What Kind of Diffusion Models Do We Need in Online Reinforcement Learning?

Zihao Wu[1], **Hongyao Tang**[1]*, **Yi Ma**[2], **Yan Zheng**[1], **Jianrong Wang**[1], **Jianye Hao**[1,3]
[1]School of Computer Software, Tianjin University
[2]School of Computer and Information Technology, Shanxi University
[3]MemoraX AI

Abstract

Score matching and flow matching exhibit powerful expressive capabilities in continuous control tasks. However, since the log probability densities of their generated actions cannot be directly accessed, they introduce substantial challenges to the implementation and optimization of Maximum-Entropy Reinforcement Learning (RL). To address this critical limitation, the RL community has developed a wide range of diffusion-based RL algorithms, leading to a widespread perception that the integration of diffusion models and RL has achieved remarkable success. Yet, is this optimistic conclusion truly well-founded? In this paper, we propose a systematic taxonomy for existing mainstream diffusion RL methods that consists of three distinct categories, and we conduct in-depth theoretical analyses of their inherent limitations and core defects.Our analysis reveals that existing methods either collapse the intended stochastic generative policy into action maximization, require costly long-horizon sampling in high-dimensional action spaces, or rely heavily on proposal distributions whose coverage fundamentally limits policy improvement. Building on the comprehensive analysis above, we propose a novel method **Coupled Flow**. The core insight of our method is to leverage linear ordinary differential equations to maintain fully tractable log-likelihood calculation, while adopting coupled generation mechanisms to effectively capture complex distributions. In principle, **Coupled Flow** integrates into any maximum-entropy RL framework. With SAC we evaluate 5 MuJoCo tasks, 7 DeepMind Control tasks, and 3 HumanoidBench tasks, raising the return by 108% on `h1-stair-v0` and 28% on `h1-slide-v0` relative to SAC. With XQC we evaluate 5 MuJoCo tasks, 7 DeepMind Control tasks, 14 HumanoidBench tasks, and 10 MyoSuite tasks, raising the return by 117% on `humanoid-run` relative to DIME and 37% on `h1-balance_hard-v0` relative to SimbaV2. We make our code available at https://github.com/wzhhasadream/diffusionrl.

1 Introduction

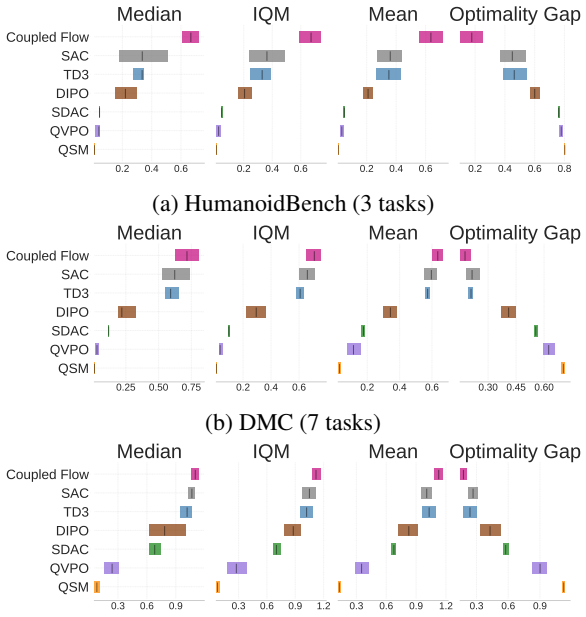
Iterative generative policies built on flow matching and score matching (Song et al., 2021; Lipman et al., 2023) have become a major paradigm in reinforcement learning (RL) (Wang et al., 2024; Yang et al., 2023; Lv et al., 2025; Psenka et al., 2025; Ma et al., 2025; Ding et al., 2024). Instead of emitting an action in one pass as a Gaussian actor does (Haarnoja et al., 2018), they define the policy through a stepwise generative process that transports a simple base noise into a state-conditioned action distribution. The added expressivity captures multimodal behavior that unimodal policies cannot represent, which is what high-dimensional control needs.

*Corresponding author: tanghongyao@tju.edu.cn.

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

Bringing such policies into online RL runs into two difficulties. The sampling process they define does not admit a tractable likelihood, so likelihood-based MaxEnt policy optimization and density-ratio correction are not directly available. Offline training (Park et al., 2025; Tiofack et al., 2026; Agrawalla et al., 2025) can lean on high-quality reference actions; online RL cannot, and the policy has to discover effective behavior from scratch. A large literature has grown around these constraints, including *SDAC* (Ma et al., 2025), *FlowRL* (Lv et al., 2025), *DIPO* (Yang et al., 2023), *DACER* (Wang et al., 2024), *QSM* (Psenka et al., 2025), *DIME* (Celik et al., 2025), and *QVPO* (Ding et al., 2024).

To understand the essential advantages and defects of these existing methods beneath their diverse technical designs, in this paper we first propose a systematic taxonomy that categorizes these existing methods into three major groups: (i) **Diffusion models as reparameterized policies** (Celik et al., 2025; Wang et al., 2024): This branch treats the sampler as a differentiable map and backpropagates through Euler integration. (ii) **Q-guided sampling based on overdamped Langevin dynamics** (Psenka et al., 2025; Yang et al., 2023): These methods build the diffusion framework on the classical overdamped Langevin equation, which yields multimodal policies. (iii) **Q-guided sampling via resampling** (Ma et al., 2025; Ding et al., 2024; Lv et al., 2025): These methods keep the vanilla diffusion training objective and reweight its loss by Q-values, so that the actor favors high-quality actions.



(a) HumanoidBench (3 tasks)

(b) DMC (7 tasks)

(c) MuJoCo (5 tasks)

Figure 1: **Coupled Flow performs the best across 15 HumanoidBench, DMC, MuJoCo tasks**. Aggregate *rliable* metrics (Agarwal et al., 2021) on each suite's reference-normalized scores (Appendix E.2), with 95% stratified bootstrap CIs.

This raises the question we take as our starting point: which of the three families delivers the best policy, and why? We answer it with a theoretical derivation and a controlled empirical study. Our analysis builds on the classical expansion framework for non-log-concave sampling (Huang et al., 2024). From this perspective we prove that the performance of the latter two families is governed by the dimensionality of the action space: their error grows with the action dimension, an ordering that our experiments reproduce. Reparameterized policies are therefore the only route to a genuinely stochastic diffusion policy in high dimensions.

That route comes with a condition: reparameterized diffusion policies are only usable in MaxEnt RL if the action log-density can be computed. To meet it, we propose **Coupled Flow**, a policy defined by a linear ordinary differential equation (ODE) whose drift is coupled across latent components. The coupling keeps the action distribution multimodal, while linearity makes the log-density available in closed form from the trace of the drift. **The result is a reparameterized actor that plugs into the MaxEnt objective directly, without auxiliary proposal distributions or long Langevin chains.**

To validate the analysis and evaluate **Coupled Flow**, we run experiments on four benchmarks: MuJoCo (Todorov et al., 2012), the DeepMind Control (DMC) Suite (Tassa et al., 2018), HumanoidBench (Sferrazza et al., 2024), and MyoSuite (Caggiano et al., 2022). On the SAC backbone we compare with SAC (Haarnoja et al., 2018), TD3 (Fujimoto et al., 2018), QSM (Psenka et al., 2025), QVPO (Ding et al., 2024), DIPO (Yang et al., 2023), and SDAC (Ma et al., 2025); on the XQC backbone we compare with XQC (Palenicek et al., 2025), SimbaV2 (Lee et al., 2025), TD-MPC2 (Hansen et al., 2024), MR.Q (Fujimoto et al., 2025), SAC Flow (Zhang et al., 2026), DIME (Celik et al., 2025), and PPO (Schulman et al., 2017). Figure 1 reports the aggregate rliable metrics of the 15-task SAC comparison on the same normalized scale, where **Coupled Flow** ranks first on the interquartile mean of all three suites: 1.12 against 1.05 for SAC on MuJoCo, 0.71 against 0.66 on DMC, and 0.67 against 0.36 on HumanoidBench. Figure 2 repeats the comparison on the stronger XQC backbone, where

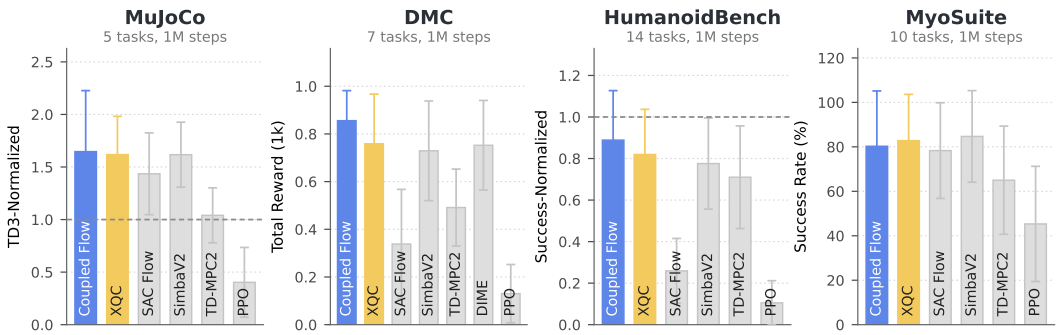


Figure 2: **Coupled Flow scored on four benchmarks with the reference normalization of each suite**. Bars report the score at 1M environment steps: TD3-normalized return on MuJoCo, return in thousands on DeepMind Control, success-normalized return on HumanoidBench, and success rate on MyoSuite. Coupled Flow is integrated with XQC. Error bars are 95% bootstrap intervals over tasks. **Coupled Flow** reaches 0.86 on DeepMind Control and 0.89 on HumanoidBench, the highest among the compared methods.

Coupled Flow leads DMC (0.86 against 0.76 for XQC) and HumanoidBench (0.89 against 0.82), is on par with XQC and SimbaV2 on MuJoCo, and raises `humanoid-run` by 150% and `h1-run-v0` by 118% in raw return. Appendix H reports every curve.

The contributions of this paper are summarized as follows:

- We analyze the three families of diffusion-based policies for online RL, propose a taxonomy grounded in what each family optimizes, and identify the failure mode of each.
- We propose **Coupled Flow**, a flow model whose log-density is available in closed form and whose coupled drift retains the expressivity needed for multimodal policies.
- We integrate **Coupled Flow** into the maximum-entropy RL framework and evaluate it on four benchmarks. On the SAC backbone it attains the highest normalized score on MuJoCo, DMC, and HumanoidBench; on the XQC backbone it leads DMC and HumanoidBench.

2 Related Work

Reparameterized Policies in Continuous Control Reparameterized stochastic policies have become a standard design choice in continuous-control RL because they permit low-variance policy-gradient estimation while keeping action sampling differentiable (Kingma & Welling, 2013). Representative maximum-entropy algorithms such as SAC (Haarnoja et al., 2018) instantiate this idea with tractable Gaussian policies, whereas alternative bounded distributions such as Beta policies have also been explored to better match action constraints (Chou et al., 2017). Despite their efficiency and clean likelihood evaluation, these conventional policy families are usually limited in expressivity and often struggle to represent strongly multimodal action distributions. This limitation motivates the use of more expressive generative policies, but also raises a central question: how can one preserve the optimization benefits of reparameterized policies while moving beyond simple unimodal distributions?

Diffusion and Flow Policies Score matching and flow matching provide powerful tools for modeling complex distributions through iterative generative processes (Song et al., 2021; Lipman et al., 2023). Their strong expressive capacity has inspired a rapidly growing body of RL methods that replace direct Gaussian actors with diffusion- or flow-based action samplers (Wang et al., 2024; Yang et al., 2023; Lv et al., 2025; Psenka et al., 2025; Ma et al., 2025; Ding et al., 2024). In offline RL, such models can be trained from high-quality action data or behavior datasets, making diffusion policies a natural fit for policy extraction and behavior-regularized improvement (Park et al., 2025; Tiofack et al., 2026; Agrawalla et al., 2025). Online RL is fundamentally different: there is no fixed expert action dataset, the target action distribution changes throughout training, and policy improvement must be driven by critic feedback collected from scratch. As a result, the success of diffusion policies in offline settings does not directly transfer to online maximum-entropy RL.

Online Diffusion RL and Likelihood Bottlenecks Recent online diffusion RL algorithms address the absence of reference data and tractable action likelihoods through different approximations. Methods such as *DIME* (Celik et al., 2025) and *DACER* (Wang et al., 2024) treat diffusion sampling as a reparameterized policy and introduce surrogate entropy or approximate density estimates. *QSM* (Psenka et al., 2025) and *DIPO* (Yang et al., 2023) instead guide action generation through Langevin-style sampling from critic-induced energy landscapes. Other methods, including *SDAC*, *QVPO*, and *FlowRL* (Ma et al., 2025; Ding et al., 2024; Lv et al., 2025), construct Q-weighted resampling objectives to bias generative training toward high-value actions. These approaches demonstrate that expressive iterative samplers can be integrated with actor-critic learning.

