Foundation Pareto Flow Policy for Multi-Objective Reinforcement Learning

Zhanjiang Yang[1,2*] **Lijun Sun**[1*] **Yueming Li**[1] **Meng Li**[1] **Yijun Yang**[3†]
[1]Shenzhen Technology University [2]Shenzhen University [3]Tsinghua University
2410263030@mails.szu.edu.cn sunlijun@sztu.edu.cn
202301510025@stumail.sztu.edu.cn limeng2@sztu.edu.cn
yijun.steven.yang@gmail.com

Abstract

The overarching goal of reinforcement learning (RL) is to efficiently optimize the agent's policy for various environments, tasks, dynamics, and even user preferences, which often consist of multiple and potentially conflicting rewards/objectives. This necessitates the learning of Pareto-optimal policies that can navigate different trade-offs among objectives. However, existing multi-objective RL (MORL) methods, whether trained online or offline, are typically limited to task- or objective-specific policy learning and do not address generalization of learned trade-offs to unseen tasks/objectives. In this work, we propose the **F**oundation **P**areto **F**low **P**olicy (**FP2**) for offline MORL in continuous robotic control, inspired by the pre-mid-post-training paradigm underlying recent foundation models. Specifically, we instantiate FP2 as a multi-modal diffusion transformer equipped with a likelihood-free flow matching loss, enabling unified, scalable, and instructable learning of diverse trade-off trajectories. We further develop a three-stage training pipeline to gradually optimize the policy: (1) pre-training on a large-scale MORL dataset, (2) mid-training on a carefully curated high-quality dataset, and (3) post-training on few collected trajectory data from new tasks and trade-offs. Empirical results demonstrate that FP2 outperforms previous state-of-the-art methods on most tasks of the D4MORL benchmark, achieving superior zero-shot interpolation and extrapolation on complex Pareto manifolds as well as efficient few-shot adaptation of the learned trade-off prior to new continuous-control tasks. Code is available at `https://github.com/FibonacciRabbit/FP2`.

1 Introduction

Multi-objective reinforcement learning (MORL) Roijers et al. [2013], Roijers and Whiteson [2017] solves real-world decision-making that involves multiple, often conflicting objectives Vamplew et al. [2022] through learning Pareto-optimal policies capturing diverse trade-offs among objectives. The preferred trade-off of a decision-making task, such as maximizing antibacterial activity while minimizing toxicity and production costs in drug design Stokes et al. [2020] or balancing competing objectives in complex continuous robot control Xu et al. [2020], Zhu et al. [2023], is typically unpredictable and can evolve over time. For example, faster robotic movement may be preferred when energy is abundant, whereas careful, energy-efficient strategy is required when energy is limited in locomotion tasks. Hence, MORL needs to learn Pareto policies that can handle different tasks, scenarios, and preferences, as illustrated by the proof-of-concept result in Fig. 1.

*Equal contribution.
†Corresponding author.

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

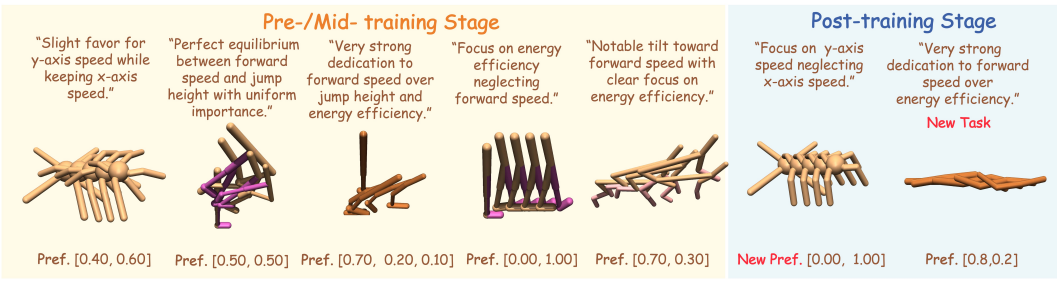


Figure 1: **F**oundation **P**areto **F**low **P**olicy (**FP2**) for MORL. FP2 is a rectified flow-based policy conditioned on numerical and textual user preferences. Through principled pre- and mid-training on diverse datasets (yellow region), FP2 learns a reusable multi-objective trade-off prior that generalizes to out-of-distribution preferences and can be efficiently adapted to new locomotion tasks (blue region) via few-shot post-training.

To this end, we introduce a generalizable offline MORL policy, named **F**oundation **P**areto **F**low **P**olicy (**FP2**), which is built upon the multi-modal diffusion transformer (MMDiT) Peebles and Xie [2023] equipped with a rectified flow matching loss Esser et al. [2024], Liu et al. [2023], Ma et al. [2025], Black et al. [2024], enabling unified and instructable sequence modeling of diverse Pareto behavior trajectories. Furthermore, inspired by the pre-training–post-training paradigm underlying recent foundation models, we develop a three-stage training pipeline to gradually optimize the policy FP2 from rich yet suboptimal trajectory data to scarce but high-quality data. As shown in Fig. 2, we pre-train FP2 in Stage 1 on a large-scale, diverse, but noisy dataset including trajectory data accessible from several offline MORL datasets. In Stage 2, we design a novel mid-training stage that continuously fine-tunes the pre-trained policy (i.e., FP2-Pre) using a curated set of high-quality Pareto-optimal trajectories, sharpening the policy's action distribution for Pareto-front behaviors while improving policy–preference alignment. This stage mitigates the scarcity of expert data by amplifying their supervisory effect for learning fine-grained representations, leading to improved performance. In Stage 3, we post-train the FP2-Mid policy with a very small amount of task-specific data to efficiently adapt it to unseen preference configurations or environment dynamics. Moreover, to boost the policy's robustness to noisy and suboptimal data that exists in commonly-used MORL datasets, FP2 introduces a contrastive alignment regularization to effectively overcome the potential misalignment between numerical/textual preferences and practical behavior trajectories. In experiments, we conduct extensive comparison with strong MORL baselines on the D4MORL benchmark Zhu et al. [2023], demonstrating the superiority of FP2 in terms of zero-shot interpolation and extrapolation on complex Pareto manifolds, as well as few-shot adaptation of a shared multi-objective trade-off prior across held-out environments.

Our contributions are summarized below:

- A pre-trained offline MORL policy that provides a reusable multi-objective trade-off prior for continuous control, generalizing zero-shot to unseen preferences and few-shot transferring of this prior to new control tasks.
- A principled pipeline including pre-, mid-, and post-training that takes the same spirit as mainstream training recipes adopted by advanced foundation models.
- To the best of our knowledge, FP2 is the first offline MORL method that exhibits excellent interpolation/extrapolation abilities on complex Pareto manifolds and transfer of learned multi-objective trade-off structure across locomotion tasks.

2 Related work

Multi-objective optimization (MOO). MOO seeks Pareto-optimal solutions balancing multiple, often conflicting objectives. These solutions form the Pareto set of undominated decisions in the decision space and induce the Pareto front in the objective space Felten et al. [2024], Hayes et al. [2022], Chen et al. [2025]. Existing MOO methods can be broadly categorized by whether preferences are specified *a priori*. When preferences are known, a single Pareto-optimal solution can be obtained by optimizing the corresponding scalarized objective, obviating the need to recover the entire Pareto

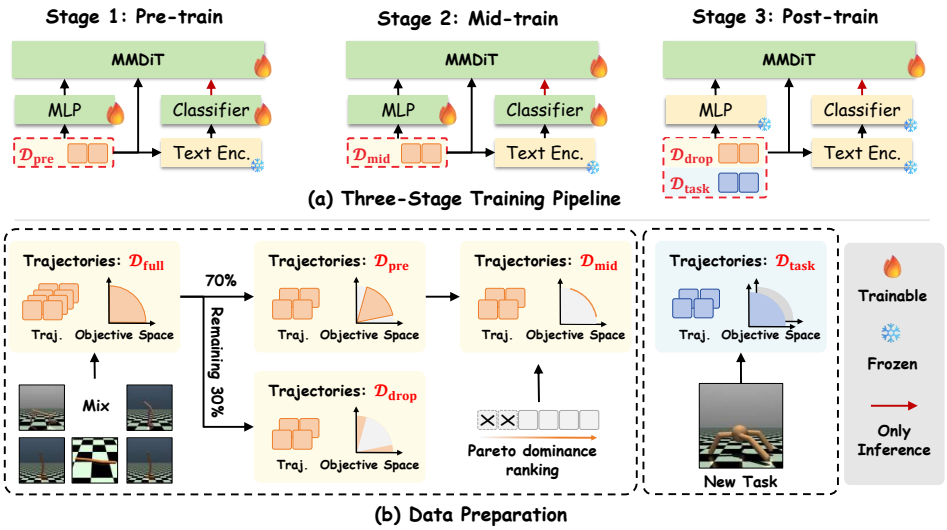


Figure 2: Three-stage training pipeline of **F**oundation **P**areto **F**low **P**olicy (**FP2**). In Stage 1, FP2 is pre-trained to learn diverse long-horizon trade-off behaviors based on large-scale trajectories $\mathcal{D}_{\text{pre}}$. In Stage 2, FP2 is continuously mid-trained to align policy behavior with the target preference using curated high-quality Pareto non-dominated trajectories $\mathcal{D}_{\text{mid}}$. In Stage 3, FP2 can be further post-trained to efficiently adapt to unseen preference configurations or new tasks with the remaining trajectories $\mathcal{D}_{\text{drop}}$ or a very small amount of new task data $\mathcal{D}_{\text{task}}$. (See Sec. 4.3 for more details.)

set Roijers et al. [2013], Van Moffaert et al. [2013], Roijers et al. [2018]. When preferences are unknown, two classes of approaches enable preference-specific solutions at test time: finite Pareto-set methods and continuous Pareto-set methods. Finite Pareto-set methods learn a collection of policies that span diverse objective preferences, yielding a discrete approximation of the Pareto set Xu et al. [2020], Yang et al. [2021], Janmohamed et al. [2024], Liu et al. [2024]. Continuous Pareto-set methods model a preference–solution mapping that captures the underlying continuous Pareto structure through a unified preference-conditioned policy Yuan et al. [2024], Du et al. [2025].

MORL builds upon the MOO framework by incorporating sequential decision-making. In such settings, meaningful behaviors arise only through cumulative trade-offs over an entire rollout, inducing complex temporal correlations and making Pareto-optimal policy learning fundamentally more difficult Annadani et al. [2025].

Generative models for offline MORL. MORL is generally framed as a sequential decision-making problem involving state transitions. Traditional RL algorithms typically learn policies using value-based or policy-based methods Sutton et al. [2018], Lin [2005], Mossalam et al. [2016], Yang et al. [2019], Xu et al. [2020], Liu et al. [2025], Li et al. [2026a,b]. Recently, conditional generative models have emerged as an alternative paradigm for offline decision-making by formulating policy learning as return-conditioned sequence modeling. These approaches can be broadly categorized into two classes. The first class consists of autoregressive sequence modeling methods, such as Decision Transformer (DT) Chen et al. [2021] and Prompt-DT Xu et al. [2022], which model the joint distribution over states, actions, and returns and generate actions autoregressively conditioned on the return. The second class comprises non-autoregressive trajectory generation methods, typically based on diffusion models, such as Diffuser Janner et al. [2022] and Decision Diffuser (DD) Ajay et al. [2023]. They jointly model state dynamics and action distributions through a unified sampling-and-planning process, offering a flexible framework for incorporating environment- and task-level constraints beyond reward signals. However, most generative approaches have focused on single-objective RL settings.

In the context of MORL, PEDA Zhu et al. [2023] extends single-objective return-conditioned models to multi-objective settings and introduces preference–return–conditioned generative methods, including MODT(P) and MORvS(P). MODULI Yuan et al. [2024] further improves generalization to out-of-distribution (OOD) preferences. Despite these advances, generalization to unseen preference configurations remains limited, and learned multi-objective trade-off behaviors do not readily transfer across heterogeneous environment dynamics. In this direction, and in line with recent efforts to bring foundation models to embodied decision-making Yang et al. [2024], Li et al. [2024], our work advances diffusion-based generative policies toward a foundation-level Pareto policy, enabling a

transferable multi-objective trade-off prior that improves generalization to OOD preferences and few-shot adaptation to unseen continuous-control tasks.

3 Preliminaries

Multi-objective reinforcement learning (MORL). In MORL, the underlying multi-objective sequential decision problem is typically modeled as a multi-objective Markov decision process (MOMDP) $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \Omega \rangle$, where $\mathcal{S} = \{\boldsymbol{s}_t\}$ is the state space; $\mathcal{A} = \{\boldsymbol{a}_t\}$ is the action space; $\mathcal{P}(\boldsymbol{s}_{t+1} \mid \boldsymbol{s}_t, \boldsymbol{a}_t)$ is a transition function mapping the current state-action pair $(\boldsymbol{s}_t, \boldsymbol{a}_t)$ at time t to a distribution over the next state $\boldsymbol{s}_{t+1}$ at time $t+1$; $\boldsymbol{r}_t = \mathcal{R}(\boldsymbol{s}_t, \boldsymbol{a}_t) \in \mathbb{R}^{D_o}$ is the vector-valued reward function and D_o is the number of objectives; $\gamma \in [0, 1)$ is a discount factor that controls the relative importance of future rewards; $\Omega = \{\, \boldsymbol{\omega} = [\omega_i]_{i=1}^{D_o} \in \mathbb{R}^{D_o} \mid \omega_i \in [0, 1],\ \sum_{i=1}^{D_o} \omega_i = 1 \,\}$ is the normalized preference space.

The goal of preference-conditioned MORL is to learn a policy $\pi(\boldsymbol{a}_t \mid \boldsymbol{s}_t, \boldsymbol{\omega})$ that maximizes the expected return vector $\boldsymbol{G}^\pi = \mathbb{E}_{\boldsymbol{a}_t \sim \pi}[\sum_{t=0}^{T} \gamma^t \boldsymbol{r}_t]$, where T denotes the episode termination time. A policy π_p is *Pareto-optimal (undominated)* if there exists no policy π_q such that $\boldsymbol{G}_i^{\pi_q} \geq \boldsymbol{G}_i^{\pi_p}$ for all i and $\boldsymbol{G}_j^{\pi_q} > \boldsymbol{G}_j^{\pi_p}$ for at least one j. The set of all policies defines the *decision space*, and the set of all achievable return vectors defines the *objective space*. The *Pareto set (PS)* is the set of Pareto-optimal policies, and the corresponding set of objective vectors forms the *Pareto front* ($\mathcal{F}$).

Rectified flow (RF) is a kind of *flow matching* Liu et al. [2022], Albergo and Vanden-Eijnden [2023], Lipman et al. [2023] method. Given a dataset $\mathcal{D}$ consisting of continuous data points $\boldsymbol{x}$ drawn from an unknown target data distribution p_1, RF aims to learn a transformation from a simple distribution p_0 to p_1. This is achieved by modeling the data distribution evolution through an *ordinary differential equation (ODE)* defined over the time domain $t \sim P(t) \in [0, 1]$:

$$\frac{\mathrm{d}\boldsymbol{z}_t}{\mathrm{d}t} = v_\theta(\boldsymbol{z}_t, t), \quad \boldsymbol{z}_0 \sim p_0, \tag{1}$$

where θ denotes the parameters of the velocity neural network $v_\theta(\cdot)$, which is used to characterize the data distribution evolution. In practice, p_0 is typically chosen as a standard Gaussian distribution $\mathcal{N}(0, \boldsymbol{I})$ due to its analytical tractability and ease of sampling.

