MODEL-BASED FLOW POLICY OPTIMIZATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative control policies (GCPs), such as diffusion and flow matching policies can represent expressive, multi-modal action distributions, but training them with reinforcement learning is difficult: their likelihoods are expensive to compute, and backpropagating through the sampling chain is costly and unstable. Existing methods sidestep this with model-free, on-policy RL, which is prohibitively sample-inefficient, especially from pixels. We observe that a learned world model resolves this tension. Likelihood-free objectives such as advantage-weighted flow matching require abundant on-policy data, which a latent world model can generate cheaply in imagination. We propose MB-FPO, which trains flow policies by regressing onto imagined actions weighted by their advantages, with no likelihoods and no backpropagation through sampling. With temporally correlated exploration noise, MB-FPO is substantially more sample efficient than prior flow policy optimization methods on the DeepMind Control and Meta-World benchmarks, outperforms model-free baselines on most pixel-based tasks, and matches or exceeds a model-based Gaussian policy trained on the same world.

1 INTRODUCTION

Flow policies (Lipman et al., 2023) have emerged as a powerful class of policies for continuous control, particularly in robot learning. Their expressiveness lets them fit multimodal demonstrations, and the iterative nature of flow matching helps them produce the fine-grained actions required for precise manipulation. Flow policies build on continuous normalizing flows (CNFs) (McAllister et al., 2025), which represent a complex distribution by transporting samples from a Gaussian through a learned velocity field. However, while flow policies excel in imitation learning, training them with reinforcement learning (RL) has proven difficult. Computing the likelihood of an action requires integrating the velocity-field divergence along the sampling trajectory, which is prohibitively expensive inside an RL loop. The alternative, backpropagating through the multi-step sampling process, is costly in memory and compute and often unstable (Yi et al., 2026; Gao et al., 2026).

Recent model-free methods sidestep the likelihood problem. Flow Policy Optimization (FPO) (McAllister et al., 2025) replaces the likelihood ratio in proximal policy optimization (PPO) (Schulman et al., 2017) with a surrogate built from conditional flow matching losses. FPO++ (Yi et al., 2026) stabilizes this surrogate with per-sample ratios and an asymmetric trust region, training flow policies for legged locomotion and humanoid control. However, these methods rely on massively parallel simulation to make on-policy sampling viable. Furthermore, they depend on clipped importance ratios and multi-epoch updates to maximize gradient steps per rollout, a demand compounded by the complexity of expressive flow policies. Off-policy methods for diffusion and flow policies (Psenka et al., 2024; Wang et al., 2024) reuse data more efficiently, but typically require differentiating through the sampling chain, making them computationally expensive and data inefficient (McAllister et al., 2025). When interaction is expensive, as in real-world robot learning (Dulac-Arnold et al., 2021), the sample inefficiency of such methods becomes prohibitive.

Model-based reinforcement learning offers a different route. A learned world model can generate many imagined rollouts cheaply, so on-policy data becomes abundant rather than scarce. Importance ratios are then no longer needed to reuse stale data, and a much simpler RL objective suffices. We train flow policies with advantage-weighted flow matching on trajectories imagined in a world model's latent space: the policy is regressed onto its sampled actions, and each action is weighted by its advantage. Because the objective is a weighted flow-matching loss, it needs neither action

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.

