International Conference on Machine Learning*. PMLR, 2019.

[45] Yudeng Lin, Qingtian Zhang, Bin Gao, Jianshi Tang, Peng Yao, Chongxuan Li, Shiyu Huang, Zhengwu Liu, Ying Zhou, Yuyi Liu, et al. Uncertainty quantification via a memristor bayesian deep neural network for risk-sensitive reinforcement learning. *Nature Machine Intelligence*, 5(7):714–723, 2023.

[46] José Luis Pérez, Javier Corrochano, Javier García, Rubén Majadas, Cristina Ibanez-Llano, Sergio Pérez, and Fernando Fernández. Discrete uncertainty quantification for offline reinforcement learning. *Journal of Artificial Intelligence and Soft Computing Research*, 13(4):273–287, 2023.

[47] Rahul Rahaman and Alexandre H Thiery. Uncertainty quantification and deep ensembles. In *Advances in Neural Information Processing Systems*, 2021.

[48] Kamyar Ghasemipour, Shixiang Shane Gu, and Ofir Nachum. Why so pessimistic? estimating uncertainties for offline RL through ensembles, and why their independence matters. In *Advances in Neural Information Processing Systems*, 2022.

[49] Prateek Jain, Sham M Kakade, Rahul Kidambi, Praneeth Netrapalli, and Aaron Sidford. Parallelizing stochastic gradient descent for least squares regression: Mini-batching, averaging, and model misspecification. *Journal of Machine Learning Research*, 18(223):1–42, 2018.

[50] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.

[51] Stephen P Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.

[52] Abbas Abdolmaleki, Sandy H Huang, Giulia Vezzani, Bobak Shahriari, Jost Tobias Springenberg, Shruti Mishra, Dhruva TB, Arunkumar Byravan, Konstantinos Bousmalis, Andras Gyorgy, et al. On multi-objective policy optimization as a tool for reinforcement learning: Case studies in offline RL and finetuning. *arXiv preprint arXiv:2106.08199*, 2021.

[53] 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*. IEEE, 2012.

[54] Yunhai Feng, Nicklas Hansen, Ziyan Xiong, Chandramouli Rajagopalan, and Xiaolong Wang. Finetuning offline world models in the real world. In *Conference on Robot Learning*, 2023.

Organization of Appendix

The appendix is organized as follows. In Section A, we discuss the potential societal impacts of our work. In Section B, we provide additional theoretical and algorithmic details of HiP3, including the convergence analysis of MOSAIC, uncertainty estimation, the environment model, background on multi-objective optimization, comparison with linear scalarization, and the bandit selection mechanism. In Section C, we describe the experimental setup, implementation details, computing infrastructure and cost, and hyper-parameters used in our experiments. In Section D, we present additional empirical analyses, including validation of the optimism-pessimism crossover behavior, results on additional domains, transient performance analysis, hyperparameter sensitivity studies, additional ablations, policy-selection analyses, complete benchmark results, and extended baseline comparisons.

A Societal Impacts

This work contributes to offline-to-online reinforcement learning by explicitly modeling the trade-off between optimistic exploration and pessimistic conservatism, a central challenge when adapting policies under distribution shift. By enabling flexible policy selection during online interaction, the proposed approach has the potential to reduce unnecessary exploration and improve sample efficiency, which may benefit real-world learning systems where online interaction is costly or risky.

At the same time, as with many offline-to-online RL methods, transient performance degradation during early online fine-tuning highlights the need for cautious deployment, particularly in safety-critical settings or under severe distribution shifts. While our experiments focus on standard continuous control benchmarks, applying the proposed framework in real-world systems would require careful validation, monitoring, and safeguards to ensure stable behavior. We hope this work encourages further research into principled uncertainty-aware adaptation mechanisms that balance performance and safety in online learning.

B Theory and Algorithmic Details

B.1 Convergence Analysis

Assumption 1. Suppose the m objectives $f_1, f_2, \ldots, f_m$ are differentiable and their gradients are L_i-Lipschitz continuous with constant $L_i > 0$.

Assumption 2. For any $i \in \{1, 2, \cdots, m\}$, $\{g_i(\theta_t)\}_{t=1}^{\infty}$ is an independent stochastic gradient sequence satisfying:

$$\mathbb{E}[g_i(\theta_t)] = \nabla f_i(\theta_t),\ Var(g_i(\theta_t)) = \mathbb{E}[||g_i(\theta_t) - \nabla f_i(\theta_t)||^2] \leq \sigma^2.$$

MGDA Update: In the implementation, MOSAIC updates the policy by minimizing a convex combination of three surrogate losses. For the analysis below, we work in the equivalent objective-space view and let $f_1, \ldots, f_m$ denote the corresponding objectives, so that descent on each surrogate loss is equivalent to ascent on its paired objective. In the ascent stage of MOSAIC (see Alg. 2), define the combined objective-space update direction

$$d_t = \sum_{i=1}^{m} \beta_{i,t}\, g_i(\theta_t),$$

where the weights $\beta_{i,t} \geq 0$ are chosen by solving the MGDA subproblem (Eq. 7), ensuring that no individual objective is decreased at that step (i.e., $\nabla f_i(\theta_t)^\top d_t \geq 0, \forall i$). The stochastic policy update in objective space is then:

$$\theta_{t+1} = \theta_t + \eta d_t.$$

Let $\bar{d}_t = \mathbb{E}_t[d_t]$ denote the conditional expected update direction.

Theorem B.1 (First-order convergence of the ascent stage)**.** *Suppose that Assumptions 1 and 2 hold, and let $L = \max_i L_i$. Assume further that, for each objective f_i, there exists a finite upper reference value $f_i^\star$ such that $f_i(\theta_t) \leq f_i^\star$ for all iterates generated by the ascent stage. Define*

$$\Delta_i = f_i^\star - f_i(\theta_0), \qquad \Delta = \max_i \{\Delta_i\}.$$

For any fixed T, choose the constant stepsize

$$\eta = \frac{c}{\sqrt{T}}, \qquad 0 < c < \frac{2}{L}.$$

Then

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|\bar{d}_t\|^2] = O\left(\frac{\Delta}{c\sqrt{T}} + \frac{cL\sigma^2}{\sqrt{T}}\right).$$

Consequently, for any $\epsilon > 0$, after

$$T = O\left(\frac{1}{\epsilon^2}\left(\frac{\Delta}{c} + cL\sigma^2\right)^2\right)$$

iterations, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[\|\bar{d}_t\|^2] \leq \epsilon.$$

Thus, the ascent stage enjoys a first-order convergence guarantee toward Pareto stationarity in expectation.

Proof. By L_i-smoothness and the ascent update $\theta_{t+1} = \theta_t + \eta d_t$, we have for each f_i:

$$\begin{aligned} f_i(\theta_{t+1}) &\geq f_i(\theta_t) + \nabla f_i(\theta_t)^\top(\theta_{t+1} - \theta_t) - \frac{L_i}{2}\|\theta_{t+1} - \theta_t\|^2 \\ &= f_i(\theta_t) + \eta\, \nabla f_i(\theta_t)^\top d_t - \frac{L_i}{2}\eta^2\|d_t\|^2. \end{aligned}$$

Taking conditional expectation $\mathbb{E}_t[\cdot]$ on both sides yields

$$\mathbb{E}_t[f_i(\theta_{t+1})] \geq f_i(\theta_t) + \eta\, \nabla f_i(\theta_t)^\top \bar{d}_t - \frac{L_i}{2}\eta^2\mathbb{E}_t[\|d_t\|^2], \tag{8}$$

where $\bar{d}_t = \mathbb{E}_t[d_t]$.

Also,

$$\begin{aligned} \mathbb{E}_t\left[\|d_t\|^2\right] &= \left\|\bar{d}_t\right\|^2 + \mathbb{E}_t\left[\left\|d_t - \bar{d}_t\right\|^2\right] \\ &\leq \left\|\bar{d}_t\right\|^2 + \sigma^2\sum_{i=1}^m \beta_{i,t}^2 \\ &\leq \left\|\bar{d}_t\right\|^2 + \sigma^2\left(\sum_{i=1}^m \beta_{i,t}\right)^2 \\ &= \left\|\bar{d}_t\right\|^2 + \sigma^2, \end{aligned} \tag{9}$$

where we use Assumption 2 and the independence of the stochastic gradients. The second inequality uses the fact

$$\sum_{i=1}^m \beta_{i,t}^2 \leq \left(\sum_{i=1}^m \beta_{i,t}\right)^2$$

along with $\sum_{i=1}^m \beta_{i,t} = 1$.