In particular, the training objective of $v_\theta(\cdot)$ for the interpolated state $\boldsymbol{z}_t = t\boldsymbol{x} + (1-t)\boldsymbol{z}_0$ is:

$$\min_\theta \mathbb{E}_{\substack{\boldsymbol{z}_0 \sim p_0,\, \boldsymbol{x} \sim p_1, \\ t \sim P(t)}} \left[\| v_\theta(\boldsymbol{z}_t, t) - (\boldsymbol{x} - \boldsymbol{z}_0) \|_2^2 \right] \tag{2}$$

Guidance-based conditional generation. Compared to generating arbitrary data from the distribution p_1, conditional generation aims to produce samples consistent with a specified property $\boldsymbol{c}$ and maximize the conditional probability $p(\boldsymbol{x} \mid \boldsymbol{c})$ for each data-condition pair $(\boldsymbol{x}, \boldsymbol{c})$. Two widely used approaches are the *classifier guidance (CG)* Dhariwal and Nichol [2021] and the *classifier-free guidance (CFG)* Ho and Salimans [2022]. CG trains an auxiliary classifier model $C_\phi(\boldsymbol{z}_t, t, \boldsymbol{c}) := \log p_\phi(\boldsymbol{c} \mid \boldsymbol{z}_t, t)$ parameterized by ϕ and updates the vector field with the guidance control weight w^G at inference time:

$$\tilde{v}_\theta(\boldsymbol{z}_t, t \mid \boldsymbol{c}) = v_\theta(\boldsymbol{z}_t, t) + w^G \nabla_{\boldsymbol{z}_t} C_\phi(\boldsymbol{z}_t, t, \boldsymbol{c}) \tag{3}$$

CFG typically learns a single neural network to parameterize both the unconditional model $v_\theta(\boldsymbol{z}_t, t \mid \varnothing)$ and the conditional model $v_\theta(\boldsymbol{z}_t, t \mid \boldsymbol{c})$ by randomly setting $\boldsymbol{c} = \varnothing$ with probability η, where $\varnothing$ denotes the null condition. The training objective is given by:

$$\mathcal{L}_{RF} = \mathbb{E}[\| v_\theta(\boldsymbol{z}_t, t \mid \boldsymbol{c}) - (\boldsymbol{x} - \boldsymbol{z}_0) \|^2] \tag{4}$$

At inference time, the guided vector field is obtained by:

$$\tilde{v}_\theta(\boldsymbol{z}_t, t \mid \boldsymbol{c}) = (1 - w^G) v_\theta(\boldsymbol{z}_t, t \mid \varnothing) + w^G v_\theta(\boldsymbol{z}_t, t \mid \boldsymbol{c}) \tag{5}$$

4 Method

In this section, we introduce the problem setup (Sec. 4.1), network architecture (Sec. 4.2, Fig. 3), training pipeline (Sec. 4.3, Fig. 2), and training objective (Sec. 4.4) of the **F**oundation **P**areto **F**low **P**olicy (**FP2**). Together, these components address the generalization challenges of offline MORL, such as insufficient data coverage in exponentially growing multi-objective space, prevalence of suboptimal or noisy trajectories, and scarce high-quality Pareto-front data.

4.1 Problem setup

We consider an offline MORL setting with a static trajectory dataset $\mathcal{D} = \{\tau^i\}$, where $\tau^i = \{\boldsymbol{\omega}^i, \boldsymbol{s}_t^i, \boldsymbol{a}_t^i, \boldsymbol{r}_t^i\}_{t=0}^T$. The goal is to learn a policy $\pi(\boldsymbol{a}_t \mid \boldsymbol{s}_t, \boldsymbol{\omega}, \boldsymbol{G}_t)$ conditioned on a preference vector $\boldsymbol{\omega}$ and the vector-valued return-to-go (RTG) $\boldsymbol{G}_t = \sum_{k=t}^T \gamma^{k-t} \boldsymbol{r}_k$. The policy can also be viewed as a conditional generative model over the joint distribution of state–action sequences, guided by $\boldsymbol{\omega}$ and the RTG condition.

4.2 Unified framework overview

Generative backbone. Fig. 3 illustrates the overall network architecture of FP2. It adopts a rectified flow-based generative model parameterized by a multi-modal diffusion transformer (MMDiT) Peebles and Xie [2023],which provides a scalable sequence-modeling backbone for continuous trajectory generation. MMDiT is conditioned on three signals: numerical and textual preferences, together with a task-specific return vector.

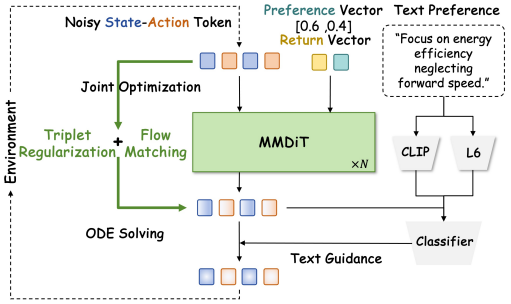


Figure 3: Network architecture of **F**oundation **P**areto **F**low **P**olicy (**FP2**) for MORL. FP2 is a rectified flow-based model with a multi-modal diffusion transformer (MMDiT) backbone that denoises state-action chunks and is guided by multi-modal conditions including numerical and textual preferences and multi-objective returns. FP2 is trained offline and deployed as an online policy.

Dual encoders for numerical and textual preferences. We encode the numerical preference $\boldsymbol{\omega}$ and return vector $\boldsymbol{G}_t$ using a lightweight MLP, and incorporate the resulting tokens via classifier-free guidance (CFG). To enable fuzzy control using textual preferences (e.g., "run aggressively" versus "run economically"), we adopt dual text encoders following prior work Esser et al. [2024], Polyak et al. [2024]. Specifically, we combine all-MiniLM-L6-v2 (L6) Reimers and Gurevych [2019] and CLIP Radford et al. [2021]: L6 captures fine-grained semantic representations, while CLIP emphasizes high-level, action-centered concepts. Here, numerical and textual preferences play different roles: numerical inputs enable precise and efficient control through CFG, while textual preferences are encoded as semantic embeddings and incorporated through CG at inference time. This design avoids re-training the MMDiT backbone for novel or personalized textual instructions, allowing FP2 to leverage both exact numerical preferences and robust semantic instructions.

Return vector. Following prior work Chen et al. [2021], we use the discounted return-to-go (RTG) vector as an additional condition to stabilize training. At inference time, prior works adopt different RTG conditioning strategies, such as predicting returns from preferences Zhu et al. [2023]. Here, we set the RTG condition to the maximum normalized return vector $\boldsymbol{G} = \mathbf{1}^{D_o}$ using the multi-objective preference-aware return normalization in Yuan et al. [2024].

4.3 Three-stage training pipeline

Fig. 2 provides an overview of the proposed three-stage training pipeline. While multi-stage training is common in the era of foundation models, our goal is to adapt this recipe to offline MORL scenarios, which present some unique challenges. First, data sampling in multi-objective spaces grows exponentially with the number of objectives, leading to insufficient coverage in practice and poor generalization. For example, sampling 100 points along each objective dimension in an n-dimensional space yields 100^n samples, making full-space coverage impractical. Second, offline MORL datasets' effective supervision for preference alignment remains sparse: actions in most trajectories are sub-optimal or noisy, while high-quality Pareto-front samples are scarce both in absolute number and relative proportion. These motivate a staged design that first learns broad multi-objective control priors and then enhances fine-grained preference alignment.

Data preparation. As shown in Fig. 2, we derive all datasets from D4MORL Zhu et al. [2023], which contains six multi-objective robotic control environments. We denote the full dataset as $\mathcal{D}_{\text{full}}$ and associate each trajectory with both a numerical preference vector and a textual preference description. Since D4MORL does not provide language annotations, text prompts are deterministically generated

from $\boldsymbol{\omega}$ using fixed templates (see Appendix B.1). Following Yuan et al. [2024], we split $\mathcal{D}_{\text{full}}$ into $\mathcal{D}_{\text{pre}}$ by retaining 70% of the data, with the remaining 30% forming $\mathcal{D}_{\text{drop}}$. From $\mathcal{D}_{\text{pre}}$, we further construct the alignment dataset $\mathcal{D}_{\text{mid}}$ by retaining trajectories from the top two fronts identified via non-dominated sorting. These fronts are dataset-relative and do not imply ground-truth optimality, which may be unavailable in practice. For cross-task experiments, we train on five environments and evaluate on the held-out environment, denoted as $\mathcal{D}_{\text{task}}$.

Stage 1: Pre-training. FP2 is pre-trained on the large-scale dataset $\mathcal{D}_{\text{pre}}$ containing both Pareto-optimal and suboptimal trajectories. This stage focuses on learning fundamental multi-objective trade-off behaviors, facilitating faster policy learning in later stages.

Stage 2: Mid-training. While pre-training yields a strong multi-objective representation prior, sub-optimal and noisy trajectories in Stage 1 can weaken preference–behavior alignment, motivating a tailored mid-training stage to enhance fine-grained alignment. More importantly, Stage 2 does not require ground-truth Pareto-optimal trajectories; instead, it uses dataset-relative high-quality trajectories to fine-tune FP2 and reduces the potential preference-behavior inconsistency introduced in Stage 1. Formally, we define preference alignment as follows.

Definition 1 (Preference Alignment)**.** *Let $\boldsymbol{\omega}_-, \boldsymbol{\omega}, \boldsymbol{\omega}_+ \in \mathbb{R}^{D_o}$ be preference vectors such that $\boldsymbol{\omega}_- \preceq_\omega \boldsymbol{\omega} \preceq_\omega \boldsymbol{\omega}_+$, where $\preceq_\omega$ denotes a component-wise partial order in preference space. Let $\boldsymbol{y}^*(\boldsymbol{\omega}_-), \boldsymbol{y}^*(\boldsymbol{\omega}), \boldsymbol{y}^*(\boldsymbol{\omega}_+) \in \mathcal{F}$ denote objective outcomes on the Pareto frontier $\mathcal{F}$. A predicted frontier point $\hat{\boldsymbol{y}}$ for $\boldsymbol{y}^*(\boldsymbol{\omega})$ is said to be **aligned** with preference $\boldsymbol{\omega}$ if $\boldsymbol{y}^*(\boldsymbol{\omega}_-) \preceq \hat{\boldsymbol{y}} \preceq \boldsymbol{y}^*(\boldsymbol{\omega}_+)$, where $\preceq$ denotes a partial order (e.g., Pareto dominance) over objective space.*

Accordingly, Stage 2 is trained on the dataset $\mathcal{D}_{\text{mid}}$. This design retains local structure along the Pareto manifold, encouraging consistent and monotonic preference–behavior mappings.

Stage 3: Post-training. In the final stage, FP2 is adapted to unseen preferences or new environments using a small number of trajectories drawn from either the new preference dataset $\mathcal{D}_{\text{drop}}$ or the new task dataset $\mathcal{D}_{\text{task}}$. To avoid catastrophic forgetting, we adopt a parameter-efficient post-training method that only finetunes the MMDiT backbone while freezing all remaining modules.

4.4 Training objective

The three stages in our training pipeline use a unified objective comprising the rectified flow matching loss, a triplet regularization term, and a post-training penalty:

$$\mathcal{L} = \mathcal{L}_{\text{RF}} + \lambda_{\text{tri}} \mathcal{L}_{\text{triplet}} + \mathbf{1}_{\text{post}} \mathcal{L}_{\text{weight}} \tag{6}$$

where λ_{tri} controls the regularization strength, and $\mathbf{1}_{\text{post}} = 1$ during post-training and 0 otherwise.

Rectified flow matching loss. Given a pair of samples $(\boldsymbol{z}_0, \boldsymbol{x}) \in \mathcal{D}$ and an interpolation time $t \sim \mathcal{U}(0, 1)$, rectified flow training minimizes the discrepancy between the predicted velocity and the target displacement $\boldsymbol{x} - \boldsymbol{z}_0$:

$$\mathcal{L}_{\text{RF}} = \mathbb{E}_{\boldsymbol{z}_0, \boldsymbol{x}, t} \left[\| v_\theta(\boldsymbol{z}_t, t \mid \boldsymbol{c}) - (\boldsymbol{x} - \boldsymbol{z}_0) \|_2^2 \right], \tag{7}$$

where $\boldsymbol{z}_t = (1 - t)\boldsymbol{z}_0 + t\boldsymbol{x}$, and $\boldsymbol{c} = [\boldsymbol{\omega}, \boldsymbol{G}_t]$.

Triplet velocity regularization. Preferences lie in a continuous space, where nearby preferences should induce similar yet distinguishable behaviors. However, these fine-grained behavioral relations can be easily misaligned or incorrectly ordered during training. To address this, we introduce a velocity-space regularization that constructs hard negatives from nearest neighbors in preference space, yielding a triplet formulation that enforces both similarity and distinguishability. For a mini-batch sample $b \sim \mathcal{B}$, we define the positive distance as $d_{\text{pos}}^{(b)} = \frac{1}{D} \| \boldsymbol{v}_{\text{pred}}^{(b)} - \boldsymbol{v}_{\text{tgt}}^{(b)} \|_2^2$ with $\boldsymbol{v}_{\text{tgt}} = \boldsymbol{x} - \boldsymbol{z}_0$ as in Eq. (7), and select the nearest neighbor in preference space $b' = \arg\min_{j \neq b} \| \boldsymbol{\omega}^{(j)} - \boldsymbol{\omega}^{(b)} \|_2$, yielding the negative distance $d_{\text{neg}}^{(b)} = \frac{1}{D} \| \boldsymbol{v}_{\text{pred}}^{(b)} - \boldsymbol{v}_{\text{pred}}^{(b')} \|_2^2$. Finally, the triplet loss is defined as:

$$\mathcal{L}_{\text{triplet}} = \mathbb{E}_{b \sim \mathcal{B}} \left[\max \left(0,\ d_{\text{pos}}^{(b)} - \lambda\, d_{\text{neg}}^{(b)} \right) \right], \tag{8}$$

where $\lambda > 0$ controls the margin, and the loss focuses on hard samples when $d_{\text{neg}}^{(b)} < d_{\text{pos}}^{(b)} / \lambda$.

Post-training penalty. During post-training, we apply an ℓ_2 regularization term to mitigate catastrophic forgetting:

$$\mathcal{L}_{\text{weight}} = \|\theta - \theta_{\text{mid}}\|_2^2. \tag{9}$$

Here, θ denotes the post-trained model weights, and θ_{mid} denotes the weights after mid-training.

5 Experiments

In this section, we first evaluate the effectiveness of FP2 and its three-stage training pipeline on task-specific locomotion tasks (Sec. 5.1 and Sec. 5.2). We then examine its foundational learning and transfer capabilities in cross-task settings (Sec. 5.3), followed by ablation studies (Sec. 5.4).

Dataset. We evaluate our proposed method on the D4MORL dataset Zhu et al. [2023], a benchmark specifically designed for offline MORL, as summarized in Table 1. It comprises 1.8 million annotated demonstrations across six multi-objective (MO) MuJoCo task environments, each featuring two to three objectives. For each environment, D4MORL provides two datasets, `Expert` and `Amateur`, with the latter serving as a noisier counterpart of the former; together, they correspond to the full dataset $\mathcal{D}_{\text{full}}$. Furthermore, to assess generalization, we adopt the `Shattered` and `Narrow` preference-deficient configurations $\mathcal{D}_{\text{pre}}$ from MODULI Yuan et al. [2024], which remove 30% of the data.

Table 1: Summary of the D4MORL dataset.

TASK	#OBJ	S-D	A-D	STEP-E/A	#OPT-E/A	#PRE
Ant	2	27	8	500/500	102/65	
HalfCheetah	2	17	6	499/482	1547/459	
Hopper	2	11	3	499/387	223/52	
Walker2d	2	17	6	499/442	2055/411	
Swimmer	2	8	2	500/500	1475/58	
Hopper-3obj	3	11	3	499/442	8945/1925	

Note. #OBJ: number of objectives. S/A-D: state/action dimensions. STEP-E/A and #OPT-E/A: average episode length and Pareto-front trajectories for `Expert`/`Amateur`. #PRE: range of preferences; only the first two dimensions are shown for 3-objective tasks.