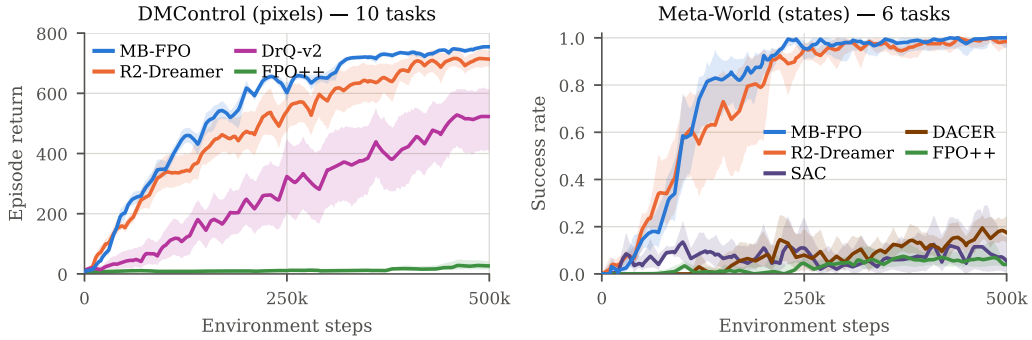


Figure 1: Mean performance over DMC and Meta-World environments. **Left:** mean episode return over ten pixel-based dm_control tasks, against the Gaussian actor of R2-Dreamer (Morihira et al., 2026), DrQ-v2 (Yarats et al., 2022) and FPO++ (Yi et al., 2026). **Right:** mean success rate over six state-based Meta-World tasks, against R2-Dreamer, SAC (Haarnoja et al., 2018), DACER (Wang et al., 2024) and FPO++. Line: interquartile mean (IQM) over all task–seed runs at each step (30 on DMC, 18 on Meta-World); band: 95% confidence interval (Agarwal et al., 2021).

likelihoods nor backpropagation through the sampling chain, while retaining the full expressiveness of the flow policy. Since the flow's own sampling noise can contract during training, we pair the objective with temporally correlated action noise for exploration (Eberhard et al., 2023).

Contributions. We present MB-FPO, a model-based reinforcement learning algorithm for training flow policies. MB-FPO learns a latent world model to model the environment and uses it to generate on-policy trajectories and trains the flow policy on the imagined trajectories with advantage-weighted flow matching without action likelihoods or backpropagation through the sampling chain, and using temporally correlated noise for exploration. On pixel-based DeepMind Control Suite tasks and state-based Meta-World manipulation tasks, MB-FPO is substantially more sample-efficient than the model-free methods like FPO++, DrQ-v2 and DACER and matches or improves on our model-based baseline R2-Dreamer with a Gaussian policy (Figure 1).

2 Preliminaries

World model. In MB-FPO we learn a latent dynamics model to generate on policy samples for the policy optimization. We use the latent dynamics model introduced in R2-Dreamer (Morihira et al., 2026), which builds on the recurrent state space model (RSSM) presented by Hafner et al. (2019) to model POMDPs and use them in an MPC scheme. Dreamer (Hafner et al., 2020) used the RSSM as a learned simulator to learn a policy π_θ and a value function V_ψ. The RSSM is a latent dynamics model that models the underlying state of the POMDP s_t using a recurrent neural network's hidden state h_t and a latent state representation z_t. To generate on-policy rollouts, the RSSM learns a transition model $p_\phi(\hat{z}_t \mid h_t)$ and predictors of the reward $p_\phi(r_t \mid h_t, z_t)$ and continuation $p_\phi(c_t \mid h_t, z_t)$. R2-Dreamer keeps that structure but replaces pixel reconstruction as the grounding signal: the latent is instead tied to the observation by a Barlow Twins redundancy-reduction loss (Zbontar et al., 2021) between a learned projection of the RSSM state and the encoder embedding of the same frame. The model consists of

- RNN $h_t = f_\phi(h_{t-1}, z_{t-1}, a_{t-1})$
- Inference Network $z_t \sim q_\phi(z_t|h_t, o_t)$
- Transition Network $\hat{z}_t \sim p_\phi(\hat{z}_t|h_t)$
- Projector $u_t = g_\phi(h_t, z_t)$.

The model is trained with the objective

$$\mathcal{L}(\phi) \doteq \mathbb{E}_{q_\phi}\left[\sum_{t=1}^{T}\left(\beta_{\text{pred}}\mathcal{L}_{\text{pred}} + \beta_{\text{dyn}}\mathcal{L}_{\text{dyn}} + \beta_{\text{rep}}\mathcal{L}_{\text{rep}} + \beta_{\text{bt}}\mathcal{L}_{\text{bt}}\right)\right],$$

$$\text{where, } \mathcal{L}_{\text{pred}} = -\ln p_\phi(r_t \mid z_t, h_t) - \ln p_\phi(c_t \mid z_t, h_t)$$