3 Preliminaries

Maximum-Entropy RL We consider a continuous-control Markov Decision Process (MDP) $\mathcal{M} = (\mathbb{S}, \mathbb{A}, p, r, \gamma)$ (Bellman, 1957), where $\mathbb{S} \subseteq \mathbb{R}^{d_s}$ and $\mathbb{A} \subseteq \mathbb{R}^{d_a}$ denote the state and action spaces. Throughout this paper, we use h to index environment interaction steps and reserve $t \in [0, 1]$ for the internal time of diffusion or flow samplers.

Maximum-entropy RL augments the standard return objective with an entropy regularizer:

$$J_{\text{MaxEnt}}(\pi) = \mathbb{E}_\pi \left[\sum_{h=0}^{\infty} \gamma^h \big(r(s_h, a_h) + \alpha \mathcal{H}(\pi(\cdot \mid s_h)) \big) \right],$$

where $\mathcal{H}(\pi(\cdot \mid s)) = -\mathbb{E}_{a \sim \pi(\cdot|s)}[\log \pi(a \mid s)]$ and $\alpha > 0$ is the temperature coefficient. For a fixed critic Q, the corresponding soft policy improvement target is the Boltzmann distribution

$$\pi_{\text{B}}(a \mid s) = \frac{1}{Z(s)} \exp\left(\frac{Q(s, a)}{\alpha} \right), \qquad Z(s) = \int_{\mathbb{A}} \exp\left(\frac{Q(s, a)}{\alpha} \right) da.$$

Thus, online maximum-entropy RL requires both high-quality action generation and access to the policy log-density $\log \pi(a \mid s)$.

Implicit Generative Policies Diffusion and flow policies define actions through an implicit generative process rather than a closed-form density. Let $\pi_t^{\rightarrow}(\cdot \mid s)$ denote the time-evolving conditional marginal distribution over actions, where $\pi_0^{\rightarrow}(\cdot \mid s)$ is a simple base distribution and $\pi_1^{\rightarrow}(\cdot \mid s)$ is the final policy distribution. A general SDE sampler can be written as

$$dA_t^{\rightarrow} = b_t(A_t^{\rightarrow}, s)dt + \sigma_t(A_t^{\rightarrow}, s)dW_t, \qquad A_0 \sim \pi_0^{\rightarrow}(\cdot \mid s),$$

where W_t is a Brownian motion. Score-based policies instantiate the drift using a learned score network, while flow-based policies use deterministic velocity fields. In the ODE case,

$$dA_t^{\rightarrow} = v_t(A_t^{\rightarrow}, s)dt, \qquad A_0 \sim \pi_0^{\rightarrow}(\cdot \mid s), \quad t \in [0, 1]. \tag{1}$$

Given the initial noise, Equation (1) induces a deterministic mapping $a = f_\theta(s, \epsilon)$, where $\epsilon \sim p(\epsilon)$ and the mapping is implemented by numerical integration. This viewpoint will be used to connect diffusion or flow policies with reparameterized policy optimization in the next section.

Reparameterized Policies A common way to parameterize stochastic policies in continuous control is to express action sampling as a differentiable transformation of exogenous noise:

$$a = f_\theta(s, \epsilon), \qquad \epsilon \sim p(\epsilon).$$

This representation induces the policy distribution as the pushforward measure

$$\pi_\theta(\cdot \mid s) = (f_\theta(s, \cdot))_{\#} p(\epsilon).$$

When the density of this pushforward distribution is available, as in Gaussian policies used by SAC (Haarnoja et al., 2018). A typical instantiation used is

$$a = \tanh\big(\mu_\theta(s) + \sigma_\theta(s) \odot \epsilon\big), \qquad \epsilon \sim \mathcal{N}(0, I),$$

where $\odot$ denotes elementwise multiplication.

4 Theoretical Analysis: Three Types of Diffusion-based Online RL Methods

We analyze existing diffusion-based online RL methods through three representative paradigms: reparameterized policy optimization, Langevin-based Q-guided sampling, and Q-weighted resampling. Although these methods differ substantially in implementation, they all attempt to bypass the same obstacle: expressive diffusion or flow policies usually do not provide tractable action likelihoods for maximum-entropy RL. A more detailed discussion is provided in Appendix C.

4.1 Treating ODE/SDE Sampling as A Reparameterized Policy

This paradigm regards ODE/SDE sampling as a reparameterized policy: the action is generated by a differentiable map $a = f_\theta(s, \epsilon)$, where the map is implemented by numerical simulation rather than a single feedforward network. Representative methods include *DACER* and *DIME* (Wang et al., 2024; Celik et al., 2025). The corresponding actor objective can be written in either a TD3-style or SAC-style form:

$$\text{TD3-style:} \quad \max_\theta \ \mathbb{E}_{s\sim\mathcal{D}} \, \mathbb{E}_{\epsilon\sim p(\epsilon)} \Big[Q\big(s, f_\theta(s,\epsilon)\big) \Big] \tag{2}$$

$$\text{SAC-style:} \quad \max_\theta \ \mathbb{E}_{s\sim\mathcal{D}} \, \mathbb{E}_{\epsilon\sim p(\epsilon)} \Big[Q\big(s, f_\theta(s,\epsilon)\big) - \alpha \log \left((f_\theta(\cdot, s))_{\#} p(\epsilon) \right) \Big] \tag{3}$$

Here, $\mathcal{D}$ denotes the replay buffer distribution and $(f_\theta(\cdot, s))_{\#} p(\epsilon)$ denotes the pushforward policy distribution. The SAC-style objective requires the exact log-density of this pushforward distribution, which is typically expensive for general ODE/SDE samplers. **Therefore, existing methods often adopt TD3-style objectives or surrogate entropy.**

Proposition 1 (Augmented-state equivalence)**.** *Consider the optimization objective*

$$\max_\theta \ \mathbb{E}_{s\sim\mathcal{D},\ \epsilon\sim p(\epsilon)} \Big[Q\big(s, f_\theta(s,\epsilon)\big) \Big].$$

We define the augmented state $\tilde{s} := (s, \epsilon)$ and the corresponding induced deterministic policy $\tilde{\pi}_\theta(\tilde{s}) := f_\theta(s, \epsilon)$. Maximizing the aforementioned objective is equivalent to searching for an action that maximizes the Q-value for every augmented state $\tilde{s}$:

$$\tilde{\pi}^\star(\tilde{s}) \in \arg\max_{a\in\mathbb{A}} Q(s, a).$$

> **Takeaway 1.** *Why the TD3-style paradigm is infeasible.* In fact, the formulation in Equation (2) fails to produce genuine multimodal policies and is essentially identical to standard deterministic policies.

4.2 Q-guided Sampling via Overdamped Langevin Dynamics

This family aims to sample actions from the critic-induced Boltzmann distribution $\pi_{\text{Boltz}}(a \mid s) \propto \exp(Q(s, a)/\alpha)$. Given a fixed state s, the overdamped Langevin SDE over the action space is

$$\mathrm{d}A_t^{\rightarrow} = \nabla_a Q\big(s, A_t^{\rightarrow}\big)\,\mathrm{d}t + \sqrt{2\alpha}\,\mathrm{d}W_t, \qquad t \geq 0, \tag{4}$$

where W_t is Brownian motion and α is the temperature. Under favorable log-concave conditions, Langevin dynamics mix rapidly. For example, if $Q(s, \cdot)$ is ρ-strongly concave, then

$$\mathrm{KL}(\pi_t \,\|\, \pi_{\text{Boltz}}^{\rightarrow}) \leq e^{-2\rho t}\, \mathrm{KL}(\pi_0 \,\|\, \pi_{\text{Boltz}}^{\rightarrow}).$$

However, this guarantee no longer captures the hard cases encountered in multimodal control. When $Q(s, \cdot)$ is non-concave, mixing can become exponentially slow in the action dimension. Even for locally quadratic multimodal landscapes such as

$$Q(s, a) = \exp\left(-\sum_i \|a - a_i\|^2 \right),$$

sampling from the corresponding Boltzmann distribution is computationally intractable in the worst case. Practical methods therefore increase the number of diffusion steps or use accelerated variants such as momentum or Metropolis correction (Dwivedi et al., 2019). For example, *QSM* adopts multiple Langevin iterations per state, whereas *DIPO* boosts computational efficiency by caching prior Langevin outputs as warm-start policies, thereby indirectly extending the diffusion steps.

> **Takeaway 2.** *Why the overdamped Langevin dynamics paradigm is infeasible.* Essentially, this category of methods directly conducts gradient ascent on actions. However, unlike the approach presented in Section 4.1, all states share a unified set of parameters. This causes gradient ascent to be executed within the parameter latent space, extending the diffusion/gradient steps to span the entire environmental horizon.

4.3 Q-guided Sampling via Resampling

Resampling-based methods also target high-value actions, but avoid simulating Langevin dynamics. They draw samples from a tractable proposal distribution $q(\cdot \mid s)$ and reweight each action by $w(s, a)$, typically derived from the critic. This induces the target policy form

$$\pi_1^{\rightarrow}(\cdot \mid s) \propto w(s, \cdot) q(\cdot \mid s).$$

The following theorem formalizes the corresponding weighted score/flow matching objective.

Theorem 1. *Let $f_\theta(s, a_t, t)$ be a neural network parameterizing the conditional score function or probability flow velocity field. Sample state s from distribution $\mathcal{D}$, action a_1' from $q(\cdot \mid s)$, and time t uniformly over $[0, 1]$. Each (s, a_1') is assigned a non-negative weight $w(s, a_1') \geq 0$. a_t is generated by SDE or ODE conditioned on (s, a_1'), with $a_1 \propto w(s, \cdot) q(\cdot|s)$ for its marginal distribution.*

For the stochastic differential equation (SDE) formulation, we adopt a weighted conditional denoising score matching objective:

$$\mathcal{L}^{\mathrm{SDE}} = \mathbb{E}_{\substack{s\sim\mathcal{D},\, t\sim\mathrm{Uniform}[0,1], \\ a_1'\sim q(\cdot|s),\; a_t\sim\pi_{t|1}^{\rightarrow}}} \Big[\lambda(t)\, w(s, a_1') \left\| f_\theta(s, a_t, t) - \nabla_{a_t} \log \pi_t^{\rightarrow}(a_t \mid s, a_1') \right\|^2 \Big], \tag{5}$$

where $\lambda(t)$ denotes a time-dependent weighting factor that compensates for scale variations of the score across different noise levels, and $w(s, a_1')$ is a reweighting term (e.g., derived from a soft policy improvement step).

For the probability flow ODE formulation, the corresponding reweighted velocity matching objective is:

$$\mathcal{L}^{\mathrm{ODE}} = \mathbb{E}_{\substack{s\sim\mathcal{D},\, t\sim\mathrm{Uniform}[0,1], \\ a_1'\sim q(\cdot|s),\; a_t\sim\pi_{t|1}^{\rightarrow}}} \Big[w(s, a_1') \left\| f_\theta(s, a_t, t) - v_t(a_t \mid s, a_1') \right\|^2 \Big]. \tag{6}$$

This paradigm avoids gradients of Q and does not require long-run convergence to a stationary distribution. Representative methods include *QVPO*, *SDAC*, and *FlowRL*. Its main difficulty is instead shifted to the choice of w and q. A broad proposal such as a uniform distribution can cover high-value regions but requires exponentially many samples in high-dimensional action spaces (He & Zhang, 2025); a narrow proposal such as a replay-buffer or behavior policy is efficient but may miss important Q-value modes. Thus, resampling methods face a coverage-efficiency trade-off.

> **Takeaway 3.** *Why resampling is infeasible.* At the Q-guidance stage, resampling diffusion uses a **zero-order (function-value-only) oracle**: it ranks or reweights action candidates using $Q(s, a)$, without accessing an action gradient or subgradient of Q. The actor is still trained by backpropagation through the resulting weighted diffusion loss. The resulting limitation is proposal coverage: without directional information from Q, a proposal that misses narrow high-value regions cannot recover them, while a sufficiently broad proposal can require exponentially many samples in high-dimensional action spaces.

5 Coupled Flow for Tractable Likelihood and Flexible Expressivity

In summary, our research motivation is well-defined: we aim to **develop a diffusion model whose likelihood can be computed exactly**, so that entropy-regularized policy optimization applies without a surrogate objective.

Drawing inspiration from the most fundamental linear ordinary differential equations (ODEs), in this work we propose to integrate multiple linear ODEs to formulate a composite ordinary differential equation. More specifically, the constructed ODE adopts a distinctive structure where each individual component evolves under the constraint of a linear ODE, meanwhile enabling dynamic interactions among different components. We term this type of flow framework as **Coupled Flow**.

