STREAMING FLOW POLICY OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Proximal policy optimization (PPO) with a diagonal Gaussian actor is the standard method for massively parallel robot learning, yet its exploration is memoryless: independent action noise is injected at every control step. Physical behaviors, by contrast, unfold as temporally coherent trajectories, so exploration should be coherent as well. We introduce Streaming Flow Policy Optimization (SFPO), which builds temporal persistence into the actor while preserving PPO's simplicity and efficiency. Flow and diffusion policies generate structured action trajectories, but their flow time indexes the noise level: policy gradients must score a K-step denoising chain or a surrogate, not the executed action. Streaming flow policies, designed for imitation learning, instead identify flow time with execution time, advancing the flow one deterministic Euler step per control step. SFPO replaces this step with an Euler–Maruyama step, so each control-step transition is an exact diagonal Gaussian over the executed action. Augmenting the state with the previous action preserves PPO's exact per-step likelihood, importance ratio, entropy, and KL divergence at one network evaluation per control step. A single retention parameter interpolates between memoryless PPO and persistent streaming integration, giving a prior for correlated exploration without restricting the learned policy. On 13 of 14 continuous-control, manipulation, and locomotion tasks, with policies trained from scratch, SFPO matches or exceeds PPO and recent diffusion- and flow-policy methods at PPO's cost, with PPO's hyperparameters and one fixed retention setting. These results position SFPO as a drop-in replacement for PPO with temporally coherent exploration.

1 INTRODUCTION

Massively parallel simulation has made on-policy reinforcement learning a standard recipe for robot control, with proximal policy optimization (PPO; Schulman et al., 2017) and a diagonal Gaussian actor as its workhorse (Rudin et al., 2022). This recipe has produced strong results in locomotion, whole-body control, and dexterous manipulation, yet its policy class remains simple: conditioned on the current state, each action is sampled independently from a Gaussian. Exploration is therefore memoryless, even though physical control is inherently temporal. Robot behaviors unfold as coherent trajectories, so exploration that persists across control steps should be better matched to physical interaction than independent per-step perturbations.

This motivates a natural question: *can we give PPO temporally structured exploration without giving up what makes PPO effective at scale?* The challenge is that PPO's simplicity is also its strength. Its clipped objective, entropy regularization, and KL-adaptive learning rate all rely on evaluating a stored action under both the behavior policy and the updated policy. A diagonal Gaussian makes these quantities tractable: its likelihood, entropy, and KL divergence are available in closed form, while producing and scoring an action requires only one network evaluation per control step. Together with generalized advantage estimation (Schulman et al., 2016), this yields a mature optimization recipe whose hyperparameters transfer across tasks and robots (Section 3.1). A more structured policy is compelling only if it preserves these advantages rather than trading them for a large increase in sampling or optimization cost.

Flow-matching and diffusion policies (Chi et al., 2023), built on diffusion and flow-matching generative models (Ho et al., 2020; Lipman et al., 2023; Liu et al., 2023), are a natural source of structured action trajectories and are widely used for imitation learning. Standard flow and diffusion policies, however, use flow time to index the *noise level*: producing an action or an action chunk requires K sequential denoising steps. This raises two obstacles for large-scale on-policy RL. First, each

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

action costs K network evaluations. Second, the executed action is the endpoint of a deterministic transformation of noise, and its density requires integrating a divergence along the flow (Chen et al., 2018). Recent policy-gradient methods therefore score the intermediate denoising transitions or a likelihood-free surrogate rather than the exact density of the action executed in the environment (Section 2).

We take a different route. Our key observation is that *flow time need not index the noise level*. Streaming Flow Policy (SFP; Jiang et al., 2025) instead identifies flow time with *execution time*: the flow trajectory is the action trajectory, and each control step advances the flow by one Euler step (Section 3.2). This gives a temporally coupled policy with one network evaluation per executed action. SFP, however, was introduced for imitation learning, and its deterministic Euler transition has no density, so PPO cannot be applied to it directly.

We show that a minimal change turns this construction into an exactly scoreable stochastic policy. Replacing the deterministic Euler update with an Euler–Maruyama step (Kloeden & Platen, 1992) makes each control-step transition an exact diagonal Gaussian while preserving the carried flow state. We call the resulting method *Streaming Flow Policy Optimization* (SFPO). Augmenting the state with the previous action and the flow phase, $\tilde{s}_t = (s_t, a_{t-1}, \tau_t)$, yields an MDP whose transitions do not depend on the policy parameters (Proposition 1). Consequently, the action actually executed in the environment has an exact per-step likelihood, importance ratio, entropy, and KL divergence, all available from the single network evaluation that produced it. SFPO therefore keeps PPO's one-evaluation-per-control-step cost and directly inherits its clipped objective, entropy regularization, KL target, and optimization machinery.

More importantly, the construction gives PPO a built-in temporal structure for exploration. The SFPO policy takes the form:

$$a_t = (1 - k\Delta t)\, a_{t-1} + \Delta t\, v_\theta(a_{t-1}, \tau_t \mid s_t) + \sigma\sqrt{\Delta t}\, \zeta_t, \qquad \zeta_t \sim \mathcal{N}(0, I), \tag{1}$$

where v_θ is a network that takes the observation s_t, the previous action a_{t-1} and the flow phase τ_t as inputs, and $\sigma \in \mathbb{R}^A_{>0}$ is a learnable diffusion coefficient. The step size Δt is the flow time assigned to one control step, not a physical time step, and the gain $k \geq 0$ is the coefficient of SFP's stabilizing term (SFP uses $k > 0$), a fixed hyperparameter (Section 4.3). The gain enters the policy only through the product $k\Delta t \in [0, 1]$, which sets how strongly stochastic perturbations persist across control steps. At $k\Delta t = 1$ the carried term vanishes, and the policy is PPO's Gaussian actor with the previous action as an extra network input. If the network is not given that input, the policy is exactly PPO's memoryless actor (Appendix A.11). At $k = 0$ it becomes a pure streaming integrator. Between these extremes the carry induces first-order autoregressive (AR(1)) correlation with an analytically controlled correlation time. This persistence is a *prior*, not a restriction on the policy class: because the previous action is an input to the network, the network can cancel, preserve, or amplify the carry as the task requires. SFPO thus biases exploration toward temporally coherent trajectories without constraining the learned mean policy. PPO is therefore the memoryless special case of SFPO. Unlike flow- and diffusion-policy methods, SFPO needs no denoising loop and no surrogate likelihood.

The streaming-flow view fixes the range of the retention $k\Delta t$, gives the carry an Ornstein–Uhlenbeck interpretation, and connects SFPO to SFP-pretrained policies (Appendices A.5, A.12 and B), a setting we leave to future work. SFPO's gains do not come from a larger policy class. On our robotics tasks, PPO's actor already observes the previous action, so PPO could represent every SFPO mean. The difference is that SFPO builds the carry $(1 - k\Delta t)\, a_{t-1}$ into the mean, so exploration noise persists across control steps from the start of training, whereas PPO would have to learn that dependence. In our experiments, v_θ is trained from scratch by the policy gradient alone, without demonstrations or flow-matching pretraining, and the flow phase τ_t is held constant.

Contributions.

- We introduce SFPO, a temporally structured stochastic policy with an exact per-step likelihood, importance ratio, entropy, and KL divergence. Augmenting the state with the carried action and flow phase gives an MDP on which PPO's gradient estimator is exact, so PPO's objective and machinery apply unchanged (Sections 4.1 and 4.4).
- We show that a single retention parameter $k\Delta t$ continuously connects memoryless PPO, temporally correlated AR(1) exploration, and streaming integration. The resulting persistence is a structured exploration prior that does not restrict the learned mean policy (Section 4.3).

Table 1: The policy classes for continuous-control RL. K is the number of denoising steps.

method	flow time indexes	evals / action	likelihood for RL	optimization unit
PPO (Gaussian)	— (memoryless)	1	exact	executed action
ARP (Gaussian)	—	1	exact	executed action
DPPO / Flow-GRPO / ReinFlow	noise level	K	exact transitions	denoising chain
FPO / FPO++	noise level	K	CFM surrogate	proxy
SFPO	execution time	1	exact	executed action

- Across 14 continuous-control, manipulation, and locomotion tasks, SFPO matches or exceeds PPO, ARPPO, DPPO and FPO++ in mean final return on 13, with gains of 46–65% over PPO on velocity tracking, humanoid skateboarding and in-hand reorientation. On eight robotics tasks, an SFPO update costs 0.90–1.11× a PPO update, against a median 5.0× for DPPO and 13× for FPO++. SFPO reuses PPO's optimization hyperparameters and adds only the retention $k\Delta t = 0.5$, fixed across tasks (Section 5).

2 Related Work

Temporally structured exploration. Several methods replace independent Gaussian action noise with perturbations that persist across control steps: Ornstein–Uhlenbeck (OU) noise (Lillicrap et al., 2016), autoregressive policies (ARP) that filter the action noise with a stationary AR process (Korenkevych et al., 2019), generalized state-dependent exploration (Raffin et al., 2022), colored noise (Eberhard et al., 2023), temporally correlated noise in the policy's latent state (Lattice; Chiappa et al., 2023), and movement-primitive trajectories with segment-wise updates (TCE; Li et al., 2024). SFPO builds this persistence into the policy dynamics: its retention coefficient interpolates between memoryless Gaussian exploration and AR(1) dynamics, with OU dynamics as the continuous-time limit (Section 4.3). ARP is the closest: it also scores an exact Gaussian on a history-augmented MDP, so PPO applies unchanged, but it replaces only the white-noise component around the Gaussian mean with a stationary AR process, whose history term is a function of past states and actions. In SFPO, the executed action is instead an input to the network, so the mean depends on it as well. Our ARPPO baseline is ARP trained with PPO (Appendix D.1).

Generative policies for robot control. Diffusion and flow-matching policies (Chi et al., 2023), built on the corresponding generative models (Ho et al., 2020; Lipman et al., 2023; Liu et al., 2023), represent multimodal action distributions and are widely used in imitation learning, but their flow time indexes the noise level, so sampling takes several sequential network evaluations per action or action chunk. Streaming Diffusion Policy (Høeg et al., 2025) staggers noise levels across an action buffer so that each executed chunk takes only a fraction of the denoising steps, and SFP (Jiang et al., 2025) identifies flow time with execution time (Section 3.2). SFPO builds on SFP, replacing its deterministic Euler update with an Euler–Maruyama transition and its imitation objective with on-policy reinforcement learning, which gives each environment step an exact density.

Policy optimization for diffusion and flow policies. Recent methods fine-tune diffusion and flow models with reinforcement learning while their flow time still indexes the noise level. They take three routes around the missing action likelihood. DPPO (Ren et al., 2025) treats the K reverse diffusion steps as an inner MDP with Gaussian transitions. Flow-GRPO (Liu et al., 2025) turns the deterministic sampler into an SDE that preserves the pretrained marginals and has Gaussian transitions. ReinFlow (Zhang et al., 2025) adds noise from a learned noise network to each Euler step, which also gives Gaussian transitions. FPO (McAllister et al., 2026) replaces the likelihood ratio with a surrogate built from the flow-matching loss; we compare against its updated version, FPO++ (Yi et al., 2026). These methods score a K-step denoising chain, or a surrogate for the action likelihood, whose latents are not part of any environment state. SFPO instead carries the previous executed action in the augmented state, so each environment step costs one network evaluation and is one Gaussian transition with an exact density (Proposition 1; Table 1). Unlike Flow-GRPO, SFPO does not need to preserve marginals: that construction keeps a pretrained model's distribution fixed, whereas SFPO's policy gradient is meant to change it, and its noise scale σ is a learned policy parameter rather than a sampler setting (Appendix A.8).

3 PRELIMINARIES

We consider a discounted Markov decision process (MDP) $(\mathcal{S}, \mathbb{R}^A, P, r, \gamma)$ with states $s_t \in \mathcal{S}$, executed actions $u_t \in \mathbb{R}^A$, transition kernel $P(s_{t+1} \mid s_t, u_t)$, reward $r_t = r(s_t, u_t)$ and discount $\gamma \in (0, 1)$. Policies are trained on-policy from rollouts of N parallel environments over H control steps per iteration. We write t for the control-step index and τ for flow time.

3.1 PROXIMAL POLICY OPTIMIZATION

PPO's actor (Schulman et al., 2017), following TRPO (Schulman et al., 2015), is a diagonal Gaussian with a state-dependent mean and a state-independent, learnable standard deviation,

$$\pi_\theta(u \mid s) = \mathcal{N}\big(u;\ \mu_\theta(s),\ \mathrm{diag}(\sigma^2)\big), \qquad \sigma \in \mathbb{R}^A_{>0}, \tag{2}$$

whose log-density, entropy $\mathcal{H}(\sigma)$ and Kullback–Leibler (KL) divergence between two parameter settings are all closed form (Appendix A.1). PPO improves the policy by re-scoring each *stored* action under the current parameters and clipping the resulting importance ratio $\rho_t(\theta) = \pi_\theta(u_t \mid s_t)/\pi_{\theta_{\mathrm{old}}}(u_t \mid s_t)$, with advantages from generalized advantage estimation (GAE; Schulman et al., 2016) and an entropy bonus on σ (Appendix A.2). A common companion (Rudin et al., 2022) is a KL-adaptive learning rate that scales the step size by a fixed factor whenever the batch-mean $D_{\mathrm{KL}}(\pi_{\theta_{\mathrm{old}}} \,\|\, \pi_\theta)$ leaves a dead band around a target $\mathrm{KL}^\star$ (Appendix A.3). The ratio, the entropy bonus and the controller all require the exact density of a stored action.

3.2 STREAMING FLOW POLICY

SFP (Jiang et al., 2025) is an imitation-learning method that identifies flow time τ with the *execution* time of an action chunk, so that the flow lives in $\mathbb{R}^A$ and the action trajectory *is* the flow trajectory. Given an observation history h and a demonstration trajectory $\xi\colon [0, 1] \to \mathbb{R}^A$, it constructs the conditional velocity field and initial distribution

$$v_\xi(a, \tau) = \dot{\xi}(\tau) - k\big(a - \xi(\tau)\big), \qquad p^0_\xi(a) = \mathcal{N}\big(a;\ \xi(0),\ \sigma_0^2 I\big), \tag{3}$$

the trajectory's own velocity plus a proportional *stabilizing* term with gain $k > 0$ that pulls the state back toward the demonstration. A network $v_\theta(a, \tau \mid h)$ is trained by conditional flow matching against this field (Appendix A.4). At inference the chunk is *streamed*: the state is initialized at the previous action, σ_0 is set to zero, and each action is one Euler step $a \leftarrow a + v_\theta(a, \tau \mid h)\,\Delta\tau$ that is sent to the controller as soon as it is computed.

We build on two features of SFP: the quantity being integrated *is* the executed action, and one control step costs one network evaluation. What SFP does not provide is a likelihood: its inference step is a deterministic Euler step and its objective is demonstration flow matching, so PPO cannot be applied to it as it stands.

4 STREAMING FLOW POLICY OPTIMIZATION

SFPO is developed in four steps. Section 4.1 adds the previous action and a flow phase to the state. On the resulting MDP, the per-step score and importance ratio that PPO uses are exact for any policy whose density can be evaluated exactly. Section 4.2 turns SFP's velocity field into an SDE on the action. Section 4.3 takes one Euler–Maruyama step of this SDE per control step; the step is a Gaussian, so its density is exact. Section 4.4 runs PPO with this Gaussian as the policy. As in PPO, each control step involves one network evaluation, one Gaussian, one exact ratio and one advantage. Appendix B gives the network and the implementation, and Algorithm 1 in Appendix A.10 gives the full iteration.

4.1 THE AUGMENTED MDP

The actor keeps, for each environment, the previous action $a_{t-1} \in \mathbb{R}^A$ and a counter n_t of control steps since the last reset. The augmented state is

$$\tilde{s}_t = (s_t,\ a_{t-1},\ \tau_t) \in \widetilde{\mathcal{S}} = \mathcal{S} \times \mathbb{R}^A \times [0, 1), \qquad \tau_t = \begin{cases} (n_t \bmod T)/T, & T \geq 1, \\ 0, & T = 0, \end{cases} \tag{4}$$

where τ_t is the flow phase and T is the chunk period. With $T \leq 1$ the phase is always zero. Our default is $T = 0$, which means no chunk structure; $T > 1$ recovers SFP's flow time within a chunk (Section 4.3). The carried action is not reset at multiples of T, and the stream runs until the episode ends. The transition of $\tilde{s}_t$ consists of the environment step with $u_t = a_t$, the carry $a_{t-1} \leftarrow a_t$ (optionally clamped), the counter increment, and a reset of (a_{t-1}, n_t) to zero when an episode ends. None of these depends on the policy parameters.

Proposition 1 (Exactness on the augmented MDP). *Let P_{aug} be the transition of $\tilde{s}_t$ described above, and let $\pi_\theta(a_t \mid \tilde{s}_t)$ be any policy on $\widetilde{\mathcal{S}}$ with a density differentiable in θ. Then $(\widetilde{\mathcal{S}}, \mathbb{R}^A, P_{\text{aug}}, r, \gamma)$ is an MDP, P_{aug} does not depend on θ, and the trajectory density factorizes as $p_\theta = p(\tilde{s}_0) \prod_t \pi_\theta(a_t \mid \tilde{s}_t)\, P_{\text{aug}}(\tilde{s}_{t+1} \mid \tilde{s}_t, a_t)$. Consequently the policy-gradient theorem holds with the per-step score $\nabla_\theta \log \pi_\theta(a_t \mid \tilde{s}_t)$, and $\pi_\theta(a_t \mid \tilde{s}_t)/\pi_{\theta_{\text{old}}}(a_t \mid \tilde{s}_t)$ is the exact per-step importance ratio.*

The proof is in Appendix A.9. PPO therefore applies on $\widetilde{\mathcal{S}}$ to any policy whose per-step density can be evaluated exactly. The next two subsections build such a policy from SFP.