$$\mathcal{L}_{\text{dyn}} = \mathrm{KL}\left[\mathrm{sg}(q_\phi(z_t \mid h_t, o_t)) \parallel p_\phi(z_t \mid h_t)\right],\ \mathcal{L}_{\text{rep}} = \mathrm{KL}\left[q_\phi(z_t \mid h_t, o_t) \parallel \mathrm{sg}(p_\phi(z_t \mid h_t))\right],$$

$$\mathcal{L}_{\text{bt}} = \textstyle\sum_i \left(1 - C_{ii}\right)^2 + \lambda_{\text{bt}} \sum_{i \neq j} C_{ij}^2, \quad C_{ij} = \frac{1}{B}\sum_b \tilde{u}_i^{(b)}\, \tilde{e}_j^{(b)}.$$

Here $e_t = \mathrm{enc}(o_t)$ is the encoder embedding, $\tilde{u}$ and $\tilde{e}$ are u and $\mathrm{sg}(e)$ standardized to zero mean and unit variance over the batch of B latent-frame pairs, and C is their cross-correlation matrix; λ_{bt} is a weighting constant. The diagonal term aligns each projected latent dimension with its embedding counterpart and the off-diagonal term decorrelates dimensions, preventing the collapse the removed decoder would otherwise guard against; the stop gradient on e keeps the encoder from learning a trivial embedding. $\mathcal{L}_{\text{dyn}}$ optimizes the transition model applying the stop gradient operator (sg) to q_ϕ, and $\mathcal{L}_{\text{rep}}$ regularizes the inference network applying sg to p_ϕ. Dreamer uses the latent dynamics model to generate model rollouts to train a policy and a value function, where the value function is trained using TD(λ) returns as targets

$$\min_\psi \mathbb{E}_{z_{t'} \sim p_\phi, a_{t'} \sim \pi_\theta}\left[\sum_{t'=t}^{t+H} \frac{1}{2}\left\|V_\psi(z_{t'}) - R(z_{t'})\right\|^2\right], \text{ where} \tag{1}$$

$$R(z_t) = r_t + \gamma(1 - d_t)\left((1-\lambda)V_\psi(z_{t+1}) + \lambda R(z_{t+1})\right),\ R(z_{t+H}) = V_\psi(z_{t+H}),$$

and H is the horizon of the model trajectories. The policy π is trained with an objective using TD(λ) returns

$$\max_{\pi_\theta}\ \mathbb{E}_{a_{t'} \sim \pi, z_{t'} \sim p_\phi}\Big[R(z_t) \mid z_t\Big]. \tag{2}$$

Flow matching policies. In some use cases such as robotics, Gaussian policies are not expressive enough to represent the action distribution that solves a task (Chi et al., 2023). A flow-matching policy represents the action distribution as a Gaussian transported through a learned velocity field: an action is generated by integrating $\dot{\boldsymbol{x}} = v_\theta(\boldsymbol{x}, \tau, \boldsymbol{z})$ from $\boldsymbol{x}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ over $\tau \in [0, 1]$ with N Euler steps, followed by a squashing to the action bounds. Training uses the conditional flow-matching (CFM) loss on the linear (optimal-transport) path (Lipman et al., 2023): for a state–action pair $(\boldsymbol{z}, \boldsymbol{a})$ with endpoint $\boldsymbol{x}_1 = \boldsymbol{a}$ and draws $\tau \sim \mathrm{U}[0, 1]$, $\boldsymbol{x}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, the path point is $\boldsymbol{x}_\tau = (1-\tau)\boldsymbol{x}_0 + \tau\boldsymbol{x}_1$ and

$$\mathcal{L}_{\text{CFM}}(\theta; \boldsymbol{z}, \boldsymbol{a}, \tau, \boldsymbol{x}_0) = \left\|v_\theta(\boldsymbol{x}_\tau,\ \tau,\ \boldsymbol{z}) - (\boldsymbol{x}_1 - \boldsymbol{x}_0)\right\|^2. \tag{3}$$

In this work, we follow McAllister et al. (2025); Yi et al. (2026); Gao et al. (2026); Zhang et al. (2025) and use flow-matching policies in the reinforcement learning setting, where we train the policy to maximize expected returns by regressing onto its own imagined actions weighted by their advantages.