Substituting Eq. (9) into Eq. (8) gives

$$\begin{aligned} \mathbb{E}_t[f_i(\theta_{t+1})] &\geq f_i(\theta_t) + \eta\, \nabla f_i(\theta_t)^\top \bar{d}_t - \frac{L_i}{2}\eta^2\left(\left\|\bar{d}_t\right\|^2 + \sigma^2\right) \\ &\geq f_i(\theta_t) + \eta\, \|\bar{d}_t\|^2 - \frac{L_i}{2}\eta^2\left(\left\|\bar{d}_t\right\|^2 + \sigma^2\right), \end{aligned}$$

where we use the fact $\nabla f_i^\top \bar{d}_t \geq \|\bar{d}_t\|^2$ (Lemma 2.1 in [37]) in the second inequality.

Rearranging the above inequality and then taking total expectation yields

$$\left(\eta - \frac{L_i}{2}\eta^2\right)\mathbb{E}[\|\bar{d}_t\|^2] \leq \mathbb{E}\left[f_i(\theta_{t+1}) - f_i(\theta_t)\right] + \frac{L_i}{2}\eta^2\sigma^2.$$

Summing over $t = 0$ to $T-1$ and dividing both sides by T gives

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\bar{d}_t\|^2\right] \leq \frac{\mathbb{E}[f_i(\theta_T)] - f_i(\theta_0)}{T\left(\eta - \frac{L_i}{2}\eta^2\right)} + \frac{L_i\eta^2\sigma^2}{2\left(\eta - \frac{L_i}{2}\eta^2\right)}.$$

Since $\mathbb{E}[f_i(\theta_T)] \leq f_i^\star$ by assumption, it follows that

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\bar{d}_t\|^2\right] \leq \frac{\Delta_i}{T\left(\eta - \frac{L_i}{2}\eta^2\right)} + \frac{L_i\eta^2\sigma^2}{2\left(\eta - \frac{L_i}{2}\eta^2\right)}.$$

Setting $\eta = \frac{c}{\sqrt{T}}$ with $0 < c < \frac{2}{L}$ and using $L = \max_i L_i$, we obtain

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\bar{d}_t\|^2\right] = O\left(\frac{\Delta}{c\sqrt{T}} + \frac{cL\sigma^2}{\sqrt{T}}\right).$$

Therefore, for any $\epsilon > 0$, after

$$T = O\left(\frac{1}{\epsilon^2}\left(\frac{\Delta}{c} + cL\sigma^2\right)^2\right)$$

iterations, we have

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left[\|\bar{d}_t\|^2\right] \leq \epsilon.$$

□

B.2 Uncertainty Estimation u in HiP3

The predicted uncertainty u of the environment model M is estimated as the maximum standard deviation among the ensemble, i.e., $u(s_t, a_t) = \max_{i=1}^{N} ||\sigma_i(s_t, a_t)||$. This ensemble-based uncertainty estimation strategy follows established practices in model-based offline RL such as MOPO [6] and MOReL [7]. This method is favored for its scalability, ease of implementation, and effectiveness in representing epistemic uncertainty in deep neural networks. Although alternative techniques such as Bayesian methods [45] and unsupervised representation-based approaches [46] are available, recent studies [47, 48] have demonstrated that deep ensembles tend to provide more accurate and better-calibrated uncertainty estimates. These results indicate that ensemble methods strike an effective balance between predictive performance and computational cost, making them a reliable and practical choice for uncertainty estimation in HiP3.

A limitation of using predicted return and uncertainty directly is that they may not match the true dynamics in out-of-distribution regions, especially when the offline dataset has limited coverage. Conservative approaches [5, 6] address this by penalizing high-uncertainty predictions. While effective in reducing overestimation, this requires careful tuning of penalty coefficients and may restrict exploration during online fine-tuning.

HiP3 avoids the use of fixed penalties by formulating optimism and pessimism as two separate objectives in a bi-objective optimization problem. This formulation produces a Pareto set of offline policies with different optimism-pessimism levels. During online fine-tuning, HiP3 performs state-dependent policy selection over these policies.

B.3 Environment Model M

The environment model used by HiP3 consists of a bootstrap ensemble of N environment models: $M = \{M_i\}_{i=1}^{N}$. Each member i of the ensemble is a probabilistic neural network, taking state-action pairs (s_t, a_t) as input and outputting a Gaussian distribution over the next state

and reward: $\hat{P}_i(s_{t+1}, r|s_t, a_t) = \mathcal{N}(\mu_i(s_t, a_t), \sigma_i(s_t, a_t))$. The predicted reward is the ensemble mean of the reward components $\hat{r}(s_t, a_t) = \frac{1}{N}\sum_{i=1}^{N} \mu_{i,r}(s_t, a_t)$, where $\mu_{i,r}(s_t, a_t)$ denotes the reward component of the joint predicted mean $\mu_i(s_t, a_t)$. The predicted uncertainty u of the environment model M is estimated as the maximum standard deviation among the ensemble, i.e., $u(s_t, a_t) = \max_{i=1}^{N} ||\sigma_i(s_t, a_t)||$.

This environment model is trained according to the design of MOPO [6]. Specifically, we trained an ensemble of 7 models, from which the 5 best models ($N = 5$) in terms of the prediction errors are selected. Each model of the ensemble is independently trained using the maximum likelihood estimation with mini-batch stochastic gradient descent [49]. During the inference stage, the policy interacts with a randomly selected model from the N ensemble models, generating state-reward pairs from the corresponding distributions. This approach introduces variability by sampling different transitions from different models within a single episode, leading to a highly stochastic environment model. Previous works [6, 15] have shown that this environment model effectively mitigates the model exploitation problem in offline RL scenarios.

The hyper-parameters of the environment model M are provided in Table 5. For the MuJoCo tasks, we adopt the standard MBPO/MOPO world-model configuration [6, 15] given their proven effectiveness in similar continuous control environments.

Hyper-parameter	**Value**
Ensemble size	7
Number of elite models	5
Model architecture	FC(200, 200, 200, 200)
Batch size	256
Optimizer	Adam
Learning rate	1×10^{-3}
Holdout ratio	0.2

Table 5: The hyper-parameters used for environment model M

B.4 Multi-objective Optimization

Multi-objective optimization (MOO) considers the simultaneous optimization of two or more potentially conflicting objectives [50]. Formally, given a decision space $\mathcal{X}$ and d objective functions $f_i : \mathcal{X} \to \mathbb{R}$ for $i \in \{1, \ldots, d\}$, we consider the following vector optimization problem:

$$\max_{x \in \mathcal{X}} \mathbf{f}(x) \triangleq \big(f_1(x), f_2(x), \ldots, f_d(x)\big).$$

The trade-offs among objectives arise because improving one objective may degrade another, so a single solution that simultaneously maximizes all objectives need not exist. MOO may therefore admit multiple Pareto-optimal solutions representing different trade-offs. We use the following standard definitions.

Dominance. Given two solutions $x_1, x_2 \in \mathcal{X}$, we say that x_1 dominates x_2 (denoted $x_1 \succ x_2$) if

$$f_i(x_1) \geq f_i(x_2), \quad \forall i \in \{1, \ldots, d\},$$

and

$$f_j(x_1) > f_j(x_2)$$

for at least one objective j.

Pareto optimality. A solution $x^\star \in \mathcal{X}$ is Pareto-optimal if there does not exist another solution $x' \in \mathcal{X}$ such that $x' \succ x^\star$.

Pareto set and Pareto front. The Pareto set is the set of Pareto-optimal solutions in the decision space:

$$\mathcal{P}^\star = \{x \in \mathcal{X} \mid \nexists\, x' \in \mathcal{X} \text{ such that } x' \succ x\}.$$

The corresponding Pareto front is the image of the Pareto set in the objective space:

$$\mathcal{F}^\star = \{\mathbf{f}(x) \mid x \in \mathcal{P}^\star\}.$$

By providing Pareto-optimal solutions spanning different regions of the Pareto front, MOO enables different trade-offs to be preserved for subsequent selection according to task requirements.

B.5 Comparison with linear scalarization

Linear scalarization (LS) provides a natural baseline for multi-objective RL by combining multiple objectives through a fixed scalarization weight. In the context of model-based O2O RL, fixed linear scalarization introduces two structural limitations that are particularly relevant to optimism-pessimism trade-offs:

First, linear scalarization can recover supported Pareto-optimal solutions by varying the scalarization weight, but it cannot in general recover unsupported Pareto-optimal solutions when the attainable objective set is non-convex [51]. HiP3 instead uses reference-vector-guided constrained optimization together with MGDA, and is therefore not restricted to the supported solutions induced by fixed linear scalarization weights. This allows MOSAIC to target trade-off regions that linear scalarization may miss.

Second, a single LS objective fixes a global weight λ and therefore a single optimism-pessimism balance. This can be extended by training multiple LS policies with different weights and selecting among them online, as in our HiP3-LS baseline. However, each candidate remains tied to a predefined scalarization weight, and different weights can map to the same or similar supported solutions, which can produce redundant candidates and limit the diversity available for online selection. HiP3 instead directly targets distinct trade-off regions through reference vectors and then performs state-dependent policy selection over the resulting pool.