Baselines. We compare our pre-trained model (**FP2-Pre**), mid-trained model (**FP2-Mid**), post-trained model (**FP2-Post**), and foundation model (**FP2-Fou**) trained on five MO task environments, with strong offline MORL baselines: (1) MODT(P) Chen et al. [2021], Zhu et al. [2023], a Decision Transformer-based approach; (2) MORvS(P) Emmons et al. [2022], Zhu et al. [2023], a supervised reinforcement learning method; (3) BC(P) Emmons et al. [2022], Zhu et al. [2023], a behavior cloning algorithm; (4) CQL(P) Kumar et al. [2020], Zhu et al. [2023], a conservative Q-learning variant; and (5) MODULI Yuan et al. [2024], the current state-of-the-art offline MORL algorithm. Since COLA Li et al. [2026a] and Q-chunking Li et al. [2026b] are designed for online and offline-to-online RL, respectively, we report comparisons with their fixed-data adaptations separately in Appendix D.5.

Metric. To comprehensively assess the performance of FP2 and baselines, we adopt metrics capturing Pareto-front convergence and diversity, as well as preference alignment. **(1)** Hypervolume (**HV**)↑ Zitzler and Thiele [2002], Auger et al. [2012]: HV is widely-used to measure the volume of the objective space dominated by the obtained Pareto front solutions $\mathcal{F}$. $\mathcal{HV}(\mathcal{F}) = \int_{\mathbb{R}^{D_o}} \mathbf{1}_{H(\mathcal{F})}(\boldsymbol{y})\mathrm{d}y$, where D_o is the number of objectives, $H(\mathcal{F}) = \{\boldsymbol{y} \in \mathbb{R}^{D_o} \mid \exists \boldsymbol{y}' \in \mathcal{F}, \boldsymbol{y}_0 \preceq \boldsymbol{y} \preceq \boldsymbol{y}'\}$, $\boldsymbol{y}_0$ is a reference point in the objective space ($\mathbf{0}^{D_o}$ for maximization problems), and $\boldsymbol{y}_0 \preceq \boldsymbol{y}$ means $\boldsymbol{y}_0$ weakly dominates $\boldsymbol{y}$, $\mathbf{1}_{H(\mathcal{F})}(\boldsymbol{y}) = 1$ iff $\boldsymbol{y} \in H(\mathcal{F})$ and 0 otherwise. A higher HV is better. **(2)** Sparsity (**SP**)↓ Zhu et al. [2023]: SP indicates how sparsely the obtained solutions cover the Pareto front: $\mathcal{SP}(\mathcal{F}) = \frac{1}{|\mathcal{F}|}\sum_{i=1}^{n}\sum_{k=1}^{|\mathcal{F}|-1}(\widetilde{\mathcal{F}}_i(k) - \widetilde{\mathcal{F}}_i(k+1))^2$, where $\widetilde{\mathcal{F}}_i$ is the list of Pareto front solutions sorted by the i-th objective value. A lower SP is better. **(3)** Preference alignment score (**PAS**)↑: We propose a cosine similarity-based preference alignment metric PAS. For a preference vector $\boldsymbol{\omega}$, a reference Pareto frontier point $\boldsymbol{y}_{\text{gt}}$, and a predicted point $\boldsymbol{y}_{\text{pred}}$, it is defined as: $\mathcal{PAS} = \frac{1}{2}\left(1 + \frac{(\boldsymbol{y}_{\text{gt}}-\boldsymbol{\omega})\cdot(\boldsymbol{y}_{\text{pred}}-\boldsymbol{\omega})}{\|\boldsymbol{y}_{\text{gt}}-\boldsymbol{\omega}\|_2 \cdot \|\boldsymbol{y}_{\text{pred}}-\boldsymbol{\omega}\|_2}\right) \in [0,1]$, where larger values indicate better alignment.

5.1 Effective preference generalization via pre- and mid-training

Table 2 compares FP2-Pre, FP2-Mid, and prior methods on `Expert-Shattered/Narrow`. FP2-Mid outperforms MODULI on $11/12$ tasks in HV and $10/12$ in SP. The gain from FP2-Pre to FP2-Mid on most tasks highlights the role of mid-training in enhancing preference–behavior alignment from dataset-relative high-quality trajectories. Fig. 4 further shows improved interpolation and extrapolation of FP2-Mid on Pareto manifolds (Hopper on Amateur-Shattered and Swimmer on Amateur-Narrow). We summarize all 36 comparisons in Appendix D: FP2-Mid achieves median gains of +0.98% on HV, +29.44% on SP, and +1.04% on PAS, and favorable Win/Tie/Loss results of 30/3/3, 29/4/3, and 24/9/3, respectively (Table 9). The gains are larger in SP than in HV, indicating

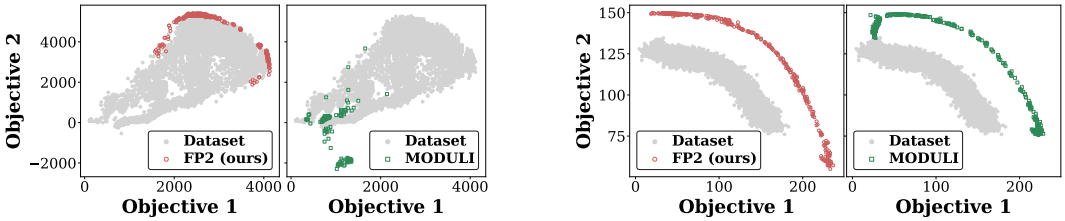


Figure 4: Preference generalization, with OOD points in color and dataset points in gray.

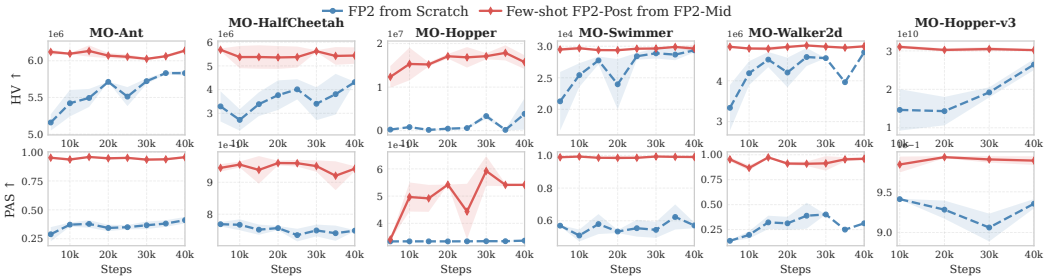


Figure 5: Training performance comparison for out-of-distribution (OOD) preference generalization in Table 3: FP2 trained from scratch vs. few-shot FP2-Post initialized from FP2-Mid.

that FP2 mainly improves the coverage and uniformity of the obtained front, while HV, bounded by the quality of the offline data, is maintained or modestly improved.

Table 2: Task-specific preference generalization on `Expert-Shattered` (S) and `Expert-Narrow` (N). We report average scores over three seeds; full mean ± std results are provided in Appendix D.2.

Metric	Method	Ant-S	HalfC.-S	Hopper-S	Walker-S	Swim.-S	Hopper3-S	Ant-N	HalfC.-N	Hopper-N	Walker-N	Swim.-N	Hopper3-N
HV↑	MODT(P)	5.88	5.69	1.95	5.01	3.20	2.83	5.05	5.04	1.85	4.75	3.03	3.18
	MORvS(P)	**6.38**	5.73	2.06	5.14	3.24	3.28	6.06	5.46	1.98	4.85	3.10	3.32
	BC(P)	4.76	5.66	1.49	3.47	3.19	1.95	4.90	4.88	1.29	1.88	3.06	2.32
	MODULI	**6.38**	5.78	2.07	5.20	3.24	3.33	6.18	5.74	2.07	5.13	3.16	3.15
	FP2-Pre	6.17	5.76	2.01	5.16	3.22	**3.80**	6.06	5.74	2.01	5.10	3.16	3.39
	FP2-Mid	6.35	**5.79**	**2.08**	**5.22**	**3.26**	3.70	**6.24**	**5.79**	**2.10**	**5.22**	**3.26**	**3.81**
SP↓	MODT(P)	2.22	0.18	0.85	0.49	15.50	0.09	0.88	0.04	0.34	0.34	3.65	0.11
	MORvS(P)	0.75	0.15	0.20	0.27	7.36	0.06	0.79	0.05	0.16	0.19	4.38	0.22
	BC(P)	5.16	0.34	2.58	16.61	9.05	0.17	1.56	**0.02**	0.59	10.76	5.53	0.14
	MODULI	0.53	0.11	0.13	0.11	5.63	0.10	0.96	0.04	0.28	0.14	5.84	0.16
	FP2-Pre	1.15	0.13	**0.12**	0.12	7.11	0.07	0.82	0.05	0.47	0.12	6.12	**0.03**
	FP2-Mid	**0.51**	**0.07**	0.13	**0.05**	**2.46**	**0.05**	**0.75**	0.05	**0.08**	**0.06**	**1.68**	0.05
PAS↑	MODULI	0.94	**0.99**	0.43	**0.99**	**0.99**	0.98	**0.93**	**0.99**	0.92	0.93	**0.99**	0.96
	FP2-Pre	0.92	**0.99**	**0.92**	0.92	0.92	0.95	0.90	**0.99**	0.90	0.92	0.89	0.91
	FP2-Mid	**0.96**	**0.99**	0.56	**0.99**	**0.99**	**0.99**	**0.92**	0.98	**0.96**	**0.98**	**0.99**	**0.98**

Note. HV/SP are rescaled for readability. HV: $\times 10^6$ for Ant, HalfC., Walker; $\times 10^7$ for Hopper; $\times 10^4$ for Swim.; $\times 10^{10}$ for Hopper3. SP: $\times 10^4$ for Ant, HalfC., Swim.; $\times 10^5$ for Hopper, Hopper3; $\times 10^0$ for Walker. HalfC.: HalfCheetah; Swim.: Swimmer; Hopper3: Hopper-3obj.

5.2 Fast adaptation to new preferences via post-training

We evaluate few-shot adaptation to unseen preference configurations using a fixed budget of 1K gradient steps. Specifically, we adapt on Pareto-front samples from the 30% held-out preference regions in the `Expert/Amateur-Shattered/Narrow` datasets, and all metrics are computed on these held-out regions. Table 3 compares FP2 trained from scratch (Scratch), zero-shot FP2-Mid, and post-trained FP2 (FP2-Post). FP2-Post outperforms Scratch in HV and PAS on all six tasks, often by a large margin; notably, this all-task advantage over Scratch is also observed for zero-shot FP2-Mid, indicating that pre- and mid-training provide a strong initialization for fast adaptation. Fig. 5 further shows that FP2-Post converges faster than Scratch. Post-training does not always improve over zero-shot FP2-Mid; for example, HV slightly decreases on MO-Hopper (1.95 to 1.89).

5.3 Cross-task transfer of a reusable trade-off prior

We evaluate whether FP2 can learn a reusable multi-objective trade-off prior that transfers across environments. To cover different locomotion characteristics, we adopt a systematic leave-one-task-out protocol: for each target environment, the model is trained on the other five multi-objective

Table 3: Few-shot post-training (1K steps) on the held-out 30% preference regions of `Expert-Shattered`; all metrics are computed on the held-out regions.

Environment	Scratch			FP2-Mid (Zero-shot)			FP2-Post (Few-shot)		
	HV ↑	SP ↓	PAS ↑	HV ↑	SP ↓	PAS ↑	HV ↑	SP ↓	PAS ↑
MO-Ant	5.04	3.53	0.36	5.70	3.37	0.85	**5.83**	**1.69**	**0.91**
MO-HalfCheetah	4.00	0.34	0.79	5.69	0.09	**0.98**	**5.73**	**0.08**	0.97
MO-Hopper	0.91	**0.07**	0.28	**1.95**	0.14	0.54	1.89	0.13	**0.60**
MO-Walker2d	4.80	1.21	0.17	4.87	0.20	**0.99**	**5.05**	**0.15**	**0.99**
MO-Swimmer	2.79	71.33	0.67	3.03	11.13	**0.99**	**3.09**	**9.15**	**0.99**
MO-Hopper-3obj	2.13	0.10	0.70	3.63	0.60	**0.97**	**3.66**	**0.03**	**0.97**

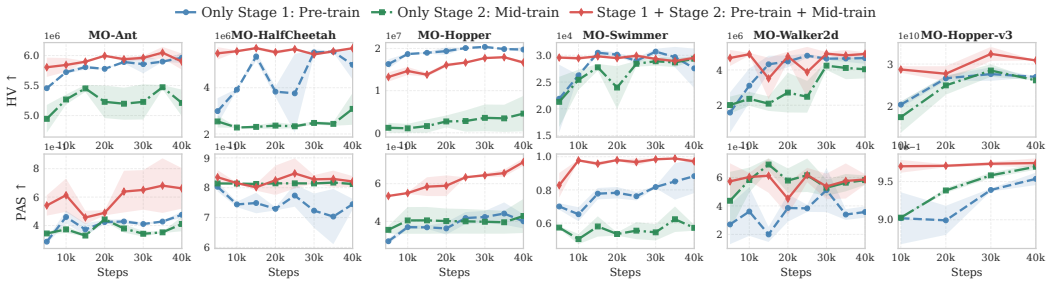


Figure 7: Ablation study. Pre- and mid-training together achieve the best and fastest convergence, demonstrating the effectiveness of the multi-stage training paradigm.

MuJoCo tasks and then adapted to the held-out task with a fixed budget of 1K gradient steps. This out-of-distribution (OOD) task generalization setting is more challenging than OOD preference generalization because the target environment has different dynamics and is never observed during training. The FP2 model trained on mixed-environment data is denoted as **FP2-Fou**, and its post-trained variant as **FP2-Fou-Post**.

For comparison, we adapt MODULI to the cross-task setting using the same zero-padding, feature/temporal masking, mixed-environment sampling, leave-one-task-out split, target-task data, and 1K-step budget as FP2. Fig. 6 illustrates the adaptation behavior of FP2-Fou-Post, training from scratch (Scratch), zero-shot FP2-Fou, and MODULI, while Table 13 reports quantitative results. Among these comparisons, FP2-Fou-Post vs. Scratch isolates the effect of pre- and mid-training, FP2-Fou vs. FP2-Fou-Post isolates post-training, and MODULI serves as an algorithmic offline-MORL reference. FP2-Fou-Post consistently outperforms all baselines, demonstrating the effective cross-task transfer capability of FP2. As an exploratory extension beyond locomotion, Appendix D.6 further evaluates zero-shot FP2 on ten multi-objective Meta-World manipulation environments Yu et al. [2020] with control-level speed and cost objectives.

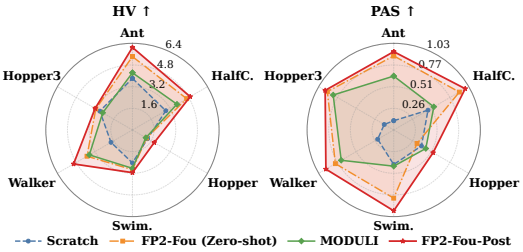


Figure 6: Cross-task performance comparison.

5.4 Ablation

Multi-stage training paradigm. Fig. 7 compares training-stage variants to isolate their contributions. Stage 1 provides a strong generative prior; Stage 2 alone is constrained by its limited data coverage. Their combination best improves preference–behavior alignment and performs strongest overall.

Generative backbone. To isolate the effect of Rectified Flow (RF), Table 4 compares RF with DDIM Song et al. [2021] under the same training stages (Pre and Pre+Mid) on Expert-Shattered. The two backbones perform comparably overall, with the clearest RF gain on MO-Hopper-3obj after pre-training (3.80 vs. 3.37); mid-training improves DDIM on all six tasks and RF on five, indicating that FP2's gains mainly arise from the staged training pipeline rather than from RF alone.