3 METHOD

Overview. MB-FPO trains a flow-matching actor entirely inside a learned world model (Figure 2). We use R2-Dreamer's world model (Section 2), due to its strong performance without a reconstruction objective. The actor is trained with advantage-weighted conditional flow matching (CFM) on rollouts imagined by the world model, and the critic is trained on λ-returns as in Dreamer (Hafner et al., 2025). We describe the algorithm in detail below and summarize it in Algorithm 1.

Model Rollouts. In each policy optimization step we sample latent states $\boldsymbol{z}_t = (h_t, z_t)$ from the replay buffer, and we roll the flow policy out in the world model's transition for H steps, sampling $\boldsymbol{a}_t \sim \pi_\theta(\cdot \mid \boldsymbol{z}_t)$ and $\boldsymbol{z}_{t+1} \sim p_\phi(\cdot \mid \boldsymbol{z}_t, \boldsymbol{a}_t)$, and read predicted rewards and continuation probabilities off the reward and continuation heads. Unlike the model-free case, generating a rollout costs one world-model unroll, so for every gradient step on the actor we can afford to sample a fresh rollout. Thus, this simplifies the on-policy policy optimization drastically as we do not need to consider a trust region. Advantages are estimated on the imagined rollout as in Dreamer: the critic V_ψ is trained on λ-returns (Eq. 1), and the advantage of an imagined step is its λ-return minus the critic's value, normalized to unit scale to give $\hat{A}_t$.

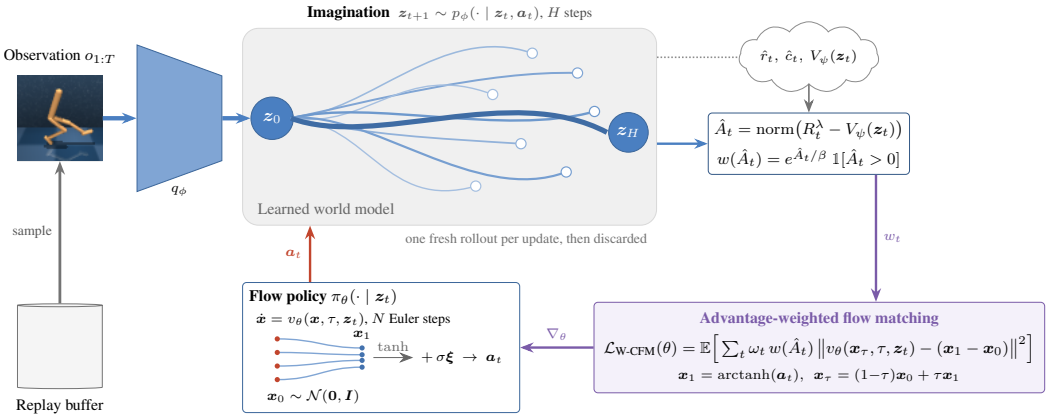


Figure 2: Overview of MB-FPO. Observations sampled from the replay buffer are encoded into posterior latent states $\boldsymbol{z}_0$, from which the world model imagines H-step rollouts under the flow policy. Predicted rewards, continuations and the critic give a normalized advantage $\hat{A}_t$ for every imagined step, weighted by $w(\hat{A}_t) = e^{\hat{A}_t/\beta}\,\mathbb{1}[\hat{A}_t > 0]$, so steps with non-positive advantage get zero weight. The flow policy is then regressed onto its own imagined actions with the advantage-weighted CFM loss, and the rollout is discarded.