For a controlled comparison, we construct HiP3-LS by replacing MOSAIC with fixed linear scalarization objectives. HiP3-LS trains five policies by maximizing

$$wJ^{\hat{r}}(\pi_\theta) + (1-w)J^{\hat{u}}(\pi_\theta), \qquad w \in \{0.9, 0.7, 0.5, 0.3, 0.1\},$$

which matches the first coordinates of HiP3's reference vectors. Each policy receives an equal share of the 1M-step offline budget and is used directly as one arm of the same LinUCB selector.

To provide an additional comparison with an existing multi-objective RL approach beyond linear scalarization, we also combine our bandit selector with DiME [52]. Since no open-source implementation is available, we implement DiME on top of AWAC following the original paper.

Method	**HiP3**	**HiP3-LS**	**DiME+AWAC**	**AWAC**
halfcheetah-random-v2	63.5±2.2	47.5±17.1	28.4±1.6	33.2±0.4
halfcheetah-medium-replay-v2	69.4±3.6	57.8±15.4	36.0±4.1	29.9±3.8
hopper-random-v2	95.2±5.5	62.3±21.9	7.0±4.9	9.2±0.1
hopper-medium-replay-v2	100.1±11.0	82.9±6.0	51.2±8.0	29.6±2.8
mean	82.1±5.5	62.6±15.1	30.7±4.7	25.5±1.8

Table 6: Comparison with Linear Scalarization Method

Under the matched protocol, HiP3 achieves a mean score of 82.1 compared with 62.6 for HiP3-LS. DiME+AWAC improves over vanilla AWAC on several novel online tasks, while HiP3 achieves substantially higher aggregate performance under the same offline and online optimization budgets. These results support the motivation of Eq. 2: avoiding restriction to fixed linear scalarization enables more flexible offline trade-off coverage and provides a stronger policy pool for subsequent online adaptation.

B.6 Contextual Policy Selection with LinUCB

We use LinUCB [33] as a lightweight contextual selector for state-dependent policy selection during online adaptation. Each policy in the online candidate set is treated as an arm, while the current

environment state serves as the context. Formally, we denote the candidate policy set by $\{\pi^k\}_{k=1}^K$ and write the LinUCB statistics compactly as $B(A, b)$, where $A = \{A_k\}_{k=1}^K$ and $b = \{b_k\}_{k=1}^K$, with each $A_k \in \mathbb{R}^{d\times d}$ initialized to I_d and each $b_k \in \mathbb{R}^d$ initialized to 0_d. At each online interaction step t, LinUCB computes a confidence-adjusted score for each candidate policy from the current state s_t:

$$\begin{aligned} &u_t^k \leftarrow (A_k^{-1} b_k)^T s_t + \omega\sqrt{s_t^T A_k^{-1} s_t}, k = 1, ..., K \\ &P_t \leftarrow \mathrm{Softmax}(u_t^1, \ldots, u_t^K), \\ &k_t \leftarrow \arg\max_k P_t(k). \end{aligned} \tag{10}$$

Here P_t denotes the policy-preference vector obtained by softmax-normalizing the LinUCB scores. Softmax preserves the ranking of arms, so selecting $\arg\max_k P_t(k)$ executes the policy with the highest raw LinUCB score. Figures 5 and 7 visualize P_t throughout online adaptation. After executing the selected policy, HiP3 observes the immediate environment reward r_t and updates only the statistics of the selected arm:

$$\begin{aligned} A_{k_t} &\leftarrow A_{k_t} + \eta_B s_t s_t^T, \\ b_{k_t} &\leftarrow b_{k_t} + \eta_B r_t s_t, \end{aligned} \tag{11}$$

where $k_t = \arg\max_k P_t(k)$ and $\eta_B = 0.05$ is the update rate used in our implementation to stabilize the online adaptation of the bandit statistics.

We use the observed environment reward as direct target-task feedback, avoiding additional model- or critic-based value estimation, while long-horizon policy improvement is handled separately by the SAC updates during online fine-tuning. The selector is applied throughout the 100K-step online fine-tuning stage used in our experiments. A comparison with a nonlinear KernelUCB selector is reported in Appendix D.5.2, showing that increasing selector capacity does not improve performance under the same online protocol.

C Experimental Setup

C.1 MuJoCo Simulation Environment

We conduct the experiments on continuous control tasks using the MuJoCo [53] simulation environment. For offline pre-training, we use the standard D4RL Gym benchmark [9], which provides datasets with diverse qualities. Our HiP3 and all the baseline methods are tested on three tasks: Walker2d, Hopper, and HalfCheetah. For each task, these algorithms are evaluated on datasets with four different quality levels: random-v2, medium-v2, medium-replay-v2, and medium-expert-v2. These datasets are generated as follows. "Random" is generated by collecting 1 million steps of rollouts from a randomly initialized policy. "Medium" is generated by collecting 1 million steps of rollouts from a partially trained SAC. "Medium-replay" is generated by recording all samples in the replay buffer observed during training until the policy reaches a medium level of performance. "Medium-expert" is generated by mixing 1 million steps of expert-level data with 1 million steps of sub-optimal data.

In the online stage, we categorize the experiments into two types of online tasks: the same online tasks and novel online tasks. For the former, the same offline task environments are used for the online tasks, while for the latter we introduce changes in both dynamics and reward structure, following common practices for inducing distribution shifts in continuous control benchmarks. Specifically, we modify the environments as follows:

- Gravity increased from $9.8 \rightarrow 11.5$
- Friction coefficients scaled by $1.2\times$
- Forward-reward weight scaled by 0.85
- Control-cost weight increased to 0.2
- Noise scales are set to 0.2, 0.1, and 0.05 for HalfCheetah, Hopper, and Walker2d, respectively.

These modifications introduce noticeable distribution shifts and increase task difficulty compared to the standard settings.

C.2 Implementation Details

We first describe the baseline implementations. Cal-QL [2], AWAC [20], IQL [40] and CQL [11] are evaluated in the offline-to-online settings provided by CORL [41], using the official CORL implementations. For MBPO [15], PEX [3], WSRL [24], SO2 [27], FQL [30] and EDIS [29], we use the official implementations released by their authors. When conducting the comparison for PEX, we employed the same O2O setting provided by the official code. For MBPO, we learned the policies from scratch with the hyper-parameters provided by the official implementation. For FamO2O [23], we have referenced their published results, which involved running 1M steps for online fine-tuning.

All O2O methods except FamO2O are pretrained for 1M offline steps and then fine-tuned for 100K online steps. MBPO is trained from scratch for 100K steps. For HiP3, candidate initialization evaluates each offline policy with one target-task rollout and selects the top-K candidates, using a rollout horizon of at most 500 steps per policy. The ranking trajectories are used only for candidate initialization and are not added to the online replay buffer or used for policy or model updates. The resulting candidate set is then fine-tuned for 100K online steps following the procedure described in Section 4. The reported 100K-step budget counts only environment transitions used for online fine-tuning. The one-time rollout-based candidate ranking uses additional target-environment interactions outside this fine-tuning budget.

C.3 Computing Infrastructure

Given the large number of experiments, our experiments were conducted using a mix of GPUs, including NVIDIA RTX-3090, NVIDIA V100, NVIDIA A100, and NVIDIA A6000. NVIDIA RTX-3090 was run on Ubuntu operating system, while the other GPUs used CentOS. The software dependencies, implementation details, and instructions for reproducing our experiments are provided in the README of our public code repository.

C.4 Computational Cost

HiP3 constructs multiple policies in the offline stage, but this does not multiply the total offline optimization budget by the number of policies. All three matched variants use the same total 1M-step offline policy-optimization budget and 100K-step online fine-tuning budget. For HiP3 and HiP3-LS, the offline budget is distributed across the policy pool rather than assigned independently to each policy. This matched protocol allows us to separate differences in wall-clock cost from differences in the total amount of policy optimization.

Table 7 reports the end-to-end wall-clock cost of the three variants under the same model-based backbone, rollout configuration, hardware, and optimization budgets. The single-policy variant requires 5.8h for offline training and 8.4h for online fine-tuning, for a total of 14.2h. HiP3-LS increases the offline time to 6.3h while keeping the online time unchanged at 8.4h. Since the total offline optimization budget is fixed, this increase reflects the additional computation associated with maintaining and optimizing multiple candidate policies rather than additional policy-optimization steps. Full HiP3 requires 7.1h offline and 8.5h online, for a total of 15.6h. Relative to HiP3-LS, the additional offline computation is associated with MOSAIC: HiP3-LS uses fixed linear scalarization, whereas MOSAIC additionally evaluates reference alignment and combines multiple objective gradients through MGDA during optimization.

MOSAIC further expands the policy pool through warm-started neighborhood search rather than training every additional candidate from scratch. After optimizing a policy for a reference vector, HiP3 reuses the resulting policy as the initialization for nearby reference vectors. This allows additional trade-off regions to be populated without linearly increasing the optimization budget with the number of candidate policies. The total offline policy-optimization budget therefore remains fixed at 1M steps.