5.1 Coupled Flow

Definition 1 (Coupled Flow)**.** *Let $\boldsymbol{x} \in \mathbb{R}^d$ and partition it into n disjoint components*

$$\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n), \qquad \boldsymbol{x}_i \in \mathbb{R}^{d_i}, \qquad \sum_{i=1}^{n} d_i = d.$$

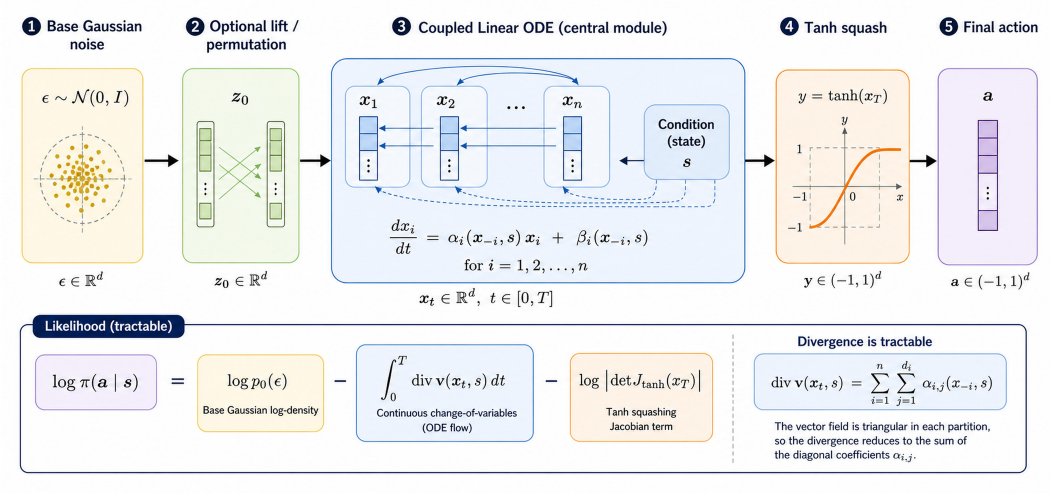


Figure 3: Schematic illustration of **Coupled Flow**. The policy keeps a reparameterized sampling path while computing likelihood through the tractable divergence of the coupled linear ODE.

Denote by $\boldsymbol{x}_{-i}$ the collection of all components except $\boldsymbol{x}_i$, i.e.,

$$\boldsymbol{x}_{-i} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{i-1}, \boldsymbol{x}_{i+1}, \ldots, \boldsymbol{x}_n).$$

A Coupled Flow *is defined as the ODE system*

$$\frac{d\boldsymbol{x}_i}{dt} = \boldsymbol{\alpha}_i(\boldsymbol{x}_{-i}, t) \odot \boldsymbol{x}_i + \boldsymbol{\beta}_i(\boldsymbol{x}_{-i}, t), \qquad i = 1, \ldots, n,$$

where $\boldsymbol{\alpha}_i(\cdot, t), \boldsymbol{\beta}_i(\cdot, t) : \mathbb{R}^{d-d_i} \to \mathbb{R}^{d_i}$ are vector-valued functions, and $\odot$ denotes elementwise multiplication.

Proposition 2 (Tractable divergence and likelihood)**.** *Let $v(\boldsymbol{x}, t)$ be the velocity field associated with a Coupled Flow. Then its divergence admits the closed-form expression*

$$\nabla \cdot v(\boldsymbol{x}, t) = \sum_{i=1}^{n} \sum_{j=1}^{d_i} \alpha_{i,j}(\boldsymbol{x}_{-i}, t),$$

where $\alpha_{i,j}$ denotes the j-th coordinate of $\boldsymbol{\alpha}_i$.

Consequently, the log-density evolves according to

$$\frac{d}{dt} \log p_t(\boldsymbol{x}_t) = -\nabla \cdot v(\boldsymbol{x}_t, t),$$

and hence

$$\log p_0(\boldsymbol{x}_0) = \log p_T(\boldsymbol{x}_T) + \int_0^T \nabla \cdot v(\boldsymbol{x}_t, t)\, dt.$$

Therefore, the likelihood of the Coupled Flow can be computed exactly.

The key reason is structural: although $\boldsymbol{\alpha}_i$ and $\boldsymbol{\beta}_i$ may depend on $\boldsymbol{x}_{-i}$, they do not depend on $\boldsymbol{x}_i$. Hence, the diagonal Jacobian contribution of the i-th component is exactly $\boldsymbol{\alpha}_i$, and the full divergence is obtained by summing these coordinates. This avoids Hutchinson-style stochastic trace estimation (Grathwohl et al., 2019) and makes entropy-regularized policy optimization directly applicable.

5.2 Practical Instantiation

Our method defines **Coupled Flow** as a time-homogeneous conditional flow policy governed by two hyperparameters: *num_ode* and *num_step*. *num_ode* sets the number of coupling blocks for latent action partitioning, and *num_step* denotes Euler discretization steps for applying the shared conditional velocity field. Without explicit time embedding, all steps reuse parameters, yielding a time-homogeneous, parameter-efficient ODE system.

Note that the case of *num_ode*=1 and *num_step*=1 simplifies the flow to a single observation-conditioned diagonal affine transform on standard Gaussian noise, identical to a vanilla diagonal Gaussian policy. Thus our framework strictly generalizes traditional Gaussian actors.

When *num_ode* is larger than the action dimension, we pad and permute latent variables into a higher-dimensional auxiliary space for coupled transformation, then project back to the original action space. This enhances coupling pattern flexibility while keeping the model lightweight. The full sampling and log-likelihood process is outlined in Algorithm 1.

6 Experiments

The core objective of the experiment is to validate the efficacy of the proposed **Coupled Flow** approach, verify the findings of our previous theoretical analysis under long-horizon training settings, and quantify its performance advantages through multi-dimensional analyses. Specifically, we focus on the following four key research questions:

> **Q1.** Can **Coupled Flow** effectively generate multimodal policies?
>
> **Q2.** Does the method consistently improve the training performance of MaxEnt RL algorithms on diverse continuous control tasks, and how does it perform against two typical multimodal policy learning methods?
>
> **Q3.** How sensitive is the proposed **Coupled Flow** to hyperparameter settings, and how do different heuristic designs affect the experimental results?
>
> **Q4.** Can **Coupled Flow** be integrated with the current strongest SOTA algorithm, XQC (Palenicek et al., 2025), to obtain stronger performance?

6.1 Multimodal Capabilities (Q1)

For Q1, the ability of **Coupled Flow** to generate multimodal policies depends on the entropy coefficient α. If the entropy coefficient is too small, effective exploration cannot be achieved, and the model will easily converge to a single modality. Refer to Appendix D for detailed experimental results.

6.2 Main Results (Q2)

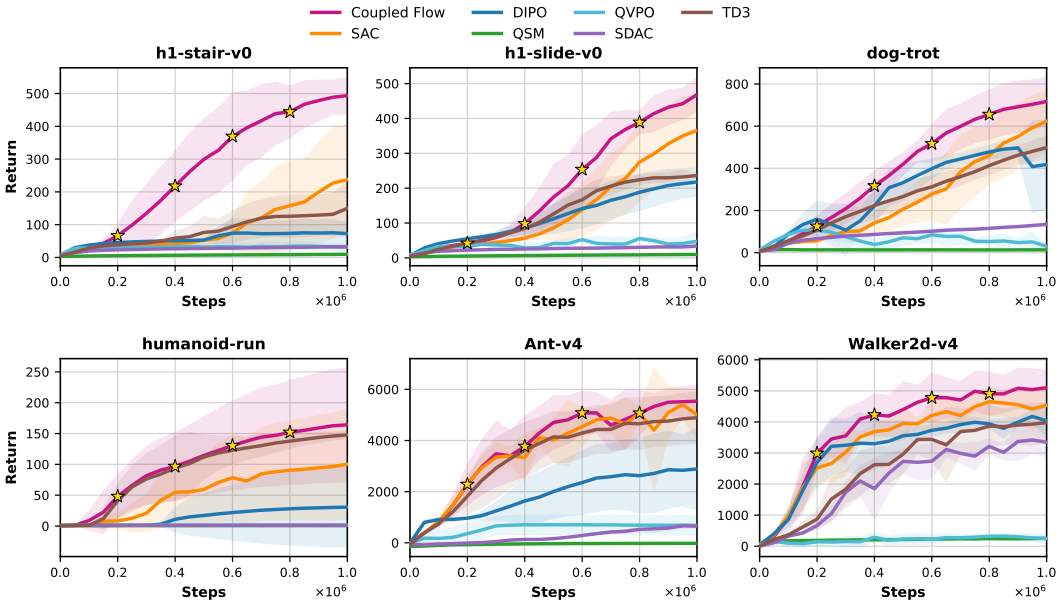


Figure 4: Main results on six representative high-dimensional tasks, two per benchmark: Humanoid-Bench (`h1-stair-v0`, `h1-slide-v0`), DMC (`dog-trot`, `humanoid-run`), and MuJoCo (`Ant-v4`, `Walker2d-v4`). Curves report mean $\pm$ standard deviation across seeds; the full per-environment results are in Appendix H.

For Q2, we evaluate **Coupled Flow** against representative baselines across five MuJoCo, seven DMC, and three HumanoidBench tasks. As shown in Fig. 4, **Coupled Flow** achieves stronger final performance and sample efficiency, especially in high-dimensional control. Consistent with our analysis, resampling-based and Langevin-based policies often become unstable in these settings and may collapse during training.

6.3 Sensitive Hyperparameter Analysis (Q3)

For Q3, we run a 5×5 grid over *num_ode*, *num_step*$\in \{2, \ldots, 6\}$ on Humanoid Run, with five seeds per configuration. Fig. 5 shows that **Coupled Flow** remains stable across the grid, while the Gaussian SAC reaches only 99.95. The absence of severe failures across 125 trials indicates that the method is not overly sensitive to these two hyperparameters.

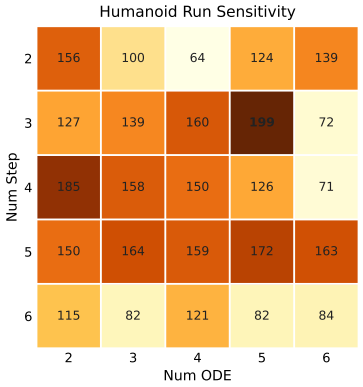


Figure 5: Parameter sensitivity on Humanoid Run.

6.4 Integration with XQC (Q4)

XQC is currently among the strongest model-free methods on proprioceptive continuous control. For Q4, we integrate **Coupled Flow** with this SOTA stack and compare against XQC together with *SimbaV2* (Lee et al., 2025), *TD-MPC2* (Hansen et al., 2024), *MR.Q* (Fujimoto et al., 2025), *DIME* (Celik et al., 2025), *SAC Flow* (Zhang et al., 2026), and *PPO* (Schulman et al., 2017). Fig. 6 collects six representative high-dimensional tasks, and **Coupled Flow** improves on XQC on every one of them.

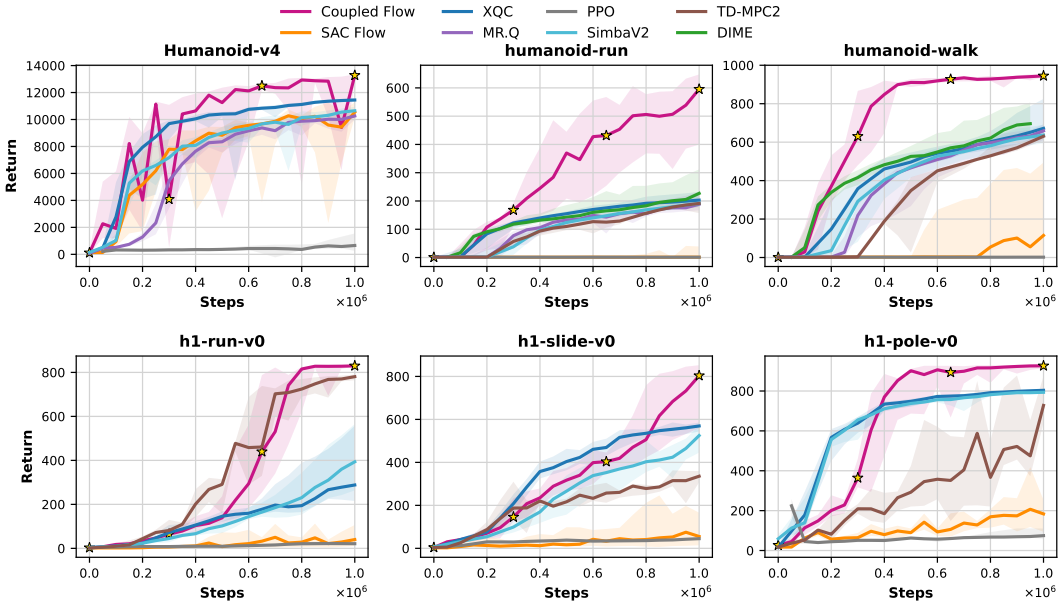


Figure 6: Comparison with XQC and other recent SOTA methods on representative high-dimensional tasks. Curves show IQM over seeds with 95% percentile bootstrap intervals.