Flow Policy Optimization. We adopt the latent-action parameterization of FMER (Gao et al., 2026), which trains the flow in an unbounded space and then projects the generated sample to the action space. The actor is a velocity network $v_\theta(\boldsymbol{x}, \tau, \boldsymbol{z})$ conditioned on the latent state $\boldsymbol{z}$. We draw $\boldsymbol{x}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, take N Euler steps of $\dot{\boldsymbol{x}} = v_\theta(\boldsymbol{x}, \tau, \boldsymbol{z})$ to obtain $\boldsymbol{x}_1$, and execute $\boldsymbol{a} = \tanh(\boldsymbol{x}_1)$. Training therefore happens in the latent space: we use the CFM loss of Eq. 3 with endpoint $\boldsymbol{x}_1 = \mathrm{arctanh}(\boldsymbol{a})$ instead of $\boldsymbol{x}_1 = \boldsymbol{a}$. The actor minimizes the CFM loss weighted by the advantage of the target action

$$\mathcal{L}_{\text{W-CFM}}(\theta) = \mathbb{E}\Big[\textstyle\sum_t \omega_t\, w(\hat{A}_t)\, \hat{\mathcal{L}_{\text{CFM}}}(\theta; \boldsymbol{z}_t, \boldsymbol{a}_t)\Big], \qquad w(\hat{A}) = \exp\big(\hat{A}/\beta\big)\,\mathbb{1}[\hat{A} > 0], \tag{4}$$

where $\hat{\mathcal{L}_{\text{CFM}}}$ averages Eq. 3 (with $\boldsymbol{x}_1 = \mathrm{arctanh}(\boldsymbol{a})$) over n_{mc} draws of $(\tau, \boldsymbol{x}_0)$, $\omega_t = \prod_{i \le t} \gamma \hat{c}_i$ is the cumulative discounted continuation probability along the rollout (as in Dreamer), and β is a temperature. As in Dreamer's policy objective (Eq. 2), the actor learns from imagined rollouts; where REINFORCE needs the action log-likelihood, which a flow does not provide cheaply, the CFM loss serves as its stand-in. Eq. 4 is advantage-weighted regression (AWR) (Peng et al., 2019) with the CFM loss in place of the negative log-likelihood, the form FMER (Gao et al., 2026) derives for off-policy flow training; Appendix B gives the derivation. The actor regresses toward actions with positive advantage, with weight growing exponentially in the advantage; actions with non-positive advantage carry no weight. The weight is non-negative everywhere, so the objective never maximizes the CFM loss, which is unbounded above. This is where it departs from Flow Policy Optimization (McAllister et al., 2025). FPO's surrogate uses the ratio $\hat{r} = \exp(\hat{\mathcal{L}_{\text{CFM}}}^{\text{old}} - \hat{\mathcal{L}_{\text{CFM}}})$, which equals one at the start of each update, where its gradient is $-\hat{A}\,\nabla_\theta \hat{\mathcal{L}_{\text{CFM}}}$: a signed weight that raises the CFM loss of negative-advantage actions and must be kept in check by ratio clipping and a trust region (Yi et al., 2026). Because the world model supplies a fresh rollout for every gradient step, we take a single step per rollout and need no ratio, clipping thresholds or stored old policy, and we replace the signed $\hat{A}$ by the non-negative weight $w(\hat{A})$.

Exploration. The base draw $\boldsymbol{x}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ is the flow's only intrinsic source of stochasticity, and it is not a dependable one as nothing in the objective stops the learned transport from contracting, and a narrow π_θ starves the update of the very proposals it regresses toward. We therefore add explicit, temporally correlated exploration noise in action space,

$$\boldsymbol{a} = \mathrm{clip}\big(\tanh(\boldsymbol{x}_1) + \sigma\boldsymbol{\xi},\ -1,\ 1\big). \tag{5}$$