Table 4: Ablation on (1) generative backbone, (2) preference guidance, (3) contrastive learning, and (4) post-training strategy. HV is reported in environment-specific scales.

	Method	Ant	HalfCheetah	Hopper	Swimmer	Walker2d	Hopper-3obj
(1)	**FP2-Pre**	**6.17**	5.76	**2.01**	3.22	5.16	**3.80**
	⟳ DDIM-Pre	6.15	**5.77**	2.00	**3.24**	**5.17**	3.37
	FP2-Mid	6.35	**5.79**	**2.08**	**3.26**	**5.22**	**3.70**
	⟳ DDIM-Mid	**6.37**	**5.79**	**2.08**	**3.26**	5.21	3.69
(2)	**FP2-Mid**	**6.35**	**5.79**	**2.08**	**3.26**	**5.22**	**3.70**
	w/o Text.Pref.	6.33	5.76	**2.08**	3.24	**5.22**	3.53
	w/o Num.Pref.	6.07	5.74	1.92	3.16	5.14	3.21
(3)	**FP2-Pre**	**6.17**	**5.76**	2.01	**3.22**	**5.16**	**3.80**
	w/o Triplet	6.11	5.74	1.96	3.20	5.15	3.33
	⟳ InfoNCE	5.63	5.46	**2.08**	3.13	4.97	3.57
(4)	**FP2-Post**	5.83	**5.73**	**1.89**	**3.09**	**5.05**	3.66
	w/o ℓ_2-Post	5.65	5.10	1.35	3.02	4.88	2.11
	⟳ LoRA	**5.85**	5.71	1.88	3.08	**5.05**	**3.67**

Numerical and textual preferences. Table 4 shows that removing numerical preferences causes a clear drop, whereas removing textual preferences changes performance only slightly except on MO-Hopper-3obj (3.70 vs. 3.53), indicating that numerical preferences drive trade-off control, while textual preferences mainly serve as a semantic interface.

Effect of contrastive objective. Table 4 shows that removing contrastive learning (w/o Triplet) degrades performance. Triplet loss outperforms InfoNCE Oord et al. [2018], which treats all other in-batch samples as negatives, by using nearby-preference negatives to capture subtle behavior differences, improving preference awareness.

Effect of post-training strategies. FP2-Post uses the ℓ_2 penalty in Eq. (9), with LoRA Hu et al. [2022] as an alternative. As shown in Table 4, both outperform w/o ℓ_2-Post on all six tasks and perform comparably, suggesting that FP2 mainly benefits from post-training itself rather than from a specific fine-tuning technique.

6 Conclusion and limitation

We studied generalization across preferences and continuous-control tasks in offline multi-objective reinforcement learning. Accordingly, we proposed **F**oundation **P**areto **F**low **P**olicy (**FP2**), an offline MORL foundation policy that provides a reusable multi-objective trade-off prior for continuous control. FP2 is built on rectified flow and trained via a principled three-stage pipeline, with mid-training and a contrastive triplet loss to enhance fine-grained preference alignment. Experiments show that FP2 generalizes to unseen preferences and supports few-shot cross-task adaptation of a shared multi-objective trade-off prior with a unified policy. These results highlight the benefits of mid-training with high-quality Pareto-optimal data in data-scarce offline settings and the triplet regularization over nearby negative samples for fine-grained control in continuous preference space.
Limitations. There are two main limitations. First, textual preferences are constructed from templates, which ensures consistency through deterministic rules but limits flexibility for free-form language expressions. Second, FP2 is evaluated only in state-based continuous control, and its cross-task gains are not yet disentangled into multi-objective structure versus general trajectory pretraining; extending it to richer modalities, such as image inputs, and isolating these factors remain interesting directions for future work.

Acknowledgments and Disclosure of Funding

This work was supported by the Natural Science Foundation of Top Talent of SZTU under Grant No. GDRC202521.

References

Anurag Ajay, Yilun Du, Abhi Gupta, Joshua B Tenenbaum, Tommi S Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision making? In *International Conference on Learning Representations*, 2023.

Michael Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *International Conference on Learning Representations*, 2023.

Yashas Annadani, Syrine Belakaria, Stefano Ermon, Stefan Bauer, and Barbara E Engelhardt. Preference-guided diffusion for multi-objective offline optimization. *arXiv preprint arXiv:2503.17299*, 2025.

Anne Auger, Johannes Bader, Dimo Brockhoff, and Eckart Zitzler. Hypervolume-based multiobjective optimization: Theoretical foundations and practical implications. *Theoretical Computer Science*, 425:75–103, 2012.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. pi_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in Neural Information Processing Systems*, 34:15084–15097, 2021.

Weiyu Chen, Baijiong Lin, Xiaoyuan Zhang, Xi Lin, Han Zhao, Qingfu Zhang, and James T Kwok. Gradient-based multi-objective deep learning: Algorithms, theories, applications, and beyond. *arXiv preprint arXiv:2501.10945*, 2025.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in neural information processing systems*, 34:8780–8794, 2021.

Yongkang Du, Jieyu Zhao, Yijun Yang, and Tianyi Zhou. Controllable pareto trade-off between fairness and accuracy. *arXiv preprint arXiv:2509.13651*, 2025.

Scott Emmons, Benjamin Eysenbach, Ilya Kostrikov, and Sergey Levine. Rvs: What is essential for offline rl via supervised learning? In *International Conference on Learning Representations*, 2022.

Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *International Conference on Machine Learning*, 2024.

Florian Felten, El-Ghazali Talbi, and Grégoire Danoy. Multi-objective reinforcement learning based on decomposition: A taxonomy and framework. *Journal of Artificial Intelligence Research*, 79: 679–723, 2024.

Conor F Hayes, Roxana Radulescu, Eugenio Bargiacchi, Johan Kallstrom, Matthew Macfarlane, Mathieu Reymond, Timothy Verstraeten, Luisa M Zintgraf, Richard Dazeley, Fredrik Heintz, et al. A practical guide to multi-objective reinforcement learning and planning. *Autonomous Agents and Multi-Agent Systems*, 36(1):59, 2022.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*, 2022.

Edward J Hu, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, Weizhu Chen, et al. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Hannah Janmohamed, Maxence Faldor, Thomas Pierrot, and Antoine Cully. Preference-conditioned gradient variations for multi-objective quality-diversity. *arXiv preprint arXiv:2411.12433*, 2024.

Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pages 9902–9915. PMLR, 2022.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

Kanxue Li, Baosheng Yu, Qi Zheng, Yibing Zhan, Yuhui Zhang, Tianle Zhang, Yijun Yang, Yue Chen, Lei Sun, Qiong Cao, Li Shen, Lusong Li, Dapeng Tao, and Xiaodong He. Muep: A multimodal benchmark for embodied planning with foundation models. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI 2024, Jeju, South Korea, August 3-9, 2024*, pages 129–138. ijcai.org, 2024. URL `https://www.ijcai.org/proceedings/2024/15`.

Pengyi Li, Hongyao Tang, Yifu Yuan, Jianye Hao, Zibin Dong, and Yan Zheng. Cola: Towards efficient multi-objective reinforcement learning with conflict objective regularization in latent space. *Advances in Neural Information Processing Systems*, 38:80891–80914, 2026a.

Qiyang Li, Zhiyuan Paul Zhou, and Sergey Levine. Reinforcement learning with action chunking. *Advances in Neural Information Processing Systems*, 38:55518–55553, 2026b.

JiGuan G Lin. On min-norm and min-max methods of multi-objective optimization. *Mathematical Programming*, 103(1):1–33, 2005.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Erlong Liu, Yu-Chang Wu, Xiaobin Huang, Chengrui Gao, Ren-Jian Wang, Ke Xue, and Chao Qian. Pareto set learning for multi-objective reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 18789–18797, 2025.

Ruohong Liu, Yuxin Pan, Linjie Xu, Lei Song, Jiang Bian, Pengcheng You, and Yize Chen. C-morl: Multi-objective reinforcement learning through efficient discovery of pareto front. *arXiv preprint arXiv:2410.02236*, 2024.

Xingchao Liu, Chengyue Gong, et al. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2022.

Xingchao Liu, Xiwen Zhang, Jianzhu Ma, Jian Peng, and Qiang Liu. Instaflow: One step is enough for high-quality diffusion-based text-to-image generation. *arXiv preprint arXiv:2309.06380*, 2023.

Yiyang Ma, Xingchao Liu, Xiaokang Chen, Wen Liu, Chengyue Wu, Zhiyu Wu, Zizheng Pan, Zhenda Xie, Haowei Zhang, Xingkai Yu, et al. Janusflow: Harmonizing autoregression and rectified flow for unified multimodal understanding and generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7739–7751, 2025.

Hossam Mossalam, Yannis M Assael, Diederik M Roijers, and Shimon Whiteson. Multi-objective deep reinforcement learning. *arXiv preprint arXiv:1610.02707*, 2016.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4195–4205, 2023.

Adam Polyak, Amit Zohar, Andrew Brown, Andros Tjandra, Animesh Sinha, Ann Lee, Apoorv Vyas, Bowen Shi, Chih-Yao Ma, Ching-Yao Chuang, et al. Movie gen: A cast of media foundation models. *arXiv preprint arXiv:2410.13720*, 2024.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, 2019.

Diederik M Roijers and Shimon Whiteson. *Multi-objective decision making*. Springer, 2017.

Diederik M Roijers, Peter Vamplew, Shimon Whiteson, and Richard Dazeley. A survey of multi-objective sequential decision-making. *Journal of Artificial Intelligence Research*, 48:67–113, 2013.

Diederik M Roijers, Denis Steckelmacher, and Ann Nowé. Multi-objective reinforcement learning for the expected utility of the return. In *Proceedings of the Adaptive and Learning Agents Workshop at FAIM*, volume 2018, 2018.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021.

Jonathan M Stokes, Kevin Yang, Kyle Swanson, Wengong Jin, Andres Cubillos-Ruiz, Nina M Donghia, Craig R MacNair, Shawn French, Lindsey A Carfrae, Zohar Bloom-Ackermann, et al. A deep learning approach to antibiotic discovery. *Cell*, 180(4):688–702, 2020.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*. MIT press Cambridge, 2018.

Peter Vamplew, Benjamin J Smith, Johan Källström, Gabriel Ramos, Roxana Rădulescu, Diederik M Roijers, Conor F Hayes, Fredrik Heintz, Patrick Mannion, Pieter JK Libin, et al. Scalar reward is not enough: A response to silver, singh, precup and sutton (2021). *Autonomous Agents and Multi-Agent Systems*, 36(2):41, 2022.

Kristof Van Moffaert, Madalina M Drugan, and Ann Nowé. Scalarized multi-objective reinforcement learning: Novel design techniques. In *2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*, pages 191–199. IEEE, 2013.

Jie Xu, Yunsheng Tian, Pingchuan Ma, Daniela Rus, Shinjiro Sueda, and Wojciech Matusik. Prediction-guided multi-objective reinforcement learning for continuous robot control. In *International Conference on Machine Learning*, pages 10607–10616. PMLR, 2020.

Mengdi Xu, Yikang Shen, Shun Zhang, Yuchen Lu, Ding Zhao, Joshua Tenenbaum, and Chuang Gan. Prompting decision transformer for few-shot policy generalization. In *International Conference on Machine Learning*, pages 24631–24645. PMLR, 2022.

Runzhe Yang, Xingyuan Sun, and Karthik Narasimhan. A generalized algorithm for multi-objective reinforcement learning and policy adaptation. *Advances in Neural Information Processing Systems*, 32, 2019.

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.

Yijun Yang, Tianyi Zhou, Kanxue Li, Dapeng Tao, Lusong Li, Li Shen, Xiaodong He, Jing Jiang, and Yuhui Shi. Embodied multi-modal agent trained by an llm from a parallel textworld. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 26265–26275. IEEE, 2024.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020.

Yifu Yuan, Zhenrui Zheng, Zibin Dong, and Jianye HAO. MODULI: Unlocking preference generalization via diffusion models for offline multi-objective reinforcement learning. In *International Conference on Machine Learning*, 2024.

Baiting Zhu, Meihua Dang, and Aditya Grover. Scaling pareto-efficient decision making via offline multi-objective rl. In *International Conference on Learning Representations*, 2023.

Eckart Zitzler and Lothar Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 2002.

A Dataset

A.1 Raw data

Table 5: Basic statistics of the full `Expert` and `Amateur` datasets. Preference ranges are reported for the first two dimensions of the preference vector.

$\mathcal{D}_{\text{full}}$	#Objectives	Preference range	#Preference	#Optimal traj.	Total traj.
MO-Ant	2	$[0.2391, 0.2979] \rightarrow [0.7021, 0.7609]$	501	102/65	50k
MO-HalfCheetah	2	$[0.1668, 0.3768] \rightarrow [0.6232, 0.8332]$	501	1547/459	50k
MO-Hopper	2	$[0.3208, 0.4353] \rightarrow [0.5647, 0.6792]$	501	223/52	50k
MO-Walker2d	2	$[0.1668, 0.2583] \rightarrow [0.7417, 0.8332]$	501	2055/411	50k
MO-Swimmer	2	$[0.0774, 0.0950] \rightarrow [0.9050, 0.9226]$	501	1475/58	50k
MO-Hopper-3obj	3	$[0.1511, 0.0007] \rightarrow [0.6078, 0.5919]$	501	8945/1925	50k

We use the **D4MORL benchmark** Zhu et al. [2023], which consists of six multi-objective robotic control environments with two or three objectives. Each environment provides trajectories generated under a dense set of preference vectors.

Table 5 summarizes the basic statistics of the full dataset, which we denote as $\mathcal{D}_{\text{full}}$. Following common practice, trajectories are segmented into fixed-length windows during training; therefore, the number of trajectories does not directly correspond to the number of training samples.

A.2 Dataset construction

All datasets used in this work are derived from the full dataset $\mathcal{D}_{\text{full}}$, consistent with the description in the main paper.

Pre-train split. Following the protocol of Yuan et al. [2024], we randomly retain 70% of trajectories from $\mathcal{D}_{\text{full}}$ to construct the pre-training dataset $\mathcal{D}_{\text{pre}}$, with the remaining 30% forming the dropped subset $\mathcal{D}_{\text{drop}}$. The dropped data are not used during pre-training.

Mid-train dataset. From $\mathcal{D}_{\text{pre}}$, we further construct a dataset $\mathcal{D}_{\text{mid}}$ by retaining only the trajectories on top two Pareto fronts ranked by dominance. Specifically, we first compute the Pareto-optimal set for each environment, remove it, and then recompute the next Pareto front to obtain a set of near-optimal trajectories. The union of these two fronts constitutes $\mathcal{D}_{\text{mid}}$, which is used for preference alignment in Stage 2. Table 6 reports the number of Pareto-optimal and suboptimal trajectories retained in $\mathcal{D}_{\text{mid}}$ for each environment.

Table 6: Basic statistics of $\mathcal{D}_{\text{mid}}$, filtered from the `Expert` and `Amateur` datasets (abbreviated as "E" and "A", respectively, in the table).

$\mathcal{D}_{\text{mid}}$	#Optimal traj. (E/A)	#Suboptimal traj. (E/A)	Total traj. (E/A)
MO-Ant	98/58	166/92	264/150
MO-HalfCheetah	970/393	1245/531	2215/924
MO-Hopper	182/47	250/71	432/118
MO-Walker2d	1688/360	2077/590	3765/950
MO-Swimmer	1032/55	1230/92	2262/147
MO-Hopper-3obj	6573/1583	5179/1911	11752/3494

Task-level split. For foundation models, we use the data of five environments for training and leave the remaining one environment for out-of-distribution evaluation, denoted as $\mathcal{D}_{\text{task}}$.

A.3 Preprocessing

Before slicing trajectories into fixed-horizon windows, we apply the following preprocessing steps to the raw trajectories.