At the online stage, HiP3 fine-tunes only the policy selected at the current interaction step rather than updating all candidate policies. The policy pool therefore does not multiply the online policy-optimization cost. LinUCB is computationally lightweight, maintaining a matrix $A \in \mathbb{R}^{n\times d\times d}$ and a vector $b \in \mathbb{R}^{n\times d}$, where $n < 10$ is the number of candidate policies and $d < 50$ is the observation dimension. Consistent with this design, the online wall-clock times of the three matched variants are nearly identical in Table 7. The reported online time for HiP3 also includes the one-time

rollout ranking used to initialize the candidate set, which takes less than 10 minutes in our MuJoCo experiments on the same NVIDIA A6000 setup.

Variant	Offline Time (h)	Online Time (h)	End-to-End Time (h)
Single-policy	5.8h	8.4h	14.2h
HiP3-LS	6.3h	8.4h	14.7h
HiP3	7.1h	8.5h	15.6h

Table 7: End-to-end computational cost of matched HiP3 variants. All variants use the same total 1M-step offline policy-optimization budget and 100K-step online fine-tuning budget. Wall-clock times are averaged over 3 seeds on a common NVIDIA A6000 GPU.

For additional runtime context, Tables 8 and 9 report stage-wise wall-clock times for representative baselines evaluated on the same NVIDIA A6000 platform. These measurements provide absolute resource references across different algorithmic implementations, while Table 7 provides the controlled comparison used to isolate the computational cost introduced by HiP3 itself.

Method	HiP3 (offline)	Cal-QL (offline)	PEX (offline)	MOPO (offline)
Hopper	7.5h	9.2h	2.9h	5.9h
HalfCheetah	8.3h	8.7h	4.0h	7.0h
Walker2d	5.6h	7.5h	2.7h	5.2h

Table 8: Computation cost comparison for offline stage

Method	HiP3 (online)	MBPO (online)
Hopper	7.7h	7.8h
HalfCheetah	8.5h	8.9h
Walker2d	9.4h	9.7h

Table 9: Computation cost comparison for online stage

C.5 The Hyper-parameters of HiP3

For the training of HiP3 on the MuJoCo tasks, the hyper-parameters for the offline and the online stages can be found in Table 10(a) and Table 10(b), respectively.

D Additional Empirical Results

D.1 Generality of Optimism-Pessimism Crossover

Figure 1 provides an illustrative example showing that optimistic policies can adapt better to novel online tasks, while pessimistic policies can perform better when the online task remains close to the offline distribution. To examine whether this phenomenon is specific to a single setting, we conduct a controlled study across four representative offline datasets.

We consider two MuJoCo domains (HalfCheetah and Hopper) and two offline dataset qualities (random and medium-replay). For each offline policy pool, we identify an optimistic and a pessimistic policy according to their optimism and pessimism objectives, respectively. The two policies are selected before observing the online task and are then independently fine-tuned for 100K online steps using the same online optimization procedure. No Top-K ranking, LinUCB selection, or policy switching is applied in this analysis.

Hyper-parameter	Value
Number of hidden units	256
Number of hidden layers	2
Optimizer	Adam
Rollout length	Walker2d: 1
	HalfCheetah: 3
	Hopper: 5
Rollout batch size	50
Updates per step	1
Autotune SAC α	True
SAC α learning rate	3×10^{-4}
Max SAC α	1.0
Uncertainty-squashing constant κ	1.5
Learning rate	3×10^{-4}
Discount factor	0.99
Reference-vector similarity threshold ψ	0.999
Target update rate	5×10^{-3}
Batch size	256
Number of reference vectors	5
Environment steps	$5 \times (180K + 2 \times 10K)$
(τ_a, τ_b) of reference vectors	$(0.1, 0.9)$
Number of offline policies	100

(a) Hyper-parameters for offline RL

Hyper-parameter	Value
Number of hidden units	256
Number of hidden layers	2
Optimizer	Adam
Rollout length	1
Rollout batch size	5×10^4
Model training frequency	500
Model retain epochs	5
Updates per step	15
Autotune SAC α	True
SAC α learning rate	3×10^{-4}
Max SAC α	1.0
Learning rate	3×10^{-4}
Discount factor	0.99
Target update rate	5×10^{-3}
Batch size	256
Number of epochs	100
Environment steps per epoch	1000
Number of online policies	5

(b) Hyper-parameters for online RL

Table 10: The hyper-parameters of HiP3 used for offline and online learning on the MuJoCo tasks.

Offline Dataset	Same Online Task		Novel Online Task	
	Optimistic	Pessimistic	Optimistic	Pessimistic
halfcheetah-r	48.4	**56.1**	**57.6**	12.8
halfcheetah-mr	56.2	**85.9**	**54.5**	53.0
hopper-r	45.7	**100.6**	**74.2**	31.6
hopper-mr	78.6	**102.2**	**86.5**	37.7
mean	57.2	**86.2**	**68.2**	33.8

Table 11: Generality of the optimism-pessimism crossover behavior. Optimistic and pessimistic policies are selected before observing the online task and independently fine-tuned for 100K online steps without Top-K ranking, LinUCB selection, or policy switching. The results show a consistent crossover between adaptation on the same and novel online tasks.

Across all four settings, the relative ordering consistently reverses after introducing the online distribution shift: pessimistic policies perform better on the same online tasks, while optimistic policies achieve higher performance on the novel online tasks. These results suggest that the preferred optimism-pessimism trade-off depends on the online task rather than being fixed during offline training.

D.2 Results on Additional Domains

We further evaluate HiP3 on additional domains beyond standard D4RL locomotion tasks, including high-dimensional manipulation (Adroit) and navigation (Maze2D). As shown in Table 12, HiP3 achieves the highest mean across the two Adroit tasks and the highest Maze2D score among the compared O2O baselines. These results extend the empirical evaluation to manipulation and navigation settings.

Domain	Task	HiP3	Cal-QL	AWAC	CQL
Adroit	pen-human-v1	35.8±10.2	27.6±24.8	**36.6±10.6**	1.2±5.0
	pen-cloned-v1	**54.6±4.7**	-3.2±0.1	47.3±11.7	13.2±6.4
	mean	**45.2±7.5**	12.2±12.5	42.0±11.2	7.2±5.7
Maze2D	medium-dense-v1	**142.4±21.2**	77.3±21.2	48.7±26.4	59.2±24.1

Table 12: Generalization to additional domains. We evaluate HiP3 on high-dimensional manipulation tasks (Adroit) and navigation tasks (Maze2D), and compare with widely-used O2O RL baselines (Cal-QL, AWAC, CQL). HiP3 achieves competitive or superior performance across domains.

D.3 Transient Performance Degradation and Recovery Analysis

We further analyze the transient degradation observed during early online fine-tuning. This analysis complements the relative performance-drop comparison in Table 13 by measuring the absolute minimum performance and recovery behavior.

Method	HiP3	MOPO+MBPO	Cal-QL
halfcheetah-r-v2	8.3	18.7	17.2
halfcheetah-mr-v2	7.6	15.0	11.6
hopper-r-v2	14.6	2.8	3.4
hopper-mr-v2	20.4	28.5	21.5
mean	12.7	16.3	13.4

(a) Novel online tasks

Method	HiP3	MOPO+MBPO	Cal-QL
halfcheetah-r-v2	1.2	3.6	2.1
halfcheetah-mr-v2	1.4	9.8	4.8
hopper-r-v2	12.1	1.5	4.2
hopper-mr-v2	16.7	23.1	2.6
mean	7.9	9.5	3.4

(b) Same online tasks

Table 13: Relative performance drop comparison on same and novel online tasks. The drop is measured between the pre-fine-tuning performance and the minimum performance during online adaptation. (Notations: r = random, mr = medium-replay.)

For each run, we define the minimum performance as the lowest normalized score observed during online fine-tuning. The recovery time is defined as the first checkpoint after degradation where the performance returns to the pre-fine-tuning level.

Task	Minimum Normalized Score	Recovery Step, Median [Range]
Same-HC-r	33.2 ± 1.3	$8K$ $[7K, 9K]$
Same-HC-mr	46.6 ± 15.7	$11K$ $[7K, 18K]$
Same-Hop-r	14.5 ± 1.0	$24K$ $[15K, 31K]$
Same-Hop-mr	43.0 ± 5.9	$18K$ $[12K, 28K]$
Novel-HC-r	27.0 ± 2.8	$7K$ $[6K, 10K]$
Novel-HC-mr	39.3 ± 21.2	$9K$ $[7K, 12K]$
Novel-Hop-r	11.2 ± 1.8	$26K$ $[13K, 38K]$
Novel-Hop-mr	27.7 ± 8.6	$17K$ $[10K, 24K]$

Table 14: Absolute transient degradation and recovery analysis during online fine-tuning. Minimum normalized scores are averaged over three seeds. Recovery steps are reported as median [minimum, maximum] across seeds.