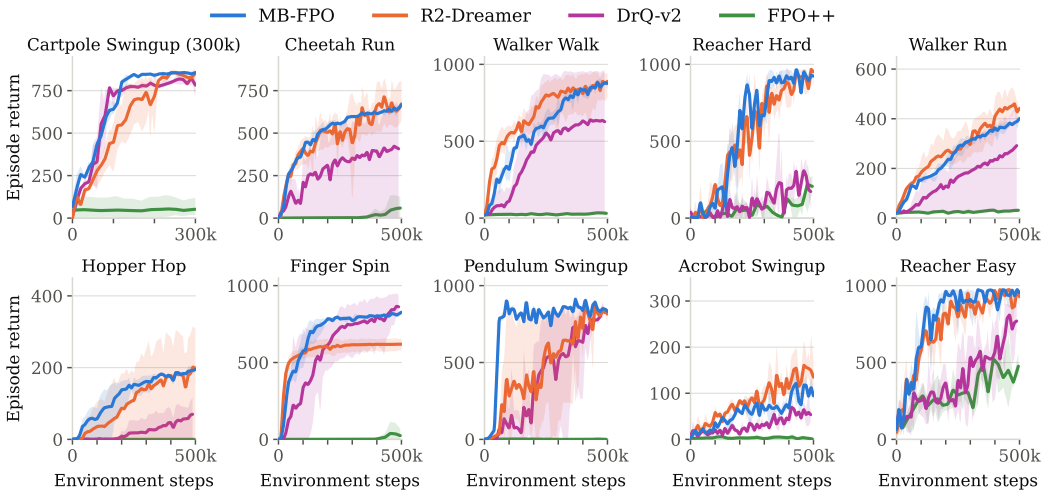


Figure 3: Learning curves on ten pixel-based dm_control tasks, 3 seeds each. MB-FPO and the Gaussian R2-Dreamer actor share an identical world model and differ only in the actor objective; FPO++ and DrQ-v2 are model-free.

When acting in the environment $\boldsymbol{\xi}$ is temporally correlated, as constructed below; in imagination it is white, $\boldsymbol{\xi} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$, in every experiment. Since $\boldsymbol{a}$ in Eq. 5 is the action in the CFM target $\operatorname{arctanh}(\boldsymbol{a})$, the actor regresses toward the perturbed action. With no entropy term in the objective, this perturbation is our only exploration mechanism. Following Eberhard et al. (2023), we use temporally correlated noise rather than i.i.d. draws. Viewing a noise sequence over an episode as a sum of sine waves, the weight of low-frequency waves, which make nearby timesteps similar, relative to high-frequency ones sets how correlated the noise is over time. We construct the noise $\boldsymbol{\xi}$ for each action dimension

$$\boldsymbol{\xi} \;=\; \mathcal{F}^{-1}\Big[\, f^{-\alpha/2}\,(\boldsymbol{u} + i\,\boldsymbol{v})\,\Big], \qquad \boldsymbol{u}, \boldsymbol{v} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}), \tag{6}$$

where f is the frequency of each wave, $\boldsymbol{u} + i\,\boldsymbol{v}$ gives each wave a random size and starting point, $f^{-\alpha/2}$ makes slower waves larger, and the inverse Fourier transform $\mathcal{F}^{-1}$ sums the waves into one value per timestep. $\boldsymbol{\xi}$ is rescaled to unit variance so that σ in Eq. 5 alone sets its magnitude. With $\alpha = 0$ all waves have equal weight and the noise is i.i.d.; it jitters in place because consecutive perturbations cancel. With $\alpha = 2$ slow waves dominate, giving Brownian noise similar to DDPG's Ornstein-Uhlenbeck process, which drifts in one direction for too long. We use $\alpha = 1$ (pink noise), which Eberhard et al. (2023) found to be a better exploration default than either extreme.

4 Experiments

In our experiments, we evaluate our method in comparison with the baselines on ten pixel-based DeepMind Control Suite tasks (Tassa et al., 2018) as well as six state-based Meta-World (Yu et al., 2019) manipulation tasks.

Baselines. For the visual RL tasks we use FPO++ (Yi et al., 2026), a model-free on-policy baseline. In addition, we compare to DrQ-v2 (Yarats et al., 2022), an off-policy model-free baseline for visual RL; we run the authors' implementation with its default configuration. We also compare with the Gaussian REINFORCE actor of R2-Dreamer. For the Meta-World tasks we compare our method to SAC (Haarnoja et al., 2018) with the hyperparameters TD-MPC2 tuned for it (Hansen et al., 2024), DACER (Wang et al., 2024), an off-policy diffusion actor-critic that differentiates through its sampling chain, run with the authors' code and default hyperparameters, as well as FPO++ and the Gaussian R2-Dreamer policy. We do not compare with FMER (Gao et al., 2026), which optimizes the same objective off-policy, because no implementation is publicly available.