4.2 From SFP's field to an SDE

SFP's conditional field (3) is affine in the action, the sum of a drive and a linear contraction:

$$v_\xi(a, \tau) = \underbrace{\dot{\xi}(\tau) + k\,\xi(\tau)}_{\text{drive}} - \underbrace{k\,a}_{\text{contraction}}. \tag{5}$$

The contraction is the same for every demonstration, flow time and observation, so it is known in closed form. SFPO keeps it and learns only the drive, which it replaces with a network output $v_\theta(a, \tau \mid s)$. The resulting drift is

$$f_\theta(a, \tau \mid s) = v_\theta(a, \tau \mid s) - k\,a, \qquad f_\theta : \mathbb{R}^A \times [0, 1) \times \mathcal{S} \to \mathbb{R}^A. \tag{6}$$

Because a is also an input to v_θ, every drift has this form, so the split is a reparameterization, not a restriction (Appendix A.5). SFP's optimal field has the same form (Lemma 2), but v_θ is trained by the policy gradient, not regressed onto demonstrations.

A deterministic Euler step of this drift has no density. Adding diffusion gives the SDE

$$da_\tau = f_\theta(a_\tau, \tau \mid s)\, d\tau + \sigma \odot dW_\tau, \tag{7}$$

where W_τ is a standard Wiener process on $\mathbb{R}^A$ in flow time and the diffusion coefficient $\sigma \in \mathbb{R}^A_{>0}$ acts elementwise. The SDE models the policy's action trajectory, not the environment.

4.3 The control-step kernel

One control step is one Euler–Maruyama step of (7) with step size Δt (Kloeden & Platen, 1992), started at the carried action:

$$a_t = a_{t-1} + f_\theta(a_{t-1}, \tau_t \mid s_t)\,\Delta t + \sigma\sqrt{\Delta t}\,\zeta_t, \qquad \zeta_t \sim \mathcal{N}(0, I). \tag{8}$$

With the drift (6), this is SFPO's policy on the augmented MDP:

$$\begin{aligned} \mu_\theta(\tilde{s}_t) &= (1 - k\Delta t)\, a_{t-1} + \Delta t\, v_\theta(a_{t-1}, \tau_t \mid s_t), \\ \pi_\theta(a_t \mid \tilde{s}_t) &= \mathcal{N}\big(a_t;\ \mu_\theta(\tilde{s}_t),\ \operatorname{diag}(\bar{\sigma}^2)\big), \end{aligned} \tag{9}$$

with per-step standard deviation $\bar{\sigma} = \sigma\sqrt{\Delta t}$. We call $\beta = 1 - k\Delta t$ the *retention coefficient* and write $\mu_t = \mu_\theta(\tilde{s}_t)$ when the dependence on θ is not at issue. The kernel is a diagonal Gaussian, so the log-density of the executed action is exact and costs one network evaluation:

$$\log \pi_\theta(a_t \mid \tilde{s}_t) = -\frac{1}{2} \sum_{i=1}^{A} \left(\frac{a_{t,i} - \mu_{\theta,i}(\tilde{s}_t)}{\bar{\sigma}_i} \right)^2 - \sum_{i=1}^{A} \log \bar{\sigma}_i - \frac{A}{2} \log 2\pi. \tag{10}$$

Its entropy and KL divergence are also closed form (Section 4.4 and Appendix A.6).

Three regimes. The product $k\Delta t$ is constrained to $[0, 1]$, which keeps the retention β in $[0, 1]$ (Appendix A.5). There are three regimes:

- $k\Delta t = 1$ ($\beta = 0$): the carried term vanishes and $\mu_\theta(\tilde{s}_t) = \Delta t\, v_\theta(a_{t-1}, \tau_t \mid s_t)$. This is PPO's memoryless Gaussian with the previous action as an extra network input. If the network is not given that input, it is exactly PPO's policy (Appendix A.11).
- $k = 0$ ($\beta = 1$): $\mu_\theta(\tilde{s}_t) = a_{t-1} + \Delta t\, v_\theta$, a pure streaming integrator. Exploration noise accumulates, so its variance grows linearly in the steps since a reset and has no stationary value.
- $0 < k\Delta t < 1$: with v_θ held fixed, the carried action is a first-order autoregressive (AR(1)) process with coefficient β. Its stationary standard deviation and lag-ℓ autocorrelation are

$$\sigma_{\text{stat}} = \frac{\bar{\sigma}}{\sqrt{1-\beta^2}}, \qquad \operatorname{corr}(a_t, a_{t+\ell}) = \beta^\ell, \tag{11}$$

  so noise persists for about $\ell_c = -1/\log\beta$ control steps.

Choosing Δt. Δt is the flow time that elapses in one control step. It is not a physical time step. The kernel depends on k, σ and v_θ only through $k\Delta t$, $\sigma\sqrt{\Delta t}$ and $\Delta t\, v_\theta$, so a change of Δt is undone by rescaling k, σ and the network output. The policy class is therefore the same for every Δt, and the hyperparameter that matters is the retention $\beta = 1 - k\Delta t$. The density (10) is exact at any Δt, so a smaller Δt gives no greater accuracy. In practice we set $\Delta t = 1$, so σ is the per-step standard deviation and PPO's initial standard deviation carries over unchanged. Appendix A.7 gives the details, including how $1/k$ can be read as a time constant when Δt is set to the control period.

The chunk period. With $T > 1$ the phase τ_t becomes an active input that cycles through $[0, 1)$ once every T control steps, as flow time does within an SFP chunk; SFP also sets $\Delta t = 1/T$. The kernel and the objective are unchanged, the entropy coefficient and KL target need no retuning, and the policy stays closed-loop (Appendix A.13). Our main results keep the phase off.

Unlike autoregressive policies (Korenkevych et al., 2019), which replace only the white-noise component around the Gaussian mean with a stationary AR process, the carried action is an input to SFPO's network, so the learned mean depends on it too. The carry adds no latency: the mean conditions on the current observation at every step, and the network can cancel the carry. The coefficient β therefore sets how sampled noise propagates by default, without constraining the mean (Appendix A.12).

4.4 PPO ON THE AUGMENTED MDP

The rollout stores, for each transition, the augmented state $\tilde{s}_t$, the action a_t, its log-density and its mean. The update re-scores each stored action at its stored augmented state. This gives an exact ratio as long as the replay sees the same kernel inputs as the rollout (Appendix B covers the observation normalizer). With the ratio

$$\rho_t(\theta) = \exp\big(\log \pi_\theta(a_t \mid \tilde{s}_t) - \log \pi_{\theta_{\text{old}}}(a_t \mid \tilde{s}_t)\big), \tag{12}$$

the objective is PPO's clipped objective (Appendix A.2) on the augmented MDP:

$$\mathcal{L}_{\text{SFPO}}(\theta) = \frac{1}{B}\sum_t \max\big(-\hat{A}_t\rho_t,\ -\hat{A}_t \operatorname{clip}(\rho_t, 1-\epsilon, 1+\epsilon)\big) + c_V\,\mathcal{L}^V - c_{\mathcal{H}}\,\mathcal{H}(\bar{\sigma}), \tag{13}$$

$$\mathcal{H}(\bar{\sigma}) = \sum_{i=1}^{A} \log\bar{\sigma}_i + \frac{A}{2}\log(2\pi e), \tag{14}$$

where B is the number of transitions in a minibatch, $\hat{A}_t$ the GAE advantage (A.4), ϵ the clip range, $\mathcal{L}^V$ the value loss with weight c_V, and $c_{\mathcal{H}}$ the entropy coefficient. Advantages are computed on environment transitions, one per control step, with no inner discount. The entropy does not depend on the state and acts on σ as PPO's bonus acts on its standard deviation.

The KL divergence between the stored and the current kernels is also closed form:

$$D_{\text{KL}}\big(\pi_{\theta_{\text{old}}}(\cdot \mid \tilde{s}_t) \,\|\, \pi_\theta(\cdot \mid \tilde{s}_t)\big) = \sum_{i=1}^{A}\Big[\log\frac{\bar{\sigma}'_i}{\bar{\sigma}_i} + \frac{\bar{\sigma}_i^2 + \big(\mu_i - \mu'_i\big)^2}{2\,\bar{\sigma}_i'^2} - \frac{1}{2}\Big], \tag{15}$$

where primes mark the current parameters, $\mu = \mu_{\theta_{\text{old}}}(\tilde{s}_t)$ and $\mu' = \mu_\theta(\tilde{s}_t)$. Its batch mean drives the KL-adaptive learning rate (A.6). It measures the change in the distribution of the executed action, as PPO's KL does, so PPO's target $\text{KL}^\star$ applies without re-measurement (Appendix A.13).

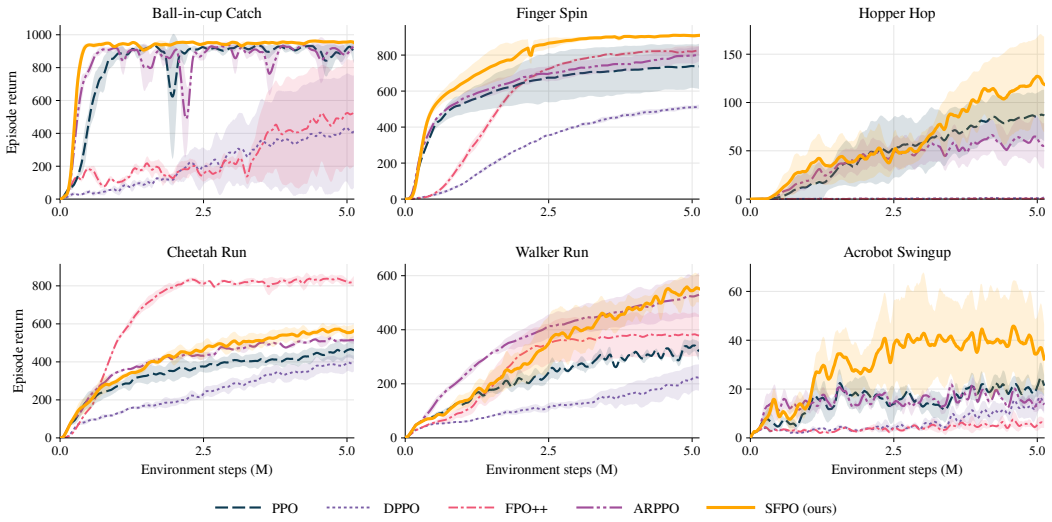

Figure 1: **DeepMind Control Suite, grouped by reward structure.** Episode return against environment steps, mean over three seeds with a ± 1 standard deviation band; SFPO runs its default kernel ($k\Delta t = 0.5$, phase off). *Top*: sparse and gated reward tasks, *Bottom*: dense reward tasks.

The critic is $V_\phi(s_t)$ on the observation alone. Any function of $\tilde{s}_t$ is a valid baseline, so the policy gradient stays unbiased. The TD residuals, however, bootstrap from a value that does not see (a_{t-1}, τ_t), so GAE with $\lambda < 1$ is biased; a critic on $\tilde{s}_t$ removes the bias.

5 Experiments

We evaluate SFPO across several domains. Our experiments cover (i) six environments (2 sparse, 1 gated, and 3 dense-reward) adapted from the DeepMind Control Suite (Tunyasuvunakool et al., 2020) (Appendix C); (ii) eight robotics control tasks (Appendix D); and (iii) deployment of two robotics tasks in the real world. Results are reported in Figure 1, Figure 2, and Figure 4.

The six DeepMind Control tasks are grouped by how often the reward is nonzero. Ball-in-cup Catch and Finger Spin give a reward of 1 only while the ball is in the cup or the spinner exceeds a speed threshold. Hopper Hop gives its speed reward only while the torso is above a standing height. Cheetah Run, Walker Run and Acrobot Swingup are shaped at every step. The eight robotics tasks run in Genesis (Genesis Authors, 2024): velocity tracking on rough terrain with a LimX Oli humanoid, tracking a motion-capture dance with a Unitree G1 (Liao et al., 2026), skateboarding with a G1 (Han et al., 2026), lifting a cube with a Franka arm and a 12-DoF hand, turning a cube to target orientations in a 20-DoF hand, playing a Chopin prelude with two hands in the style of RoboPianist (Zakka et al., 2023), picking up YCB objects (Calli et al., 2015) from a wrist RGB camera, and a Go1 quadruped handstand (Zakka et al., 2025). Appendix D gives the setup of each task. We designed our experiments to answer the following questions.

Q: How does SFPO compare with state-of-the-art methods? SFPO matches or exceeds every baseline in mean final return on 13 of the 14 tasks in Figure 1 and Figure 2. We compare against PPO, DPPO (Ren et al., 2025), FPO++ (Yi et al., 2026), and ARPPO, which is PPO with the autoregressive policy of Korenkevych et al. (2019). All methods train from scratch, although DPPO and FPO++ are designed mainly for fine-tuning pretrained policies. On the robotics tasks, PPO's observation includes the previous action, so SFPO's gains there come without an extra input; on DeepMind Control, only SFPO's actor receives it, through the carry. We report the return of the stochastic training rollouts, averaged over seeds and over the last 50 training iterations on DeepMind Control or the last 100 on the robotics tasks. Each method has three seeds per task. The one task where SFPO trails is Cheetah Run, where FPO++ ends at 822 against SFPO's 558. On Hopper Hop, neither DPPO nor FPO++ exceeds a return of 2. On the robotics tasks, SFPO ties PPO on bimanual piano (65.4 against 65.6) and has the highest mean return on the other seven, with quadruped handstand close to a tie. Its largest gains over PPO, beyond the seed spread, are on velocity tracking (44.5 against 26.9),

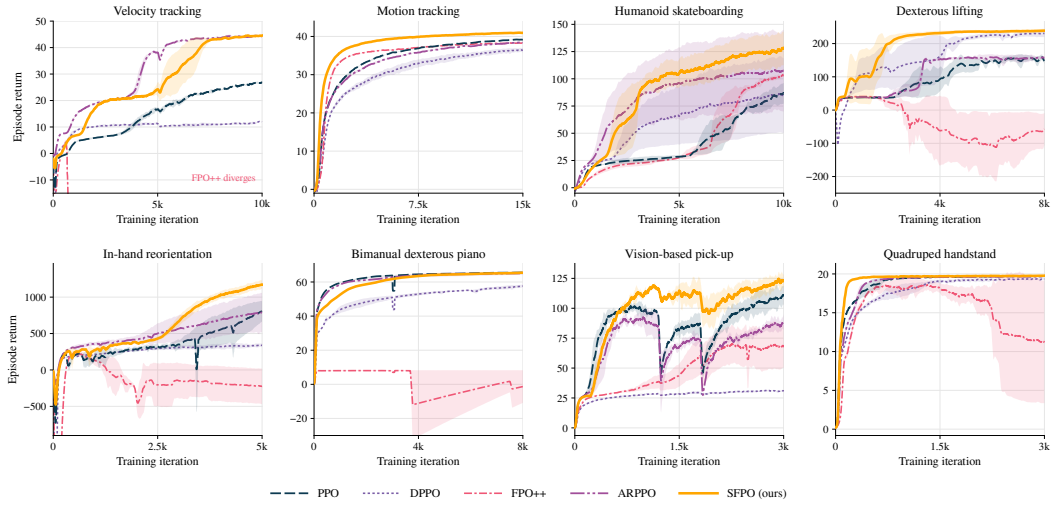

Figure 2: **Robotics tasks.** Episode return against training iteration; SFPO at $k\Delta t = 0.5$, phase off. Bands are ± 1 s.d. over three seeds. Task visuals and details are in Appendix D.

humanoid skateboarding (128 against 87) and in-hand reorientation (1170 against 800). On these seven tasks, SFPO reaches the return PPO ends training at in 20–65% of PPO's training iterations. ARPPO, the closest competitor, ends far below SFPO on in-hand reorientation (785 against 1170) and dexterous lifting (158 against 239), and level with it on velocity tracking (44.2 against 44.5), where it reaches that return sooner, and on Walker Run. DPPO ends below PPO on seven of the eight robotics tasks, and FPO++ ends below its own peak return on five.

Q: Does SFPO keep PPO's cost and PPO's hyperparameters? On every task, SFPO runs with PPO's optimizer settings unchanged: the same clip ϵ, KL target, entropy coefficient, learning rate and schedule, and number of epochs and minibatches. It adds one setting of its own, the retention $k\Delta t = 0.5$, fixed across all tasks. DPPO and FPO++ use their own settings. ARPPO keeps PPO's settings too and adds two, the order and smoothing parameter of its autoregressive noise, fixed across tasks. Table 2 gives the wall-clock time of one training iteration on the robotics tasks, relative to PPO's. Every method runs on every task on the same GPU, one run at a time, with the settings of its training runs, and we take the median over 50 iterations after 10 of warm-up. On the eight tasks, one SFPO update costs 0.90–1.11× PPO's and one rollout 0.95–1.06×. DPPO's update costs a median 5.0× PPO's, FPO++'s 13× and ARPPO's 1.52×. The baselines' ratios vary across tasks because they follow each baseline's own settings. FPO++'s update grows with its 16 epochs and 16 flow-matching samples per action, and ARPPO's update back-propagates through the mean at the current and p past states (Appendix D.1). DPPO and FPO++ also evaluate the network K times per control step, so their rollouts take a median 1.08× and 1.22× PPO's time, while SFPO evaluates it once.