Across the 24 runs, the median recovery time is 12K online steps: 18/24 runs recover within 20K steps, and all runs recover within 38K steps. The degradation is generally smaller on HalfCheetah tasks, whereas Hopper tasks exhibit larger and longer transient drops.

These results suggest that although HiP3 achieves strong final adaptation performance, practical deployment may require safeguards during early fine-tuning, such as retaining the offline policy as a fallback or delaying aggressive online updates.

D.4 Hyperparameter Sensitivity Analysis

We further evaluate the sensitivity of HiP3 to its method-specific hyperparameters, including the uncertainty-squashing constant κ, reference-vector range (τ_a, τ_b), perturbation size ϵ, and similarity threshold ψ.

All experiments use the same protocol as the main experiments, and only one hyperparameter is changed at a time while keeping all other configurations fixed. We evaluate representative novel online tasks across different domains and report the mean normalized return. The default configurations are shared across all evaluation tasks without task-specific adjustment.

Sweep	Value	HC-r	HC-mr	Hop-r	Hop-mr	Mean
κ	0.5	66.8±3.0	66.4±3.9	92.2±5.1	94.6±8.3	80.0
κ	**1.5**	**63.5±2.2**	**69.4±3.6**	**95.2±5.5**	**100.1±11.0**	**82.1**
κ	2.5	61.9±0.4	68.7±1.6	96.7±7.3	97.4±10.6	81.2
(τ_a, τ_b)	(0.05,0.95)	64.1±2.6	69.3±2.0	88.0±8.2	97.6±17.5	79.8
(τ_a, τ_b)	**(0.1,0.9)**	**63.5±2.2**	**69.4±3.6**	**95.2±5.5**	**100.1±11.0**	**82.1**
(τ_a, τ_b)	(0.2,0.8)	63.9±2.3	70.2±3.1	89.5±3.5	95.8±16.7	79.9

Table 15: Sensitivity analysis of the uncertainty-squashing constant κ and reference-vector range (τ_a, τ_b). Default configurations are bolded. Each non-default setting changes only the corresponding hyperparameter while keeping all other configurations fixed.

Sweep	Value	HC-r	HC-mr	Hop-r	Hop-mr	Mean
ϵ	0.025	62.7±1.5	67.3±3.6	91.4±5.1	96.0±4.3	79.4
ϵ	**0.05**	**63.5±2.2**	**69.4±3.6**	**95.2±5.5**	**100.1±11.0**	**82.1**
ϵ	0.075	58.7±3.6	72.3±2.5	98.5±5.0	94.2±7.9	80.9
ψ	0.995	64.1±4.8	72.5±3.3	94.3±7.2	94.7±12.1	81.4
ψ	**0.999**	**63.5±2.2**	**69.4±3.6**	**95.2±5.5**	**100.1±11.0**	**82.1**
ψ	0.9995	63.0±2.7	68.7±4.8	90.6±4.6	104.5±6.4	81.7

Table 16: Sensitivity analysis of the perturbation size ϵ and similarity threshold ψ. Default configurations are bolded. The same experimental protocol is used across all settings.

Across all tested hyperparameters, the maximum variation in mean normalized return is 2.7 points, and no task-specific adjustment is performed.

D.5 Complete Benchmark Results and Additional Analyses

This section presents additional empirical analyses in six parts: model and training sensitivity, online policy-selection analysis, complete learning curves on the same and novel online tasks, additional ablations on policy-pool construction and selection, complete benchmark tables, and extended baseline comparisons.

D.5.1 Model and Training Sensitivity

We first examine the sensitivity of HiP3 to world-model training. Table 17 varies the hyper-parameter max-model-epoch-since-update controlling model update epochs ($\mathrm{mmepoch} \in \{1, 5, 10\}$). The default setting ($\mathrm{mmepoch} = 5$) performs comparably to longer training while clearly outperforming the under-trained model.

Additionally, Figure 6 studies the number of candidate policies K. Too many candidates receive insufficient online updates under the limited interaction budget, whereas too few candidates reduce policy diversity. We therefore use $K = 5$, which performs best on most evaluated tasks.

Method	HiP3 (default, mmepoch=5)	HiP3 (mmepoch=1)	HiP3 (mmepoch=10)
halfcheetah-random-v2	63.5±2.2	48.9±3.7	63.1±2.6
halfcheetah-medium-replay-v2	69.4±3.6	56.7±3.1	66.3±2.3
hopper-random-v2	95.2±5.5	25.0±9.8	92.3±9.0
hopper-medium-replay-v2	100.1±11.0	26.1±8.2	100.8±6.9
mean	82.1±5.5	39.2±6.2	80.6±5.2

Table 17: Ablation study on world model training impact

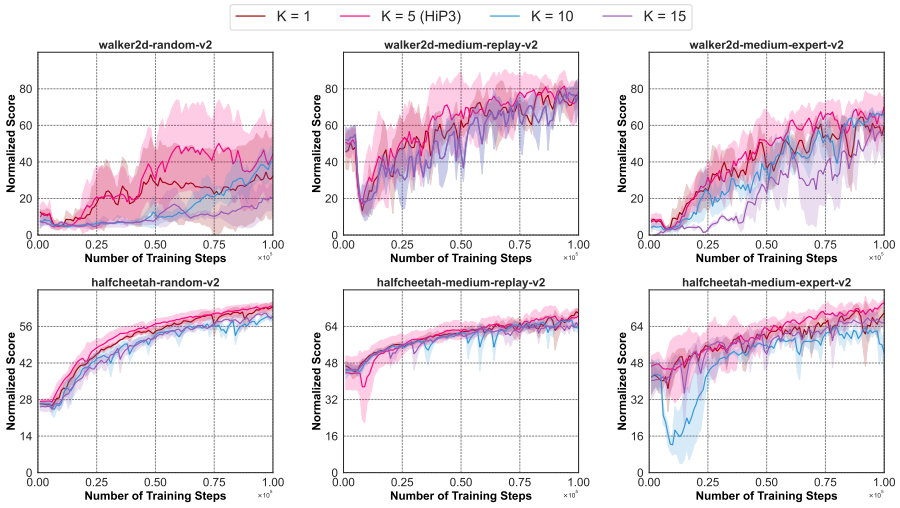


Figure 6: Ablation study on the number of policies K when fine-tuning the policy pool on six novel online tasks. The normalized score is averaged over 5 random seeds (each with 10 evaluations). The curves exclusively illustrate the performance during the online stage, where LinUCB is used for policy selection from the pool.

D.5.2 Online Policy-Selection Analysis

We first examine whether state-dependent policy selection benefits from a more expressive nonlinear selector. Table 18 compares the default linear LinUCB with a nonlinear KernelUCB variant based on random Fourier features. Under the same online adaptation protocol and bandit feedback, LinUCB achieves a mean score of 82.1 compared with 77.8 for KernelUCB, indicating that increasing the selector's function-class capacity does not improve performance in the evaluated settings. We then analyze how the resulting LinUCB preferences evolve during online adaptation. Figure 5 visualizes the policy preferences P_t obtained by softmax-normalizing the LinUCB scores and used for state-dependent policy selection. To summarize executed selections over the full online run, Table 4 reports the dominant-policy selection frequency,

$$\max_k \frac{1}{T} \sum_{t=1}^{T} \mathbf{1}[k_t = k],$$

defined as the fraction of online steps assigned to the most frequently selected candidate.

Table 18: Impact of bandit model capacity on online adaptation performance

Method	HiP3 (LinUCB)	HiP3 (KernelUCB)
halfcheetah-random-v2	63.5±2.2	62.0±2.8
halfcheetah-medium-replay-v2	69.4±3.6	63.5±4.9
hopper-random-v2	95.2±5.5	86.8±11.7
hopper-medium-replay-v2	100.1±11.0	99.0±2.8
mean	82.1±5.5	77.8±5.6