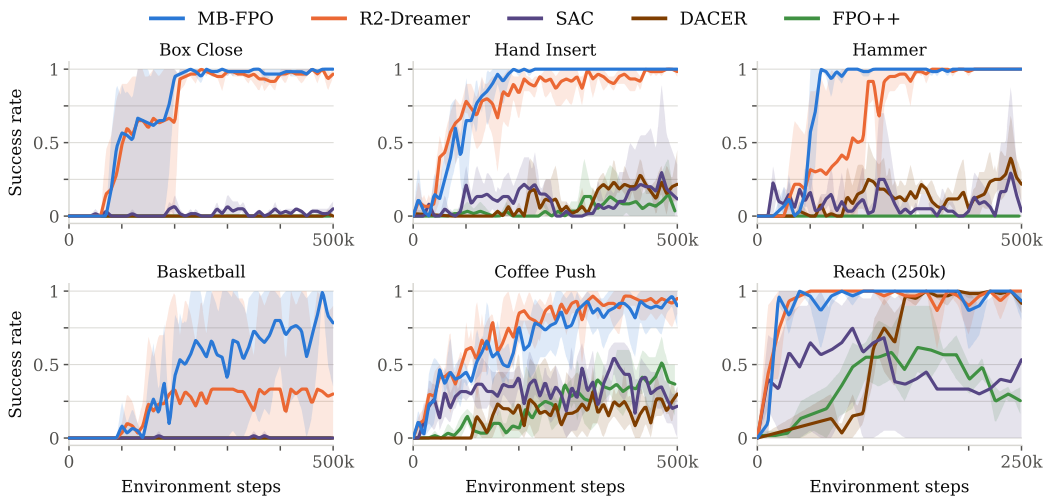


Figure 4: Learning curves on six state-based Meta-World tasks. MB-FPO and the Gaussian R2-Dreamer actor share the same world-model architecture and hyperparameters and differ only in the actor objective; SAC, DACER and FPO++ are model-free Gaussian, diffusion and flow policy baselines.

4.1 DMC

From the results in Figure 3 we see that FPO++ is not effective in the sample regime we are considering, which is to be expected from an on policy approach. MB-FPO is also ahead of DrQ-v2 on most tasks, which is also consistent with the R2-Dreamer results, and shows the usefulness of leveraging imagination in a world model and the representation learning benefits of world modeling. Against the Gaussian actor in R2-Dreamer, we see that our approach is mostly more stable than the Gaussian approach and is more sample efficient on tasks with sparse rewards due to the use of the more expressive nature of flow, where we iterate multiple times to infer the best action as well as the use of time correlated noise leading to better exploration.

4.2 Meta-World

We evaluate our method in comparison with the baselines on six state-based Meta-World tasks in Figure 4. On four of the six tasks both MB-FPO and R2-Dreamer Gaussian actor have identical performance. On hand-insert and hammer MB-FPO has stronger sample efficiency and is able to solve the task faster than the Gaussian baseline. Both model-free baselines fall short within this environment interaction budget. FPO++ and SAC are similarly limited, they stay near zero on box-close, hammer and basketball. This showcases that the combination of using a world model for sampling on policy rollouts for a flow policy and using noise correlated noise leads to stable and strong exploration.

4.3 Multimodality and policy analysis

Generative control policies (GCPs) have been hypothesized to be useful in control because they can represent multimodal action distributions, which are common in continuous control tasks (Chi et al., 2023). This hypothesis has, however, come under scrutiny (Pan et al., 2026) recently. We investigate, this hypothesis in the context of reinforcement learning for flow policies. Several of the environments we consider admit multiple optimal actions: in pendulum_swingup the pendulum can be swung up from either side, and in cartpole_swingup the pole can be swung up from either side of the cart. At symmetric states in these tasks, the optimal action distribution is therefore bimodal. Flow policies are expressive enough to represent such distributions, but nothing in the loss forces them to do so. Whether the learned policy is multimodal depends on the task and on the critic: when the critic has a single maximum over actions, the policy tends to concentrate a single mode there;