Table 2: **Iteration relative to PPO**, median over the eight robotics tasks on one GPU, with the update's range; K is network evaluations per control step.

method	K	update	rollout
PPO	1	1.00	1.00
DPPO	8	5.0 (1.6–8.2)	1.08
FPO++	16–64	13 (3.4–20)	1.22
ARPPO	1	1.52 (1.08–2.53)	1.04
SFPO	1	**0.98** (0.90–1.11)	**1.00**

Q: How do the retention $k\Delta t$ and the chunk period T affect learning? Figure 3 sweeps the retention $k\Delta t$ over its full range $\{0, 0.25, 0.5, 0.75, 1\}$ and the chunk period $T \in \{1, 2, 4, 8, 16, 32\}$ on in-hand reorientation, with three seeds per cell. Each cell sets $\Delta t = 1/T$ and $k = (k\Delta t)\,T$, and initializes $\bar{\sigma}$ at the value that gives a stationary spread (11) of 0.5; at $k = 0$, which has no stationary spread, $\bar{\sigma}$ takes its $k\Delta t = 0.5$ value. Along a row, T therefore changes only the phase τ_t and the factor $1/T$ on $\Delta t\, v_\theta$. Rows with $k\Delta t > 0$ are thus matched on stationary spread, the width of the action distribution, and differ in temporal correlation. They cannot also be matched on per-step noise, which is 0.33, 0.43, 0.48 and 0.5 at $k\Delta t = 0.25$, 0.5, 0.75 and 1; the integrator and $k\Delta t = 0.5$ are matched on per-step noise instead.

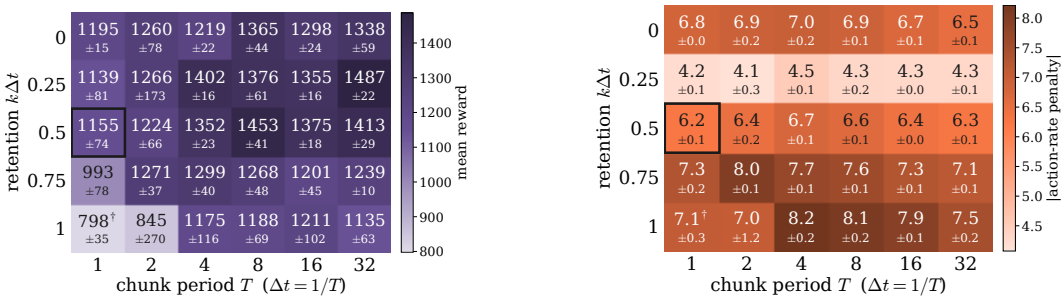

Figure 3: **Retention and chunk-period sweep on in-hand reorientation.** Each cell is the mean over three seeds of the last 100 of 5000 iterations, with the seed standard deviation below it. *Left*: mean reward. *Right*: magnitude of the action-rate penalty; lower is smoother. The outlined cell is the setting used in Figure 2. †Two seeds; the third diverged.

Intermediate retention gives the highest reward. At $T \geq 4$, the rows $k\Delta t = 0.25$ and 0.5 average 1405 and 1398, the integrator ($k = 0$) 1305, and the memoryless kernel ($k\Delta t = 1$) 1178. At matched stationary spread, $k\Delta t = 0.5$ thus leads the memoryless kernel by about 220, and at matched per-step noise it leads the integrator by 75–134 at $T \geq 4$. At every retention, $T = 1$ gives the lowest reward. At $k\Delta t = 0.5$, $T = 8$ raises it from 1155 to 1453, and the 50-iteration moving average reaches 1000 after 2290–2650 iterations instead of 3860–4270. The main results keep the phase off, so their gains do not depend on it; separating the phase input from the factor $1/T$ is left to future work. The learned σ ends at its lower bound in every run: a stationary spread of 0.2, PPO's standard-deviation floor on this task (the integrator uses the $k\Delta t = 0.5$ value). The rows thus stay matched on stationary spread throughout training, and the action-rate penalty, which barely changes with T, follows their per-step noise.

Q: Do SFPO policies transfer to hardware? We deployed SFPO policies zero-shot on two physical humanoids: velocity tracking on a LimX Oli and motion tracking on a Unitree G1 (Figure 4). We present the hardware results qualitatively. In simulation, SFPO's training rollouts, exploration noise included, violate joint limits less and change actions more smoothly than PPO's: over the final 100 iterations, its joint-limit and action-rate penalties are 45% and 12% below PPO's on velocity tracking and 19% and 14% below on motion tracking.

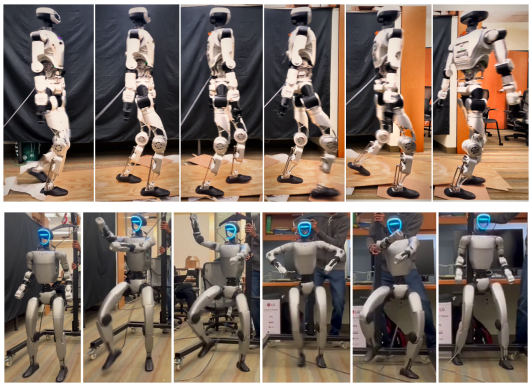

Figure 4: **Real-world deployment.** *Top*: velocity tracking on uneven terrain, LimX Oli. *Bottom*: motion tracking, Unitree G1.

6 Limitations and Conclusion

We introduced Streaming Flow Policy Optimization (SFPO), an extension of PPO that adds temporal structure while keeping PPO's exact likelihoods and optimization recipe. SFPO carries the previous action forward and updates it with an Euler–Maruyama step, so every executed action has an exact diagonal-Gaussian density and costs one network evaluation per control step. A single retention coefficient moves between memoryless PPO, AR(1) exploration, and pure integration; at $k\Delta t = 1$, when the previous action is not an input to the network, SFPO reduces exactly to PPO (Proposition 2, Appendix A.11). Across 14 tasks, SFPO has the highest return on 12 and ties PPO on a thirteenth, against PPO, ARPPO, DPPO and FPO++, at essentially PPO's computational cost.

The main limitation is that each control step is a single Gaussian, so SFPO cannot represent multimodal action distributions within a step, and because the Gaussian has full support, distinct trajectory-level behaviors are not strictly separated. Conditioning on a latent variable sampled once per trajectory would give an exact mixture over trajectory modes while keeping the exact per-step density (Appendix A.14); we leave this to future work.

REFERENCES

Adam Block, Ali Jadbabaie, Daniel Pfrommer, Max Simchowitz, and Russ Tedrake. Provable guarantees for generative behavior cloning: Bridging low-level stability and high-level behavior. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M. Dollar. The YCB object and model set: Towards common benchmarks for manipulation research. In *International Conference on Advanced Robotics (ICAR)*, pp. 510–517, 2015. doi: 10.1109/ICAR.2015.7251504.

Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems (RSS)*, 2023.

Alberto Silvio Chiappa, Alessandro Marin Vargas, Ann Zixiang Huang, and Alexander Mathis. Latent exploration for reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Carles Domingo-Enrich, Michal Drozdzal, Brian Karrer, and Ricky T. Q. Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control. In *International Conference on Learning Representations (ICLR)*, 2025.

Onno Eberhard, Jakob Hollenstein, Cristina Pinneri, and Georg Martius. Pink noise is all you need: Colored noise exploration in deep reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2023.

Genesis Authors. Genesis: A generative and universal physics engine for robotics and beyond. `https://github.com/Genesis-Embodied-AI/genesis-world`, December 2024.

Jinrui Han, Dewei Wang, Chenyun Zhang, Xinzhe Liu, Ping Luo, Chenjia Bai, and Xuelong Li. HUSKY: Humanoid skateboarding system via physics-aware whole-body control. In *Robotics: Science and Systems (RSS)*, 2026. doi: 10.15607/RSS.2026.XXII.019.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

Sigmund H. Høeg, Yilun Du, and Olav Egeland. Fast policy synthesis with variable noise diffusion models. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4821–4828, 2025. doi: 10.1109/ICRA55743.2025.11127858.

Sunshine Jiang, Xiaolin Fang, Nicholas Roy, Tomás Lozano-Pérez, Leslie Pack Kaelbling, and Siddharth Ancha. Streaming flow policy: Simplifying diffusion/flow-matching policies by treating action trajectories as flow trajectories. In *Conference on Robot Learning (CoRL)*, pp. 238–257, 2025.

Peter E. Kloeden and Eckhard Platen. *Numerical Solution of Stochastic Differential Equations*. Springer, 1992.

Dmytro Korenkevych, A. Rupam Mahmood, Gautham Vasan, and James Bergstra. Autoregressive policies for continuous control deep reinforcement learning. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 2754–2762, 2019. doi: 10.24963/ijcai.2019/382.

Ge Li, Hongyi Zhou, Dominik Roth, Serge Thilges, Fabian Otto, Rudolf Lioutikov, and Gerhard Neumann. Open the black box: Step-based policy updates for temporally-correlated episodic reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2024.

Qiayuan Liao, Takara E. Truong, Xiaoyu Huang, Yuman Gao, Guy Tevet, Koushil Sreenath, and C. Karen Liu. BeyondMimic: From motion tracking to versatile humanoid control via guided diffusion. *Science Robotics*, 11(117):eadx8924, 2026. doi: 10.1126/scirobotics.adx8924.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2016.

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

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-GRPO: Training flow matching models via online RL. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations (ICLR)*, 2023.

David McAllister, Songwei Ge, Brent Yi, Chung Min Kim, Ethan Weber, Hongsuk Choi, Haiwen Feng, and Angjoo Kanazawa. Flow matching policy gradients. In *International Conference on Learning Representations (ICLR)*, 2026.

Antonin Raffin, Jens Kober, and Freek Stulp. Smooth exploration for robotic reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2022.

Allen Z. Ren, Justin Lidard, Lars L. Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. In *International Conference on Learning Representations (ICLR)*, 2025.

Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to walk in minutes using massively parallel deep reinforcement learning. In *Conference on Robot Learning (CoRL)*, pp. 91–100, 2022.

John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization. In *International Conference on Machine Learning (ICML)*, pp. 1889–1897, 2015.

John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations (ICLR)*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Richard S. Sutton, David McAllester, Satinder Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 1999.

Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020.

George E. Uhlenbeck and Leonard S. Ornstein. On the theory of the Brownian motion. *Physical Review*, 36(5):823–841, 1930.

Brent Yi, Hongsuk Choi, Himanshu Gaurav Singh, Xiaoyu Huang, Takara E. Truong, Carmelo Sferrazza, Yi Ma, Rocky Duan, Pieter Abbeel, Guanya Shi, Karen Liu, and Angjoo Kanazawa. Flow policy gradients for robot control. *arXiv preprint arXiv:2602.02481*, 2026.

Kevin Zakka, Philipp Wu, Laura Smith, Nimrod Gileadi, Taylor Howell, Xue Bin Peng, Sumeet Singh, Yuval Tassa, Pete Florence, Andy Zeng, and Pieter Abbeel. RoboPianist: Dexterous piano playing with deep reinforcement learning. In *Conference on Robot Learning (CoRL)*, pp. 2975–2994, 2023.

Kevin Zakka, Baruch Tabanpour, Qiayuan Liao, Mustafa Haiderbhai, Samuel Holt, Jing Yuan Luo, Arthur Allshire, Erik Frey, Koushil Sreenath, Lueder A. Kahrs, Carmelo Sferrazza, Yuval Tassa, and Pieter Abbeel. Demonstrating MuJoCo Playground. In *Robotics: Science and Systems (RSS)*, 2025. doi: 10.15607/RSS.2025.XXI.020.

Tonghe Zhang, Chao Yu, Sichang Su, and Yu Wang. ReinFlow: Fine-tuning flow matching policy with online reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

A Derivations

This appendix collects the derivations behind Section 3 and Section 4. Throughout, $\bar{\sigma} = \sigma\sqrt{\Delta t}$ is the per-step standard deviation, $\beta = 1 - k\Delta t$ the retention coefficient, and for $w \in \mathbb{R}^A_{>0}$ we write $\langle x, y\rangle_{w^{-2}} = \sum_i x_i y_i / w_i^2$ and $\|x\|^2_{w^{-2}} = \langle x, x\rangle_{w^{-2}}$.

A.1 Diagonal Gaussian identities

Let $p = \mathcal{N}(\mu, \mathrm{diag}(w^2))$ and $q = \mathcal{N}(\mu', \mathrm{diag}(w'^2))$ on $\mathbb{R}^A$ with $w, w' \in \mathbb{R}^A_{>0}$ (the letter s is reserved for states throughout). Then

$$\log p(x) = -\tfrac{1}{2}\|x - \mu\|^2_{w^{-2}} - \sum_{i=1}^{A} \log w_i - \tfrac{A}{2}\log 2\pi, \tag{A.1}$$

$$\mathcal{H}(p) = -\mathbb{E}_p[\log p(x)] = \sum_{i=1}^{A} \log w_i + \tfrac{A}{2}\log(2\pi e), \tag{A.2}$$

$$D_{\mathrm{KL}}(p \,\|\, q) = \mathbb{E}_p[\log p(x) - \log q(x)] = \sum_{i=1}^{A} \Big[\log \frac{w_i'}{w_i} + \frac{w_i^2 + (\mu_i - \mu_i')^2}{2 w_i'^2} - \frac{1}{2} \Big]. \tag{A.3}$$

For (A.2), $\mathbb{E}_p\|x-\mu\|^2_{w^{-2}} = A$. For (A.3), $\mathbb{E}_p[(x_i-\mu_i')^2] = w_i^2 + (\mu_i-\mu_i')^2$ and $\mathbb{E}_p[(x_i-\mu_i)^2] = w_i^2$; substituting into $\mathbb{E}_p[\log p - \log q]$ term by term gives the sum. When $w' = w$ the KL reduces to the mean drift $\frac{1}{2}\|\mu - \mu'\|^2_{w^{-2}}$, and when $\mu' = \mu$ to $\sum_i \big[\log(w_i'/w_i) + w_i^2/(2w_i'^2) - \frac{1}{2}\big]$. Equations (10), (14) and (15) of the main text are (A.1)–(A.3) with $w = \bar{\sigma}$ and $\mu = \mu_\theta(\tilde{s}_t)$.

A.2 PPO's advantage estimator and objective

The critic $V_\phi \colon \mathcal{S} \to \mathbb{R}$ is a network with parameters ϕ that estimates the state value $V^\pi(s) = \mathbb{E}_\pi\big[\sum_{l\ge 0}\gamma^l r_{t+l} \mid s_t = s\big]$ of the current policy. The advantage $A^\pi(s,u) = Q^\pi(s,u) - V^\pi(s)$ measures how much better action u is than the policy's average action at s. Advantages are estimated by GAE (Schulman et al., 2016) on the stored rollout, by a backward pass over each environment's H steps:

$$\delta_t = r_t + \gamma(1-d_t)\,V_\phi(s_{t+1}) - V_\phi(s_t), \qquad \hat{A}_t = \delta_t + \gamma\lambda(1-d_t)\,\hat{A}_{t+1}, \qquad \hat{R}_t = \hat{A}_t + V_\phi(s_t). \tag{A.4}$$

All three quantities are scalars, one per environment and control step. The temporal-difference residual δ_t is a one-step estimate of the advantage. The GAE advantage $\hat{A}_t$ is an exponentially weighted sum of future residuals, $\hat{A}_t = \sum_{l\ge 0}(\gamma\lambda)^l\delta_{t+l}$ within an episode. The parameter $\lambda \in [0,1]$ trades bias for variance: $\lambda = 0$ gives the one-step residual, which is biased whenever V_ϕ is inaccurate, and $\lambda = 1$ gives the Monte Carlo return minus $V_\phi(s_t)$, which is unbiased but has high variance. The return target $\hat{R}_t$ is the regression target for the critic. The flag $d_t \in \{0,1\}$ marks episode termination and stops the recursion. A time-limit truncation is not a termination: at a truncated step the reward is increased by $\gamma V_\phi(s_t)$ as a stand-in for the value of the unobserved next state. The rollout is bootstrapped with $V_\phi(s_H)$ after its last step. With the importance ratio ρ_t of Section 3.1, the clipped surrogate is

$$\mathcal{L}_{\mathrm{PPO}}(\theta) = \mathbb{E}_t\Big[\max\big(-\hat{A}_t\,\rho_t,\; -\hat{A}_t\,\mathrm{clip}(\rho_t, 1-\epsilon, 1+\epsilon)\big)\Big] + c_V\,\mathcal{L}^V - c_{\mathcal{H}}\,\mathcal{H}(\sigma), \tag{A.5}$$

where $\mathbb{E}_t$ is the average over a minibatch of stored transitions and ϵ is the clip range. Clipping removes the incentive to move ρ_t outside $[1-\epsilon, 1+\epsilon]$. The value loss $\mathcal{L}^V$ is the squared error between $V_\phi(s_t)$ and $\hat{R}_t$, with the change in V_ϕ clipped to ϵ in the same way, and c_V is its weight. The entropy bonus $\mathcal{H}(\sigma)$ of (A.2), with weight $c_{\mathcal{H}}$, keeps σ from collapsing early.

A.3 THE KL-ADAPTIVE LEARNING RATE

The companion to (A.5) used throughout is the KL-adaptive learning rate of Rudin et al. (2022), as implemented in their released code.[1] After every minibatch it compares the batch mean $\overline{D_{\mathrm{KL}}}$ of $D_{\mathrm{KL}}(\pi_{\theta_{\mathrm{old}}} \,\|\, \pi_\theta)$ with a target $\mathrm{KL}^\star$ and moves the step size η by a fixed factor,

$$\eta \leftarrow \begin{cases} \max(\eta_{\min},\ \eta/1.5) & \text{if } \overline{D_{\mathrm{KL}}} > 2\,\mathrm{KL}^\star, \\ \min(\eta_{\max},\ 1.5\,\eta) & \text{if } 0 < \overline{D_{\mathrm{KL}}} < \mathrm{KL}^\star/2, \\ \eta & \text{otherwise.} \end{cases} \tag{A.6}$$

It is a bang-bang rule with a dead band $[\mathrm{KL}^\star/2,\ 2\,\mathrm{KL}^\star]$, and it reads $\overline{D_{\mathrm{KL}}}$ as a measure of how far the parameters moved.