7 Conclusion

We analyzed diffusion-based online reinforcement learning (RL) methods under three paradigms and identified their core bottlenecks: deterministic collapse, high-dimensional mixing overhead, and proposal-coverage sensitivity. Driven by this analysis, we propose **Coupled Flow**, a coupled linear ODE policy that enjoys tractable likelihood computation and flexible expressive capability. On the SAC backbone it attains the highest normalized score on MuJoCo, DMC, and HumanoidBench. On the stronger XQC backbone it leads DeepMind Control and HumanoidBench, matches XQC and SimbaV2 on MuJoCo, and more than doubles the raw return on the high-dimensional running tasks.

Limitations Our main evaluation covers MuJoCo, DMC, and HumanoidBench, with a supplementary MyoSuite study in Appendix H.4; extending **Coupled Flow** to real-world control is left for future work.

Acknowledgments

This work is supported by the National Natural Science Foundation of China (Grant Nos. 62422605, 62533021, 92370132, 62541610), the Tianjin Natural Science Foundation Project (No. 25JCQNJC01720), the National Key Research and Development Program of China (Grant No. 2026ZD0126104), the Fundamental Research Program of Shanxi Province (Serial No. 202503021212091), and the MoE Key Laboratory of Brain-inspired Intelligent Perception and Cognition, University of Science and Technology of China (No. 2521006). The authors declare no competing interests.

References

Agarwal, R., Schwarzer, M., Castro, P. S., Courville, A., and Bellemare, M. G. Deep reinforcement learning at the edge of the statistical precipice. *NeurIPS*, 2021.

Agrawalla, B., Nauman, M., Agrawal, K., and Kumar, A. floq: Training critics via flow-matching for scaling compute in value-based rl, 2025. URL https://arxiv.org/abs/2509.06863.

Bellman, R. A markovian decision process. *Indiana University Mathematics Journal*, 6(4):15, 1957.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym, 2016. URL https://arxiv.org/abs/1606.01540.

Caggiano, V., Wang, H., Durandau, G., Sartori, M., and Kumar, V. Myosuite: A contact-rich simulation suite for musculoskeletal motor control. In *Proceedings of the 4th Annual Learning for Dynamics and Control Conference (L4DC)*, volume 168 of *Proceedings of Machine Learning Research*, pp. 492–507, 2022.

Celik, O., Li, Z., Blessing, D., Li, G., Palenicek, D., Peters, J., Chalvatzaki, G., and Neumann, G. Dime: Diffusion-based maximum entropy reinforcement learning. *arXiv preprint arXiv:2502.02316*, 2025.

Chou, P.-W., Maturana, D., and Scherer, S. Improving stochastic policy gradients in continuous control with deep reinforcement learning using the beta distribution. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 834–843. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/chou17a.html.

Ding, S., Hu, K., Zhang, Z., Ren, K., Zhang, W., Yu, J., Wang, J., and Shi, Y. Diffusion-based reinforcement learning via q-weighted variational policy optimization, 2024. URL https://arxiv.org/abs/2405.16173.

Dwivedi, R., Chen, Y., Wainwright, M. J., and Yu, B. Log-concave sampling: Metropolis-hastings algorithms are fast, 2019. URL https://arxiv.org/abs/1801.02309.

Foster, D. J. and Rakhlin, A. Foundations of reinforcement learning and interactive decision making, 2023. URL https://arxiv.org/abs/2312.16730.

Fujimoto, S., Van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In *ICLR*, 2018.

Fujimoto, S., D'Oro, P., Zhang, A., Tian, Y., and Rabbat, M. Towards general-purpose model-free reinforcement learning. In *ICLR*, 2025.

Grathwohl, W., Chen, R. T. Q., Bettencourt, J., Sutskever, I., and Duvenaud, D. Ffjord: Free-form continuous dynamics for scalable reversible generative models. In *International Conference on Learning Representations*, 2019.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *ICML*, 2018.

Hansen, N., Su, H., and Wang, X. Td-mpc2: Scalable, robust world models for continuous control. In *ICLR*, 2024.

He, Y. and Zhang, C. On the query complexity of sampling from non-log-concave distributions, 2025. URL https://arxiv.org/abs/2502.06200.

Huang, X., Zou, D., Dong, H., Ma, Y., and Zhang, T. Faster sampling without isoperimetry via diffusion-based monte carlo, 2024. URL https://arxiv.org/abs/2401.06325.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes, 2013. URL https://arxiv.org/abs/1312.6114.

Langford, J. and Zhang, T. The epoch-greedy algorithm for contextual multi-armed bandits, 2007. Technical report.

Lee, H., Lee, Y., Seno, T., Kim, D., Stone, P., and Choo, J. Hyperspherical normalization for scalable deep reinforcement learning. In *ICML*, 2025.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling, 2023. URL https://arxiv.org/abs/2210.02747.

Lv, L., Li, Y., Luo, Y., Sun, F., Kong, T., Xu, J., and Ma, X. Flow-based policy for online reinforcement learning, 2025. URL https://arxiv.org/abs/2506.12811.

Ma, H., Chen, T., Wang, K., Li, N., and Dai, B. Efficient online reinforcement learning for diffusion policy, 2025. URL https://arxiv.org/abs/2502.00361.

Palenicek, D., Vogt, F., Watson, J., Posner, I., and Peters, J. Xqc: Well-conditioned optimization accelerates deep reinforcement learning, 2025. URL https://arxiv.org/abs/2509.25174.

Park, S., Li, Q., and Levine, S. Flow q-learning, 2025. URL https://arxiv.org/abs/2502.02538.

Psenka, M., Escontrela, A., Abbeel, P., and Ma, Y. Learning a diffusion model policy from rewards via q-score matching, 2025. URL https://arxiv.org/abs/2312.11752.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Sferrazza, C., Huang, D.-M., Lin, X., Lee, Y., and Abbeel, P. Humanoidbench: Simulated humanoid benchmark for whole-body locomotion and manipulation. *arXiv Preprint arxiv:2403.10506*, 2024.

Song, Y., Sohl-Dickstein, J., Kingma, D. P., Kumar, A., Ermon, S., and Poole, B. Score-based generative modeling through stochastic differential equations, 2021. URL https://arxiv.org/abs/2011.13456.

Tassa, Y., Doron, Y., Muldal, A., Erez, T., Li, Y., Casas, D. d. L., Budden, D., Abdolmaleki, A., Merel, J., Lefrancq, A., et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Tiofack, F. N., Hellard, T. L., Schramm, F., Perrin-Gilbert, N., and Carpentier, J. Guided flow policy: Learning from high-value actions in offline reinforcement learning, 2026. URL https://arxiv.org/abs/2512.03973.

Todorov, E., Erez, T., and Tassa, Y. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

Towers, M., Kwiatkowski, A., Terry, J., Balis, J. U., De Cola, G., Deleu, T., Goulão, M., Kallinteris, A., Krimmel, M., KG, A., et al. Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*, 2024.

Wang, Y., Wang, L., Jiang, Y., Zou, W., Liu, T., Song, X., Wang, W., Xiao, L., Wu, J., Duan, J., and Li, S. E. Diffusion actor-critic with entropy regulator, 2024. URL https://arxiv.org/abs/2405.15177.

Yang, L., Huang, Z., Lei, F., Zhong, Y., Yang, Y., Fang, C., Wen, S., Zhou, B., and Lin, Z. Policy representation via diffusion probability model for reinforcement learning, 2023. URL https://arxiv.org/abs/2305.13122.

Zhang, Y., Yu, S., Zhang, T., Guang, M., Hui, H., Long, K., Wang, Y., Yu, C., and Ding, W. Sac flow: Sample-efficient reinforcement learning of flow-based policies via velocity-reparameterized sequential modeling. In *ICLR*, 2026.

Appendix Contents

A **The Use of Large Language Models** **14**

B **Proof** **14**

B.1 Reparameterized Policy 14

B.2 Resampling 14

B.3 Coupled Flow 16

C **Further Discussions** **17**

D **Strategic Expression Capability** **18**

E **Preliminaries and Baselines** **18**

E.1 Baseline Algorithms 18

E.2 Environment Details 20

F **Implementation Details** **23**

F.1 Pseudocode of the Coupled Flow Actor 23

G **Experimental Details** **24**

H **Full Results** **24**

H.1 MuJoCo 25

H.2 DeepMind Control 26

H.3 HumanoidBench 27

H.4 MyoSuite 28

A The Use of Large Language Models (LLMs)

Large Language Models (LLMs) were utilized to support the writing and refinement of this manuscript. Specifically, an LLM was employed to assist in enhancing language clarity, improving readability, and ensuring coherent expression across different sections of the paper. It aided in tasks like rephrasing sentences, checking grammar, and optimizing the overall textual flow.

It should be noted that the LLM played no role in the conception of research ideas, the formulation of research methodologies, or the design of experiments. All research concepts, ideas, and analyses were independently developed and carried out by the authors. The LLM's contributions were strictly limited to elevating the linguistic quality of the paper, without any involvement in the scientific content or data analysis.

The authors fully assume responsibility for the entire content of the manuscript, including any text generated or polished with the help of the LLM. We have verified that the text produced with the LLM complies with ethical guidelines and does not lead to plagiarism or any form of scientific misconduct.

B Proof

B.1 Proof of Proposition 1

In this section, we present the detailed proof of Proposition 1.

Proof. Consider the objective

$$J(\theta) \coloneqq \mathbb{E}_{s\sim\mathcal{D},\ \epsilon\sim p(\epsilon)}\big[Q\big(s, f_\theta(s,\epsilon)\big)\big].$$

Define the augmented state

$$\tilde{s} \coloneqq (s,\epsilon),$$

and let the induced distribution over augmented states be

$$\tilde{\mathcal{D}}(\tilde{s}) = \mathcal{D}(s)\, p(\epsilon).$$

By construction, the induced deterministic policy on the augmented state space is

$$\tilde{\pi}_\theta(\tilde{s}) \coloneqq f_\theta(s,\epsilon).$$

Therefore, the original objective can be rewritten as

$$J(\theta) = \mathbb{E}_{\tilde{s}\sim\tilde{\mathcal{D}}}\big[Q\big(s, \tilde{\pi}_\theta(\tilde{s})\big)\big].$$

Now observe that for each fixed augmented state $\tilde{s} = (s,\epsilon)$, the quantity being optimized is simply

$$Q\big(s, \tilde{\pi}_\theta(\tilde{s})\big),$$

where the action is chosen by the deterministic mapping $\tilde{\pi}_\theta$. Hence, maximizing $J(\theta)$ over θ is equivalent to searching, for each augmented state $\tilde{s}$, for an action $a \in \mathbb{A}$ that maximizes $Q(s,a)$. In other words, an optimal augmented-state policy $\tilde{\pi}^\star$ satisfies

$$\tilde{\pi}^\star(\tilde{s}) \in \arg\max_{a\in\mathbb{A}} Q(s,a).$$

Thus, after augmenting the state with the noise variable ϵ, the stochastic optimization problem induced by the reparameterized policy becomes equivalent to a deterministic action-selection problem on the augmented state space. This proves the claim. □

B.2 Proof of Theorem 1

In this section, we prove Theorem 1. We only present the proof for the ODE case, since the proof for the SDE case is almost identical.

Proof. For each fixed state s, we aim to construct an ordinary differential equation (ODE) that transports the initial distribution $a_0 \sim \pi_0(\cdot \mid s)$ to the target conditional distribution

$$\pi_1(\cdot \mid s) = \frac{w(s,\cdot)\, q(\cdot \mid s)}{z(s)}, \qquad z(s) = \int w(s,a)\, q(a \mid s)\, da.$$

Consider the ODE

$$dA_t = v_t(A_t, s)\, dt, \qquad A_0 \sim \pi_0(\cdot \mid s), \qquad t \in [0,1], \tag{7}$$

whose marginal law at time t is denoted by $\pi_t(\cdot \mid s)$.

Under this framework, the conditional flow matching objective is defined as

$$\mathcal{L}^{\text{ode}} = \mathbb{E}_{s\sim\mu,\ t\sim\text{Unif}[0,1],\ a_1\sim q(\cdot|s),\ a_t\sim p_t(\cdot|a_1,s)}\Big[w(s,a_1)\, \|f_t^\theta(s,a_t) - v_t(a_t \mid s,a_1)\|^2\Big].$$

We now show that the minimizer of this objective exactly recovers the marginal velocity field $v_t(s,a_t)$.