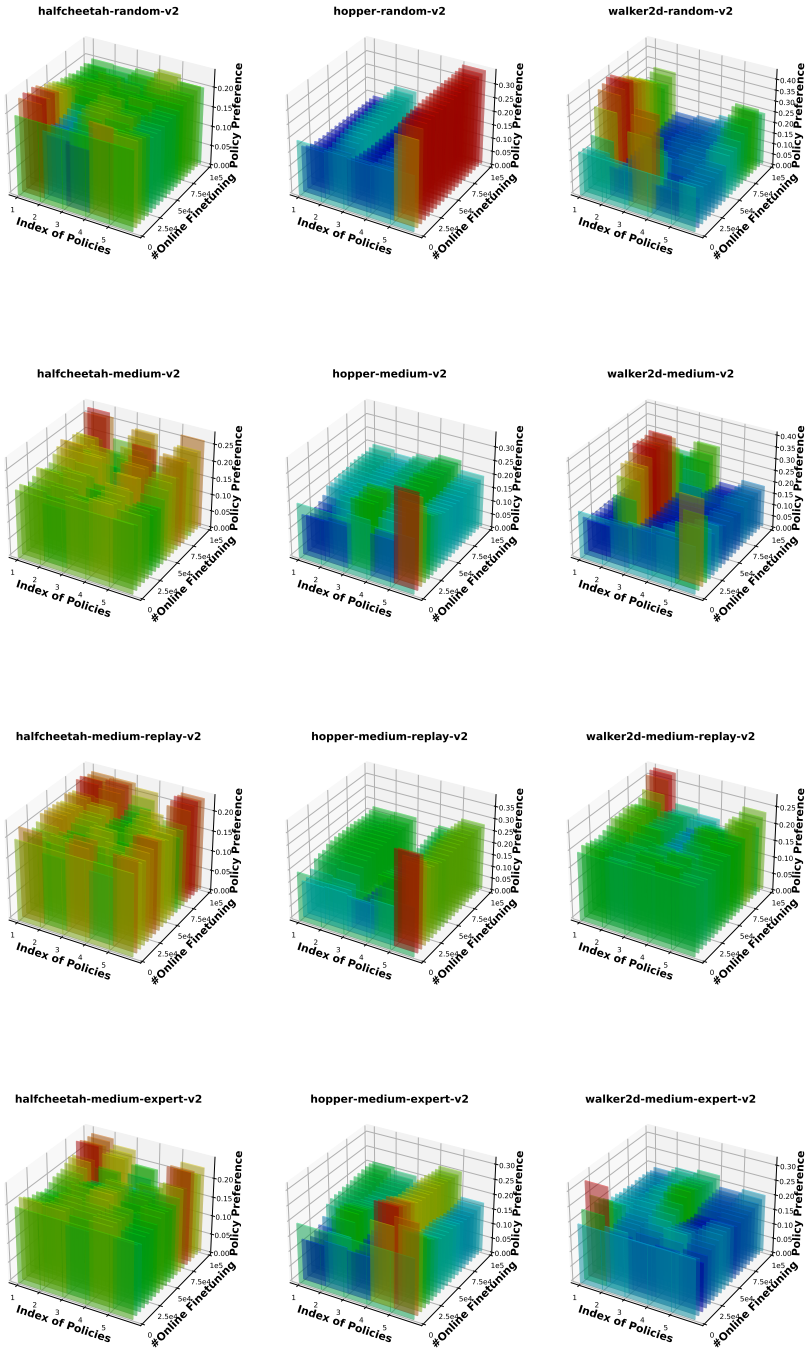


Figure 7: Online policy-selection dynamics across the novel online tasks. The heat maps visualize the policy preferences P_t obtained by softmax-normalizing the LinUCB scores at each online interaction step. The visualizations show whether the selector develops a concentrated preference for one candidate or maintains comparable preferences across multiple candidates during adaptation.

D.5.3 Complete Learning Curves on Same and Novel Online Tasks

We provide the full offline-to-online learning curves on the same and novel online tasks in Figures 8 and 9, respectively. These figures complement the representative plots reported in the main paper and show that the overall trend in favor of HiP3 is consistent across datasets.

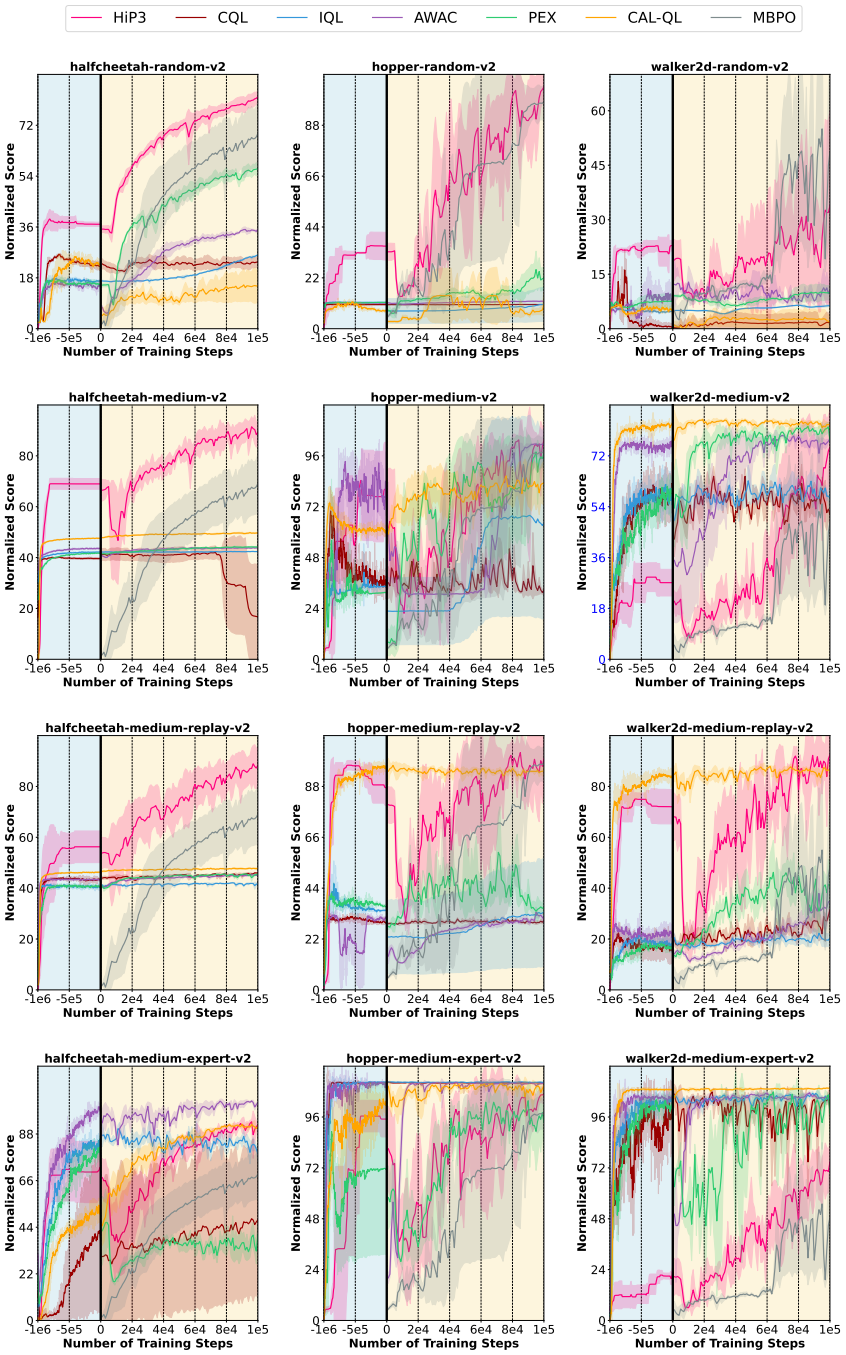


Figure 8: Offline-to-online learning curves on **the same online tasks**. Curves show mean over five seeds; shaded regions denote standard deviation.