A.4 THE STABILIZING CONDITIONAL FLOW OF SFP

SFP trains v_θ by conditional flow matching against the field (3),

$$\mathcal{L}_{\mathrm{SFP}}(\theta) = \mathbb{E}_{(h,\xi)\sim p_{\mathcal{D}}}\, \mathbb{E}_{\tau\sim\mathrm{Unif}[0,1]}\, \mathbb{E}_{a\sim p_\xi(\cdot|\tau)}\, \big\| v_\theta(a,\tau \mid h) - v_\xi(a,\tau) \big\|^2, \tag{A.7}$$

on demonstrations of horizon T_{pred}, executed receding-horizon over $T_{\mathrm{chunk}} \le T_{\mathrm{pred}}$ steps. The stabilizing term is motivated by Block et al. (2023), who show that with a low-level stabilizing controller and a sufficiently expressive generative model, behaviour cloning approximately matches the per-time-step distribution of the expert trajectories, as measured by an optimal-transport cost. Jiang et al. (2025) report that $k > 0$ outperforms $k = 0$. The two lemmas below restate SFP's Theorems 1 and 2, the second together with the form of the minimizer given in their Eq. 6: the conditional marginal is a Gaussian tube that narrows along the chunk, and the minimizer of (A.7) induces at every τ the mixture of those tubes over $p_{\mathcal{D}}(\xi \mid h)$. SFP therefore matches per-time marginals rather than the joint distribution over trajectories.

Lemma 1 (Gaussian tube; Theorem 1 of Jiang et al. (2025))**.** *Let $\xi\colon [0,1] \to \mathbb{R}^A$ be differentiable and $k \ge 0$. The flow of $\dot a = v_\xi(a,\tau) = \dot\xi(\tau) - k\big(a - \xi(\tau)\big)$ started from $a(0) \sim \mathcal{N}(\xi(0), \sigma_0^2 I)$ has marginals $a(\tau) \sim \mathcal{N}\big(\xi(\tau), \sigma_0^2 e^{-2k\tau} I\big)$.*

Proof. Let $e(\tau) = a(\tau) - \xi(\tau)$. Then $\dot e = \dot a - \dot\xi = -k\,e$, so $e(\tau) = e^{-k\tau} e(0)$ and

$$a(\tau) = \xi(\tau) + e^{-k\tau}\big(a(0) - \xi(0)\big). \tag{A.8}$$

The map $a(0) \mapsto a(\tau)$ is affine with slope $e^{-k\tau}$, so the pushforward of $\mathcal{N}(\xi(0), \sigma_0^2 I)$ is $\mathcal{N}(\xi(\tau), \sigma_0^2 e^{-2k\tau} I)$. □

The same conclusion follows from the continuity equation: for $p = \mathcal{N}(\xi(\tau), \varsigma(\tau)^2 I)$ a direct computation gives $\partial_\tau p + \nabla\cdot(p\,v_\xi) = p\,\big(\dot\varsigma/\varsigma + k\big)\big(\|a - \xi(\tau)\|^2/\varsigma^2 - A\big)$, which vanishes identically iff $\dot\varsigma = -k\varsigma$, i.e. $\varsigma(\tau) = \sigma_0 e^{-k\tau}$.

Lemma 2 (Marginal field; Theorem 2 and Eq. 6 of Jiang et al. (2025))**.** *The minimizer of (A.7) over all measurable v is the posterior mean of the conditional fields,*

$$v^*(a,\tau \mid h) = \mathbb{E}\big[v_\xi(a,\tau) \,\big|\, a,\tau,h\big] = \frac{\int v_\xi(a,\tau)\, p_\xi(a \mid \tau)\, p_{\mathcal{D}}(\xi \mid h)\, d\xi}{\int p_\xi(a \mid \tau)\, p_{\mathcal{D}}(\xi \mid h)\, d\xi}, \tag{A.9}$$

and the flow of v^* started from $p^*(\cdot \mid 0, h)$ has marginals $p^*(a \mid \tau, h) = \int p_\xi(a \mid \tau)\, p_{\mathcal{D}}(\xi \mid h)\, d\xi$ for all $\tau \in [0,1]$.

Proof. Writing the expectation over (ξ, a) as an expectation over $a \sim p^*(\cdot \mid \tau, h)$ followed by the posterior over ξ given (a,τ,h), the objective is $\mathbb{E}_h\mathbb{E}_\tau\mathbb{E}_a\mathbb{E}_{\xi|a,\tau,h}\|v(a,\tau \mid h) - v_\xi(a,\tau)\|^2$, a squared error whose pointwise minimizer in $v(a,\tau \mid h)$ is the conditional mean (A.9). For the marginals, each p_ξ satisfies $\partial_\tau p_\xi = -\nabla\cdot(p_\xi v_\xi)$ by Lemma 1; integrating against $p_{\mathcal{D}}(\xi \mid h)$ and using (A.9), $\partial_\tau p^* = -\int \nabla\cdot(p_\xi v_\xi)\, p_{\mathcal{D}}(\xi \mid h)\, d\xi = -\nabla\cdot\big(p^* v^*\big)$, so $p^*(\cdot \mid \tau, h)$ solves the continuity equation of v^* from the stated initial condition; this is the conditional-to-marginal argument of Lipman et al. (2023, Theorem 1). □

[1] `https://github.com/leggedrobotics/rsl_rl`. The rule is also stated in Appendix A.4 of the arXiv version of the paper.

Two remarks. First, because $v_\xi(a,\tau) = \big[\dot\xi(\tau) + k\xi(\tau)\big] - ka$ is affine in a with an ξ-independent slope, (A.9) gives $v^*(a,\tau \mid h) = \mathbb{E}\big[\dot\xi(\tau) + k\xi(\tau)\,\big|\,a,\tau,h\big] - ka$: the optimal field is a posterior-averaged drive minus the *same* contraction, which is the form SFPO adopts in (6). Second, the lemma matches the marginals $p^*(\cdot \mid \tau, h)$ at each τ; the joint law of a trajectory integrated under v^* need not equal the law of the demonstrations, which Jiang et al. (2025) discuss as a limitation of the method.

A.5 Affine separation and the retention coefficient

Reparameterization, not restriction. For any drift f on $\mathbb{R}^A$, define $v = f + ka$; then $f = v - ka$, so the family $\{v_\theta(a,\tau \mid s) - ka\}$ with v_θ ranging over all functions of (a,τ,s) is the family of all drifts. Likewise, for any $\beta \in [0,1]$ the family of kernel means $\{\beta a + \Delta t\, v_\theta(a,\tau \mid s)\}$ is the family of all functions of $\tilde s = (s,a,\tau)$: given a target mean $\mu^\star(\tilde s)$, take $v_\theta = (\mu^\star - \beta a)/\Delta t$. The retention coefficient therefore changes the parameterization of the mean—its value at initialization, and the way sampled noise propagates under a given v_θ—but not the class of policies. The same holds across methods: with the phase off, a PPO actor that observes a_{t-1}, as on all eight robotics tasks, can represent every SFPO mean. SFPO's gains there therefore do not come from a larger policy class. What SFPO changes is the parameterization, which sets the initial mean and how exploration noise propagates: the exploration prior of Section 4.3.

Euler versus exact discretization. Hold $v_\theta = v$ fixed over one step. The linear ODE $\dot a = v - ka$ has the exact one-step solution $a(\tau+\Delta t) = e^{-k\Delta t}a(\tau) + (1 - e^{-k\Delta t})\,v/k$, while the Euler step gives $(1 - k\Delta t)\,a(\tau) + \Delta t\, v$. Both have the fixed point v/k, and $\beta = 1 - k\Delta t$ is the first-order expansion of $e^{-k\Delta t}$. The exact factor is always positive; the Euler factor is positive iff $k\Delta t < 1$, alternates in sign for $1 < k\Delta t < 2$, and diverges for $k\Delta t > 2$. The constraint $k\Delta t \in [0,1]$ is thus the range in which the Euler retention has a continuous-time counterpart, with $k\Delta t = 1$ (no retention) as the boundary case. We keep the Euler form rather than the exponential one because it is what SFP integrates and because at $k\Delta t = 1$, for a network that does not take the carried action as input, it reduces to PPO exactly (Appendix A.11).

A.6 The Euler–Maruyama kernel

For the SDE $da_\tau = f_\theta(a_\tau, \tau \mid s)\,d\tau + \sigma \odot dW_\tau$ with $\sigma \in \mathbb{R}^A_{>0}$ acting elementwise, the Euler–Maruyama scheme (Kloeden & Platen, 1992) freezes the drift over a step of length Δt and uses the Brownian increment $W_{\tau+\Delta t} - W_\tau \sim \mathcal{N}(0, \Delta t\, I)$:

$$\begin{aligned} a_t &= a_{t-1} + f_\theta(a_{t-1}, \tau_t \mid s_t)\,\Delta t + \sigma \odot \big(W_{\tau_t+\Delta t} - W_{\tau_t}\big) \\ &\overset{d}{=} a_{t-1} + f_\theta(a_{t-1}, \tau_t \mid s_t)\,\Delta t + \sigma\sqrt{\Delta t}\,\zeta_t, \qquad \zeta_t \sim \mathcal{N}(0, I). \end{aligned} \tag{A.10}$$

Conditioned on $\tilde s_t$, a_t is Gaussian with mean $a_{t-1} + f_\theta \Delta t$ and covariance $\mathrm{diag}(\sigma^2 \Delta t)$. With the drift (6) the mean is (9), and (10), (14) and (15) are (A.1)–(A.3) with $w = \bar\sigma = \sigma\sqrt{\Delta t}$.

Units and the KL. In (15) the log-ratio and variance-ratio terms are Δt-free (the $\sqrt{\Delta t}$ cancels), while the mean-shift term is $\|\Delta\mu\|^2_{\sigma^{-2}}/(2\Delta t)$. A change of the velocity net by Δv moves the mean by $\Delta\mu = \Delta t\,\Delta v$, so at fixed σ the per-step KL is $\Delta t\,\|\Delta v\|^2_{\sigma^{-2}}/2$: in SFP's units $\Delta t = 1/T$ the same movement of v_θ registers a KL T times smaller than at $\Delta t = 1$. At $\Delta t = 1$ the per-step standard deviation is σ and the network output enters the mean unscaled, which is why SFPO's defaults use $\Delta t = 1$; what the scaling does and does not imply for the target $\mathrm{KL}^\star$ is Appendix A.13.

A.7 The step size as a choice of units

Invariance. For any $c > 0$, the substitution

$$(\Delta t,\ k,\ \sigma,\ v_\theta) \ \mapsto\ (c\,\Delta t,\ k/c,\ \sigma/\sqrt{c},\ v_\theta/c) \tag{A.11}$$

leaves $k\Delta t$, $\bar\sigma = \sigma\sqrt{\Delta t}$ and $\Delta t\, v_\theta$ unchanged, hence the mean and standard deviation of (9), the log-density (10), the ratio (12), the entropy (14) and the KL (15). Dividing v_θ by c scales the network's last layer, so the policy class is the same for every Δt. The step size therefore selects a parameterization, not a policy. What it does change is the scale at which parameters are initialized

and updated: at fixed k, σ and network initialization, a smaller Δt gives a longer memory and a smaller per-step noise (Appendix A.13). This is why the chunk-period experiments, which change Δt at fixed k and σ, change the policy: they change $k\Delta t$ and $\bar{\sigma}$.

Reading k in physical time. Nothing in SFPO carries a unit of flow time: v_θ is not regressed onto a time-parameterized target, as it is in SFP, and the control period Δ_c enters neither the kernel nor the objective. The invariance (A.11) then says that Δt cannot be identified from the policy, so it cannot be grounded to Δ_c or to the simulator's time step. It can still be set to Δ_c as a labelling convention. Let Δ_c be the control period in seconds and set $\Delta t = \Delta_c$. Then k is a rate in s^{-1}, σ a diffusion coefficient in action units per $\sqrt{\mathrm{s}}$, and the correlation time in control steps is $\ell_c = -1/\log(1 - k\Delta_c) \approx 1/(k\Delta_c)$, that is, $1/k$ seconds. A user who wants exploration correlated over t_c seconds sets $k\Delta t \approx \Delta_c/t_c$. At a higher control rate the same t_c gives a retention closer to one, which matches the observation of Korenkevych et al. (2019) that white-noise exploration becomes less effective as the action rate grows and that their smoothing parameter, which for an AR(1) process corresponds to $1 - k\Delta_c$, should be larger at higher action rates.

Refinement and convergence. With $\Delta t = \Delta_c$, refining the discretization at fixed k and σ means raising the control rate. With the network output held fixed, the carried action then converges in stationary variance and autocorrelation to the Ornstein–Uhlenbeck process $da_\tau = (v - ka_\tau)\, d\tau + \sigma \odot dW_\tau$ (Appendix A.12). For a general network the limit also depends on how the observation stream and v_θ change with the control rate, and we do not study it. A policy trained at one control rate is not a discretization of this limit, which is a second reason Δt is not a physical time step. In the other direction, $k\Delta t$ is bounded by 1: beyond it the Euler retention changes sign and has no continuous-time counterpart (Appendix A.5). The time constant $1/k$ therefore cannot be shorter than one control step, and $k\Delta t = 1$ is the memoryless case.

No discretization error at a fixed control rate. For an SDE solver, a smaller step reduces the error against the continuous-time solution. SFPO has no continuous-time solution to approximate: the policy is defined as the one-step kernel (9), and (10) is its exact density for every Δt. The only way to refine the discretization at a fixed control period is to take $m > 1$ Euler–Maruyama substeps of size $\Delta t/m$ per control step. That costs m network evaluations per action. It also removes the exact density: the executed action is then $a^{(m)}$ with $a^{(j)} = a^{(j-1)} + f_\theta(a^{(j-1)}, \tau_t \mid s_t)\, \Delta t/m + \sigma\sqrt{\Delta t/m}\, \zeta^{(j)}$, a nonlinear function of m Gaussian increments whenever v_θ is nonlinear in a. Its density is an integral over the $m-1$ intermediate states, which is again the situation of the denoising-chain methods of Section 2. One step per control step is therefore the choice that keeps both PPO's cost and PPO's exact ratio.

A.8 Comparison with marginal-preserving SDEs

A probability-flow ODE $\dot{a} = b_\tau(a)$ with marginals ρ_τ admits a one-parameter family of SDEs that share those marginals,

$$da = \big(b_\tau(a) + \epsilon_\tau \nabla \log \rho_\tau(a)\big)\, d\tau + \sqrt{2\epsilon_\tau}\, dW, \tag{A.12}$$

for any schedule $\epsilon_\tau \geq 0$: the Fokker–Planck equation of (A.12) is $\partial_\tau p = -\nabla \cdot \big((b + \epsilon \nabla \log \rho)p\big) + \epsilon \Delta p$, and at $p = \rho_\tau$ the two ϵ terms cancel, leaving the continuity equation of b_τ. What the members share is the marginals and not the transition densities—the ODE's transition density is a Dirac and no $\epsilon_\tau > 0$ reproduces it—and for the linear interpolant $I_\tau = (1 - \tau)x_0 + \tau x_1$ with Gaussian noise $x_0 \sim \mathcal{N}(0, I)$ the schedule $\epsilon_\tau = (1 - \tau)/\tau$ is the memoryless noise schedule of Domingo-Enrich et al. (2025), under which the noise x_0 and the sample x_1 are independent, as they are under the interpolant. Flow-GRPO (Liu et al., 2025) uses a member of (A.12) because it fine-tunes a pretrained generative model and the noise it adds for exploration then leaves that model's marginals unchanged.

SFPO's (7) is not a member of (A.12): its diffusion is a constant σ with no score term, and σ is learned. That is a choice in one respect and a necessity in two.

No reference distribution to preserve. The family exists to hold a reference distribution fixed while making its sampler stochastic. SFPO has no such reference: its drift is not regressed onto demonstrations but moved by the policy gradient, so the marginals a preserving schedule would fix are SFP's imitation marginals, which are not the objective. They are also degenerate at the point where the question arises. SFP streams at inference with $\sigma_0 = 0$ (Section 3.2), so by Lemma 1 the

tube width $\varsigma_\tau = \sigma_0 e^{-k\tau}$ vanishes and, conditioned on $\tilde{s}_t$, the flow state is a point mass at every τ. With ρ_τ a Dirac the only member of (A.12) that preserves it is $\epsilon_\tau \equiv 0$: preserving the marginals here means preserving determinism, which is what Section 4.2 exists to remove.

The score is unavailable when flow time is execution time. Even granting a diffuse ρ_τ, (A.12) needs $\nabla \log \rho_\tau$, while one evaluation of SFPO's network returns a velocity. When flow time indexes the noise level, the two are interchangeable. For a Gaussian path $p_\tau(x \mid x_1) = \mathcal{N}(\alpha_\tau x_1, \varsigma_\tau^2 I)$ the conditional field is $v = \dot{\alpha}_\tau x_1 - \dot{\varsigma}_\tau \varsigma_\tau \nabla \log p_\tau$, and Tweedie's identity $\mathbb{E}[x_1 \mid x] = \big(x + \varsigma_\tau^2 \nabla \log p_\tau\big)/\alpha_\tau$ turns the posterior average of that into

$$v(x,\tau) = \frac{\dot{\alpha}_\tau}{\alpha_\tau}\, x + \Big(\frac{\dot{\alpha}_\tau \varsigma_\tau^2}{\alpha_\tau} - \dot{\varsigma}_\tau \varsigma_\tau\Big) \nabla \log p_\tau(x), \tag{A.13}$$

an affine relation that can be solved for the score. That is what lets the methods whose flow time indexes the noise level convert their ODE at all. When flow time is execution time, the same substitution leaves a residue. By Lemma 2 the optimal field is $v^*(a, \tau \mid h) = \mathbb{E}\big[\dot{\xi}(\tau) + k\xi(\tau) \mid a, \tau, h\big] - ka$, and Tweedie applied to the tubes of Lemma 1 gives $\mathbb{E}[\xi(\tau) \mid a, \tau, h] = a + \varsigma_\tau^2 \nabla \log p^*(a \mid \tau, h)$, so

$$v^*(a, \tau \mid h) = \mathbb{E}\big[\dot{\xi}(\tau) \,\big|\, a, \tau, h\big] + k\,\varsigma_\tau^2\, \nabla \log p^*(a \mid \tau, h). \tag{A.14}$$

Tweedie returns the posterior mean of the demonstration's *position* at flow time τ; the drive carries its *velocity*, a different functional of ξ, and nothing recovers $\mathbb{E}[\dot{\xi}(\tau) \mid a, \tau, h]$ from v^* alone. The conversion is unavailable here rather than declined. The residue vanishes only in the degenerate cases $k = 0$, where there is no contraction to invert, and $\sigma_0 = 0$, where $\varsigma_\tau = 0$ and the score does not exist.