First, by the definition of $\pi_1(a_1 \mid s) = \frac{w(s,a_1)q(a_1|s)}{z(s)}$, we can rewrite the loss as

$$\begin{aligned}
\mathcal{L}^{\text{ode}} &= \int \mu(s) \int_0^1 \int q(a_1 \mid s) \int p_t(a_t \mid a_1, s)\, w(s,a_1)\, \|f_t^\theta(s,a_t) - v_t(a_t \mid s,a_1)\|^2\, da_t\, da_1\, dt\, ds \\
&= \int z(s)\mu(s) \int_0^1 \int \pi_1(a_1 \mid s) \int p_t(a_t \mid a_1, s)\, \|f_t^\theta(s,a_t) - v_t(a_t \mid s,a_1)\|^2\, da_t\, da_1\, dt\, ds \\
&= \int z(s)\mu(s) \int_0^1 \int\int p_t(a_t, a_1 \mid s)\, \|f_t^\theta(s,a_t) - v_t(a_t \mid s,a_1)\|^2\, da_1\, da_t\, dt\, ds \\
&= \int z(s)\mu(s) \int_0^1 \int\int p(a_1 \mid a_t, s)\, \pi_t(a_t \mid s)\, \|f_t^\theta(s,a_t) - v_t(a_t \mid s,a_1)\|^2\, da_1\, da_t\, dt\, ds.
\end{aligned}$$

When optimizing with respect to θ, the factors $z(s)\mu(s)$ and $\pi_t(a_t \mid s)$ are independent of f_t^θ. Therefore, for each fixed triplet (s, a_t, t), we only need to minimize

$$\int p(a_1 \mid a_t, s)\, \|f_t^\theta(s,a_t) - v_t(a_t \mid s,a_1)\|^2\, da_1.$$

This is a standard weighted least-squares problem, whose unique minimizer is given by the conditional expectation

$$f_t^{\theta^*}(s,a_t) = \mathbb{E}_{a_1\sim p(a_1|a_t,s)}\big[v_t(a_t \mid s,a_1)\big].$$

It remains to show that this conditional expectation is exactly equal to the marginal velocity field $v_t(s,a_t)$.

For each fixed s and a_1, let $p_t(\cdot \mid a_1, s)$ denote the conditional path density induced by

$$dA_t = v_t(A_t \mid s, a_1)\, dt.$$

Then $p_t(a_t \mid a_1, s)$ satisfies the continuity equation

$$\partial_t p_t(a_t \mid a_1, s) + \nabla_{a_t} \cdot \Big(p_t(a_t \mid a_1, s)\, v_t(a_t, \mid s, a_1)\Big) = 0.$$

Multiplying both sides by $\pi_1(a_1 \mid s)$ and integrating over a_1, we obtain

$$\partial_t \pi_t(a_t \mid s) + \nabla_{a_t} \cdot \left(\int p_t(a_t \mid a_1, s)\, \pi_1(a_1 \mid s)\, v_t(a_t, \mid s, a_1)\, da_1\right) = 0,$$

where

$$\pi_t(a_t \mid s) = \int p_t(a_t \mid a_1, s)\, \pi_1(a_1 \mid s)\, da_1.$$

By Bayes' rule,

$$p(a_1 \mid a_t, s) = \frac{p_t(a_t \mid a_1, s)\, \pi_1(a_1 \mid s)}{\pi_t(a_t \mid s)}.$$

Hence,

$$\int p_t(a_t \mid a_1, s)\, \pi_1(a_1 \mid s)\, v_t(a_t, \mid s, a_1)\, da_1 = \pi_t(a_t \mid s) \int p(a_1 \mid a_t, s)\, v_t(a_t, \mid s, a_1)\, da_1$$
$$= \pi_t(a_t \mid s)\, \mathbb{E}_{a_1 \sim p(a_1 \mid a_t, s)}\big[v_t(a_t, \mid s, a_1)\big].$$

Therefore,

$$\partial_t \pi_t(a_t \mid s) + \nabla_{a_t} \cdot \big(\pi_t(a_t \mid s)\, \mathbb{E}_{a_1 \sim p(a_1 \mid a_t, s)}\big[v_t(a_t, \mid s, a_1)\big]\big) = 0.$$

On the other hand, by definition, the marginal ODE

$$dA_t = v_t(A_t, s)\, dt, \qquad A_t \sim \pi_t(\cdot \mid s),$$

implies that $\pi_t(a_t \mid s)$ also satisfies

$$\partial_t \pi_t(a_t \mid s) + \nabla_{a_t} \cdot \big(\pi_t(a_t \mid s)\, v_t(a_t, s)\big) = 0.$$

Comparing the two continuity equations, we conclude that

$$\pi_t(a_t \mid s)\, v_t(a_t, s) = \pi_t(a_t \mid s)\, \mathbb{E}_{a_1 \sim p(a_1 \mid a_t, s)}\big[v_t(a_t, \mid s, a_1)\big].$$

Thus, for every a_t such that $\pi_t(a_t \mid s) > 0$,

$$v_t(a_t, s) = \mathbb{E}_{a_1 \sim p(a_1 \mid a_t, s)}\big[v_t(a_t, \mid s, a_1)\big].$$

Substituting this identity into the optimality condition yields

$$f_t^{\theta^*}(s, a_t) = v_t(a_t, s).$$

Therefore, minimizing the conditional flow matching objective exactly recovers the marginal velocity field v_t, which proves the theorem. □

B.3 Proof of Proposition 2

Proof. By definition, the velocity field of the coupled flow is

$$v(\boldsymbol{x}, t) = \big(v_1(\boldsymbol{x}, t), \ldots, v_n(\boldsymbol{x}, t)\big),$$

where each block is given by

$$v_i(\boldsymbol{x}, t) = \boldsymbol{\alpha}_i(\boldsymbol{x}_{-i}, t) \odot \boldsymbol{x}_i + \boldsymbol{\beta}_i(\boldsymbol{x}_{-i}, t), \qquad i = 1, \ldots, n.$$

For each coordinate $j = 1, \ldots, d_i$, we may write

$$v_{i,j}(\boldsymbol{x}, t) = \alpha_{i,j}(\boldsymbol{x}_{-i}, t)\, x_{i,j} + \beta_{i,j}(\boldsymbol{x}_{-i}, t).$$

We now compute the divergence of v. Since both $\alpha_{i,j}(\boldsymbol{x}_{-i}, t)$ and $\beta_{i,j}(\boldsymbol{x}_{-i}, t)$ depend only on $\boldsymbol{x}_{-i}$, they are independent of the coordinates in $\boldsymbol{x}_i$. Therefore,

$$\frac{\partial v_{i,j}(\boldsymbol{x}, t)}{\partial x_{i,j}} = \alpha_{i,j}(\boldsymbol{x}_{-i}, t).$$

Summing over all blocks and coordinates yields

$$\nabla \cdot v(\boldsymbol{x}, t) = \sum_{i=1}^{n} \sum_{j=1}^{d_i} \frac{\partial v_{i,j}(\boldsymbol{x}, t)}{\partial x_{i,j}} = \sum_{i=1}^{n} \sum_{j=1}^{d_i} \alpha_{i,j}(\boldsymbol{x}_{-i}, t),$$

which proves the first claim.

Next, let $\boldsymbol{x}_t$ be a trajectory satisfying

$$\frac{d\boldsymbol{x}_t}{dt} = v(\boldsymbol{x}_t, t).$$

By the instantaneous change-of-variables formula for continuous flows, the log-density along the trajectory evolves according to

$$\frac{d}{dt} \log p_t(\boldsymbol{x}_t) = -\nabla \cdot v(\boldsymbol{x}_t, t).$$

Substituting the above closed-form divergence expression gives

$$\frac{d}{dt}\log p_t(\boldsymbol{x}_t) = -\sum_{i=1}^{n}\sum_{j=1}^{d_i}\alpha_{i,j}((\boldsymbol{x}_t)_{-i}, t).$$

Integrating both sides from 0 to T, we obtain

$$\log p_T(\boldsymbol{x}_T) - \log p_0(\boldsymbol{x}_0) = -\int_0^T \nabla \cdot v(\boldsymbol{x}_t, t)\, dt.$$

Equivalently,

$$\log p_0(\boldsymbol{x}_0) = \log p_T(\boldsymbol{x}_T) + \int_0^T \nabla \cdot v(\boldsymbol{x}_t, t)\, dt.$$

Since the divergence is available in closed form, the likelihood can be computed exactly without estimating any Jacobian trace numerically. This completes the proof. □

C Further Discussions

In the main text, we primarily analyze the failure of weighted resampling 4.3 and Langevin dynamics 4.2 in off-policy settings from the perspective of diffusion sampling. We now shift to a pure RL perspective to address a core research question: even if an accurate Boltzmann distribution can be perfectly constructed, does it deliver stronger robustness and better exploration performance than deterministic policies enhanced with Gaussian noise? This question serves as a fundamental open problem in contemporary RL research.

We first formulate three representative Markov Decision Processes (MDPs) as follows:

$$\text{Original MDP}: \max_{\pi} \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty}\gamma^t r(s_t, a_t)\right]$$

$$\text{UCB MDP}: \max_{\pi} \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty}\gamma^t \left(r(s_t, a_t) + b(s_t, a_t)\right)\right]$$

$$\text{Entropy-regularized MDP}: \max_{\pi} \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty}\gamma^t \left(r(s_t, a_t) - \alpha \log \pi(a_t|s_t)\right)\right]$$

For the standard original MDP, the Bellman equation theoretically proves that its optimal policy is strictly greedy. However, directly deploying greedy policies as behavioral policies inevitably leads to severe insufficient exploration. To tackle this critical defect, the Upper Confidence Bound (UCB) MDP (Langford & Zhang, 2007) is developed, in which the term $b(s, a)$ functions as an explicit exploration bonus. Under the tabular MDP setup, this bonus term satisfies $b(s, a) \propto \frac{1}{N(s,a)}$, which is inversely proportional to the state-action visitation count $N(s, a)$. The behavioral policy of the UCB MDP still maintains a greedy mechanism, while the introduced bonus encourages the agent to visit rarely explored state-action pairs, thereby substantially boosting exploration ability.

In comparison, the entropy-regularized MDP adopts $-\log \pi(a|s)$ as an implicit pseudo-count regularization term. Its optimal policy is exactly the Boltzmann policy induced by the soft Q-function. According to classical RL theories, combining the original MDP with a naive Boltzmann exploration policy achieves exploration performance on par with the ϵ-greedy strategy. Both methods share a regret bound of $O(A^{\frac{1}{3}}T^{\frac{2}{3}})$, which is considerably worse than the $O(\sqrt{AT\log T})$ regret guarantee of UCB-based algorithms. Fundamentally, these two schemes belong to **stochastic random exploration**, where each exploratory action cannot effectively reduce environmental and decision uncertainty (Foster & Rakhlin, 2023). The remarkable sample efficiency of the Soft Actor-Critic (SAC) algorithm originates from reasonably taking $-\log \pi(a|s)$ as a key regularization component. The core of SAC lies in optimizing the soft Q-function, rather than simply applying the naive Boltzmann policy for exploration.

As illustrated above, the core essence of maximum-entropy RL does not reside in Boltzmann exploration. Instead, it hinges on the construction of valid proxies for approximate pseudo-counts, with the term $-\log \pi(a|s)$ serving as a typical representative. From this perspective, the

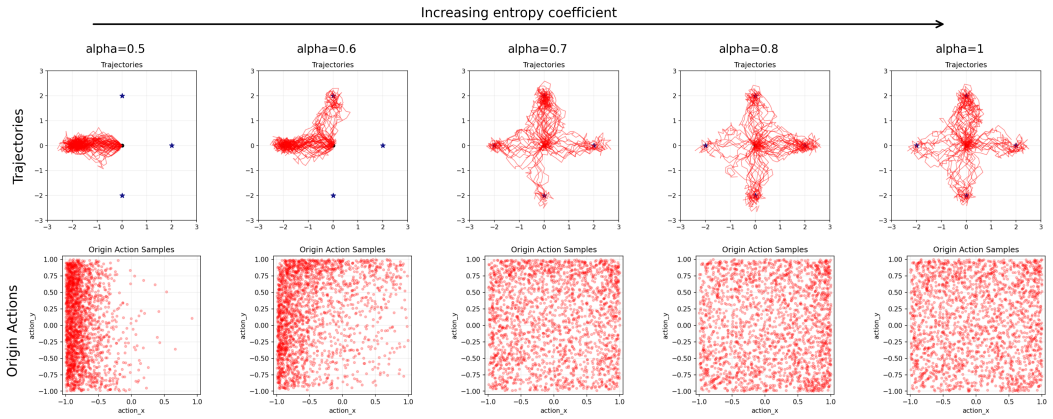


Figure 7: Trajectory Evolution with Entropy Coefficients. The first row presents trajectories starting from the origin, and the second row illustrates the actions sampled at the origin.

resampling methods 4.3 and the Langevin dynamics approach 4.2 are fundamentally equivalent to the ϵ-Greedy exploration strategy widely utilized in continuous control scenarios.