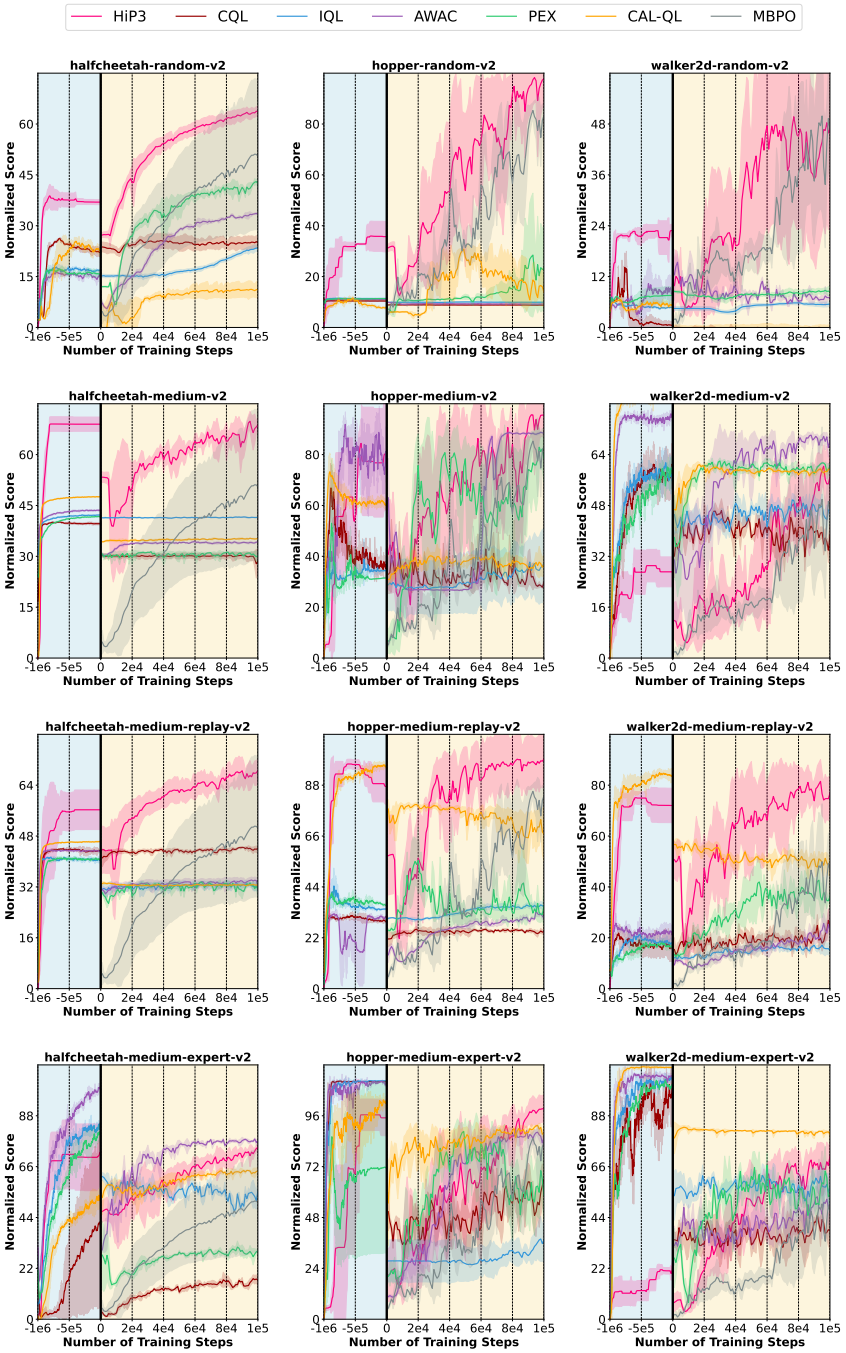


Figure 9: Offline-to-online learning curves on **the novel online tasks**. These tasks are generated via modifying the gravity, noise scales, and reward functions of the original MuJoCo tasks. Curves show mean over five seeds; shaded regions denote standard deviation.

D.5.4 Additional Ablations on Policy Pool and Policy Selection

We next report the complete ablations on policy-pool construction and policy selection. Figure 10 studies the effect of removing the policy pool, while Figure 11 compares different policy-selection algorithms. Table 19 evaluates the number of reference vectors under a fixed offline optimization budget, and Table 20 extends the single-policy comparison to all 12 novel online tasks.

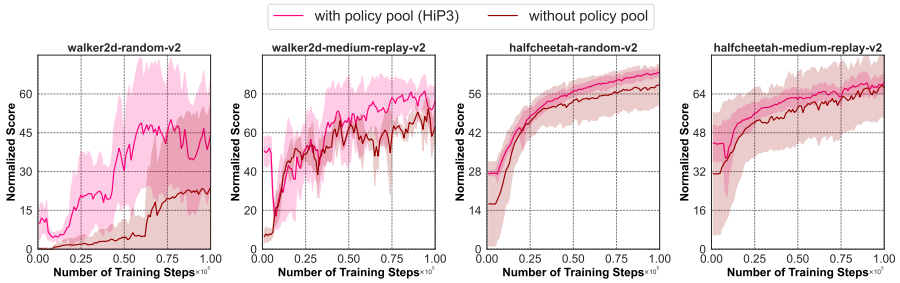


Figure 10: Complete comparison between HiP3 and the single-policy variant on four representative novel online tasks. Curves show mean and standard deviation over five seeds.

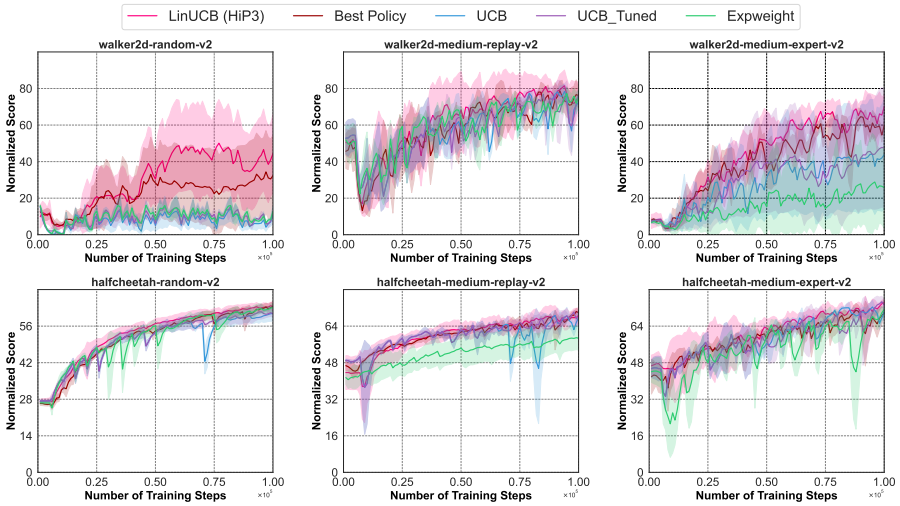


Figure 11: Ablation study on the policy selection algorithms for six novel online tasks. The normalized score is averaged over 5 random seeds (each with 10 evaluations). The curves exclusively illustrate the performance during the online stage, where various bandit algorithms are evaluated.

Reference number	**ref_vec=5 (HiP3)**	**ref_vec=4**	**ref_vec=2**
halfcheetah-r-v2	63.5±2.2	63.3±1.2	63.5±1.5
halfcheetah-mr-v2	69.4±3.6	67.7±4.5	68.4±3.8
hopper-r-v2	95.2±5.5	86.1±8.2	87.5±10.4
hopper-mr-v2	100.1±11.0	90.4±10.7	93.8±12.3
mean	82.1±5.5	76.8±6.2	78.3±7.0

Table 19: Ablation study for different number of reference vectors. For fairness, we fix the total training budget to 1M steps and allocate it evenly across reference vectors. When too many reference vectors are used, each policy receives insufficient training, leading to performance deterioration. Conversely, too few reference vectors reduce the diversity of the policy pool. The ablation study shows that using five reference vectors achieves the best balance between policy diversity and overall performance.

Novel Online Task	HiP3	Single-Policy HiP3
halfcheetah-r-v2	63.5 ± 2.2	57.1 ± 11.8
halfcheetah-medium-v2	69.3 ± 2.1	54.2 ± 8.3
halfcheetah-medium-replay-v2	69.4 ± 3.6	67.9 ± 15.4
halfcheetah-medium-expert-v2	74.7 ± 2.0	70.3 ± 4.9
hopper-r-v2	95.2 ± 5.5	83.2 ± 0.3
hopper-medium-v2	95.4 ± 3.8	88.5 ± 1.0
hopper-medium-replay-v2	100.1 ± 11.0	86.5 ± 3.0
hopper-medium-expert-v2	99.1 ± 7.6	84.0 ± 2.3
walker2d-r-v2	49.2 ± 26.7	23.1 ± 28.6
walker2d-medium-v2	59.2 ± 6.4	47.7 ± 36.4
walker2d-medium-replay-v2	77.3 ± 10.4	61.0 ± 9.7
walker2d-medium-expert-v2	74.6 ± 3.9	43.9 ± 14.6
Mean	$\mathbf{77.3 \pm 7.1}$	64.0 ± 12.9

Table 20: Complete comparison between HiP3 and its single-policy variant on all 12 novel online tasks. The single-policy variant removes the offline Pareto policy pool while retaining the same model-based online fine-tuning procedure. HiP3 uses the five seeds reported in the main results, while the single-policy variant uses three seeds per task. Both methods use the same offline data, environment model, SAC-based backbone, and online fine-tuning budget.

D.5.5 Complete Benchmark Tables

Table 21 reports the complete per-task comparison between HiP3 and the main baselines on the same and novel online tasks. These detailed tables complement the summary table in the main paper and make the performance differences across datasets explicit.