σ is a policy parameter, not a sampler setting. The schedule ϵ_τ is free, so the diffusion $\sqrt{2\epsilon_\tau}$ in (A.12) can be set to any size; what cannot be done is moving it *independently of the drift*, since the same ϵ_τ multiplies the score correction. That coupling is the defining feature of the family and it is the part SFPO cannot use. In (7) the diffusion is the exploration scale and nothing else: σ carries PPO's units at $\Delta t = 1$ (Appendix A.6), and the entropy bonus of (13) moves $\log \sigma$ while leaving the mean where it is, exactly as PPO's bonus does. Under (A.12) the same gradient would drag the drift along with it, and $c_{\mathcal{H}}$ would stop being an exploration coefficient. The reduction to PPO needs that separation twice over: the memoryless kernel of Appendix A.11 has a standard deviation free of the mean, which the coupling forbids, and a mean free of the carried action, which a drift carrying $\nabla \log \rho_\tau(a)$ does not generically give.

A.9 The augmented MDP: proof of Proposition 1

Let the environment be an episodic MDP $(\mathcal{S}, \mathbb{R}^A, P, r, \gamma)$ with termination indicator $d_t \in \{0, 1\}$ and initial distribution p_0, run as a continuing process in which a terminated environment is reset to $s \sim p_0$. Recall $\widetilde{\mathcal{S}} = \mathcal{S} \times \mathbb{R}^A \times [0, 1)$ and, given $\tilde{s}_t = (s_t, a_{t-1}, \tau_t)$ and the policy output a_t,

$$s_{t+1} \sim P(\cdot \mid s_t, u_t), \qquad u_t = a_t, \qquad d_t = d(s_t, u_t, s_{t+1}) \quad \text{(reset to } p_0 \text{ if } d_t = 1), \tag{A.15}$$

$$a'_t = (1 - d_t)\, c(a_t), \qquad c(a) = \begin{cases} \mathrm{clip}(a, -\bar{a}, \bar{a}) & \bar{a} > 0, \\ a & \text{otherwise}, \end{cases} \tag{A.16}$$

$$\tau_{t+1} = (1 - d_t)\, \phi(\tau_t), \qquad \phi(\tau) = \big((T\tau + 1) \bmod T\big)/T \quad (\phi \equiv 0 \text{ if } T = 0), \tag{A.17}$$

and $\tilde{s}_{t+1} = (s_{t+1}, a'_t, \tau_{t+1})$. The phase update is well defined on τ_t alone because $(n_t + 1) \bmod T$ is a function of $n_t \bmod T$. Every factor of the induced kernel $P_{\mathrm{aug}}(\tilde{s}_{t+1} \mid \tilde{s}_t, a_t)$—the environment step, the clamp, the counter and the reset—is either the environment's own transition or a deterministic map of $(\tilde{s}_t, a_t)$, and none of them involves θ. The reward $r(s_t, u_t)$ is a function of $(\tilde{s}_t, a_t)$. Hence $(\widetilde{\mathcal{S}}, \mathbb{R}^A, P_{\mathrm{aug}}, r, \gamma)$ is an MDP, $\pi_\theta(\cdot \mid \tilde{s}_t)$ is a Markov policy on it, and for a rollout of length H

$$p_\theta(\tilde{s}_{0:H}, a_{0:H-1}) = p(\tilde{s}_0) \prod_{t=0}^{H-1} \pi_\theta(a_t \mid \tilde{s}_t)\, P_{\mathrm{aug}}(\tilde{s}_{t+1} \mid \tilde{s}_t, a_t), \tag{A.18}$$

$$\nabla_\theta \log p_\theta(\tilde{s}_{0:H}, a_{0:H-1}) = \sum_{t=0}^{H-1} \nabla_\theta \log \pi_\theta(a_t \mid \tilde{s}_t). \tag{A.19}$$

Algorithm 1 Streaming Flow Policy Optimization (one iteration)

Require: actor $(\theta, \log\sigma)$, critic ϕ, frozen actor normalizer ψ, retention $\beta = 1 - k\Delta t$, chunk period T

1: **for** $t = 1, \dots, H$, in each of N environments **do** ▷ rollout, no gradients
2: $\tau_t \leftarrow 0$ if $T = 0$ else $(n_t \bmod T)/T$
3: $\mu_t \leftarrow \beta\, a_{t-1} + \Delta t\, v_\theta\big(a_{t-1}, \tau_t \mid \nu_\psi(s_t)\big)$ ▷ one network evaluation
4: $a_t \leftarrow \mu_t + \bar\sigma \odot \zeta_t$ with $\zeta_t \sim \mathcal{N}(0, I)$; $\ell_t \leftarrow \log \pi_\theta(a_t \mid \tilde s_t)$ by (10)
5: store $(s_t, a_{t-1}, \tau_t, a_t, \ell_t, \mu_t, V_\phi(s_t))$; execute $u_t = a_t$; observe r_t, d_t, truncation
6: carry a_t (clamped to $\pm\bar a$ if enabled), $n_t \leftarrow n_t + 1$; zero (a_t, n_t) where $d_t = 1$
7: update the critic's normalizer only; bootstrap truncated rewards with $\gamma V_\phi(s_t)$
8: **end for**
9: compute $\hat A_t, \hat R_t$ by GAE (A.4) on environment steps; standardize $\hat A$
10: **for** epoch $= 1, \dots, E$; minibatch $= 1, \dots, M$ **do**
11: re-score the stored $(s, a_{\text{prev}}, \tau, a)$ under θ: $\log \pi_\theta$ and μ' by (9)–(10)
12: $\overline{D_{\text{KL}}} \leftarrow$ batch mean of (15); if adaptive, update η by (A.6)
13: $\rho \leftarrow \exp(\log \pi_\theta - \ell)$; gradient step on $\mathcal{L}_{\text{SFPO}}$ (13)
14: **end for**
15: fold the stored observations into the actor's normalizer ψ ▷ once per iteration

The policy-gradient theorem (Sutton et al., 1999) and the surrogate objective of Schulman et al. (2015) therefore apply on $\widetilde{\mathcal{S}}$ with the per-step score $\nabla_\theta \log \pi_\theta(a_t \mid \tilde s_t)$. At rollout the behaviour policy is $\pi_{\theta_{\text{old}}}(\cdot \mid \tilde s_t)$ and the tuple $(\tilde s_t, a_t)$ is stored; at update the same a_t is scored at the same $\tilde s_t$ under θ, so $\pi_\theta(a_t \mid \tilde s_t)/\pi_{\theta_{\text{old}}}(a_t \mid \tilde s_t)$ is the exact per-step importance weight. □

Three consequences are used in the main text. (i) *Resets and clamps are free.* Because c and the reset act after a_t has been sampled and scored, they are part of P_{aug}, not of π_θ; they neither change the density of a_t nor contribute gradient terms. A runner-level clip on the executed action, if any, is likewise outside the density, exactly as in PPO. (ii) *GAE on environment steps.* There is one policy action and one reward per environment transition, so the augmented time index is the environment's and (A.4) applies with $V(\tilde s_t)$ in place of $V(s_t)$ and no inner discount. This is the structural difference from DPPO, whose K denoising steps per environment step share one reward and need a second discount. (iii) *Observation-only critic.* For any $b\colon \widetilde{\mathcal{S}} \to \mathbb{R}$, $\mathbb{E}_{a\sim\pi_\theta(\cdot|\tilde s)}\big[\nabla_\theta \log \pi_\theta(a \mid \tilde s)\, b(\tilde s)\big] = 0$, so $b(\tilde s_t) = V_\phi(s_t)$ is a valid baseline and the gradient stays unbiased. The TD residuals in (A.4), however, bootstrap from $V_\phi(s_{t+1})$ rather than from the augmented value $V^\pi(\tilde s_{t+1})$, so for $\lambda < 1$ the estimator inherits the error $V^\pi(\tilde s) - \mathbb{E}[V^\pi(\tilde s) \mid s]$; $\lambda = 1$ removes it, and so does a critic on $\tilde s_t$.

A.10 Cost and the full iteration

One iteration is PPO's, with the kernel (9) in place of (2): roll out, score by (10), carry a_t forward and zero it on reset, compute GAE (A.4) on environment steps, and re-score the stored $(\tilde s_t, a_t)$ under θ. The cost is one network evaluation per control step at rollout and one per replayed transition at update, i.e. PPO's; storage adds the carried action, the phase, the sample and the mean to PPO's buffers. There is no denoising chain, no inner MDP, no per-level noise schedule, and no surrogate ratio.

Algorithm 1 writes the loop out in full. Every line is PPO's except the kernel on line 3, the carry on line 6 and the normalizer fold on the last line.

Both special cases are recovered exactly. At $k\Delta t = 1$ with a network that does not take a_{t-1} as input, the kernel is PPO's actor (2), and every hyperparameter of (13) and (A.6) is used at PPO's value (Appendix A.11). With the diffusion term switched off, the stream is SFP's deterministic inference (Appendix A.15). Relative to SFP, SFPO changes three things: the Euler step becomes an Euler–Maruyama step and so carries a density, the objective is PPO's rather than demonstration flow matching, and σ is learnable so that the entropy bonus has a target. SFPO trades multimodality within a control step, and exact separation of trajectory branches, for the exact density; a latent conditioning variable, part of $\tilde s_t$ and drawn once per episode, would add both at no cost to that density (Appendix A.14).

A.11 PPO AS THE MEMORYLESS SPECIAL CASE

Proposition 2 (PPO is SFPO at $k\Delta t = 1$)**.** *Let $k\Delta t = 1$ and suppose $v_\theta(a, \tau \mid s) = w_\theta(s)$ does not depend on (a, τ). Then $\pi_\theta(a_t \mid \tilde{s}_t) = \mathcal{N}\big(a_t; \Delta t\, w_\theta(s_t), \mathrm{diag}(\bar{\sigma}^2)\big)$ depends on $\tilde{s}_t$ only through s_t, the process (s_t) is Markov under it, and the log-density (10), ratio (12), surrogate (13), entropy and KL (15) coincide term by term with PPO's (A.5)–(A.6) for the Gaussian actor (2) with mean $\mu_\theta(s) = \Delta t\, w_\theta(s)$ and standard deviation $\bar{\sigma}$. At $\Delta t = 1$ the two parameterizations are identical.*

Proof. With $\beta = 1 - k\Delta t = 0$ the carried term in (9) vanishes, so $\mu_\theta(\tilde{s}_t) = \Delta t\, v_\theta(a_{t-1}, \tau_t \mid s_t) = \Delta t\, w_\theta(s_t)$. The kernel is then a function of s_t alone, the marginal process on $\mathcal{S}$ is Markov with the environment's transition, and every quantity in Sections 4.3 and 4.4 is the diagonal-Gaussian identity of Appendix A.1 evaluated at $\mu = \Delta t\, w_\theta(s_t)$ and $w = \bar{\sigma}$, which is what PPO evaluates. □

Without the assumption on v_θ, $k\Delta t = 1$ gives PPO with the previous action appended to the network input: there is no additive carry, but the replay must still be evaluated at the stored a_{t-1}, which is why the augmented state is kept in all regimes.

A.12 TEMPORAL STRUCTURE OF EXPLORATION

Hold the network output fixed at $v_\theta \equiv v$ and let $0 < k\Delta t < 1$, i.e. $\beta \in (0, 1)$. The carried action follows $a_t = \beta a_{t-1} + \Delta t\, v + \bar{\sigma}\, \zeta_t$ with fixed point $a^* = \Delta t\, v/(1-\beta) = v/k$; the deviation $e_t = a_t - a^*$ is the AR(1) recursion $e_t = \beta e_{t-1} + \bar{\sigma}\, \zeta_t$.

Variance and autocorrelation. From a zero-seeded stream ($a_0 = 0$, so $e_0 = -a^*$ is deterministic), $\mathrm{Var}(e_n) = \beta^2 \mathrm{Var}(e_{n-1}) + \bar{\sigma}^2$ gives

$$\mathrm{Var}(e_n) = \bar{\sigma}^2\, \frac{1-\beta^{2n}}{1-\beta^2} \xrightarrow[n\to\infty]{} \sigma^2_{\mathrm{stat}} = \frac{\bar{\sigma}^2}{1-\beta^2}, \qquad \mathrm{Cov}(e_t, e_{t+\ell}) = \beta^\ell\, \mathrm{Var}(e_t), \tag{A.20}$$

the second because $e_{t+\ell} = \beta^\ell e_t + \sum_{j=1}^{\ell} \beta^{\ell-j} \bar{\sigma}\, \zeta_{t+j}$ with the noise terms independent of e_t. Hence $\mathrm{corr}(e_t, e_{t+\ell}) = \beta^\ell$ at stationarity, which is (11). At $k\Delta t = \frac{1}{2}$ and $\Delta t = 1$, $\sigma_{\mathrm{stat}} = \sigma/\sqrt{3/4} \approx 1.155\,\sigma$.

Boundary cases. $\beta = 0$ ($k\Delta t = 1$): $e_t = \bar{\sigma}\zeta_t$ is white with variance $\bar{\sigma}^2$. $\beta = 1$ ($k = 0$): there is no fixed point; $a_n = a_0 + n\,\Delta t\, v + \bar{\sigma} \sum_{j \le n} \zeta_j$ is a random walk with drift, $\mathrm{Var}(a_n) = n\,\bar{\sigma}^2$, and the exploration noise accumulated since the last reset is unbounded in n.

Continuous-time limit. With k and σ fixed and $\Delta t \to 0$,

$$\begin{aligned} \sigma^2_{\mathrm{stat}} &= \frac{\sigma^2 \Delta t}{1-(1-k\Delta t)^2} = \frac{\sigma^2}{2k - k^2\Delta t} \;\to\; \frac{\sigma^2}{2k}, \\ \mathrm{corr}\left(e_t, e_{t+s/\Delta t}\right) &= (1-k\Delta t)^{s/\Delta t} \;\to\; e^{-ks}, \end{aligned} \tag{A.21}$$

the stationary variance and autocorrelation of the Ornstein–Uhlenbeck process $da_\tau = (v - k a_\tau)\, d\tau + \sigma \odot dW_\tau$ (Uhlenbeck & Ornstein, 1930), which is the SDE (7) with the drift (6) at constant v. The retention coefficient is thus the discrete-time face of an OU time constant $1/k$: exploration noise is low-pass filtered with that time constant, in the spirit of the OU noise of Lillicrap et al. (2016), but produced by the policy class itself and scored exactly by (10), as in the autoregressive policies of Korenkevych et al. (2019).

Exploration noise and action-rate penalties. The carry also changes how much of the exploration noise reaches the action differences that many rewards penalize. With v_θ held fixed, the stationary deviation e_t above gives

$$\mathrm{Var}(e_t - e_{t-1}) = \frac{2\bar{\sigma}^2}{1+\beta}, \qquad \mathrm{Var}(e_t - 2e_{t-1} + e_{t-2}) = \frac{2\bar{\sigma}^2(3-\beta)}{1+\beta}, \tag{A.22}$$

per dimension. The white-noise values are $2\bar{\sigma}^2$ and $6\bar{\sigma}^2$. At $\beta = \frac{1}{2}$ and equal $\bar{\sigma}$, the carry lowers the noise contribution to the first difference by a third and to the second difference by 44%. Seven of our eight robotics rewards penalize these differences: velocity tracking, humanoid skateboarding and in-hand reorientation penalize both, and motion tracking, dexterous lifting, bimanual piano

and vision-based pick-up penalize the first. Lower penalty exposure under exploration noise could therefore contribute to SFPO's stochastic training returns on these tasks. Two observations suggest that it is not the main effect. On the reorientation sweep of Figure 3, the integrator and $k\Delta t = 0.5$ share $\bar{\sigma}$, and (A.22), which for the differences also holds at $\beta = 1$, predicts a 36% lower noise contribution to the action-smoothness penalty for the integrator ($3\bar{\sigma}^2$ against $4.7\bar{\sigma}^2$). Its measured penalty is instead slightly higher (6.8 against 6.4), which suggests that at the end of training the learned mean, not the noise filter, dominates that penalty. Across the robotics tasks, SFPO's action-rate penalty is not uniformly lower than PPO's: it is lower on velocity tracking, motion tracking and vision-based pick-up, and similar or higher on dexterous lifting, in-hand reorientation and bimanual piano.

The carried action does not introduce latency. The derivation above holds v_θ fixed. Since a_{t-1} is an input to v_θ, v_θ can cancel the carry: $v_\theta(a, \tau \mid s) = -(\beta/\Delta t)\, a + w(s)$ gives $\mu_\theta(\tilde{s}_t) = \Delta t\, w(s_t)$, a memoryless mean, and conversely at $\beta = 0$ the output $v_\theta = a/\Delta t + w(s)$ reinstates a unit carry. A network with a linear skip realizes these maps exactly and an MLP approximates them. The mean therefore conditions on the current observation at every step with no delay, and β sets the default filter through which the sampled noise passes at initialization—a prior over the temporal structure of exploration, not a constraint on the policy (Appendix A.5).