- **State normalization.** States are normalized using dataset-wide Gaussian statistics. Let μ_s and σ_s denote the mean and standard deviation of states. The normalized state at time step t is given by $\tilde{\boldsymbol{s}}_t = (\boldsymbol{s}_t - \mu_s)/\sigma_s$.

- **Action normalization.** Actions are scaled to the range $[-1, 1]$ using min–max normalization based on the dataset action bounds.
- **Objective-aware reward normalization.** To mitigate scale discrepancies across objectives and preference configurations, we adopt a neighborhood-based normalization strategy. For a trajectory τ^i with preference vector $\boldsymbol{\omega}^i$, we define its neighborhood as $\mathcal{N}(i) = \{j \mid \|\boldsymbol{\omega}^j - \boldsymbol{\omega}^i\| < \epsilon\}$ with ϵ be a constant. The normalized reward is computed as

$$\tilde{\boldsymbol{r}}_t^i = \frac{\boldsymbol{r}_t^i - \min_{j \in \mathcal{N}(i)} \boldsymbol{r}^j}{\max_{k \in \mathcal{N}(i)} \boldsymbol{r}^k - \min_{j \in \mathcal{N}(i)} \boldsymbol{r}^j + \delta},$$

  where δ is a small constant for numerical stability.

After normalization, trajectories are sliced into overlapping windows of horizon H, yielding training samples of the form $\{\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{r}_t, \boldsymbol{\omega}, \boldsymbol{e}_{\text{txt}}\}_{t=t_i}^{t_i+H}$.

B Implementation details

This section provides implementation details of **FP2** that complement the method description in Sec. 4.2 and Sec. 4.4. All components described below instantiate the architecture and training pipeline introduced in the main paper and do not introduce additional modeling assumptions. We organize the details into a **Common Setup** (Sec. B.1), shared across all training stages, and a **Mixed-Environment Training Setup** (Sec. B.2), which is enabled only when jointly training on multiple environments.

B.1 Common setup

Backbone and embeddings. All diffusion models in FP2 use a 1D rectified-flow backbone based on a multi-modal diffusion transformer (MMDiT-1D), consistent with the architecture shown in Fig. 3. The network has a model dimension $d_{\text{model}} = 384$, 6 attention heads, and 4 MMDiT blocks. Input trajectory tokens are projected to the model dimension via a linear embedding layer. Each transformer block employs adaptive layer normalization (adaLN-Zero), where the modulation parameters are conditioned on a joint time–condition embedding. The conditioning signal $\boldsymbol{c}$ (including preference, return, and text guidance) is encoded by a two-layer MLP ($128 \to 384$) and summed with the sinusoidal time embedding to form the final time–condition embedding.

Text prompt generation. D4MORL provides numerical preference vectors but no natural language annotations. For each trajectory, the preference vector ω is already stored in the dataset and indicates the objective trade-off used to generate that trajectory. Therefore, our text prompts are not inferred from data or manually annotated; they are automatically generated from the existing ω. Specifically, we convert each ω into a short text description using fixed rules. We first compare the relative weights in ω to determine which objectives are more important. We then assign the preference to several priority levels, such as balanced, moderate and strong. Preferences with similar weight ratios are mapped to the same level and share the same text template. Finally, we fill the template with environment-specific objective names. For example, in a two-objective task, $\omega = [0.5, 0.5]$ is mapped to ‘‘`balanced trade-off between speed and energy efficiency`’’, while $\omega = [0.8, 0.2]$ is mapped to ‘‘`strongly prioritize speed over energy efficiency`’’. Thus, the text prompt is a deterministic description of the same preference vector, rather than an additional source of supervision.

Text-based preference guidance. As described in Sec. 4.2, textual preferences are encoded using dual text encoders. Specifically, we concatenate a 384-dimensional SentenceTransformer embedding (`all-MiniLM-L6-v2`) and a 512-dimensional CLIP embedding (ViT-B/32), resulting in an 896-dimensional text representation $\boldsymbol{e}_{\text{txt}}$. Textual preferences are incorporated through classifier guidance (CG). We train a lightweight text-preference classifier C_ϕ, whose network f_ϕ, a small MMDiT-style 1D transformer (128-dimensional diffusion embedding, model width 192, 2 transformer blocks, and 6 attention heads), predicts the preference text embedding from a noisy trajectory $\boldsymbol{z}_t$. Following Eq. (3), the classifier models $\log p_\phi(\boldsymbol{e}_{\text{txt}} \mid \boldsymbol{z}_t, t)$ with a Gaussian likelihood and is trained by minimizing the negative log-likelihood, which reduces to a mean-squared-error objective:

$$\mathcal{L}_{\text{cls}} = -\mathbb{E}_{\boldsymbol{x},t}\left[C_\phi(\boldsymbol{z}_t, t, \boldsymbol{e}_{\text{txt}})\right], \quad C_\phi(\boldsymbol{z}_t, t, \boldsymbol{e}_{\text{txt}}) = \log p_\phi(\boldsymbol{e}_{\text{txt}} \mid \boldsymbol{z}_t, t) \propto -\left\|f_\phi(\boldsymbol{z}_t, t) - \boldsymbol{e}_{\text{txt}}\right\|_2^2,$$

where e_{txt} denotes the offline text embedding of the preference. During inference, an exponential moving average (EMA) copy of the classifier provides $\nabla_{\boldsymbol{z}_t} C_\phi$ to guide the vector field (Eq. (3)) and scores sampled trajectories by preference alignment; the scores are aggregated along the sampled horizon to select the final action.

Step-dependent loss weighting. Errors at the beginning of a generated trajectory segment affect all subsequent steps and thus have a disproportionate influence on long-horizon returns. We therefore weight the rectified-flow loss in Eq. (7) over trajectory steps and feature dimensions:

$$\mathcal{L}_{\text{RF}} = \mathbb{E}_{\boldsymbol{z}_0,\, \boldsymbol{x},\, t}\left[\left\| \big(v_\theta(\boldsymbol{z}_t, t \mid \boldsymbol{c}) - (\boldsymbol{x} - \boldsymbol{z}_0)\big) \odot \boldsymbol{W} \right\|_2^2\right],$$

where $\odot$ denotes the element-wise product and $\boldsymbol{W} \in \mathbb{R}^{H_{\max} \times D}$ assigns a weight to each step h and feature d of a (padded) segment with horizon $H_{\max}$ and feature dimension D. We set $W_{h=1,d} = 10$ for the state dimensions of the first step and $W_{h>1,d} = 1$ for all other steps; padded entries are excluded by the feature mask (Appendix B.2). This weighting emphasizes the first step of each segment and empirically stabilizes optimization across heterogeneous tasks.

B.2 Mixed-environment training setup

The following mechanisms are enabled only in the joint multi-environment training setting, where state and action spaces are heterogeneous.

Feature-space masking. To handle varying state and action dimensions across tasks, we construct a binary feature mask $\mathbf{M}_{\text{feat}} \in \{0,1\}^{H_{\max} \times (D_s^{\max} + D_a^{\max})}$. For a given environment, entries corresponding to valid features are set to 1, while those corresponding to padded dimensions are set to 0. The mask is applied element-wise to the rectified-flow loss to ensure gradients are propagated only through task-relevant features.

Temporal masking. We additionally apply temporal masking to handle variable trajectory horizons. A boolean attention mask $\mathbf{M}_{\text{attn}} \in \{0,1\}^{H_{\max}}$ is injected into the self-attention layers of the MMDiT backbone to prevent attention to padded timesteps. In parallel, the feature mask $\mathbf{M}_{\text{feat}}$ is modulated by temporal validity, such that all features at invalid timesteps are excluded from the training objective. Together, these mechanisms confine both attention and supervision to valid temporal contexts.

C Training-stage analysis

Table 7: Training hyperparameters.

Parameter	Value		
	Pre-train	**Mid-train**	**Post-train**
Learning Rate	2e-4	2e-4	1e-5
Weight Decay	1e-5	1e-5	1e-5
Batch Size/Mini Batch	2048/256	64/64	64/64
Training Steps	400K	200K	100K

The hyperparameters employed in the three-stage training are summarized in Tab. 7. We adopted a fixed learning rate and weight decay rate for each stage.

C.1 Single-environment training

A relatively large batch size is employed in the pre-training stage to accelerate convergence, where we retain the weights that maximize the Hypervolume (HV) metric to prioritize diversity coverage. During mid-training, a small batch size is adopted to enhance alignment quality—accompanied by a corresponding reduction in the HV metric for certain environments—and we retain the weights that jointly optimize both HV and Preference Alignment Score (PAS) metrics. Finally, in the post-training stage, we use the same batch size as mid-training but a smaller learning rate to enable more stable gradient updates, while retaining the weights that yield optimal HV and PAS performance.

C.2 Mixed-environment training

Mini-batches are utilized, where each batch contains an equal amount of data from different environments for gradient updates. As shown in Fig. 8, we illustrate the metric evolution across all environments during the mixed training process, where the low PAS in the Hopper environment is primarily attributed to conflicts with the Hopper-3obj environment.

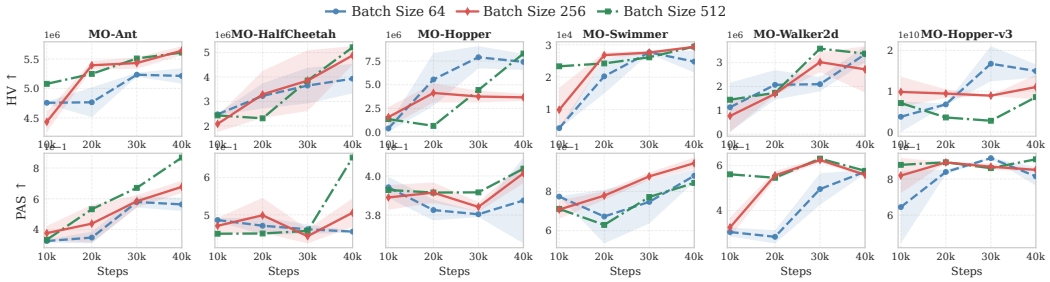


Figure 8: Ablation on the batch size of mixed-environment training.

C.3 Effect of mid-training on preference–policy alignment

Preference alignment is critical for preference generalization. As a case study, Fig. 9 compares alignment on MO-Ant `Amateur-Narrow`. By leveraging high-quality Pareto-optimal trajectories during mid-training, FP2-Mid improves preference alignment over FP2-Pre.

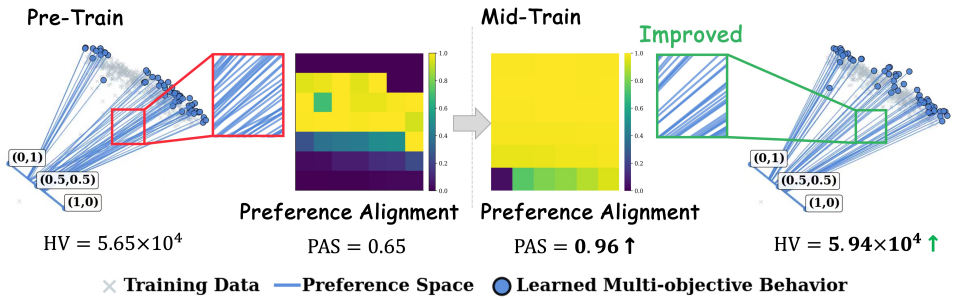


Figure 9: Effect of mid-training on preference–policy alignment. Mid-trained FP2 better aligns policy and preference than pre-trained FP2, achieving higher PAS and more uniform preference alignment distribution.

D Additional results

D.1 Controlled comparison and resource fairness

To ensure a fair comparison with prior generative MORL methods, we summarize the model capacity, training budget, and inference efficiency of our proposed FP2 and the state-of-the-art method MODULI Yuan et al. [2024] in Table 8. This resource-level comparison complements the controlled experiments in the main text, where we isolate the effects of the generative backbone and the multi-stage training pipeline.

As shown in Table 8, FP2 does not rely on a larger model or a more expensive inference procedure than MODULI. Both methods use the same DiT-1D architecture, while FP2 has fewer trainable parameters and lower inference latency. These results suggest that the empirical gains of FP2 are not simply due to increased model capacity or computational advantage.

D.2 Task-specific preference generalization comparison with state-of-the-art methods

In total, we conduct 36 comparison experiments with state-of-the-art methods across six Expert datasets and six Amateur datasets under three evaluation metrics. The statistical summary is reported in Table 9, and the complete comparison results are provided in Tables 10, 11, and 12.

Table 8: Resource-level comparison between FP2 and state-of-the-art method MODULI Yuan et al. [2024]. Both methods use a DiT-1D architecture, while FP2 uses fewer trainable parameters and achieves lower inference latency under a comparable training budget.

Aspect	**FP2**	**MODULI**
Architecture	DiT-1D	DiT-1D
Generative backbone	RF	Diffusion
Sampler	10-step Euler	10-step DDIM
Additional modules	Classifier + text encoders	ControlNet $\times 2$ + inverse-dynamics MLP
Trainable parameters	12.8M	23.4M
Training steps	600K	400K
Inference latency	112 ms / action	126 ms / action

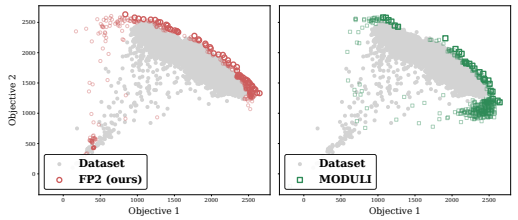
Overall, FP2-Mid achieves the best task-specific preference generalization performance. It improves over MODULI across all three metrics, with median gains of +0.98% on HV, +29.44% on SP, and +1.04% on PAS, and favorable Win/Tie/Loss results of 30/3/3, 29/4/3, and 24/9/3, respectively. These results demonstrate that FP2-Mid generalizes more robustly to task-specific preferences than prior state-of-the-art methods. Figs. 10 and 11 further visualize the OOD preference generalization of FP2 and MODULI on all tasks.

Table 9: Relative improvements (%) of FP2-Mid over state-of-the-art method MODULI across 36 comparisons. Positive values indicate improvements over MODULI Yuan et al. [2024] after accounting for the optimization direction of each metric. Positive improvements are highlighted.

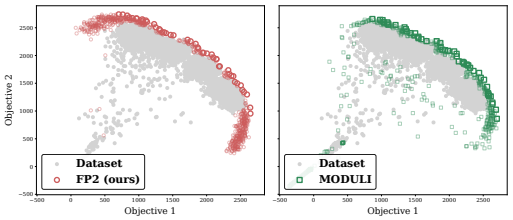
Metric	Min	P25	Median	Mean	P75	Max	Win/Tie/Loss
HV↑	-3.75	+0.35	+0.98	+2.96	+3.62	+20.95	30/3/3
SP↓	-25.00	+11.04	+29.44	+33.78	+59.52	+90.00	29/4/3
PAS↑	-7.07	+0.00	+1.04	+5.51	+3.17	+110.71	24/9/3

Table 10: Complete results for task-specific preference generalization on the full `Expert` and `Amateur` datasets. $*$ indicates results taken from Yuan et al. [2024].