D Strategic Expression Capability

In this section, we elaborate on the capability of **Coupled Flow** to generate multimodal policies. To validate this capability, we design a two-dimensional environment with four optimization objectives and integrate **Coupled Flow** into the SAC framework for model training. We also visualize the trajectory changes under different entropy coefficients, with the detailed results presented in Fig. 7.

Evidently, the multimodal representation ability of policies is highly dependent on the entropy coefficient, and more fundamentally, determined by the critic network. Even if the policy inherently possesses multimodal potential, multimodal policy generation cannot be achieved when the Q value distribution remains unimodal. This scenario corresponds to the case where the entropy coefficient α is set to an extremely small magnitude.

E Related Preliminaries

This section records every baseline used in the experiments, together with the two methods that appear in the taxonomy but are not part of the reported runs, and then the benchmark tasks.

E.1 Algorithm

TD3 In our paper, we utilize TD3 as a representative of deterministic policies. TD3, an Actor - Critic algorithm, is widely adopted as a baseline in various decision - making scenarios and has given rise to a multitude of variants, which have established new state - of - the - art (SOTA) results on numerous occasions. Different from the traditional policy gradient method DDPG (Lillicrap et al., 2015), TD3 makes use of two heterogeneous critic networks, denoted as $Q_{\theta_{1,2}}$, to alleviate the problem of over - optimization in Q - learning. Thus, the loss function of the critics is

$$\mathcal{L}_Q(\theta_i) = \mathbb{E}_{a,s,r,s'}\left[(y - Q_{\theta_i}(s,a))^2\right] \text{ for } \forall i \in \{1,2\}.$$

Where $y = r + \gamma \min_{j=1,2} Q_{\tilde{\theta}_j}(s', \pi_\phi(s'))$, $\tilde{\theta}$ denotes the target network parameters. The actor is updated according to the Deterministic Policy Gradient:

$$\nabla_\phi J(\phi) = \mathbb{E}_s\left[\nabla_a Q_{\theta_1}(s, \pi_\phi(s)) \nabla_\phi \pi_\phi(s)\right].$$

SAC We select SAC as a representative of stochastic policies and combine it with **Coupled Flow** in the main experiment. SAC is devised to maximize expected cumulative rewards while also boosting exploration via the maximumentropy principle. The actor strives to learn a stochastic policy that

outputs a distribution over actions, where the critics estimate the value of taking a specific action in a given state. This enables a more diverse range of actions, facilitating better exploration of the action space. In traditional RL, the objective is to maximize the expected return. However, SAC introduces an additional term that maximizes the entropy of the policy, encouraging exploration. The objective function for optimizing the policy is given by:

$$J(\pi) = \mathbb{E}_{s_t,a_t}\left[r(s_t,a_t) + \alpha\mathcal{H}(\pi(\cdot|s_t))\right]$$

where $H(\pi(\cdot|s_t))$ denotes the entropy of the policy, and α is a temperature parameter that balances the trade-off between the immediate reward and the policy entropy. The training procedure of SAC involves two main updates: updating the value function and updating the policy. The value function is updated by minimizing the following loss:

$$\mathcal{L}(Q) = \mathbb{E}_{(s,a,r,s')\sim D}\left[\frac{1}{2}\left(Q(s,a) - (r + \gamma V(s'))\right)^2\right]$$

where γ is the discount factor, dictating the weight assigned to future rewards. $V(s')$ denotes the value function of the next state, which is typically approximated using a separate neural network. The policy is updated by maximizing the following objective:

$$J(\pi) = \mathbb{E}_{s_t\sim D}\left[\mathbb{E}_{a_t\sim\pi}\left[Q(s_t,a_t) - \alpha\log\pi(a_t|s_t)\right]\right]$$

Here, $-\alpha\log\pi(a_t|s_t)$ represents the entropy of the policy, which serves to promote exploration.

QSM We adopt *QSM* as the representative of our second training paradigm. By leveraging the Langevin diffusion equation, we circumvent the need to solve the likelihood function via the stochastic differential equation (SDE). For concrete implementation, we refer to the official code repository[2]. In our experiments, severe policy collapse occurred with QSM across nearly all environments, which strongly validates our conclusion.

QVPO *QVPO* is a standard weighted resampling algorithm (Section 4.3). Meanwhile, to stabilize the training process, it adopts rejection sampling to accelerate convergence. Specifically, multiple candidate actions are sampled, and only the one with the maximum Q-value is retained. We refer to its official implementation[3].

DIPO *DIPO* serves as the second representative method of the Langevin training paradigm (Section 4.2) adopted in our work. Its core feature is utilizing an additional replay buffer to store actions generated from the previous Langevin iteration. We refer to the official code repository of *DIPO*[4].

SDAC *SDAC* also falls into the category of weighted resampling algorithms (Section 4.3). Different from *QVPO*, it leverages an exponential weighting strategy for parameter updating. We follow its official open-source code[5].

The XQC-backbone comparison adds seven baselines.

XQC *XQC* (Palenicek et al., 2025) is the model-free backbone of that comparison. It keeps the SAC objective and conditions the critic with batch normalization, weight normalization, and a distributional cross-entropy Bellman loss. We integrate **Coupled Flow** with XQC, and report the original XQC agent alongside.

SimbaV2 *SimbaV2* (Lee et al., 2025) is a model-free actor-critic that normalizes network weights onto a hypersphere, which is what allows the network to be scaled without the usual loss of plasticity.

TD-MPC2 *TD-MPC2* (Hansen et al., 2024) is the model-based baseline. It learns a latent world model and selects actions by planning in that latent space, so it has no explicit stochastic actor.

[2] https://michaelpsenka.io/qsm
[3] https://github.com/wadx2019/qvpo/
[4] https://github.com/BellmanTimeHut/DIPO
[5] https://github.com/mahaitongdae/diffusion_policy_online_rl

MR.Q *MR.Q* (Fujimoto et al., 2025) is a model-free actor-critic whose representation is trained with short-horizon model-based prediction targets, so one hyperparameter set transfers across domains. Actions still come from the learned policy rather than from planning.

SAC Flow *SAC Flow* (Zhang et al., 2026) is the flow-policy baseline. It keeps the SAC objective, represents the actor by a flow with a reparameterized velocity, and trains that flow by sequential modeling.

DIME *DIME* (Celik et al., 2025) is the reparameterized diffusion baseline of Section 4.1. It backpropagates through the diffusion sampler and substitutes a surrogate entropy for the intractable action log-density.

PPO *PPO* (Schulman et al., 2017) is the on-policy baseline. It updates a Gaussian policy with a clipped likelihood-ratio objective and does not use a replay buffer.

Two further methods appear in the taxonomy but are not part of the reported runs.

DACER *DACER* (Wang et al., 2024) is the other reparameterized diffusion method in Section 4.1. It likewise backpropagates through the sampler and replaces the action log-density with an entropy regulator.

FlowRL *FlowRL* (Lv et al., 2025) is the remaining Q-weighted resampling method of Section 4.3, together with QVPO and SDAC.

E.2 Environment Details

This section details the benchmark environments used in our evaluation. We list all tasks from each benchmark along with their observation and action dimensions.

DeepMind Control Suite DeepMind Control Suite (Tassa et al., 2018) serves as a standard benchmark for continuous control, encompassing a diverse spectrum of locomotion and manipulation tasks with varying levels of complexity—ranging from simple low-dimensional tasks (where the state space $s \in \mathbb{R}^3$) to highly complex high-dimensional ones (where $s \in \mathbb{R}^{223}$). For our evaluation, we select 7 tasks from the DMC benchmark, covering all environments under the "Dog" and "Humanoid" categories.

Table 1: DMC tasks. The seven Dog and Humanoid environments used in the evaluation, with observation and action dimensions.

Task	Observation Dimension $\lvert\mathcal{O}\rvert$	Action Dimension $\lvert\mathcal{A}\rvert$
Dog Run	223	38
Dog Trot	223	38
Dog Stand	223	38
Dog Walk	223	38
Humanoid Run	67	24
Humanoid Stand	67	24
Humanoid Walk	67	24

HumanoidBench HumanoidBench (Sferrazza et al., 2024) serves as a high-dimensional simulation benchmark specifically designed to drive progress in humanoid robotics research. Centered on the Unitree H1 humanoid robot integrated with dexterous hands, this benchmark encompasses a diverse set of challenging locomotion and manipulation tasks, ranging from fundamental locomotion abilities to sophisticated operations requiring high-precision manipulation. Offering a highly realistic simulated environment, it facilitates the development and systematic validation of robotic algorithms while eliminating the reliance on costly physical hardware. The SAC comparison uses three locomotion tasks, `h1-slide`, `h1-stair`, and `h1-walk`. The XQC comparison uses all fourteen tasks in Table 2.

Table 2: HumanoidBench tasks. Observation and action dimensions of the fourteen H1 environments. The SAC comparison uses `h1-slide`, `h1-stair`, and `h1-walk`; the XQC comparison uses every row.

Task	Observation Dimension $\lvert\mathcal{O}\rvert$	Action Dimension $\lvert\mathcal{A}\rvert$
h1-balance-hard	77	19
h1-balance-simple	64	19
h1-crawl	51	19
h1-hurdle	51	19
h1-maze	51	19
h1-pole	51	19
h1-reach	57	19
h1-run	51	19
h1-sit-simple	51	19
h1-sit-hard	64	19
h1-slide	51	19
h1-stair	51	19
h1-stand	51	19
h1-walk	51	19

MuJoCo Gym (Brockman et al., 2016; Towers et al., 2024) is a comprehensive suite of benchmark environments, encompassing finite Markov Decision Processes (MDPs), multi-joint dynamics simulations, and contact-aware physical simulations. It offers a diverse range of tasks, including classic Atari games, small-scale tasks (e.g., Toy Text and classic control problems), and physics-based continuous robot control scenarios. For our experiments, we focus on 5 locomotion tasks within the MuJoCo environment subset, which simulate complex physical interactions involving multi-body dynamics and contact forces. Note that we adopt the v4 version of these environments, and a complete list of the selected tasks is provided in Table 3.

Table 3: Gym-MuJoCo Tasks (v4). Selected locomotion tasks with their respective observation and action space dimensions.

Task	Observation Dimension $\lvert\mathcal{O}\rvert$	Action Dimension $\lvert\mathcal{A}\rvert$
Ant-v4	27	8
HalfCheetah-v4	17	6
Hopper-v4	11	3
Humanoid-v4	376	17
Walker2d-v4	17	6

MyoSuite MyoSuite (Caggiano et al., 2022) is a contact-rich musculoskeletal motor control suite built on MuJoCo (Todorov et al., 2012). Unlike torque-controlled benchmarks, its environments are driven by biomechanically accurate muscle–tendon units, so the policy must coordinate highly redundant activations under intermittent contacts. We adopt the dexterous-hand tasks based on the *myoHand* model, which comprises 23 joints and 39 muscle–tendon actuators, and use the fixed and random variants of five tasks: reach, pose, object hold, key turn, and pen twirl. The random ("hard") variants randomize the target at every episode, which makes them considerably harder than the fixed variants. Since these tasks are defined by task success rather than by return, we report the success rate over five seeds after 1 million environment steps, and summarize the task dimensions in Table 4.

Score Normalization Every aggregate in the main text normalizes each task before averaging over seeds or tasks. Table 5 gives the scale, and Table 6 gives the constant for every MuJoCo and HumanoidBench task in this appendix. On that scale, 0 is a random policy and 1 matches the reference.

Table 4: MyoSuite dexterous-hand tasks. The *myoHand* model is actuated by 39 muscle–tendon units, and all tasks are evaluated for 1 million environment steps.

Task	Observation Dimension $\lvert\mathcal{O}\rvert$	Action Dimension $\lvert\mathcal{A}\rvert$
myo-reach	115	39
myo-reach-hard	115	39
myo-pose	108	39
myo-pose-hard	108	39
myo-obj-hold	91	39
myo-obj-hold-hard	91	39
myo-key-turn	93	39
myo-key-turn-hard	93	39
myo-pen-twirl	83	39
myo-pen-twirl-hard	83	39

Table 5: **Score scale.** R is the return at 1M steps. A score above 1 beats the reference. MyoSuite is a success rate and is not rescaled.

Suite	**Normalized score**	**Reference**
MuJoCo	$(R - R^{\mathrm{rand}})/(R^{\mathrm{TD3}} - R^{\mathrm{rand}})$	converged TD3
DMC	$R/1000$	episode cap of 1000
HumanoidBench	$(R - R^{\mathrm{rand}})/(R^{\mathrm{succ}} - R^{\mathrm{rand}})$	success threshold
MyoSuite	success rate (%)	not rescaled

Table 6: **Reference returns.** R^{rand} is a random policy. The reference column is R^{TD3} on MuJoCo and R^{succ} on HumanoidBench. Task names match Table 3 and Table 2.