A.13 The chunk period and the scaling of Δt

Setting $T > 1$ makes the phase τ_t vary. SFP additionally couples $\Delta t = 1/T$ to it, so that flow time $[0, 1]$ spans one chunk. This subsection collects what the coupled change does, at a fixed gain k and a fixed diffusion coefficient σ; write $\beta = 1 - k\Delta t = 1 - k/T$. Nothing in the kernel, the storage or the objective changes: τ_t is already part of the augmented state (4) and is stored per transition, so the replay is exact and (10), (12), (13), (14) and (15) hold verbatim.

Exploration memory. By Appendix A.12 the carry has autocorrelation $\mathrm{corr}(e_t, e_{t+\ell}) = \beta^\ell$, hence a correlation time

$$\ell_c = -\frac{1}{\log \beta} = -\frac{1}{\log(1 - k/T)} = \frac{T}{k} + O(1). \tag{A.23}$$

At $k = \frac{1}{2}$ this grows from $\ell_c \approx 1.44$ control steps at $\Delta t = 1$ to $\approx 2T$: the exploration memory becomes chunk-long. This is what the chunk period is for. It is finite for every T, since $\beta < 1$ implies $\beta^\ell \to 0$ — a statement about the linearized carry, qualified in Appendix A.14.

Exploration scale. Two standard deviations have to be kept apart. The per-step one is $\bar{\sigma} = \sigma\sqrt{\Delta t} = \sigma/\sqrt{T}$. The trajectory-level one is the stationary spread of the carry, which by (11) with $1 - \beta^2 = k\Delta t\,(2 - k\Delta t)$ is

$$\sigma_{\mathrm{stat}} = \frac{\bar{\sigma}}{\sqrt{1 - \beta^2}} = \frac{\sigma}{\sqrt{k\,(2 - k\Delta t)}} \xrightarrow[\Delta t \to 0]{} \frac{\sigma}{\sqrt{2k}}, \tag{A.24}$$

the Ornstein–Uhlenbeck value of Appendix A.12: it depends on Δt only through the $O(k\Delta t)$ correction in the denominator. So σ, not $\bar{\sigma}$, is the Δt-free handle on how widely the policy explores, and it is the one to carry across a change of Δt unchanged. The opposite choice — holding the per-step noise fixed by taking $\sigma \mapsto \sigma\sqrt{T}$ — multiplies σ_{stat} by $\sqrt{T}$ and replaces a low-variance, persistent process with a wide random walk. At $k = \frac{1}{2}$ and $\sigma = 1$: $\sigma_{\mathrm{stat}} = 1.155$ at $\Delta t = 1$ and 1.004 at $T = 32$, against 5.68 under the rescaling.

Entropy. From (14), $\mathcal{H}(\bar{\sigma}) = \sum_i \log \sigma_i + \frac{A}{2} \log \Delta t + \frac{A}{2} \log(2\pi e)$, so at $\Delta t = 1/T$ and fixed σ the entropy is shifted down by exactly $\frac{A}{2} \log T$. The shift is an additive constant, and both sides of the bonus–surrogate balance are Δt-free in $\log \sigma$: $\partial \mathcal{H}/\partial \log \sigma_i = 1$, and by (A.1) $\partial \log \pi_\theta / \partial \log \sigma_i = \zeta_i^2 - 1$ with $\zeta = (a_t - \mu_t)/\bar{\sigma}$, which is standard normal under the behaviour policy whatever Δt is. The coefficient $c_{\mathcal{H}}$ therefore has the same effect at any Δt, even though the logged entropy does not have the same value.

The KL target. By Appendix A.6 the per-step KL at fixed σ is $\Delta t\, \|\Delta v\|^2_{\sigma^{-2}}/2$, so a given movement of v_θ registers T times smaller at $\Delta t = 1/T$. It does not follow that $\mathrm{KL}^\star$ must be rescaled, and the distinction is worth stating because the two readings differ. Both arguments of (15) are distributions over the *executed action* at one control step, whatever Δt is, and with equal standard deviations the divergence is $\frac{1}{2}\|\Delta\mu\|^2_{\bar{\sigma}^{-2}}$: half the squared shift of that distribution, measured in its own standard

deviations. That is the quantity PPO's target constrains, and the quantity (A.6) regulates; it is Δt-free. What Δt changes is the parameter movement that a given budget allows — T times more Δv at $\Delta t = 1/T$, producing the same $\Delta\mu$ — not what the budget means. The comparison that is genuinely unit-dependent is a per-step budget against a per-chunk one: T control steps accumulate T times the per-step divergence.

What the chunk period does not provide. It does not provide action chunking. The mean (9) evaluates v_θ at $\nu_\psi(s_t)$ on every control step, so a chunked stream is phase-conditioned and closed-loop: there is no open-loop segment, and therefore no K-fold latency to amortize as it is for a denoising chain. Open-loop execution in SFP's sense would require latching the observation at the chunk boundary and holding it for T steps, which replaces s_t in (4) with the latched observation and its offset—a different augmented MDP, still covered by Proposition 1 but not the one analysed here. The chunk period also does not reach multimodality; that is the subject of Appendix A.14.

A.14 Multimodality under the exact density

That a single Gaussian per step cannot represent a multimodal per-step action distribution is immediate from (9). The trajectory-level statement is less immediate and is the one that matters, because SFP is multimodal at the trajectory level without ever being multimodal per step.

Determinism gives injectivity. Under a locally Lipschitz field the flow map $\Phi_\tau\colon a(0) \mapsto a(\tau)$ is injective, by uniqueness of solutions; for SFP's conditional field (3) this is explicit in (A.8), where Φ_τ is affine with positive slope $e^{-k\tau}$. Hence $a(0) \mapsto \big(a(\tau)\big)_{\tau\in[0,1]}$ is injective, and if the initial distribution is a mixture $\sum_m w_m\, p_m^0$ with disjointly supported components, so is the induced law over trajectories: a sample belongs to one branch for its entire length, and branches cannot cross. This is how SFP produces distinct behaviours with no action noise, and why its latent-variable variant carries the latent rather than widening σ_0.

What full support preserves. SFPO's kernel has $\bar\sigma_i > 0$ for every i, so for every $\tilde s_t$ and every Borel set $B \subseteq \mathbb{R}^A$ of positive Lebesgue measure, $\pi_\theta(B \mid \tilde s_t) > 0$. Conditioning on the environment's own randomness and iterating, the law of $a_{0:H}$ has a strictly positive density on $(\mathbb{R}^A)^{H+1}$. This removes the *disjoint support* of the previous paragraph, and only that. It does not make the trajectory law unimodal, since a mixture of Gaussians also has full support. Two properties survive. Because a_{t-1} is an input to v_θ, the drift (6) is free to have several basins, and the mean carries a sample along whichever basin its carry occupies; with $\bar\sigma$ small relative to the separation between the basins, crossings are rare and a sample can follow one branch for an entire episode. The same strict positivity is what makes (10) an ordinary log-density and (12) an ordinary importance ratio, which is the entire point of replacing the Euler step with an Euler–Maruyama one.

What full support changes. Two things change. First, separation is only ever metastable: every branch has positive density from every initial condition, so there is no exact mixture decomposition and no guarantee that a sample stays where it started. Second, the carry is simultaneously the mode label and the executed action. Two samples that occupy the same action at the same phase and observation share the augmented state (4) exactly, so (9) must assign them the same distribution over what follows: branches that *cross* in action space cannot be told apart at all, however small $\bar\sigma$ is. Under SFP's injective flow map they never meet in the first place. SFPO therefore retains trajectory-level multimodality; what changes is that the mode is carried by the action it is about to execute.

A latent variable for mode selection. Selecting a mode requires a variable that is *random per sample* and *persistent*. The augmented state (4) has one of each and neither of both: τ_t is exactly persistent but deterministic — a clock, taking the same value in every environment at the same step, carrying no per-sample randomness — while a_{t-1} is random but is not free to label a mode independently of the action being executed. (Its decay at rate β^ℓ, from Appendix A.12, is the linearized statement, obtained at $v_\theta \equiv v$ around a single fixed point; it does not describe a multi-basin field.) Raising the chunk period lengthens the exploration memory to $\ell_c \approx T/k$ by (A.23) and changes neither fact. Augmenting the state with a latent $z \in \mathcal{Z}$ drawn once at episode start from a fixed distribution and held fixed — SFP's latent-variable variant — supplies exactly the missing variable. The draw is part of $p(\tilde s_0)$ and is θ-independent, so Proposition 1 goes through on $\widetilde{\mathcal{S}} \times \mathcal{Z}$ with

$\tilde{s}_t = (s_t, a_{t-1}, \tau_t, z)$, the per-step density (10) remains exact conditional on z, and the trajectory law becomes a genuine mixture over z even though each component is still per-step Gaussian.

A.15 Deterministic deployment reproduces SFP inference

Setting $\zeta_t \equiv 0$ in (8) and using the drift (6),

$$a_t = \mu_\theta(\tilde{s}_t) = \beta\, a_{t-1} + \Delta t\, v_\theta(a_{t-1}, \tau_t \mid s_t) = a_{t-1} + \Delta t\, f_\theta(a_{t-1}, \tau_t \mid s_t), \tag{A.25}$$

which is one Euler step of the ODE $\dot{a} = f_\theta(a, \tau \mid s)$ from the previous action: SFP's streaming inference with $\sigma_0 = 0$, step $\Delta\tau = \Delta t$, and its field written in the form (6). Each step is the mode of the per-step kernel (9), and at $k\Delta t = 1$ with an observation-only network it is PPO's mean action. In SFPO the deployment stream is zero-seeded at the start of an episode, so the first action is $\Delta t\, v_\theta(0, 0 \mid s_0)$, and the state is carried through the episode exactly as in training, without noise.

B Implementation details

This appendix gives the network and the implementation choices of SFPO that Section 4 leaves out. Algorithm 1 in Appendix A.10 gives the full iteration.

Network and diffusion. The velocity network takes the normalized observation, an embedding of the phase, and the carried action:

$$v_\theta(a_{t-1}, \tau_t \mid s_t) = \mathrm{MLP}_\theta\big([\,\nu_\psi(s_t),\ e(\tau_t),\ a_{t-1}\,]\big), \tag{B.1}$$

where $e(\cdot)$ is a fixed sinusoidal embedding and ν_ψ an optional observation normalizer with running statistics ψ. The diffusion coefficient is a per-dimension parameter stored as $\log\sigma$. It is learned by default, so the entropy bonus has a target as in PPO; if held fixed, it sets a floor on exploration.

Streaming state. The carried action and the counter (a_{t-1}, n_t) start at zero, are reset to zero when an environment terminates, and are not stored with the parameters. They do not depend on θ, so resets contribute no policy-gradient terms. The sampled action a_t is executed without clamping[2] and carried to the next step, where the carried copy may be clamped to $[-\bar{a}, \bar{a}]$. The clamp acts after a_t has been sampled and scored, so it bounds the carried state without changing the density.

Ratio and KL. Because σ is constant within a rollout, the stored side of (15) needs one snapshot of $\bar{\sigma}$ per iteration. The log-ratio has a one-sided straight-through clamp at 10 as a safeguard, which exact ratios rarely reach.

Relation to SFP. Switching the diffusion off recovers SFP's deterministic inference (Appendix A.15), and SFP's flow-matching loss (A.7) can pretrain the same network.

Chunk period and step size. SFP couples the step size to the chunk period through $\Delta t = 1/T$. At fixed k, Δt sets the retention $k\Delta t$. We keep T and Δt as separate settings, so the phase and the longer memory of Section 4.3 can be switched on independently.

Freezing the observation normalizer. The identity $\rho_t(\theta_{\mathrm{old}}) = 1$ requires the replay to see the same kernel inputs as the rollout. If the normalizer statistics ψ are updated during the rollout, as in standard PPO implementations, a stored observation is normalized differently at replay. The two kernels then differ at unchanged θ, so the measured KL is positive before any gradient step, and the controller (A.6) reads it as an over-large step. SFPO therefore freezes the actor's normalizer during the rollout and its replay, and updates it with the stored observations once per iteration, after the last epoch. The critic's normalizer never enters the ratio and still updates at every step.

C DeepMind Control setup

The six DeepMind Control tasks of Figure 1 use the MuJoCo implementation of the suite (Tunyasuvunakool et al., 2020) with state observations. The policy and the critic both receive the suite's flat proprioceptive observation vector, which does not contain the previous action. We change three

[2] Optionally after a deterministic constant post-scale $u_t = \alpha a_t$; we take $\alpha = 1$ throughout. A runner-level clip on the executed action, if any, sits outside the density, as in PPO.

things. The suite's time limit is removed, a wrapper cuts episodes at 1000 control steps, and every episode end is treated as a truncation, so the value is bootstrapped. The first episode of each environment starts at a random step count, so that resets are spread across environments. Actions are clipped to $[-1, 1]$. There is no action repeat, and the rewards are the suite's. Each iteration collects 64 steps from each of 64 parallel environments, for 1250 iterations or 5.12M environment steps, and each method has three seeds.

PPO and SFPO share every optimizer and network setting: a learning rate of 10^{-3} adapted to a KL target of 0.01, $\gamma = 0.99$, $\lambda = 0.95$, five epochs of four minibatches, clip $\epsilon = 0.2$, entropy coefficient 0.005, a clipped value loss with weight 1, a gradient-norm limit of 1, and ELU MLPs with three hidden layers of 256 for the actor and the critic. Every arm normalizes its observations. PPO's standard deviation and SFPO's σ are initialized to 1. SFPO runs its default kernel ($k\Delta t = 0.5$, $\Delta t = 1$, phase off), and its carried action is clipped to $[-1, 1]$, so it equals the action the environment executed. The critic observes s_t (Section 4.4). DPPO trains a $K = 8$ denoising chain from scratch with an actor learning rate of 10^{-4}, eight minibatches and a per-denoising-step clip of 0.01. FPO++ uses 16 flow-matching samples per action, 16 sampling steps, clip 0.05, a learning rate of 10^{-4} and 16 epochs. ARPPO uses the setting of Appendix D.1.

D Robotics benchmark tasks

This appendix describes the eight robotics control tasks in Figure 2. Appendix D.1 gives the settings that all eight share and Table 3 summarizes each task. The subsections that follow give each task's actions, observations, rewards, terminations and randomization. The figure for each task is a four-frame montage from a single evaluation episode of the trained SFPO policy.

D.1 Common setup

Simulation and control. All eight tasks run in the Genesis simulator (Genesis Authors, 2024) with its rigid-body solver. The policy acts at 50 Hz, a control period of $\Delta_c = 0.02$ s, except on in-hand reorientation and bimanual piano, which act at 20 Hz ($\Delta_c = 0.05$ s). Unless a task says otherwise, an action $a \in \mathbb{R}^A$ sets joint position targets $q^\star = q_0 + s \odot a$ around a default pose q_0, with a per-joint scale s, and a PD controller tracks $q^\star$. For the three humanoid tasks and the quadruped the gains follow the rule used by BeyondMimic (Liao et al., 2026): $k_p = I\omega_n^2$ and $k_d = 2\zeta I\omega_n$, with I the joint's reflected inertia, a natural frequency ω_n and a damping ratio $\zeta = 2$. The action scale is then $s_j = 0.25\,\tau_{\max,j}/k_{p,j}$.

Rewards. The reward at each control step is $r_t = \Delta_c \sum_i w_i\, r_i$, a weighted sum of the terms listed for each task. Penalties have negative weights. Tracking terms are written as $\exp(-e/\sigma^2)$ for a squared error e, or as $\exp(-e/\sigma)$ for an unsquared one.

Observations. The actor and the critic observe separate groups, except on bimanual piano, where they share one. Noise is added only to the actor's observations. Unless noted, it is uniform, $\mathrm{Unif}[-n, n]$, and we report its half-width n next to each term. The critic observes the same quantities without noise, together with privileged state that the actor does not see. Where a task stacks a history of frames, it does so for the actor only, except on in-hand reorientation.

Training. Table 4 lists the training settings. On every task the environment is the same for all methods, and PPO and SFPO share the settings in the table. All methods use GAE with $\lambda = 0.95$, and PPO, ARPPO and SFPO use clip $\epsilon = 0.2$ and a gradient-norm limit of 1. Where the table says "adaptive", the learning rate is adapted to a KL target of 0.01 (Appendix A.3). The actor and the critic are ELU MLPs. PPO's actor is a Gaussian with a state-independent standard deviation initialized to 1 (0.5 on reorientation and piano). SFPO runs its default kernel ($k\Delta t = 0.5$, $\Delta t = 1$, phase off), with σ initialized to 1 (0.5 on piano). On reorientation SFPO initializes $\sigma = 0.433$, so that its stationary spread $\sigma/\sqrt{1-\beta^2}$ matches PPO's initial 0.5 (Appendix A.12). ARPPO, on both benchmarks, is PPO with its actor replaced by the autoregressive policy of Korenkevych et al. (2019), of order $p = 3$ and smoothing parameter 0.3, the smallest value in their MuJoCo experiments. Following their finding that this learns more stably, its critic takes only s_t as input.