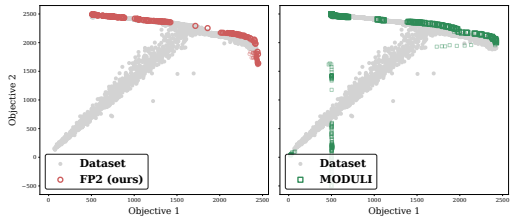
	Environment	**Metric**	**B**	**MODT(P)**	**MORvS(P)**	**BC(P)**	**CQL(P)**	**MODULI***	**MODULI**	**FP2-Pre**	**FP2-Mid**
Expert	MO-Ant	HV ($\times 10^6$)↑	6.32	6.21 ± 0.01	6.36 ± 0.02	4.88 ± 0.17	5.76 ± 0.10	6.39 ± 0.02	6.40 ± 0.01	6.26 ± 0.05	$\mathbf{6.40 \pm 0.01}$
		SP ($\times 10^4$)↓	–	8.26 ± 2.22	0.87 ± 0.19	46.20 ± 16.40	0.58 ± 0.10	0.79 ± 0.12	0.53 ± 0.12	0.50 ± 0.04	$\mathbf{0.47 \pm 0.04}$
		PAS↑	–	–	–	–	–	–	0.96 ± 0.00	0.94 ± 0.04	$\mathbf{0.97 \pm 0.02}$
	MO-HalfCheetah	HV ($\times 10^6$)↑	5.79	5.73 ± 0.00	5.78 ± 0.00	5.54 ± 0.05	5.63 ± 0.04	5.79 ± 0.00	5.79 ± 0.00	5.77 ± 0.00	$\mathbf{5.79 \pm 0.01}$
		SP ($\times 10^4$)↓	–	1.24 ± 0.23	0.67 ± 0.05	1.78 ± 0.39	0.10 ± 0.00	0.07 ± 0.00	0.09 ± 0.01	0.09 ± 0.01	$\mathbf{0.05 \pm 0.01}$
		PAS↑	–	–	–	–	–	–	0.99 ± 0.00	0.99 ± 0.00	$\mathbf{0.99 \pm 0.00}$
	MO-Hopper	HV ($\times 10^7$)↑	2.09	2.00 ± 0.02	2.02 ± 0.02	1.23 ± 0.10	0.33 ± 0.39	2.09 ± 0.01	2.09 ± 0.01	2.01 ± 0.03	$\mathbf{2.10 \pm 0.01}$
		SP ($\times 10^5$)↓	–	16.30 ± 10.60	3.03 ± 0.36	52.50 ± 4.88	2.84 ± 2.46	0.09 ± 0.01	0.12 ± 0.02	0.08 ± 0.00	$\mathbf{0.07 \pm 0.01}$
		PAS↑	–	–	–	–	–	–	0.99 ± 0.00	0.98 ± 0.00	$\mathbf{0.99 \pm 0.00}$
	MO-Walker	HV ($\times 10^4$)↑	5.21	4.89 ± 0.05	5.14 ± 0.01	3.74 ± 0.11	3.21 ± 0.32	5.20 ± 0.00	5.19 ± 0.00	5.17 ± 0.03	$\mathbf{5.22 \pm 0.01}$
		SP ($\times 10^0$)↓	–	0.99 ± 0.44	3.22 ± 0.73	75.60 ± 52.30	6.23 ± 10.70	0.11 ± 0.01	0.34 ± 0.42	0.11 ± 0.02	$\mathbf{0.06 \pm 0.01}$
		PAS↑	–	–	–	–	–	–	0.96 ± 0.00	0.90 ± 0.02	$\mathbf{0.99 \pm 0.00}$
	MO-Swimmer	HV ($\times 10^6$)↑	3.25	3.15 ± 0.02	3.24 ± 0.00	3.21 ± 0.00	3.22 ± 0.08	3.24 ± 0.00	3.25 ± 0.00	3.24 ± 0.01	$\mathbf{3.26 \pm 0.01}$
		SP ($\times 10^4$)↓	–	15.00 ± 7.49	4.39 ± 0.07	4.50 ± 0.39	13.60 ± 5.31	4.43 ± 0.38	4.43 ± 0.21	9.92 ± 2.50	$\mathbf{4.14 \pm 0.11}$
		PAS↑	–	–	–	–	–	–	0.99 ± 0.00	0.97 ± 0.00	$\mathbf{0.99 \pm 0.00}$
	MO-Hopper-3obj	HV ($\times 10^{10}$)↑	3.73	3.38 ± 0.05	3.42 ± 0.10	2.29 ± 0.07	0.78 ± 0.24	3.57 ± 0.02	3.44 ± 0.04	3.72 ± 0.01	$\mathbf{3.76 \pm 0.02}$
		SP ($\times 10^5$)↓	–	1.40 ± 0.44	2.72 ± 1.93	0.72 ± 0.09	2.60 ± 3.14	0.07 ± 0.00	0.07 ± 0.00	0.07 ± 0.02	$\mathbf{0.05 \pm 0.01}$
		PAS↑	–	–	–	–	–	–	0.98 ± 0.01	0.99 ± 0.00	$\mathbf{0.99 \pm 0.00}$
Amateur	MO-Ant	HV ($\times 10^6$)↑	5.61	5.92 ± 0.04	6.07 ± 0.02	4.37 ± 0.06	5.62 ± 0.23	6.08 ± 0.03	6.10 ± 0.01	6.10 ± 0.06	$\mathbf{6.14 \pm 0.01}$
		SP ($\times 10^4$)↓	–	8.72 ± 0.77	5.24 ± 0.52	25.90 ± 16.40	1.06 ± 0.28	0.53 ± 0.05	0.66 ± 0.08	0.67 ± 0.21	$\mathbf{0.39 \pm 0.06}$
		PAS↑	–	–	–	–	–	–	0.94 ± 0.01	0.91 ± 0.00	$\mathbf{0.97 \pm 0.00}$
	MO-HalfCheetah	HV ($\times 10^6$)↑	5.68	5.69 ± 0.01	5.77 ± 0.00	5.46 ± 0.02	5.54 ± 0.02	5.76 ± 0.00	5.76 ± 0.00	5.77 ± 0.01	$\mathbf{5.78 \pm 0.01}$
		SP ($\times 10^4$)↓	–	1.16 ± 0.42	0.57 ± 0.09	2.22 ± 0.91	0.45 ± 0.27	0.07 ± 0.02	0.07 ± 0.01	0.06 ± 0.02	$\mathbf{0.05 \pm 0.01}$
		PAS↑	–	–	–	–	–	–	0.96 ± 0.01	0.99 ± 0.00	$\mathbf{0.99 \pm 0.00}$
	MO-Hopper	HV ($\times 10^7$)↑	1.97	1.81 ± 0.05	1.76 ± 0.03	1.35 ± 0.03	1.64 ± 0.01	2.01 ± 0.01	2.01 ± 0.01	$\mathbf{2.03 \pm 0.01}$	2.01 ± 0.01
		SP ($\times 10^5$)↓	–	1.61 ± 0.29	3.50 ± 1.54	2.42 ± 1.08	3.30 ± 5.25	$\mathbf{0.10 \pm 0.01}$	0.11 ± 0.01	0.13 ± 0.01	0.13 ± 0.02
		PAS↑	–	–	–	–	–	–	0.69 ± 0.01	$\mathbf{0.92 \pm 0.01}$	0.77 ± 0.01
	MO-Walker	HV ($\times 10^4$)↑	4.99	3.10 ± 0.34	4.98 ± 0.01	3.42 ± 0.42	1.78 ± 0.33	5.06 ± 0.00	5.07 ± 0.01	5.10 ± 0.02	$\mathbf{5.15 \pm 0.01}$
		SP ($\times 10^0$)↓		164.20 ± 13.50	1.94 ± 0.06	53.10 ± 34.60	7.33 ± 5.89	0.25 ± 0.03	0.28 ± 0.02	0.10 ± 0.02	$\mathbf{0.09 \pm 0.02}$
		PAS↑	–	–	–	–	–	–	0.94 ± 0.01	0.90 ± 0.01	$\mathbf{0.99 \pm 0.00}$
	MO-Swimmer	HV ($\times 10^6$)↑	2.11	1.67 ± 0.22	2.79 ± 0.03	2.82 ± 0.04	1.69 ± 0.93	$\mathbf{3.20 \pm 0.00}$	3.12 ± 0.00	3.05 ± 0.04	3.08 ± 0.00
		SP ($\times 10^4$)↓	–	2.87 ± 1.32	$\mathbf{1.03 \pm 0.20}$	5.05 ± 1.82	9.50 ± 0.59	8.87 ± 6.24	9.72 ± 0.69	5.51 ± 0.70	5.82 ± 0.12
		PAS↑	–	–	–	–	–	–	0.95 ± 0.00	0.98 ± 0.01	$\mathbf{0.99 \pm 0.00}$
	MO-Hopper-3obj	HV ($\times 10^{10}$)↑	3.09	1.04 ± 0.16	2.77 ± 0.24	2.42 ± 0.18	0.59 ± 0.42	3.33 ± 0.06	3.30 ± 0.06	3.52 ± 0.01	$\mathbf{3.60 \pm 0.02}$
		SP ($\times 10^5$)↓	–	10.23 ± 2.78	1.03 ± 0.11	0.87 ± 0.29	2.00 ± 1.72	0.10 ± 0.00	0.12 ± 0.02	0.09 ± 0.01	$\mathbf{0.06 \pm 0.01}$
		PAS↑	–	–	–	–	–	–	0.98 ± 0.00	0.99 ± 0.00	$\mathbf{0.99 \pm 0.00}$



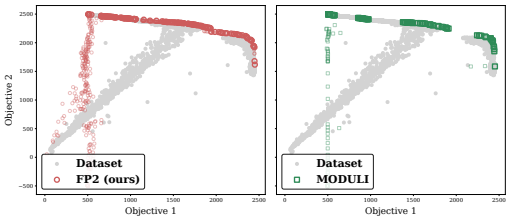
(a) MO-Ant (`Amateur-Narrow`)



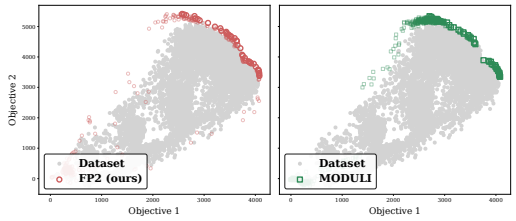
(b) MO-Ant (`Amateur-Shattered`)



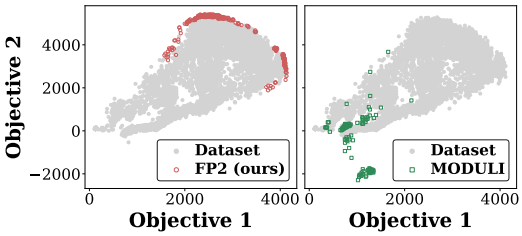
(c) MO-HalfCheetah (`Amateur-Narrow`)



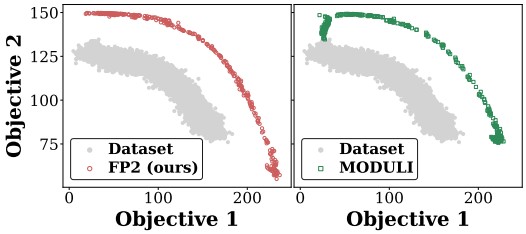
(d) MO-HalfCheetah (`Amateur-Shattered`)



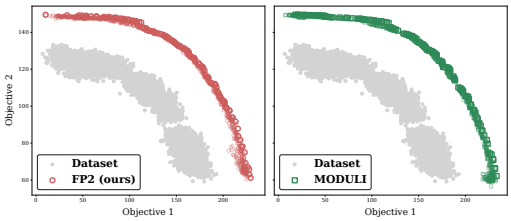
(e) MO-Hopper (`Amateur-Narrow`)



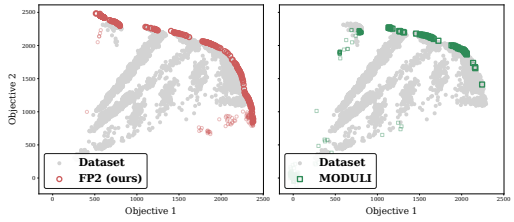
(f) MO-Hopper (`Amateur-Shattered`)



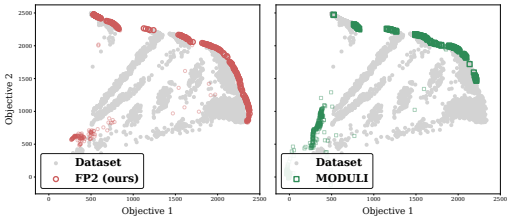
(g) MO-Swimmer (`Amateur-Narrow`)



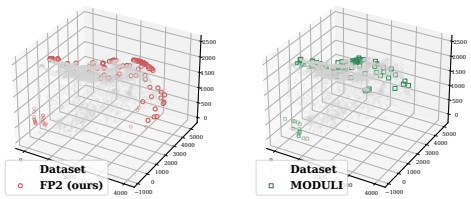
(h) MO-Swimmer (`Amateur-Shattered`)



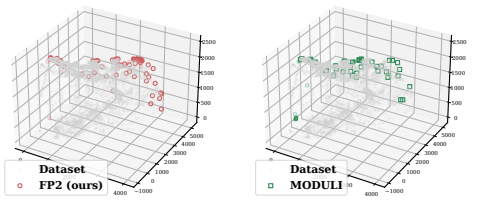
(i) MO-Walker2d (`Amateur-Narrow`)



(j) MO-Walker2d (`Amateur-Shattered`)

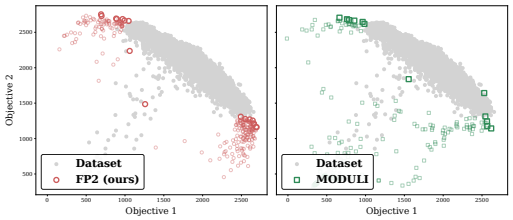


(k) MO-Hopper-3obj (`Amateur-Narrow`)

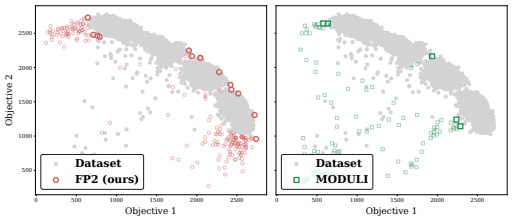


(l) MO-Hopper-3obj (`Amateur-Shattered`)

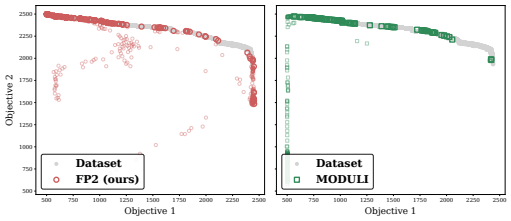
Figure 10: Preference generalization on the `Amateur` datasets. In each panel, FP2 (left) and MODULI (right) are shown with OOD points in color and dataset points in gray.