Task	**Random**	**Reference**
MuJoCo		
Ant-v4	−70.3	3,942
HalfCheetah-v4	−289.4	10,574
Hopper-v4	18.8	3,226
Humanoid-v4	120.4	5,165
Walker2d-v4	2.8	3,946
HumanoidBench		
h1-balance-hard	9.04	800
h1-balance-simple	9.39	800
h1-crawl	272.66	700
h1-hurdle	2.21	700
h1-maze	106.44	1,200
h1-pole	20.09	700
h1-reach	260.30	12,000
h1-run	2.02	700
h1-sit-simple	9.39	750
h1-sit-hard	2.45	750
h1-slide	3.19	700
h1-stair	3.11	700
h1-stand	10.55	800
h1-walk	2.38	700

F Implementation Details

F.1 Pseudocode of the Coupled Flow Actor

This section provides the pseudocode of the Coupled Flow actor as implemented. For an observation batch $s_{1:B}$, the actor draws Gaussian noise, optionally lifts it into a higher-dimensional latent space when *num_ode* exceeds the action dimension, and applies $K =$*num_step* Euler steps of the coupled velocity field. Every step reuses one shared conditioner per block, since the field is time-homogeneous and the step size is written explicitly as $1/K$. The multiplier α is a bounded direct coefficient rather than a log-standard-deviation, and the log-density is propagated with the analytic divergence of the velocity field, which is exactly the quantity used by the policy loss.

Algorithm 1 Coupled Flow Actor Sampling and Log-Likelihood

Require: Observation batch $s_{1:B}$; action dimension d_a; $M =$ *num_ode* coupling blocks; $K =$ *num_step* Euler steps; multiplier bounds $\alpha_{\min} = -10$, $\alpha_{\max} = 2$; action bounds $[a_{\min}, a_{\max}]$
Ensure: Squashed action a and log-probability $\log \pi(a \mid s)$

$d_z \leftarrow \max(d_a, M)$
Fix a permutation P of $\{1, \dots, d_z\}$ once, and set $P^{-1} \leftarrow \mathrm{argsort}(P)$
Split $\{1, \dots, d_z\}$ into M consecutive blocks $\mathcal{I}_1, \dots, \mathcal{I}_M$ of width $\lfloor d_z/M \rfloor$, the first $d_z \bmod M$ blocks receiving one extra coordinate
for $i = 1, \dots, M$ **do**
    ODE mask $m_i \leftarrow \mathbf{1}[\,\cdot \in \mathcal{I}_i\,]$, conditioning mask $\bar{m}_i \leftarrow \mathbf{1} - m_i$
end for
Sample $u \sim \mathcal{N}(0, I_{d_a})$ and set $\ell \leftarrow \log \mathcal{N}(u; 0, I_{d_a})$
if $M > d_a$ **then**
    $z \leftarrow \big[u, \mathbf{0}_{d_z - d_a}\big][P]$ ▷ lift, then permute coordinates
else
    $z \leftarrow u$
end if
for $k = 1, \dots, K$ **do**
    for $i = 1, \dots, M$ **do**
        $h_i \leftarrow g_\theta\big(s,\, z \odot \bar{m}_i\big)$ ▷ any conditioner network g_θ shared across steps
        $g_i^\alpha \leftarrow W_\alpha h_i, \quad g_i^\beta \leftarrow W_\beta h_i$
    end for
    $\alpha \leftarrow \alpha_{\min} + \dfrac{\alpha_{\max} - \alpha_{\min}}{2}\Big(\tanh\Big(\sum_{i=1}^{M} m_i \odot g_i^\alpha\Big) + 1\Big)$ ▷ one bounded coefficient per coordinate
    $\beta \leftarrow \sum_{i=1}^{M} m_i \odot g_i^\beta$
    $z \leftarrow z + \dfrac{1}{K}\big(z \odot \alpha + \beta\big)$ ▷ Euler step of $dz/dt = \alpha \odot z + \beta$
    $\ell \leftarrow \ell - \dfrac{1}{K}\sum_{j=1}^{d_z} \alpha_j$ ▷ divergence of the velocity field
end for
if $M > d_a$ **then**
    $\tilde{a} \leftarrow \big[z[P^{-1}]\big]_{1:d_a}$ ▷ unpermute, then project
else
    $\tilde{a} \leftarrow z$
end if
$\mathrm{scale} \leftarrow (a_{\max} - a_{\min})/2, \quad \mathrm{bias} \leftarrow (a_{\max} + a_{\min})/2$
$a \leftarrow \mathrm{scale} \odot \tanh(\tilde{a}) + \mathrm{bias}$
$\log J_{\tanh} \leftarrow \sum_{j=1}^{d_a} 2\big(\log 2 - \tilde{a}_j - \mathrm{softplus}(-2\tilde{a}_j)\big)$
$\log J_{\mathrm{affine}} \leftarrow \sum_{j=1}^{d_a} \log|\mathrm{scale}_j|$
return $a, \quad \log \pi(a \mid s) \leftarrow \ell - \log J_{\tanh} - \log J_{\mathrm{affine}}$

Two implementation details are worth recording. The default partition assigns contiguous coordinates to each block, whereas the random-mask ablation draws P from a fixed random seed at initialization and thereby shuffles which coordinates belong to which block, leaving all other steps unchanged. The

deterministic action used for evaluation replaces the sampled u with $\mathbf{0}$, i.e., it decodes the mean of the base Gaussian.

G Experimental Details

Our codes are implemented with Python 3.10 and JAX. All experiments were run on NVIDIA GeForce GTX 3090 GPUs. Each single training trial ranges from 2 hours to 10 hours, depending on the algorithms and environments.

SAC Implementation We implement the Soft Actor-Critic (SAC) algorithm based on the CleanRL repository, where our implementation design is mainly inspired by the official SAC code for continuous action spaces[6].

For MuJoCo benchmark tasks, we disable both layer normalization and observation normalization, and set the hyperparameters to $(\textit{num_ode}, \textit{num_step}) = (5, 3)$. By comparison, layer normalization and observation normalization are enabled for all other environments, with the hyperparameter configuration set to $(\textit{num_ode}, \textit{num_step}) = (3, 5)$. Moreover, HumanoidBench employs an extended backbone network consisting of three hidden layers with 512 neurons each, namely $(512, 512, 512)$, and adopts the quantile loss function. In contrast, all other environments utilize a lightweight two-layer backbone network $(256, 256)$ along with the standard MSE loss function. The complete and detailed hyperparameter settings are summarized in Table 7 and Table 8.

Table 7: Hyperparameters of Our SAC Implementation

Hyperparameter	**SAC (with Coupled Flow)**
Optimizer	Adam
Policy (Actor) Learning Rate	3×10^{-4}
Q-Network (Critic) Learning Rate	1×10^{-3}
Discount Factor	0.99
Batch Size	256
Warmup Steps (for Policy Update)	5000
Target Q-Network Update Rate (τ)	0.005
Target Q-Network Update Interval	1 (step)
Policy (Actor) Update Interval	2 (steps, policy_frequency)
Entropy Target	$-\lvert A\rvert$ ($\lvert A\rvert$ = action space dimension)
Network activation function	mish

Table 8: Environment-dependent SAC hyperparameters with **Coupled Flow**. LN denotes LayerNorm, ObsNorm denotes observation normalization, and QL(K) denotes quantile loss with K quantiles.

Env. Type	$(n_{\text{ode}}, n_{\text{step}})$	**LN**	**ObsNorm**	**Loss**	**Hidden Dims.**
MuJoCo	$(5, 3)$	No	No	MSE	$(256, 256)$
DMC	$(3, 5)$	Yes	Yes	MSE	$(256, 256)$
HumanoidBench	$(3, 5)$	Yes	Yes	QL($K = 100$)	$(512, 512, 512)$

H Full Results

This appendix reports the complete results behind the main-text figures. Each benchmark is presented twice: first Coupled Flow on the SAC backbone against the baselines that share that stack (SAC, DIPO, QSM, QVPO, SDAC, TD3), and then Coupled Flow integrated with XQC against the methods named in that figure's caption.

[6] https://github.com/vwxyzjn/cleanrl/blob/master/cleanrl/sac_continuous_action.py

H.1 MuJoCo

Fig. 8 compares Coupled Flow on the SAC backbone with SAC, DIPO, QSM, QVPO, SDAC, and TD3 on the five MuJoCo locomotion tasks, and Fig. 9 repeats the comparison with Coupled Flow integrated with XQC, against XQC, SAC Flow, SimbaV2, TD-MPC2, MR.Q, DIME, and PPO. DIME is the v3 release of Ant and Humanoid.

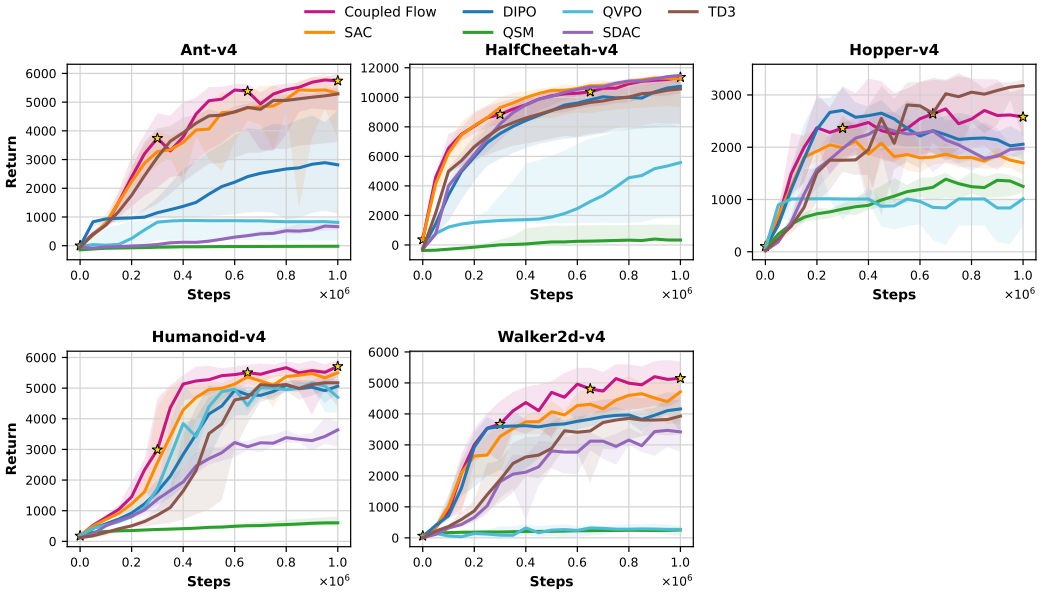


Figure 8: **MuJoCo, SAC backbone.** Learning curves comparing **Coupled Flow** with SAC, DIPO, QSM, QVPO, SDAC, and TD3 on the five MuJoCo tasks. Curves report the interquartile mean over five seeds with a 95% percentile bootstrap interval.

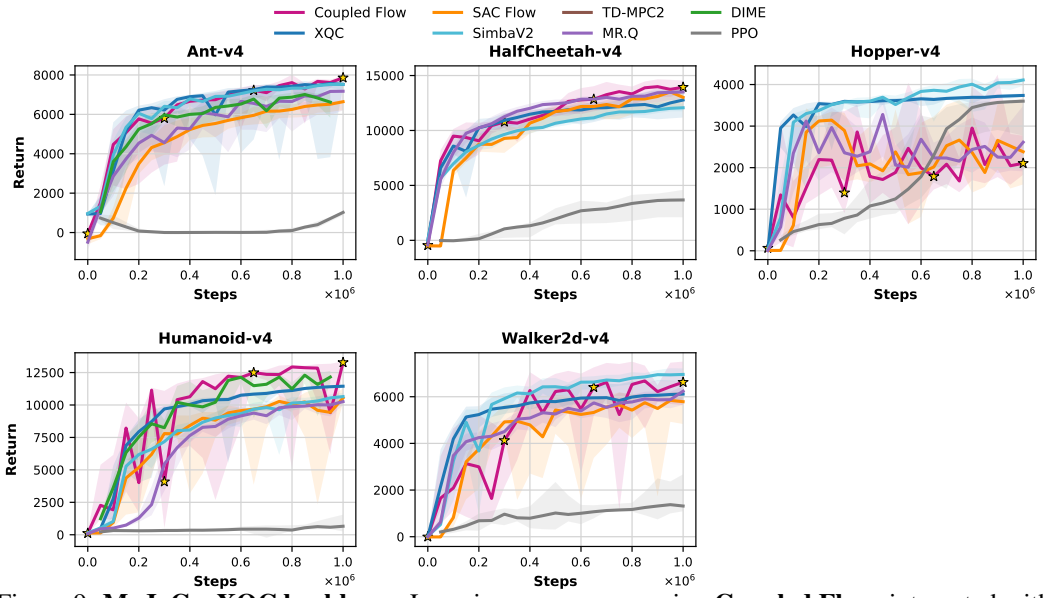


Figure 9: **MuJoCo, XQC backbone.** Learning curves comparing **Coupled Flow**, integrated with XQC, to XQC, SAC Flow, SimbaV2, TD-MPC2, MR.Q, DIME, and PPO. DIME is logged as `Ant-v3` and `Humanoid-v3`, and is not run on the other three tasks. Curves report the interquartile mean over five seeds with a 95% percentile bootstrap interval.