		CQL	IQL	FamO2O*	AWAC	PEX	Cal-QL	MBPO	HiP3
Random	HalfCheetah	23.7 ±2.1	26.4 ±0.1	–	35.6 ±0.5	56.9 ±1.2	15.4 ±5.6	66.0 ±15.4	82.6 ±2.3
	Hopper	10.5 ±0.1	10.4 ±7.3	–	11.7 ±0.2	22.2 ±4.1	12.7 ±8.5	98.2 ±6.1	105.7 ±4.1
	Walker2d	1.7 ±2.7	6.3 ±0.2	–	12.2 ±6.7	9.8 ±2.2	1.6 ±1.3	51.4 ±35.9	35.7 ±26.0
Medium	HalfCheetah	16.7 ±20.7	42.4 ±0.4	53.4	43.6 ±0.8	44.1 ±0.3	49.7 ±0.3	66.0 ±15.4	90.1 ±4.1
	Hopper	33.6 ±3.2	65.4 ±45.7	82.8	101.5 ±0.4	99.4 ±1.2	83.3 ±12.3	98.2 ±6.1	106.7 ±3.0
	Walker2d	49.1 ±8.2	56.0 ±10.2	82.8	79.0 ±1.7	72.0 ±1.2	83.1 ±2.2	51.4 ±35.9	76.4 ±16.0
Medium -replay	HalfCheetah	46.0 ±0.1	41.5 ±0.5	51.0	45.3 ±0.1	45.9 ±0.1	47.8 ±0.1	66.0 ±15.4	89.6 ±2.7
	Hopper	30.5 ±0.9	33.9 ±23.6	92.2	29.6 ±2.8	36.0 ±7.0	95.1 ±1.4	98.2 ±6.1	103.4 ±3.7
	Walker2d	33.8 ±3.6	21.3 ±1.0	88.6	34.5 ±8.4	40.6 ±7.3	85.5 ±4.9	51.4 ±35.9	94.1 ±12.1
Medium -expert	HalfCheetah	48.4 ±31.8	76.7 ±6.8	91.8	103.9 ±1.5	48.3 ±5.6	92.5 ±1.6	66.0 ±15.4	92.8 ±3.4
	Hopper	111.4 ±0.6	112.4 ±0.1	90.1	112.3 ±0.4	89.0 ±31.6	112.2 ±0.2	98.2 ±6.1	106.5 ±4.7
	Walker2d	93.5 ±14.9	105.1 ±2.9	110.6	105.8 ±1.8	89.0 ±18.0	109.6 ±0.7	51.4 ±35.9	79.3 ±10.6
Total	Mean	41.6 ±7.4	49.8 ±8.2	82.6	59.6 ±2.2	54.4 ±6.6	65.7 ±3.3	71.9 ±19.1	88.6 ±7.7

(a) Fine-tuning performance on the same online tasks

		CQL	IQL	AWAC	PEX	Cal-QL	MBPO	HiP3
Random	HalfCheetah	24.7 ±1.1	23.1 ±1.3	33.2 ±0.4	42.2 ±2.6	11.1 ±2.5	47.1 ±22.1	63.5 ±2.2
	Hopper	8.9 ±0.1	9.9 ±0.0	9.2 ±0.1	23.5 ±17.2	11.7 ±4.7	86.4 ±4.6	95.2 ±5.5
	Walker2d	-0.8 ±0.4	5.5 ±0.6	7.0 ±1.3	8.5 ±0.7	-0.6 ±1.1	44.6 ±11.4	49.2 ±26.7
Medium	HalfCheetah	29.3 ±2.5	41.5 ±0.2	33.9 ±0.4	30.4 ±0.9	35.4 ±0.1	47.1 ±22.1	69.3 ±2.1
	Hopper	31.5 ±5.4	38.4 ±12.9	88.5 ±0.1	74.8 ±15.9	37.6 ±4.6	86.4 ±4.6	95.4 ±3.8
	Walker2d	43.1 ±9.7	46.9 ±1.7	70.4 ±0.3	61.5 ±1.0	59.6 ±0.4	44.6 ±11.4	59.2 ±6.4
Medium -replay	HalfCheetah	44.5 ±0.1	32.3 ±2.1	29.9 ±3.8	30.4 ±3.5	32.6 ±0.2	47.1 ±22.1	69.4 ±3.6
	Hopper	24.7 ±1.5	35.2 ±2.0	29.6 ±2.8	39.4 ±14.7	65.9 ±14.0	86.4 ±4.6	100.1 ±11.0
	Walker2d	21.1 ±0.7	14.5 ±1.3	20.0 ±3.2	38.0 ±4.3	46.4 ±3.7	44.6 ±11.4	77.3 ±10.4
Medium -expert	HalfCheetah	16.6 ±2.9	59.3 ±5.3	77.6 ±0.7	29.9 ±1.0	64.4 ±1.9	47.1 ±22.1	74.7 ±2.0
	Hopper	68.7 ±6.8	38.7 ±11.8	88.8 ±4.4	50.9 ±9.5	88.0 ±5.5	86.4 ±4.6	99.1 ±7.6
	Walker2d	38.2 ±9.8	58.8 ±11.4	50.6 ±8.5	54.3 ±2.8	79.2 ±3.4	44.6 ±11.4	74.6 ±3.9
Total	Mean	29.2 ±3.4	33.7 ±4.2	44.9 ±4.9	40.3 ±6.2	44.3 ±3.6	59.4 ±12.7	77.3 ±7.1

(b) Fine-tuning performance on novel online tasks

Table 21: Performance comparison on the same and novel online tasks. The novel online tasks are generated by adjusting the gravity, noise scales, and reward functions of the original MuJoCo environments. $*$ indicates the results reported by the paper. Note that FamO2O reports results with a 1M-step online fine-tuning budget and is therefore not directly budget-matched to the 100K-step setting used for the other methods. We fine-tune all other methods for 100K steps, using the same hyper-parameter settings provided by the original papers. Scores for each dataset are normalized according to the D4RL benchmark [9]: **Normalized Score** $= 100 \times \frac{\text{return} - \text{random return}}{\text{expert return} - \text{random return}}$.

D.5.6 Extended Baseline Comparisons

Finally, we include additional comparisons with recent model-based reinforcement learning baselines that are relevant to HiP3. Table 22a compares HiP3 with FOWM [54] on the same online tasks, while Table 22b compares HiP3 with TD-MPC2 on novel online tasks using a 100K-step online training horizon.

As shown in Table 22a, HiP3 outperforms FOWM on the Hopper tasks, even though FOWM is fine-tuned for 500K online environment steps while HiP3 uses only 100K. For TD-MPC2 [31], we evaluate all runs for 100K online training steps, following the limited-interaction setting used in our main evaluation. We set the model size to 5, following the recommendation in the TD-MPC2 paper, and use the default values for the remaining hyper-parameters. Table 22b shows that HiP3 substantially outperforms TD-MPC2 under this limited-budget setting.

Task	HiP3	FOWM*
hopper-medium-v2	**106.7**	100.7
hopper-medium-replay-v2	**103.4**	93.5
mean	**105.1**	97.1

(a) Same online tasks with FOWM

Task	HiP3	TD-MPC2
halfcheetah	**69.2**	37.3
hopper	**97.5**	3.1
walker2d	**65.1**	2.8
mean	**77.3**	14.4

(b) Novel online tasks with TD-MPC2

Table 22: Comparisons with model-based reinforcement learning baselines. * indicates results reported by the paper. FOWM results are reported from its paper; TD-MPC2 is evaluated for 100K online training steps.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the main contribution as HiP3. These claims are supported by the method section, the main benchmark results, and the ablation studies.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper discusses limitations in the conclusion section.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The theoretical result is stated together with its assumptions, and the full proof is provided in Appendix B.1. In particular, Theorem B.1 gives the convergence statement for MOSAIC and is cross-referenced from the main method section.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The paper specifies the tasks, baselines, normalization rule, and evaluation protocol in the main text, and Appendix C provides environment details, implementation details, compute setup, and hyperparameters. Full benchmark tables and additional ablations are also included in the appendix.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We use public benchmark datasets and official baseline implementations. The data sources and baseline implementations are identified in the experimental setup and appendix.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The main paper gives the experimental protocol, baseline categories, and training budgets, while Appendix C details dataset construction, environment modifications, implementation details, and HiP3 hyperparameters. This covers the information needed to understand and reproduce the reported results.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: The main results and appendix report means over 5 random seeds, and the figures and tables specify that the shaded regions or $\pm$ values denote standard deviation. The captions also state the evaluation protocol used for these aggregated results.

   Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Appendix C.3 reports the hardware and operating systems used, and Appendix C.4 reports runtime comparisons for the offline and online stages.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work is conducted on standard simulator benchmarks and does not involve human subjects, personal data, or safety-critical deployment. Potential limitations and societal impacts are explicitly discussed, and we are not aware of any aspect of the work that conflicts with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [Yes]

Justification: The broader impacts are discussed in Appendix A. We describe potential benefits from reducing unnecessary exploration and improving sample efficiency, and we also note risks from early-stage instability and deployment under severe distribution shift.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: The paper does not release high-risk generative models, scraped datasets, or other assets with an obvious misuse profile of this kind.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We use public benchmark datasets and official baseline implementations, and we credit these assets through citations in the experimental setup and appendix.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The new asset is the implementation of HiP3, which is documented in the accompanying source-code package. The paper and appendix also provide the algorithmic description, hyperparameters, and experimental details needed to understand and use the implementation.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The work does not involve crowdsourcing or human-subject experiments.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The work does not involve human subjects or participant data.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: LLMs are not part of the proposed method, the training pipeline, or the experimental methodology.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.