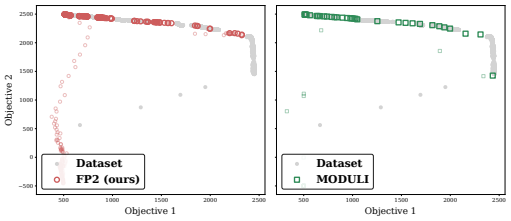
(a) MO-Ant (`Expert-Narrow`)



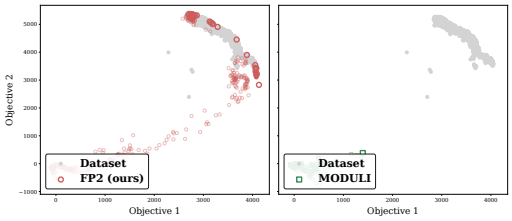
(b) MO-Ant (`Expert-Shattered`)



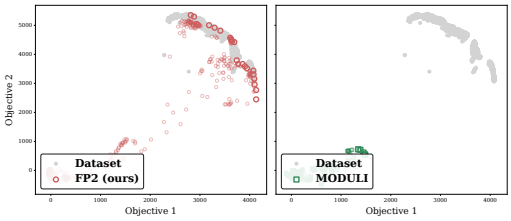
(c) MO-HalfCheetah (`Expert-Narrow`)



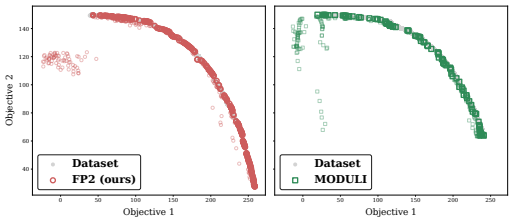
(d) MO-HalfCheetah (`Expert-Shattered`)



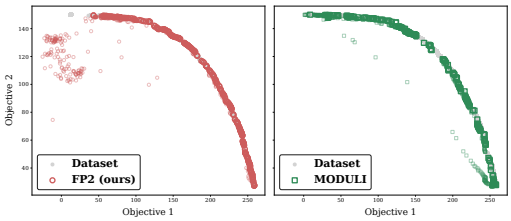
(e) MO-Hopper (`Expert-Narrow`)



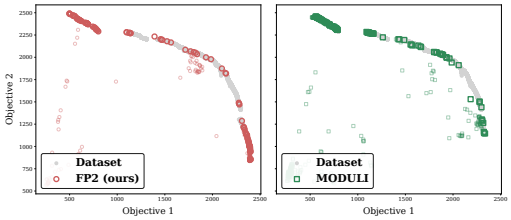
(f) MO-Hopper (`Expert-Shattered`)



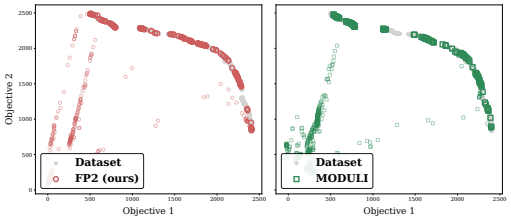
(g) MO-Swimmer (`Expert-Narrow`)



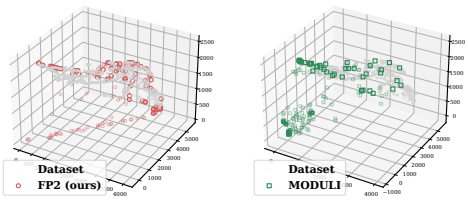
(h) MO-Swimmer (`Expert-Shattered`)



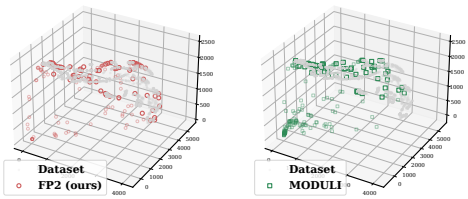
(i) MO-Walker2d (`Expert-Narrow`)



(j) MO-Walker2d (`Expert-Shattered`)



(k) MO-Hopper-3obj (`Expert-Narrow`)



(l) MO-Hopper-3obj (`Expert-Shattered`)

Figure 11: Preference generalization on the `Expert` datasets. In each panel, FP2 (left) and MODULI (right) are shown with OOD points in color and dataset points in gray.

Table 11: Complete results for task-specific preference generalization on the `Expert-Shattered` and `Expert-Narrow` datasets. $*$ indicates results taken from Yuan et al. [2024].

	Environment	Metric	MODT(P)	MORvS(P)	BC(P)	MODULI*	MODULI	FP2-Pre	FP2-Mid
Expert-Shattered	MO-Ant	HV ($\times 10^6$) ↑	5.88 ± 0.02	6.38 ± 0.01	4.76 ± 0.05	$\mathbf{6.39 \pm 0.01}$	6.38 ± 0.01	6.17 ± 0.12	6.35 ± 0.01
		SP ($\times 10^4$) ↓	2.22 ± 0.50	0.75 ± 0.13	5.16 ± 1.18	0.53 ± 0.13	0.53 ± 0.11	1.15 ± 0.65	$\mathbf{0.51 \pm 0.07}$
		PAS ↑	–	–	–	–	0.94 ± 0.01	0.92 ± 0.01	$\mathbf{0.96 \pm 0.01}$
	MO-HalfCheetah	HV ($\times 10^6$) ↑	5.69 ± 0.00	5.73 ± 0.00	5.66 ± 0.00	5.78 ± 0.00	5.78 ± 0.00	5.76 ± 0.01	$\mathbf{5.79 \pm 0.01}$
		SP ($\times 10^4$) ↓	0.18 ± 0.01	0.15 ± 0.02	0.34 ± 0.03	0.11 ± 0.01	0.11 ± 0.01	0.13 ± 0.04	$\mathbf{0.07 \pm 0.01}$
		PAS ↑	–	–	–	–	0.99 ± 0.00	0.99 ± 0.00	$\mathbf{0.99 \pm 0.00}$
	MO-Hopper	HV ($\times 10^7$) ↑	1.95 ± 0.03	2.06 ± 0.01	1.49 ± 0.01	2.07 ± 0.00	2.07 ± 0.00	2.01 ± 0.11	$\mathbf{2.08 \pm 0.00}$
		SP ($\times 10^5$) ↓	0.85 ± 0.27	0.20 ± 0.06	2.58 ± 1.01	$\mathbf{0.11 \pm 0.01}$	0.13 ± 0.01	0.12 ± 0.08	0.13 ± 0.02
		PAS ↑	–	–	–	–	0.43 ± 0.02	$\mathbf{0.92 \pm 0.01}$	0.56 ± 0.01
	MO-Walker	HV ($\times 10^4$) ↑	5.01 ± 0.01	5.14 ± 0.01	3.47 ± 0.12	5.20 ± 0.00	5.20 ± 0.00	5.16 ± 0.03	$\mathbf{5.22 \pm 0.00}$
		SP ($\times 10^0$) ↓	0.49 ± 0.03	0.27 ± 0.04	16.61 ± 13.78	0.12 ± 0.01	0.11 ± 0.01	0.12 ± 0.01	$\mathbf{0.05 \pm 0.01}$
		PAS ↑	–	–	–	–	0.99 ± 0.00	0.92 ± 0.02	$\mathbf{0.99 \pm 0.00}$
	MO-Swimmer	HV ($\times 10^6$) ↑	3.20 ± 0.00	3.24 ± 0.00	3.19 ± 0.00	3.24 ± 0.00	3.24 ± 0.00	3.22 ± 0.02	$\mathbf{3.26 \pm 0.01}$
		SP ($\times 10^4$) ↓	15.50 ± 0.33	7.36 ± 0.81	9.05 ± 0.30	5.79 ± 0.43	5.63 ± 0.47	7.11 ± 0.07	$\mathbf{2.46 \pm 0.21}$
		PAS ↑	–	–	–	–	0.99 ± 0.00	0.92 ± 0.01	$\mathbf{0.99 \pm 0.00}$
	MO-Hopper-3obj	HV ($\times 10^{10}$) ↑	2.83 ± 0.06	3.28 ± 0.07	1.95 ± 0.16	3.43 ± 0.02	3.33 ± 0.02	$\mathbf{3.80 \pm 0.08}$	3.70 ± 0.00
		SP ($\times 10^5$) ↓	0.09 ± 0.01	0.06 ± 0.00	0.17 ± 0.06	0.10 ± 0.01	0.10 ± 0.02	0.07 ± 0.02	$\mathbf{0.05 \pm 0.01}$
		PAS ↑	–	–	–	–	0.98 ± 0.00	0.95 ± 0.01	$\mathbf{0.99 \pm 0.00}$
Expert-Narrow	MO-Ant	HV ($\times 10^6$) ↑	5.05 ± 0.08	6.06 ± 0.00	4.90 ± 0.02	$\mathbf{6.36 \pm 0.00}$	6.18 ± 0.01	6.06 ± 0.19	6.24 ± 0.01
		SP ($\times 10^4$) ↓	0.88 ± 0.41	0.79 ± 0.11	1.56 ± 0.28	0.80 ± 0.25	0.96 ± 0.21	0.82 ± 0.41	0.75 ± 0.52
		PAS ↑	–	–	–	–	0.93 ± 0.01	0.90 ± 0.01	$\mathbf{0.92 \pm 0.01}$
	MO-HalfCheetah	HV ($\times 10^6$) ↑	5.04 ± 0.00	5.46 ± 0.01	4.88 ± 0.02	5.76 ± 0.00	5.74 ± 0.01	5.74 ± 0.00	$\mathbf{5.79 \pm 0.00}$
		SP ($\times 10^4$) ↓	0.04 ± 0.00	0.05 ± 0.00	$\mathbf{0.02 \pm 0.00}$	0.04 ± 0.00	0.04 ± 0.01	0.05 ± 0.01	0.05 ± 0.01
		PAS ↑	–	–	–	–	0.99 ± 0.00	$\mathbf{0.99 \pm 0.00}$	0.98 ± 0.00
	MO-Hopper	HV ($\times 10^7$) ↑	1.85 ± 0.02	1.98 ± 0.00	1.29 ± 0.01	2.04 ± 0.01	2.07 ± 0.01	2.01 ± 0.03	$\mathbf{2.10 \pm 0.00}$
		SP ($\times 10^5$) ↓	0.34 ± 0.07	0.16 ± 0.02	0.59 ± 0.24	0.25 ± 0.05	0.28 ± 0.02	0.47 ± 0.70	$\mathbf{0.08 \pm 0.01}$
		PAS ↑	–	–	–	–	0.92 ± 0.00	0.90 ± 0.00	$\mathbf{0.96 \pm 0.00}$
	MO-Walker	HV ($\times 10^4$) ↑	4.75 ± 0.01	4.85 ± 0.01	1.88 ± 0.95	5.10 ± 0.01	5.13 ± 0.01	5.10 ± 0.01	$\mathbf{5.22 \pm 0.00}$
		SP ($\times 10^0$) ↓	0.34 ± 0.09	0.19 ± 0.02	10.76 ± 11.58	0.13 ± 0.01	0.14 ± 0.01	0.12 ± 0.01	$\mathbf{0.06 \pm 0.01}$
		PAS ↑	–	–	–	–	0.93 ± 0.01	0.92 ± 0.01	$\mathbf{0.98 \pm 0.00}$
	MO-Swimmer	HV ($\times 10^6$) ↑	3.03 ± 0.00	3.10 ± 0.00	3.06 ± 0.00	3.21 ± 0.00	3.16 ± 0.00	3.16 ± 0.00	$\mathbf{3.26 \pm 0.00}$
		SP ($\times 10^4$) ↓	3.65 ± 0.22	4.38 ± 0.80	5.53 ± 0.20	3.28 ± 0.15	5.84 ± 0.01	6.12 ± 0.68	$\mathbf{1.68 \pm 0.07}$
		PAS ↑	–	–	–	–	0.99 ± 0.00	0.89 ± 0.04	$\mathbf{0.99 \pm 0.00}$
	MO-Hopper-3obj	HV ($\times 10^{10}$) ↑	3.18 ± 0.05	3.32 ± 0.00	2.32 ± 0.10	3.37 ± 0.05	3.15 ± 0.13	3.39 ± 0.29	$\mathbf{3.81 \pm 0.00}$
		SP ($\times 10^5$) ↓	0.11 ± 0.01	0.22 ± 0.02	0.14 ± 0.02	0.12 ± 0.01	0.16 ± 0.00	$\mathbf{0.03 \pm 0.02}$	0.05 ± 0.01
		PAS ↑	–	–	–	–	0.96 ± 0.00	0.91 ± 0.01	$\mathbf{0.98 \pm 0.01}$

D.3 Leave-one-task-out cross-task transfer

Table 13: Leave-one-task-out cross-task transfer under a fixed budget of **1K steps**. FP2-Fou is trained on five environments and adapted to the held-out task.

Held-out Task	Scratch		FP2-Fou (Zero-shot)		MODULI		FP2-Fou-Post		Gain vs MODULI	
	HV ↑	PAS ↑	HV ↑	PAS ↑	HV ↑	PAS ↑	HV ↑	PAS ↑	HV ↑	PAS ↑
MO-Ant	3.83	0.11	5.45	0.88	4.25	0.64	6.13	0.93	+44.2%	+45.3%
MO-HalfCheetah	2.86	0.47	4.65	0.90	3.82	0.55	4.95	0.98	+29.6%	+78.2%
MO-Hopper	1.30	0.38	1.20	0.32	1.15	0.44	1.87	0.54	+62.6%	+22.7%
MO-Swimmer	2.45	0.41	2.95	0.81	2.85	0.43	3.17	0.96	+11.2%	+123.3%
MO-Walker2d	1.84	0.22	3.90	0.80	3.70	0.72	5.05	0.93	+36.5%	+123.3%
MO-Hopper-3obj	2.80	0.13	3.14	0.91	2.55	0.83	3.20	0.94	+25.5%	+29.2%

D.4 Out-of-distribution preference adaptation

As shown in Table 14, FP2-Post outperforms FP2 trained from scratch in HV and PAS on all tasks of both datasets, demonstrating the effectiveness of few-shot adaptation from FP2-Mid to unseen preferences. Post-training does not always improve over zero-shot FP2-Mid; for example, HV slightly decreases on MO-Hopper under `Expert-Shattered` (Sec. 5.2).

D.5 Fixed-data comparison with value-based RL methods

COLA Li et al. [2026a] is an online MORL method, and Q-chunking Li et al. [2026b] is a single-objective offline-to-online method; neither is designed for learning from a fixed multi-objective dataset. For a controlled comparison, we adapt both to the fixed-data setting. For Offline-COLA, we replace its online interaction with D4MORL trajectories and add CQL-style critic regularization to mitigate out-of-distribution overestimation. For MO-Q-chunking, we append the preference vector

Table 12: Complete results for task-specific preference generalization on the `Amateur-Shattered` and `Amateur-Narrow` datasets. $*$ indicates results taken from Yuan et al. [2024].