Table 3: **Robotics tasks at a glance.** A is the action dimension. The observation columns count the actor's full input, including any stacked history, and the critic's. The vision task adds a 72×128 RGB image to both. Episode length is in control steps.

task	robot	A	actor obs.	critic obs.	control	episode
Velocity tracking	LimX Oli humanoid	31	1445	292	50 Hz	1000
Motion tracking	Unitree G1 humanoid	29	160	286	50 Hz	500
Humanoid skateboarding	Unitree G1 humanoid (23 DoF)	23	395	215	50 Hz	1000
Dexterous lifting	Franka Panda + XHand	18	84	84	50 Hz	300
In-hand reorientation	Wuji hand	20	207	403	20 Hz	1000
Bimanual piano	two Wuji hands, floating wrists	52	1212	1212	20 Hz	761
Vision-based pick-up	Franka Panda + Robotiq 2F-85	7	33 + RGB	40 + RGB	50 Hz	300
Quadruped handstand	Unitree Go1	12	45	93	50 Hz	500

Table 4: **Training settings.** "Steps" is the number of steps collected from each environment per iteration, $c_{\mathcal{H}}$ the entropy coefficient, and the widths are the actor MLP's hidden layers. The critic has the same widths, except on reorientation, where it is 512-512-256-128. The discount is $\gamma = 0.99$ except on piano (0.97). The piano runs were trained for 10k iterations and are plotted to 8k.

task	envs	steps	iters	learning rate	epochs×mb	$c_{\mathcal{H}}$	MLP widths
Velocity tracking	4096	24	10k	10^{-3}, adaptive	5×4	0.01	512-256-128
Motion tracking	4096	24	15k	10^{-3}, adaptive	5×4	0.005	512-256-128
Humanoid skateboarding	4096	24	10k	10^{-3}, adaptive	5×4	0.005	512-256-128
Dexterous lifting	4096	24	8k	10^{-3}, adaptive	5×4	0.005	512-512-256-128
In-hand reorientation	4096	40	5k	10^{-4}, fixed	4×32	0.001	512-256-128
Bimanual piano	1024	32	10k	3×10^{-4}, adaptive	5×8	0.002	512-256-128
Vision-based pick-up	1024	24	3k	10^{-3}, adaptive	5×4	0.01	256-256-128
Quadruped handstand	4096	24	3k	10^{-3}, adaptive	5×4	0	512-256-128

D.2 VELOCITY TRACKING

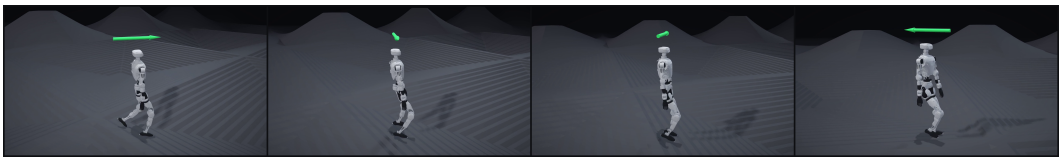

Figure 5: **Velocity tracking.** A humanoid tracks a commanded base velocity over uneven terrain, shown here holding each of the four scripted directions (the floating arrow is the command).

Task. A LimX Oli humanoid with 31 actuated joints (6 per leg, 3 in the waist, 2 in the neck, 7 per arm) tracks a commanded base velocity $c = (v_x^\star, v_y^\star, \omega_z^\star)$, expressed in its body frame, on rough terrain. The terrain is a 10×10 grid of 8×8 m tiles. Its columns are flat ground, pyramid stairs, inverted stairs, pyramid and inverted slopes, uniform height noise, and waves, and difficulty increases along its rows. There is no terrain curriculum. The command is resampled every $\mathrm{Unif}[3, 8]$ s with $v_x^\star, v_y^\star \sim \mathrm{Unif}[-1, 1]$ m/s and $\omega_z^\star \sim \mathrm{Unif}[-0.5, 0.5]$ rad/s. Ten percent of environments receive a zero command. Thirty percent receive a heading target $\psi^\star \sim \mathrm{Unif}[-\pi, \pi]$ instead of $\omega_z^\star$, and for them $\omega_z^\star = 0.5\,(\psi^\star - \psi)$, clipped to the same range. At iteration 5000 a command curriculum widens the ranges to $v_x^\star \in [-1.5, 2]$ m/s and $\omega_z^\star \in [-0.7, 0.7]$ rad/s.

Actions. The 31 actions are joint position targets with scales between 0.11 and 0.31 rad, tracked by an implicit PD controller at $\omega_n = 2\pi \cdot 5$ Hz. An episode lasts 20 s (1000 steps).

Observations. Each actor frame contains the base angular velocity (3, $n = 0.2$), the gravity vector projected into the base frame (3, $n = 0.05$), the joint positions relative to q_0 (31, $n = 0.01$), the joint velocities scaled by 0.05 (31, $n = 1.5$), the previous action (31), the command (3), and a 17×11 grid of terrain heights relative to the base, sampled at 0.1 m spacing around it and scaled by 0.2 (187,

$n = 0.1$). The actor stacks the last five frames (1445 dimensions). The critic observes one frame together with the base linear velocity (292 dimensions).

Terminations and randomization. An episode ends at the time limit, when the torso tilts more than 70° from vertical, or when the base comes within 3 m of the terrain edge; the last is treated as a time-out. At reset the base is placed at a uniform offset of up to 0.5 m with a uniform yaw. Every $\mathrm{Unif}[1, 3]$ s the base velocity receives a push of up to 0.5 m/s horizontally, 0.4 m/s vertically, 0.52 rad/s in roll and pitch and 0.78 rad/s in yaw. At startup, each environment samples a foot friction coefficient from $\mathrm{Unif}[0.3, 1.2]$, a torso center-of-mass offset of up to 2.5 cm horizontally and 3 cm vertically, and a joint-target bias of up to 0.015 rad.

Table 5: **Velocity tracking rewards.** v and ω are the base linear and angular velocity in the body frame, g^{t} is the unit gravity vector in the torso frame, and z_f, v_f are the height and velocity of foot f. The posture width σ_j is 0.05 for every joint under a zero command and ranges from 0.08 to 0.6 per joint when walking or running. The two foot terms apply only under a nonzero command.

term	weight	form
linear velocity tracking	2.0	$\exp\big(-(\lVert v^\star_{xy} - v_{xy}\rVert^2 + 2v_z^2)/0.25\big)$
angular velocity tracking	2.0	$\exp\big(-((\omega^\star_z - \omega_z)^2 + 0.05\lVert\omega_{xy}\rVert^2)/0.5\big)$
upright	1.0	$\exp\big(-\lVert g^{\mathrm{t}}_{xy}\rVert^2/0.2\big)$
posture	1.0	$\exp\big(-\frac{1}{A}\sum_j (q_j - q_{0,j})^2/\sigma_j^2\big)$
torso angular velocity	-0.05	$\lVert\omega^{\mathrm{t}}_{xy}\rVert^2$
joint limits	-1.0	distance outside 90% of each joint's range, summed
action rate	-0.1	$\lVert a_t - a_{t-1}\rVert^2$
action smoothness	-0.05	$\lVert a_t - a_{t-1}\rVert^2 + \lVert a_t - 2a_{t-1} + a_{t-2}\rVert^2$
joint acceleration	-6.25×10^{-4}	$\lVert\dot q_t - \dot q_{t-1}\rVert^2$
foot clearance	-2.0	$\sum_f \lvert z_f - 0.125\rvert\, \lVert v_{f,xy}\rVert$
foot slip	-0.1	$\sum_f \lVert v_{f,xy}\rVert^2\, \mathbb{1}[z_f < 0.05]$
self-collision	-1.0	number of self-contacts

D.3 Motion tracking

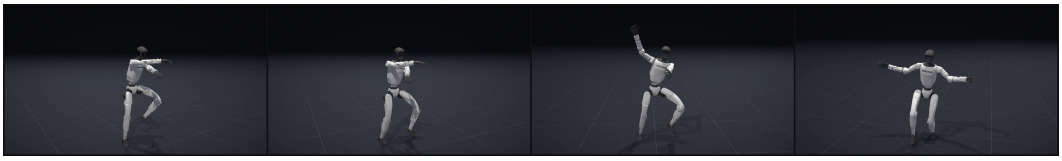

Figure 6: **Motion tracking.** A humanoid tracks a reference motion clip; the four frames sample distinct poses from one episode.

Task. A Unitree G1 humanoid with 29 actuated joints tracks one retargeted motion-capture clip with the formulation of BeyondMimic (Liao et al., 2026). The clip is a 32 s dance recorded at 60 Hz and resampled to 50 Hz. Each episode starts at a uniformly sampled phase of the clip, from the reference state plus noise. When the clip ends, the phase is resampled and the robot is placed on the reference. The ground is flat and self-collision is disabled. Fourteen bodies are tracked: the pelvis, torso, hips, knees, ankles, shoulders, elbows and wrists. The torso is the anchor. For the relative body terms, the reference bodies are moved to the robot's anchor, rotated by the heading difference and translated to its horizontal position, before the errors are computed.

Actions. The 29 actions are joint position targets tracked by an explicit PD controller, $\tau = k_p(q^\star - q) - k_d\dot q$, at $\omega_n = 2\pi \cdot 10$ Hz. The scales range from 0.07 to 0.55 rad. An episode lasts 10 s (500 steps).

Observations. The actor observes the reference joint positions and velocities (29+29), the reference anchor position (3, $n = 0.25$) and orientation (the first two columns of the rotation matrix, 6, $n = 0.05$) in the robot's anchor frame, the base linear (3, $n = 0.5$) and angular (3, $n = 0.2$) velocity, the joint positions relative to q_0 (29, $n = 0.01$) and velocities (29, $n = 0.5$), and the

previous action (29), for 160 dimensions. The critic adds the positions (42) and orientations (84) of the tracked bodies relative to the anchor (286 dimensions).

Terminations and randomization. An episode ends at the time limit, when the anchor height differs from the reference by more than 0.25 m, when the vertical component of the projected gravity differs from the reference's by more than 0.8, or when an ankle or wrist height differs from the reference by more than 0.25 m. At reset the base pose is perturbed by up to 5 cm horizontally, 1 cm vertically, 0.1 rad in roll and pitch and 0.2 rad in yaw, the base velocity by up to 0.5 m/s horizontally, 0.2 m/s vertically, 0.52 rad/s in roll and pitch and 0.78 rad/s in yaw, and each joint by up to 0.1 rad. Pushes of the same velocity ranges arrive every $\mathrm{Unif}[1, 3]$ s. At startup, each environment samples a foot friction coefficient from $\mathrm{Unif}[0.3, 1.2]$, a pelvis center-of-mass offset of up to 2.5 cm in x and 5 cm in y and z, and a joint-target bias of up to 0.01 rad.

Table 6: **Motion tracking rewards.** Hats denote the reference and tildes the reference moved to the robot's anchor. p, R, v, ω are world-frame position, orientation, linear and angular velocity, $\angle(\cdot, \cdot)$ is the angle between two orientations, and the means run over the 14 tracked bodies b.

term	weight	form
anchor position	0.5	$\exp\big(-\lVert p_{\mathrm{a}} - \hat{p}_{\mathrm{a}}\rVert^2/0.3^2\big)$
anchor orientation	0.5	$\exp\big(-\angle(R_{\mathrm{a}}, \hat{R}_{\mathrm{a}})^2/0.4^2\big)$
body position	1.0	$\exp\big(-\mathrm{mean}_b\lVert p_b - \tilde{p}_b\rVert^2/0.3^2\big)$
body orientation	1.0	$\exp\big(-\mathrm{mean}_b\angle(R_b, \tilde{R}_b)^2/0.4^2\big)$
body linear velocity	1.0	$\exp\big(-\mathrm{mean}_b\lVert v_b - \hat{v}_b\rVert^2/1.0^2\big)$
body angular velocity	1.0	$\exp\big(-\mathrm{mean}_b\lVert \omega_b - \hat{\omega}_b\rVert^2/3.14^2\big)$
action rate	-0.1	$\lVert a_t - a_{t-1}\rVert^2$
joint limits	-10.0	distance outside 90% of each joint's range, summed
self-collision	-10.0	self-contacts (always zero: self-collision is off)

D.4 HUMANOID SKATEBOARDING

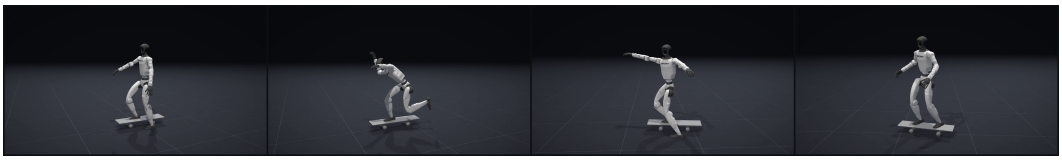

Figure 7: **Humanoid skateboarding.** A humanoid balances on a skateboard and drives it forward: upright cruise, a push crouch, a deep carve, and a recovery to balance.

Task. This task follows HUSKY (Han et al., 2026). A Unitree G1 humanoid with 23 actuated joints (the wrists are fixed) rides a passive skateboard. The board is a free body with seven passive joints: a deck tilt, two truck-roll joints and four wheels. An equality constraint couples the deck tilt to the truck roll, so leaning the deck steers the board. The command is a forward board speed $v^\star \sim \mathrm{Unif}[0, 1.5]$ m/s and a heading change $\Delta\psi^\star \sim \mathrm{Unif}[-\pi/4, \pi/4]$ relative to the board, sampled once per episode. A 6 s phase clock cycles through four segments: a push phase ($[0, 0.4)$), in which one foot strokes the ground; a push-to-steer transition ($[0.4, 0.5)$); a steer phase ($[0.5, 0.95)$), in which both feet stay on the deck and the robot leans to turn; and a steer-to-push transition ($[0.95, 1)$). The heading target is re-anchored to the board's heading at the end of each push-to-steer transition. During the two transitions, 14 robot links track targets that are interpolated from their poses at the start of the transition to recorded push and steer poses. Below $v^\star = 0.1$ m/s the clock is frozen and a stand-still reward applies. HUSKY also uses an adversarial motion prior to shape the push. We omit it, so the rewards below are the only learning signal. This makes the task harder than the published one.

Actions. The 23 actions are joint position targets with scales between 0.35 and 0.55 rad, tracked by an implicit PD controller at $\omega_n = 2\pi \cdot 10$ Hz. An episode lasts 20 s (1000 steps).

Observations. Each actor frame contains the command (2), the pelvis heading (1), the base angular velocity (3, $n = 0.2$), the projected gravity (3, $n = 0.05$), the joint positions relative to q_0 (23, $n = 0.01$) and velocities (23, $n = 1.5$), the previous action (23) and the phase (1). The actor stacks the last five frames (395 dimensions). The critic observes one frame together with the base linear velocity, the heading error, the board's pose and velocity in the pelvis frame, the deck tilt and truck angles, the 14 transition targets, the foot contact forces with the ground and the board, and a one-hot contact phase (215 dimensions).

Terminations and randomization. An episode ends at the time limit, when the pelvis tilts more than 70°, when both feet leave the board, or when the pelvis, a knee, a shoulder, an elbow or a wrist makes contact. At startup, each environment samples a friction coefficient for the robot from $\mathrm{Unif}[0.3, 1.6]$, for the feet from $\mathrm{Unif}[0.3, 1.8]$ and for the deck from $\mathrm{Unif}[0.8, 2.0]$, a torso center-of-mass offset of up to 2.5 cm horizontally and 3 cm vertically, and a deck center-of-mass offset of up to 2 cm horizontally and 1 cm vertically. Every $\mathrm{Unif}[5, 10]$ s the base receives a horizontal velocity push of up to 0.5 m/s.

Table 7: **Humanoid skateboarding rewards.** Each phase term is multiplied by the indicator of its phase. v^{b} is the board velocity in its own frame, ψ and ψ^{b} are the robot and board headings, γ is the deck tilt and γ_{ref} the tilt that turns the board onto $\psi^{\star}$ in the remaining steer time. "Push foot" is the foot that strokes the ground.

term	weight	form
push phase		
board velocity	3.0	$\exp\big(-((v^{\star}-v_x^{\mathrm{b}})^2+(v_y^{\mathrm{b}})^2+(v_z^{\mathrm{b}})^2)/0.25\big)$
heading alignment	1.0	$\exp\big(-\lvert\psi-\psi^{\mathrm{b}}\rvert/0.25\big)$
push-foot air time	3.0	$\mathbb{1}[\,0.1 <$ air time $< 0.5\,\mathrm{s}\,]$
flat push-foot contact	0.5	$\mathbb{1}$[push foot on the ground and flat]
steer phase		
feet on board	3.0	$2\cdot\mathbb{1}$[both feet on the deck] $-\,\mathbb{1}$[push foot on the ground]
steer posture	1.5	$\exp\big(-\mathrm{mean}_j(q_j-q_j^{\mathrm{steer}})^2/0.2\big)$
foot placement	1.0	$\exp(-\bar{d}/0.1)$, $\bar{d}$ the mean foot-to-marker distance on the deck
heading tracking	5.0	$\exp\big(-\lvert\psi^{\mathrm{b}}-\psi^{\star}\rvert/0.02\big)$
tilt guidance	4.0	$\exp\big(-\lvert\gamma-\gamma_{\mathrm{ref}}\rvert/0.02\big)$
transitions		
link position	10.0	$\exp\big(-\mathrm{mean}_{\ell}\lVert p_{\ell}-p_{\ell}^{\mathrm{target}}\rVert^2/0.05\big)$
link orientation	10.0	$\exp\big(-\mathrm{mean}_{\ell}\angle(R_{\ell},R_{\ell}^{\mathrm{target}})^2/0.1\big)$
push-foot contact	-0.5	$\mathbb{1}$[push foot on the ground]
all phases		
wheels on ground	0.5	$\mathbb{1}$[all four wheels in contact]
flat board	3.0	$\exp(-\lvert\gamma\rvert/0.05)$, outside the steer phase
stand still	1.0	$\exp\big(-\mathrm{mean}_j(q_j-q_{0,j})^2/0.1\big)$, when $v^{\star}<0.1$
joint limits	-5.0	distance outside the soft joint limits, summed
action smoothness	-0.1	$\lVert a_t-a_{t-1}\rVert^2+\lVert a_t-2a_{t-1}+a_{t-2}\rVert^2$
joint velocity	-10^{-3}	$\lVert\dot{q}\rVert^2$
joint acceleration	-6.25×10^{-4}	$\lVert\dot{q}_t-\dot{q}_{t-1}\rVert^2$
joint torque	-10^{-6}	$\lVert\tau\rVert^2$
self-collision	-10.0	$\mathbb{1}$[any self-contact]

D.5 DEXTEROUS LIFTING

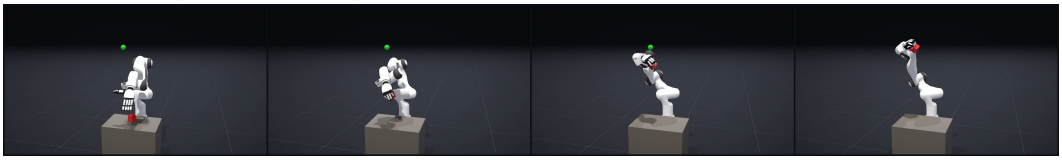

Figure 8: **Dexterous lifting.** A dexterous hand grasps a cube and lifts it to the goal: grasp on the pedestal, lift-off, mid-rise, and raised to the goal marker.