H.2 DeepMind Control

Fig. 10 covers the four dog and three humanoid variants against the SAC-stack baselines, and Fig. 11 covers the same seven tasks against the strong baselines.

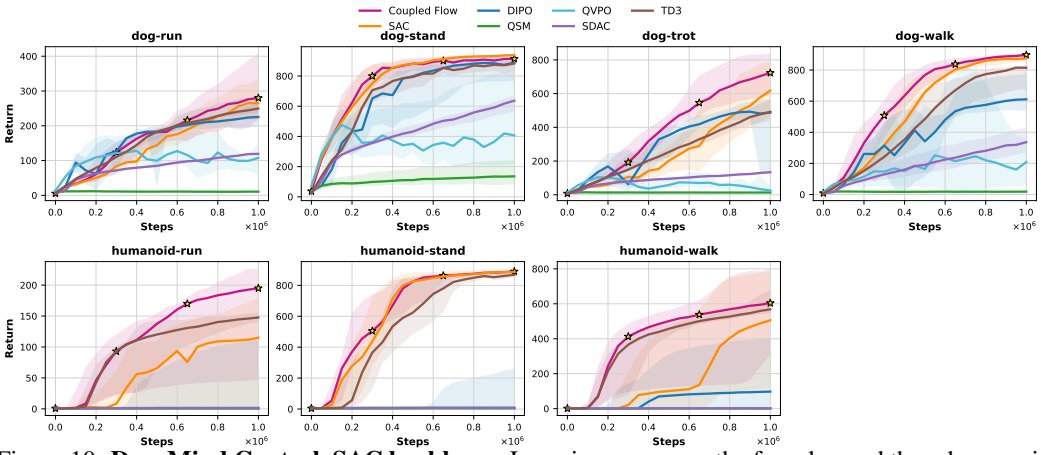


Figure 10: **DeepMind Control, SAC backbone.** Learning curves on the four dog and three humanoid tasks, comparing **Coupled Flow** with SAC, DIPO, QSM, QVPO, SDAC, and TD3. Curves report the interquartile mean over five seeds with a 95% percentile bootstrap interval.

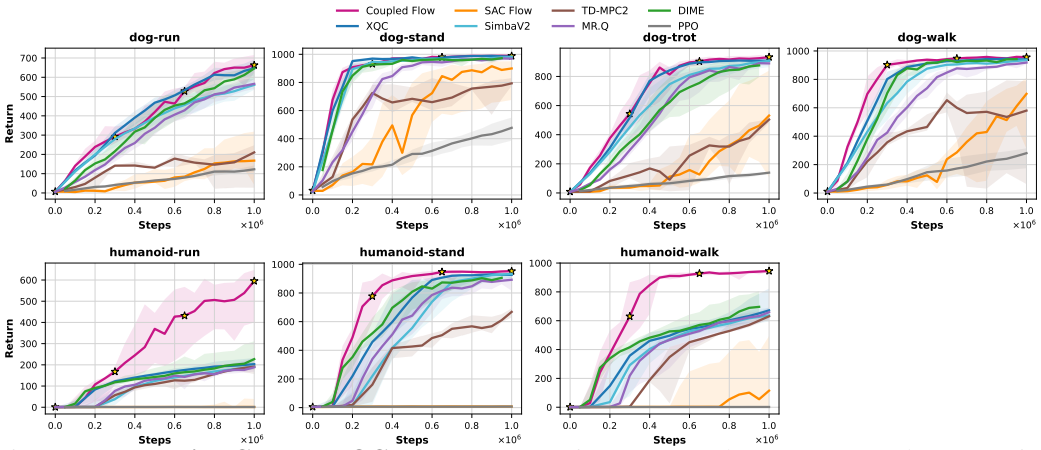


Figure 11: **DeepMind Control, XQC backbone.** Learning curves on the same seven tasks, comparing **Coupled Flow** with XQC, SAC Flow, SimbaV2, TD-MPC2, MR.Q, DIME, and PPO. Curves report the interquartile mean over five seeds with a 95% percentile bootstrap interval.

H.3 HumanoidBench

The SAC backbone was trained on the three H1 tasks used in the main text, while the XQC backbone covers all fourteen. Fig. 12 and Fig. 13 report the two comparisons respectively.

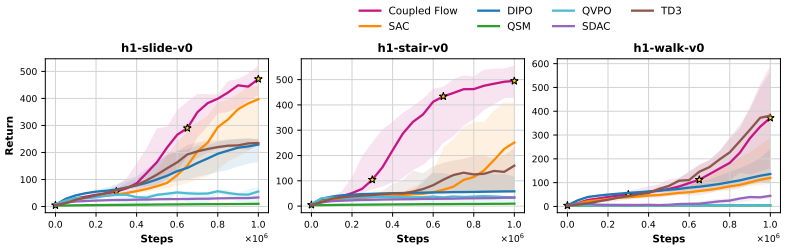


Figure 12: **HumanoidBench, SAC backbone.** Learning curves on `h1-slide-v0`, `h1-stair-v0`, and `h1-walk-v0`, comparing **Coupled Flow** with SAC, DIPO, QSM, QVPO, SDAC, and TD3. Curves report the interquartile mean over five seeds with a 95% percentile bootstrap interval.

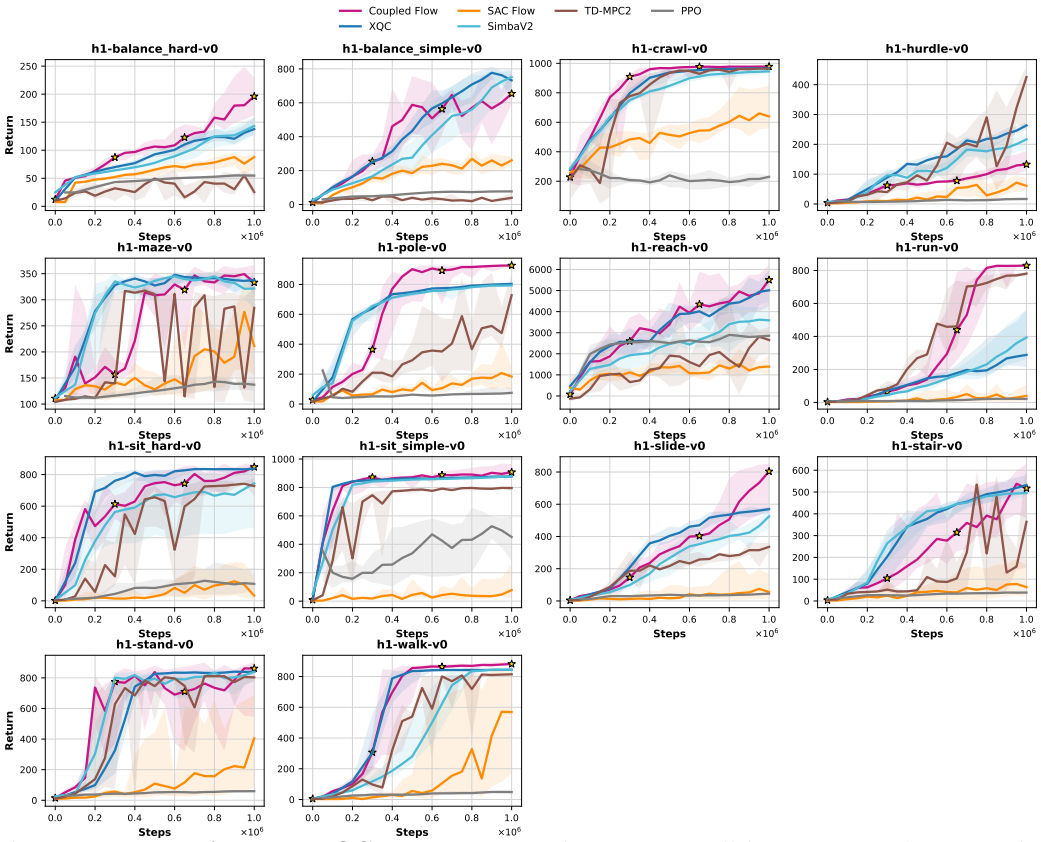


Figure 13: **HumanoidBench, XQC backbone.** Learning curves on all fourteen H1 tasks, comparing **Coupled Flow**, integrated with XQC, to XQC, SAC Flow, SimbaV2, TD-MPC2, and PPO. Curves report the interquartile mean over five seeds with a 95% percentile bootstrap interval.

H.4 MyoSuite

The ten muscle-actuated hand tasks are evaluated on the XQC backbone only, so a single comparison is reported. Because these tasks are defined by sparse task completion rather than by a dense return, Fig. 14 reports the success rate in percent.

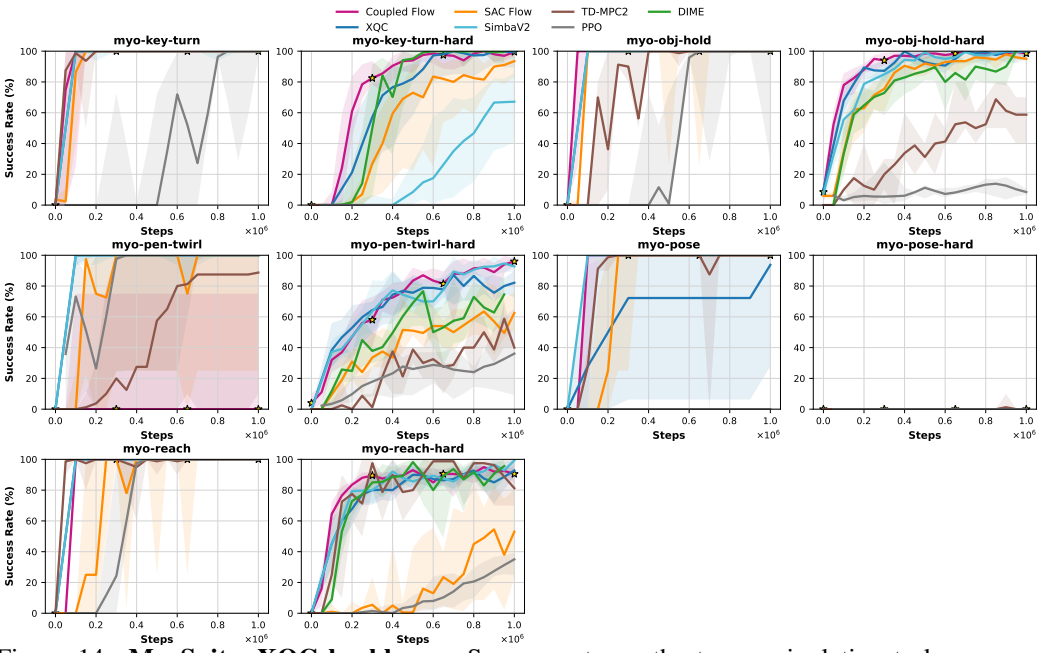


Figure 14: **MyoSuite, XQC backbone.** Success rate on the ten manipulation tasks, comparing **Coupled Flow**, integrated with XQC, to XQC, SAC Flow, SimbaV2, TD-MPC2, and PPO. DIME is shown only on the four tasks it was run on: `myo-reach-hard`, `myo-obj-hold-hard`, `myo-key-turn-hard`, and `myo-pen-twirl-hard`. Curves report the interquartile mean over five seeds with a 95% percentile bootstrap interval.

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: Please refer to abstract and introduction.

   Guidelines:

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

2. **Limitations**

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

   Answer: [Yes]

   Justification: The limitations can be seen in Conclusion of main paper.

   Guidelines:

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

3. **Theory assumptions and proofs**

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

   Answer: [Yes]

Justification: Please refer to Sec. B of the Appendix.

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 make our code available.

   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: The code is available in the supplementary materials, and the hyperparameter setting can be found in Appendix.

Guidelines:

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

6. **Experimental setting/details**

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

   Answer: [Yes]

   Justification: The experimental detail can be found in Experiment part. The hyperparameter setting can be found in Appendix G.

   Guidelines:

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

7. **Experiment statistical significance**

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

   Answer: [Yes]

   Justification: The results are accompanied by error bars, confidence intervals. We detail the calculation of error bars, running steps and the number of seeds in the text description of each figure.

   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: The Experiments Compute Resources can be found in Appendix.

   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 make sure to preserve anonymity.

   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 paper is only about promoting technological innovation and does not have social impact.

    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: Ours 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: We cite the original paper that produced the code package, and state which version of the asset is used (include a URL).

    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: This paper does not release new assets.

    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: Our paper does not involve crowdsourcing nor research with human subjects.
    Guidelines:

    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: Our paper does not involve crowdsourcing nor research with human subjects.

    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: NA.

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.