	Environment	Metric	MODT(P)	MORvS(P)	BC(P)	MODULI*	MODULI	FP2-Pre	FP2-Mid
Amateur-Shattered	MO-Ant	HV ($\times 10^6$) ↑	5.74 ± 0.04	6.09 ± 0.01	4.20 ± 0.05	6.04 ± 0.01	6.04 ± 0.01	6.07 ± 0.06	$\mathbf{6.12 \pm 0.00}$
		SP ($\times 10^4$) ↓	2.47 ± 0.61	0.79 ± 0.14	2.29 ± 1.38	0.63 ± 0.23	0.59 ± 0.05	1.15 ± 0.65	$\mathbf{0.51 \pm 0.01}$
		PAS ↑	–	–	–	–	0.96 ± 0.00	0.92 ± 0.00	$\mathbf{0.97 \pm 0.00}$
	MO-HalfCheetah	HV ($\times 10^6$) ↑	5.61 ± 0.01	5.74 ± 0.00	5.54 ± 0.00	$\mathbf{5.75 \pm 0.00}$	5.74 ± 0.01	5.74 ± 0.01	5.74 ± 0.01
		SP ($\times 10^4$) ↓	0.11 ± 0.01	0.09 ± 0.01	0.09 ± 0.01	0.13 ± 0.01	0.07 ± 0.02	0.12 ± 0.03	$\mathbf{0.07 \pm 0.02}$
		PAS ↑	–	–	–	–	0.98 ± 0.00	0.95 ± 0.04	$\mathbf{0.99 \pm 0.00}$
	MO-Hopper	HV ($\times 10^7$) ↑	1.46 ± 0.00	1.69 ± 0.01	1.63 ± 0.00	2.01 ± 0.01	2.01 ± 0.00	2.01 ± 0.04	$\mathbf{2.11 \pm 0.00}$
		SP ($\times 10^5$) ↓	0.54 ± 0.58	0.18 ± 0.01	0.11 ± 0.05	0.15 ± 0.04	0.16 ± 0.01	$\mathbf{0.14 \pm 0.02}$	0.16 ± 0.01
		PAS ↑	–	–	–	–	0.28 ± 0.00	$\mathbf{0.92 \pm 0.01}$	0.59 ± 0.01
	MO-Walker	HV ($\times 10^4$) ↑	3.86 ± 0.07	4.99 ± 0.00	3.42 ± 0.02	5.03 ± 0.00	5.04 ± 0.03	5.09 ± 0.04	$\mathbf{5.15 \pm 0.01}$
		SP ($\times 10^0$) ↓	11.13 ± 3.58	0.20 ± 0.03	4.56 ± 0.08	0.29 ± 0.01	0.12 ± 0.02	0.18 ± 0.03	$\mathbf{0.10 \pm 0.01}$
		PAS ↑	–	–	–	–	$\mathbf{0.99 \pm 0.00}$	0.90 ± 0.01	0.92 ± 0.01
	MO-Swimmer	HV ($\times 10^6$) ↑	0.75 ± 0.00	2.81 ± 0.00	2.81 ± 0.00	$\mathbf{3.17 \pm 0.00}$	3.09 ± 0.00	3.03 ± 0.02	3.05 ± 0.01
		SP ($\times 10^4$) ↓	6.75 ± 0.23	$\mathbf{1.26 \pm 0.06}$	1.89 ± 0.05	8.85 ± 0.77	4.96 ± 0.08	8.05 ± 2.92	5.04 ± 0.05
		PAS ↑	–	–	–	–	0.94 ± 0.00	0.87 ± 0.03	$\mathbf{0.99 \pm 0.00}$
	MO-Hopper-3obj	HV ($\times 10^{10}$) ↑	1.48 ± 0.02	2.84 ± 0.04	1.34 ± 0.02	3.02 ± 0.03	3.23 ± 0.01	3.46 ± 0.06	$\mathbf{3.49 \pm 0.04}$
		SP ($\times 10^5$) ↓	0.56 ± 0.08	0.13 ± 0.01	0.59 ± 0.08	0.10 ± 0.01	0.10 ± 0.00	0.07 ± 0.03	$\mathbf{0.01 \pm 0.00}$
		PAS ↑	–	–	–	–	0.97 ± 0.01	0.96 ± 0.03	$\mathbf{0.99 \pm 0.00}$
Amateur-Narrow	MO-Ant	HV ($\times 10^6$) ↑	5.17 ± 0.03	5.80 ± 0.03	4.19 ± 0.10	5.88 ± 0.03	5.86 ± 0.14	5.97 ± 0.03	$\mathbf{5.97 \pm 0.03}$
		SP ($\times 10^4$) ↓	$\mathbf{0.43 \pm 0.10}$	0.51 ± 0.08	4.42 ± 1.44	0.62 ± 0.02	0.67 ± 0.14	0.98 ± 0.07	0.46 ± 0.06
		PAS ↑	–	–	–	–	0.96 ± 0.00	0.65 ± 0.01	$\mathbf{0.96 \pm 0.00}$
	MO-HalfCheetah	HV ($\times 10^6$) ↑	4.29 ± 0.00	5.55 ± 0.00	5.08 ± 0.01	5.71 ± 0.00	5.71 ± 0.00	5.73 ± 0.00	$\mathbf{5.73 \pm 0.00}$
		SP ($\times 10^4$) ↓	1.02 ± 0.44	$\mathbf{0.02 \pm 0.00}$	0.02 ± 0.03	0.06 ± 0.02	0.10 ± 0.02	1.92 ± 0.18	0.10 ± 0.00
		PAS ↑	–	–	–	–	0.96 ± 0.01	0.92 ± 0.00	$\mathbf{0.97 \pm 0.00}$
	MO-Hopper	HV ($\times 10^7$) ↑	1.73 ± 0.01	1.69 ± 0.01	0.94 ± 0.34	1.99 ± 0.00	1.91 ± 0.07	2.07 ± 0.00	$\mathbf{2.07 \pm 0.00}$
		SP ($\times 10^5$) ↓	0.27 ± 0.07	0.25 ± 0.03	5.81 ± 8.21	0.22 ± 0.06	0.25 ± 0.01	0.08 ± 0.03	$\mathbf{0.07 \pm 0.03}$
		PAS ↑	–	–	–	–	0.80 ± 0.01	0.49 ± 0.01	$\mathbf{0.86 \pm 0.00}$
	MO-Walker	HV ($\times 10^4$) ↑	3.24 ± 0.016	4.83 ± 0.00	2.83 ± 0.01	4.91 ± 0.01	4.89 ± 0.06	$\mathbf{5.17 \pm 0.01}$	5.17 ± 0.01
		SP ($\times 10^0$) ↓	5.64 ± 3.72	0.22 ± 0.01	0.01 ± 0.00	0.24 ± 0.01	0.27 ± 0.00	0.19 ± 0.04	$\mathbf{0.09 \pm 0.00}$
		PAS ↑	–	–	–	–	0.99 ± 0.01	0.78 ± 0.01	$\mathbf{0.99 \pm 0.00}$
	MO-Swimmer	HV ($\times 10^6$) ↑	0.66 ± 0.03	2.95 ± 0.00	2.80 ± 0.00	3.15 ± 0.00	3.06 ± 0.01	3.11 ± 0.01	$\mathbf{3.11 \pm 0.01}$
		SP ($\times 10^4$) ↓	14.25 ± 3.73	$\mathbf{1.91 \pm 0.01}$	2.95 ± 0.08	5.79 ± 0.43	4.93 ± 0.81	4.39 ± 1.11	4.10 ± 0.36
		PAS ↑	–	–	–	–	$\mathbf{0.92 \pm 0.00}$	0.89 ± 0.01	0.89 ± 0.00
	MO-Hopper-3obj	HV ($\times 10^{10}$) ↑	1.85 ± 0.04	2.27 ± 0.05	2.26 ± 0.02	2.90 ± 0.05	3.05 ± 0.06	3.58 ± 0.03	$\mathbf{3.58 \pm 0.03}$
		SP ($\times 10^5$) ↓	3.12 ± 1.21	0.07 ± 0.01	0.09 ± 0.00	0.12 ± 0.02	0.08 ± 0.01	0.09 ± 0.01	$\mathbf{0.07 \pm 0.01}$
		PAS ↑	–	–	–	–	0.98 ± 0.00	0.99 ± 0.00	$\mathbf{0.99 \pm 0.00}$

Table 14: Complete fine-tuning results on out-of-distribution preferences for the `Expert-Shattered` and `Expert-Narrow` datasets.

Dataset		Expert-Shattered			Expert-Narrow		
Environment	Metric	FP2-Mid (Zero-shot)	Scratch	FP2-Post	FP2-Mid (Zero-shot)	Scratch	FP2-Post
MO-Ant	HV ($\times 10^6$) ↑	5.70 ± 0.06	5.04 ± 0.25	$\mathbf{5.83 \pm 0.06}$	5.40 ± 0.10	5.17 ± 0.10	$\mathbf{5.65 \pm 0.05}$
	SP ($\times 10^4$) ↓	3.37 ± 0.62	3.53 ± 2.66	$\mathbf{1.69 \pm 0.79}$	12.20 ± 9.83	9.12 ± 6.74	$\mathbf{6.47 \pm 3.54}$
	PAS ↑	0.85 ± 0.00	0.36 ± 0.03	$\mathbf{0.91 \pm 0.01}$	0.94 ± 0.02	0.39 ± 0.03	$\mathbf{0.95 \pm 0.00}$
MO-HalfCheetah	HV ($\times 10^6$) ↑	5.69 ± 0.03	4.00 ± 0.08	$\mathbf{5.73 \pm 0.07}$	5.47 ± 0.03	5.15 ± 0.01	$\mathbf{5.54 \pm 0.33}$
	SP ($\times 10^4$) ↓	0.09 ± 0.04	0.34 ± 0.03	$\mathbf{0.08 \pm 0.08}$	1.77 ± 0.53	2.98 ± 0.03	$\mathbf{0.04 \pm 0.00}$
	PAS ↑	$\mathbf{0.98 \pm 0.00}$	0.79 ± 0.01	0.97 ± 0.01	$\mathbf{0.99 \pm 0.00}$	0.41 ± 0.01	0.96 ± 0.03
MO-Hopper	HV ($\times 10^6$) ↑	$\mathbf{1.95 \pm 0.07}$	0.91 ± 0.55	1.89 ± 0.00	1.90 ± 0.01	0.33 ± 0.06	$\mathbf{2.00 \pm 0.00}$
	SP ($\times 10^4$) ↓	0.14 ± 0.02	$\mathbf{0.07 \pm 0.06}$	0.13 ± 0.02	1.71 ± 0.26	1.32 ± 0.38	$\mathbf{0.78 \pm 0.48}$
	PAS ↑	0.54 ± 0.07	0.28 ± 0.01	$\mathbf{0.60 \pm 0.01}$	0.98 ± 0.01	0.29 ± 0.01	$\mathbf{0.99 \pm 0.00}$
MO-Walker	HV ($\times 10^6$) ↑	4.87 ± 0.25	4.80 ± 0.11	$\mathbf{5.05 \pm 0.13}$	2.46 ± 0.14	3.65 ± 0.51	$\mathbf{3.93 \pm 0.03}$
	SP ($\times 10^4$) ↓	0.20 ± 0.18	1.21 ± 0.40	$\mathbf{0.15 \pm 0.11}$	1.94 ± 0.34	1.88 ± 0.29	$\mathbf{0.83 \pm 1.30}$
	PAS ↑	0.99 ± 0.00	0.17 ± 0.01	$\mathbf{0.99 \pm 0.01}$	0.94 ± 0.01	0.33 ± 0.02	$\mathbf{0.98 \pm 0.01}$
MO-Swimmer	HV ($\times 10^6$) ↑	3.03 ± 0.04	2.79 ± 0.02	$\mathbf{3.09 \pm 0.09}$	2.41 ± 0.14	2.46 ± 0.01	$\mathbf{2.91 \pm 0.42}$
	SP ($\times 10^4$) ↓	11.13 ± 12.01	71.33 ± 36.67	$\mathbf{9.15 \pm 13.73}$	$\mathbf{5.91 \pm 3.63}$	17.59 ± 2.85	8.77 ± 0.61
	PAS ↑	0.99 ± 0.00	0.67 ± 0.00	$\mathbf{0.99 \pm 0.00}$	0.92 ± 0.01	0.31 ± 0.00	$\mathbf{0.96 \pm 0.02}$
MO-Hopper-3obj	HV ($\times 10^6$) ↑	3.63 ± 0.09	2.13 ± 0.04	$\mathbf{3.66 \pm 0.01}$	2.42 ± 0.07	1.92 ± 0.08	$\mathbf{2.55 \pm 0.10}$
	SP ($\times 10^4$) ↓	0.60 ± 0.44	0.10 ± 0.00	$\mathbf{0.03 \pm 0.00}$	0.10 ± 0.10	0.99 ± 0.28	$\mathbf{0.04 \pm 0.03}$
	PAS ↑	0.97 ± 0.01	0.70 ± 0.01	$\mathbf{0.97 \pm 0.00}$	0.99 ± 0.00	0.76 ± 0.01	$\mathbf{0.99 \pm 0.00}$

to the inputs of its two actors and critic networks, replace online interaction with sliding-window samples from D4MORL trajectories, and remove online- and vision-specific components (best-of-n actor, non-chunking variant, and image encoders), keeping all other components unchanged. Both adaptations use the same training steps, batch size, learning rate, and seeds as FP2 pre-training (Table 7), with other hyperparameters following the official implementations. Table 15 reports HV on `Expert`-Full, the largest and highest-quality dataset in our benchmark. These results are controlled fixed-data comparisons and should not be interpreted as evaluations of the original methods in their intended online or offline-to-online regimes.

Table 15: HV comparison with fixed-data adaptations of value-based RL methods on `Expert`-Full.

Method	MO-Ant (10^6)	MO-HalfCheetah (10^6)	MO-Hopper (10^7)	MO-Swimmer (10^6)	MO-Walker2d (10^4)	MO-Hopper-3obj (10^{10})
Offline-COLA	0.65 ± 0.08	3.80 ± 0.02	0.01 ± 0.00	2.14 ± 0.38	1.99 ± 0.00	1.40 ± 0.15
MO-Q-chunking	0.58 ± 0.09	5.32 ± 0.10	0.02 ± 0.01	2.19 ± 0.19	3.08 ± 0.00	0.08 ± 0.01
FP2	$\mathbf{6.40 \pm 0.01}$	$\mathbf{5.79 \pm 0.01}$	$\mathbf{2.10 \pm 0.01}$	$\mathbf{3.26 \pm 0.01}$	$\mathbf{5.22 \pm 0.01}$	$\mathbf{3.76 \pm 0.02}$

D.6 Exploratory zero-shot transfer to multi-objective Meta-World

As an exploratory study beyond locomotion, we construct multi-objective extensions of ten Meta-World environments Yu et al. [2020], using control-level objectives analogous to those in MO-MuJoCo: movement speed and control cost. Following Meta-World, the observation consists of the positions of the end-effector, object, and goal, and the action specifies the end-effector position. All ten environments are evaluated zero-shot: FP2 and MODULI trained on all six MO-MuJoCo environments are deployed without using any Meta-World data for training or adaptation, under uniformly distributed two-objective preferences. Since no target-task data are used, Scratch is not applicable in this setting. The evaluation measures control-level trade-offs rather than task success, since neither model is trained on task-specific subgoals. Table 16 summarizes the evaluation protocol. As shown in Table 17, FP2 achieves higher HV and lower SP than MODULI on all ten environments.

Table 16: Evaluation protocol for zero-shot transfer to multi-objective Meta-World.

Item	**Setting**
Target environments	Ten Meta-World tasks: `reach-v3`, `push-v3`, `pick-place-v3`, `door-open-v3`, `drawer-open-v3`, `drawer-close-v3`, `button-press-topdown-v3`, `peg-insert-side-v3`, `window-open-v3`, `window-close-v3`
Source models	FP2 and MODULI trained on all six MO-MuJoCo environments (MO-Ant, MO-HalfCheetah, MO-Hopper, MO-Swimmer, MO-Walker2d, MO-Hopper-3obj)
Target-task data	None; all ten tasks are evaluated zero-shot
Objectives	Movement speed and control cost
Observation	Positions of the end-effector, object, and goal
Action	End-effector position
Preferences	Uniformly distributed two-objective preferences

Table 17: Zero-shot transfer from MO-MuJoCo to ten multi-objective Meta-World environments.

Metric	Method	Reach	Push	Pick-Place	Door-Open	Drawer-Open	Drawer-Close	Button-Press	Peg-Insert	Window-Open	Window-Close	Avg
HV ↑	MODULI	1.38	1.40	1.39	1.44	1.36	1.37	1.42	1.40	1.36	1.42	1.39
	FP2	**1.65**	**1.53**	**1.52**	**1.50**	**1.46**	**1.49**	**1.50**	**1.55**	**1.47**	**1.51**	**1.52**
SP ↓	MODULI	28.3	33.6	4.4	4.7	10.2	14.4	12.6	11.0	4.5	8.7	13.24
	FP2	**0.8**	**4.1**	**1.7**	**4.1**	**2.2**	**6.4**	**1.6**	**2.4**	**4.0**	**3.7**	**3.10**

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 summarize the key challenges, our contributions, and the main experimental findings.

   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: See Sec. 6.

   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: [N/A]

Justification:

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: We provide the experimental details and properly cite the open-source assets and materials needed to reproduce our experiments. The code will be made publicly available upon acceptance.

   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 a public dataset and will make our code publicly available upon acceptance.

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: See Appendix.

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: All reported experimental results are averaged over three random seeds, with standard deviations shown as error bars or reported in the form of mean $\pm$ std.

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: See Appendix B.

   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: We fully adhere to 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: [N/A]

    Justification: This work is foundational research on multi-objective reinforcement learning methods and does not target a specific real-world application or deployment scenario. As such, it does not have direct or immediate societal impacts to assess.

    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 poses no such risks.

    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: The paper properly credits the creators of all assets used, including code, data, and models, and provides appropriate citations to the relevant literature.

    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: [N/A]

    Justification: The code will be made publicly available upon acceptance. No additional assets are currently available.

    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:

    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:

    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: The core method developed in this work does not use LLMs as essential, original, or non-standard components.

    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.