Task. A 7-DoF Franka Panda arm carrying a 12-DoF RobotEra XHand grasps a 6 cm cube (43 g) from a table, lifts it to a goal position and holds it there. The goal is sampled once per episode in the arm's base frame, from $x \in [0.18, 0.40]$, $y \in [-0.15, 0.15]$ and $z \in [0.665, 0.865]$ m. A curriculum starts the goal box at a single point and widens it toward these ranges. It moves the level up by 0.05 times the fraction of finished episodes when the success rate (the cube within 8 cm of the goal at some point in the episode, averaged over the last 4096 episodes) exceeds 0.7, and down by the same amount when it falls below 0.4. The penalty weights also follow a schedule, marked in Table 8.

Actions. Six actions command an end-effector displacement of $0.05 \tanh(a)$ m in position and $0.1 \tanh(a)$ rad in orientation. Damped least squares ($\lambda = 0.1$) converts it into a joint displacement clipped to 0.2 rad per joint, and the arm's PD controller tracks the result. The other twelve actions are hand joint targets with scale 0.2 rad ($k_p = 50$, $k_d = 5$). An episode lasts 6 s (300 steps).

Observations. The actor and the critic observe the same 84 dimensions: the arm joint positions and velocities ($7 + 7$), the hand joint positions and velocities ($12 + 12$), the cube position (3, $n = 0.01$) and quaternion (4, $n = 0.01$) in the arm's base frame, the goal (3, $n = 0.01$), the grasp point, taken as the centroid of the five fingertips (3), the vector from each fingertip to the cube (15), and the previous action (18).

Terminations and randomization. An episode ends at the time limit or when the cube falls more than 4 cm below the table top. At reset the cube is offset by up to 6 cm in x and 12 cm in y with a uniform yaw, and the arm and hand joints are offset by up to 0.05 and 0.02 rad. There is no randomization of the physical parameters.

Table 8: **Dexterous lifting rewards.** d is the distance from the grasp point to the cube, h the cube's height above its resting height, d_g the cube's distance to the goal, and $G = \mathbb{1}[h > 0.02]\,\mathbb{1}[d < 0.06]$ gates the goal terms on a lifted, held cube. The two scheduled weights step up at iterations 1200 and 1800.

term	weight	form
reach	5.0	$\min(1 - \tanh(d/0.3),\ 0.7)$
reach, fine	3.0	$1 - \tanh(d/0.05)$
fingertips to cube	2.0	$\text{mean}_{\text{tips}} \exp\big(-\lVert p_{\text{tip}} - p_{\text{cube}} \rVert^2 / 0.06^2\big)$
lift height	15.0	$\text{clip}(h, 0, 0.3)$
lifted	5.0	G
goal tracking	25.0	$G\,(1 - \tanh(d_g/0.3))$
goal tracking, fine	10.0	$G\,(1 - \tanh(\max(d_g - 0.02, 0)/0.05))$
settle at goal	-6.0	$(1 - \tanh(d_g/0.05))\,\lVert \dot{q}_{\text{arm}} \rVert^2$
action rate	$-0.01 \to -0.04 \to -0.1$	$\lVert a_t - a_{t-1} \rVert^2$
arm joint velocity	$-0.01 \to -0.2 \to -0.5$	$\sum_j \max(\lvert \dot{q}_j \rvert - 0.5, 0)^2$
arm, hand joint limits	-1.0 each	distance outside the soft joint limits, summed

D.6 In-hand reorientation



Figure 9: **In-hand reorientation.** A dexterous hand reorients a cube in place; the four frames show the cube passing through distinct orientations within one episode.

Task. A right Wuji hand with 20 actuated joints (4 per finger) is mounted palm-up on a fixed base. It turns a free 5.4 cm cube (120 g) to a sequence of target orientations without dropping it. Each target is drawn uniformly on $SO(3)$ in a frame fixed to the wrist. A target counts as reached when the geodesic orientation error e stays below 0.2 rad for five consecutive steps. Twenty steps later a new target is drawn, and the cube is left where it is, so one episode is a stream of targets. After

the first 3 s of an episode, the cube receives a velocity kick in a uniformly random direction every $\mathrm{Unif}[0.6, 1.8]$ s. The kick speed is drawn from $\mathrm{Unif}[0.05, v_{\max}]$ m/s with $v_{\max} = 0.05 + 0.1\,\rho\,\kappa$. The ramp ρ grows linearly from 0 at iteration 250 to 1 at iteration 4000. The factor $\kappa \in [0.05, 1]$ starts at 0.05; it grows with episodes that last 800 steps and shrinks with episodes that end early.

Actions. The 20 actions are joint offsets from a grasp pose. The raw target $\mathrm{clip}(q_0 + 0.5\,\mathrm{clip}(a, -1, 1))$ is clipped to 90% of each joint's range and passed through a first-order filter, $q^\star_t = 0.5\,q^{\mathrm{raw}}_t + 0.5\,q^\star_{t-1}$. The hand holds its default pose for the first 0.4 s of an episode. An implicit PD controller with the manufacturer's per-joint gains (k_p from 0.18 to 0.69 N m/rad) tracks the target. An episode lasts 50 s (1000 steps).

Observations. Each actor frame contains the joint positions normalized to $[-1, 1]$ by the joint limits (20, $n = 0.06$), the error between the joint positions and the current target (20), the cube position in the wrist frame (3, $n = 0.008$ m), the rotation from the target to the cube as a 6D representation (6, Gaussian noise with standard deviation 0.05), and the previous action (20). On 2% of steps the cube position and orientation are replaced by random values. The actor stacks the last three frames (207 dimensions). The critic observes the same three frames without noise, and adds three frames of the joint positions and velocities and the fingertip positions, the cube's linear and angular velocity, the palm orientation, the progress through the current target, a drop counter, the current disturbance, and the randomized physical parameters (403 dimensions).

Terminations and randomization. An episode ends at the time limit or when the cube leaves a box around the palm and fingers for long enough. A counter rises by 1 on each step outside the box and falls by 0.5 on each step inside it, and the episode ends when it reaches 10. At each reset, gravity is tilted by a pitch drawn from $\mathrm{Unif}[-0.4, 0.1]$ rad, which stands in for the wrist's pitch. Three of the four joints of each finger start at an offset from $\mathrm{Unif}[-0.3, -0.1]$ rad. The cube starts within 1 cm of a fixed point above the palm, in a uniformly random orientation. At startup, each environment samples the hand friction from $\mathrm{Unif}[0.5, 1.0]$, scales k_p by $\mathrm{Unif}[0.75, 1.5]$, k_d by $\mathrm{Unif}[0.5, 2]$ and the armature by a log-uniform factor in $[0.75, 1.3]$, samples a joint damping from $\mathrm{Unif}[0, 0.002]$ and a target bias of up to 0.01 rad, and scales the hand's link masses by $\mathrm{Unif}[0.4, 1.5]$ and the cube's mass by $\mathrm{Unif}[0.4, 1.6]$.

Table 9: **In-hand reorientation rewards.** e is the geodesic orientation error to the current target and k the number of steps the cube has stayed on target since it was reached. The action-smoothness term is the action-rate penalty plotted in Figure 3.

term	weight	form
orientation alignment	15	1 if $e \le 0.2$, else $\max(0,\ 1 - (e - 0.2)/\pi)$
hold	11.4	k while $e < 0.2$ within the 20 steps after reaching a target
fingertip contact	0.5	$\mathbb{1}$[a distal phalanx touches the cube and the palm does not]
hand pose	-0.2	$\sum_j (q_j - q_{0,j})^2$
action smoothness	-1.0	$\lVert a_t - a_{t-1}\rVert^2 + \lVert a_t - 2a_{t-1} + a_{t-2}\rVert^2$
torque	-24	$\sum_j (\tau_j/\tau_{\max,j})^2$
fingertip slip	-0.3	$\sum_{\text{tips in contact}} \lVert v_{\text{tip}} - v_{\text{cube}}\rVert$
cube escape	-500	$0.4\,c \cdot \mathbb{1}$[cube outside the box], c the drop counter
finger collision	-1	number of contacts between fingers

D.7 BIMANUAL PIANO



Figure 10: **Bimanual piano.** Two Wuji hands play Chopin's Prelude Op. 28 No. 7, keys that are pressed are drawn in green.

Task. Two Wuji hands with 20 finger joints each play a piece on an 88-key keyboard, in the style of RoboPianist (Zakka et al., 2023). Each hand sits on a floating wrist that is driven in all six axes, and there is no arm. The piece is Chopin's Prelude Op. 28 No. 7. It has 167 notes over 25 distinct keys, with at most nine keys down at once. With a 1 s silent lead-in, it lasts 761 control steps (38 s). A key counts as pressed at 87% of its travel. The command is the piano roll: at each step, the set of keys that should be down.

Actions. Each hand has 26 actions, 52 in total. Six set the wrist pose relative to a rest pose above the keys, with ranges of 0.04 m forward, 0.03 m up, 0.16 or 0.25 m along the keyboard (right or left hand), and 0.15, 0.15 and 0.30 rad of rotation. A PD controller on the free base tracks the wrist pose. The other twenty set finger joint targets $q_{\text{rest}} + 0.5\,a$, tracked by a PD controller ($k_p = 50$, $k_d = 5$).

Observations. The actor and the critic observe the same 1212 dimensions, without noise or history: the piano roll for the current step and the next ten (11×88), the travel of every key (88), the previous action (52), and, for each hand, the finger joint positions relative to rest (20) and velocities scaled by 0.1 (20), the wrist pose relative to rest (6), and the wrist's linear and angular velocity scaled by 0.1 (6).

Terminations and randomization. An episode ends only when the piece ends. There is no randomization and no curriculum.

Table 10: **Bimanual piano rewards.** $g(e; b, m)$ equals 1 for $e \le b$ and decays as a Gaussian in $e - b$ to 0.1 at $e = b + m$. $u_k \in [0, 1]$ is the travel of key k and $\mathcal{K}_t$ the set of keys due at step t. The fingertip term uses the keys due within the next five steps and the fingertip nearest to each.

term	weight	form
key press	1	$\frac{1}{2}\,\text{mean}_{k\in\mathcal{K}_t}\, g(1-u_k; 0.05, 1) + \frac{1}{2}\,\mathbb{1}[\text{no key} \notin \mathcal{K}_t \text{ pressed}]$
fingertip to key	1	$\text{mean}_k\, g(d_k; 0.01, 0.1)$, d_k the distance from the nearest fingertip to key k
energy	-5×10^{-4}	$\sum_j \lvert\tau_j \dot{q}_j\rvert$ over fingers and wrists
rest pose	-0.02	$\lVert q - q_{\text{rest}}\rVert^2$ plus the squared wrist displacement and rotation
action rate	-10^{-3}	$\lVert a_t - a_{t-1}\rVert^2$

D.8 VISION-BASED PICK-UP

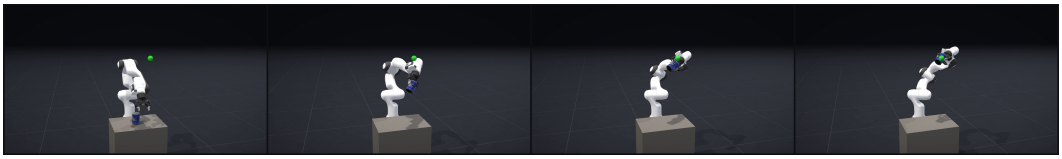

Figure 11: **Vision-based pick-up.** From an RGB observation, an arm and gripper pick a YCB object off the pedestal and lift it to the goal marker.

Task. A 7-DoF Franka Panda arm with a Robotiq 2F-85 parallel gripper picks up an object from a table and carries it to a goal, observing the scene through a wrist-mounted RGB camera. The actor does not observe the object's pose. Each environment holds one of eight YCB objects (Calli et al., 2015) for the whole run, in equal numbers: the master chef can, tomato soup can, tuna fish can, pudding box, gelatin box, potted meat can, foam brick and apple. The goal, the goal curriculum and the penalty schedule are those of dexterous lifting (Appendix D.5). Images are rendered with the Madrona batch renderer.

Actions. Six actions command the end-effector displacement exactly as in dexterous lifting. The seventh is binary: a positive value closes the gripper and a non-positive value opens it. An episode lasts 6 s (300 steps).

Observations. The actor observes the arm joint positions and velocities ($7 + 7$), the gripper joint positions (6), the goal (3, $n = 0.01$), the grasp point between the finger pads (3) and the previous action (7), for 33 dimensions, together with a 72×128 RGB image from the wrist camera (80° field of view). The critic observes the same image and adds the object's position and quaternion (40

dimensions). The actor and the critic each encode the image with their own two-layer convolutional network (5×5 kernel, stride 2, 16 channels, then 3×3, stride 2, 32 channels, ELU), followed by a spatial soft-argmax that returns 32 image keypoints (64 features). The features are concatenated with the vector observation and passed to the MLP.

Terminations and randomization. Terminations and reset randomization are those of dexterous lifting, without the hand joints. There is no randomization of mass, friction, lighting, texture or camera pose.

Table 11: **Vision-based pick-up rewards.** Notation as in Table 8, with d measured from the midpoint of the finger pads and h from each object's own resting height. The two reach terms, the lift-height and lifted terms, the two goal-tracking terms and the settle penalty have the weights and forms of Table 8 and are not repeated. The fingertip term is replaced by a grasp bonus, where $c \in [0, 1]$ is the gripper closure.

term	weight	form
grasp	2.0	$\mathbb{1}[d < 0.03]\, c$
action rate	$-0.01 \to -0.04 \to -0.1$	$\lVert a_t - a_{t-1} \rVert^2$
arm joint velocity	$-0.01 \to -0.2 \to -0.5$	$\sum_j \max(\lvert \dot{q}_j \rvert - 0.5, 0)^2$
arm joint limits	-1.0	distance outside the soft joint limits, summed

D.9 QUADRUPED HANDSTAND

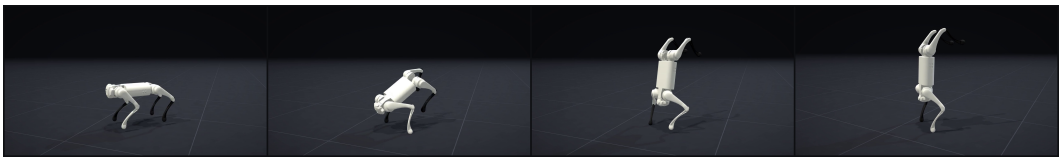

Figure 12: **Quadruped handstand.** A quadruped rears from a four-legged stance onto its front legs and balances in an inverted handstand.

Task. This task ports the Go1 handstand of MuJoCo Playground (Zakka et al., 2025). A Unitree Go1 quadruped with 12 actuated joints starts standing on flat ground, rises onto its front legs and balances with its body's forward axis pointing down and the torso at 0.55 m.

Actions. The actions are clipped to $[-1, 1]$ and accumulate into the joint targets, $q_t^\star = \mathrm{clip}(q_{t-1}^\star + 0.3\, a_t)$, with the clip at 90% of each joint's range and $q^\star$ reset to q_0 at the start of an episode. An implicit PD controller tracks the targets. The action is therefore already an increment, and SFPO's carry acts on the increments. Under exploration noise the joint target then integrates an AR(1) process instead of white noise. With v_θ held fixed and at equal per-step noise, its increments are smoother than PPO's, but its long-run variance grows $1/(1-\beta)^2 = 4$ times as fast, until the joint-range clip binds. We use the same retention $k\Delta t = 0.5$ here as on the other tasks. An episode lasts 10 s (500 steps).

Observations. The actor observes the base linear (3, $n = 0.1$) and angular (3, $n = 0.2$) velocity, the projected gravity (3, $n = 0.05$), the joint positions relative to q_0 (12, $n = 0.01$) and velocities (12, $n = 1.5$), and the previous action (12), for 45 dimensions. The critic adds the noise-free base velocities in the body and world frames, the absolute joint positions, the joint velocities and torques, and the base position (93 dimensions).

Terminations and randomization. An episode ends at the time limit, when the body's vertical axis points more than 0.25 below the horizontal (the z component of the body z-axis falls below -0.25), or when a front hip, thigh or calf comes within 5 cm of the ground. At reset the base is placed at a horizontal offset of up to 0.5 m with a uniform yaw, and each component of the base velocity is drawn from $\mathrm{Unif}[-0.5, 0.5]$. Unlike the Playground task, which randomizes the floor friction, joint friction loss, armature, link masses and torso center of mass, the dynamics are not randomized.

Table 12: **Quadruped handstand rewards.** z is the torso height, e_x the body's forward axis in the world frame, and the pose term runs over the six rear-leg joints.

term	weight	form
height	1.0	$\exp\big(-(0.55-\min(z,0.55))\big)$
orientation	1.0	$\big(\frac{1}{2}\,e_x\cdot(0,0,-1)+\frac{1}{2}\big)^2$
rear-foot contact	-0.1	$\mathbb{1}[\text{a rear foot below } 3\,\text{cm}]$
joint limits	-0.5	distance outside 90% of each joint's range, summed
rear-leg pose	-0.1	$\sum_{j\in\text{rear}}(q_j-q_{0,j})^2$