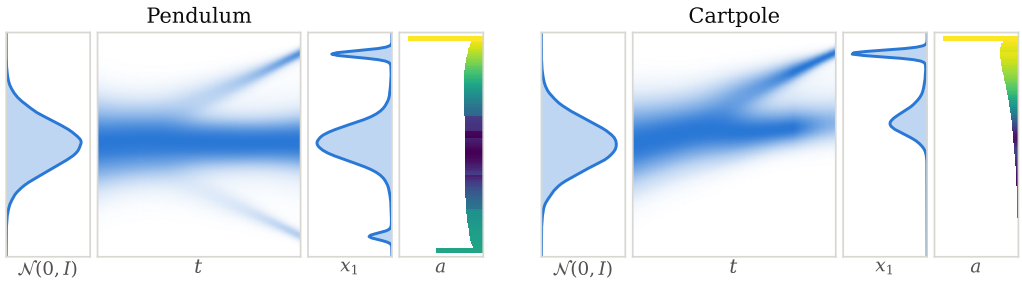


Figure 5: MB-FPO at symmetric states in pendulum and cartpole environments. The base $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$, the density of the Euler path over flow time t, the resulting pre-squash action x_1, and the executed action $a = \tanh x_1$, coloured by the model's own predicted return $\hat{Q}(a) = \hat{r}(\boldsymbol{z}') + \gamma\, \hat{c}(\boldsymbol{z}')\, V_\phi(\boldsymbol{z}')$ at that action. At the pendulum state $\hat{Q}$ has two maxima, one at each saturation, and the flow places a mode on each; at the cartpole state it has one maximum and the flow places one mode there. Note that pre-squash modes are not action modes: the squash can end up merging different latent action modes as we see in cartpole where the two x_1 modes both land on the positive side and merge into one.

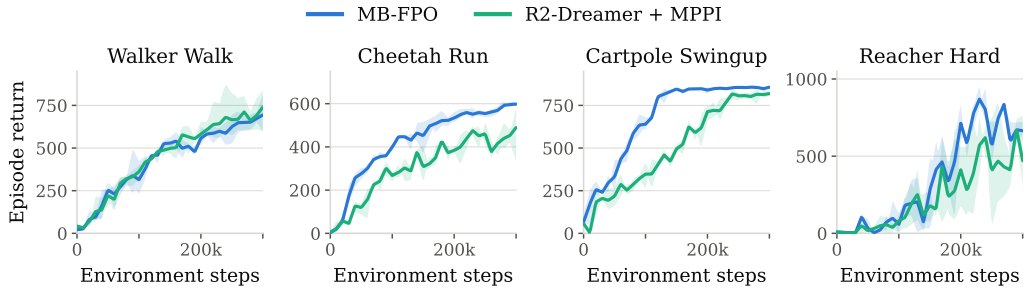


Figure 6: MB-FPO against action-space MPPI planning on the same world model, 3 seeds each, first 300k environment steps. Solid line: seed mean; band: seed min–max envelope per step; no smoothing. y-scales differ per panel.

when it has several maxima, the policy can place a mode on each. To test this, we visualize the flow at a symmetric state in each of pendulum_swingup and cartpole_swingup, alongside the predicted return

$$\hat{Q}(s, a) = \hat{r}(s, a) + \gamma\, \hat{c}(s, a)\, V(\hat{s}'), \tag{7}$$

computed over the action range from the model's predicted reward $\hat{r}$, predicted continuation probability $\hat{c}$, and the critic's value of the predicted next state $\hat{s}'$. Results are shown in Figure 5. In pendulum_swingup, $\hat{Q}$ has two maxima at the action saturation points, and MB-FPO places a mode on each. In cartpole_swingup, $\hat{Q}$ has a single maximum, and MB-FPO places a single mode there. Thus MB-FPO represents multimodal policies when the critic indicates that multiple actions are optimal, and collapses to a single mode otherwise.

4.4 Flow policy vs Planning

Flow policies are conceptually close to model predictive path integral (MPPI) planning, as used by TD-MPC2 (Hansen et al., 2024). At each step, MPPI samples Gaussian action candidates, scores them with the world model and critic, and refits the Gaussian to the exponentially weighted elites. MB-FPO performs the same weighting during training, where the indicator in $w(\hat{A}_t)$ plays the role of the elite set, but it amortizes the result into the flow network instead of recomputing it at every

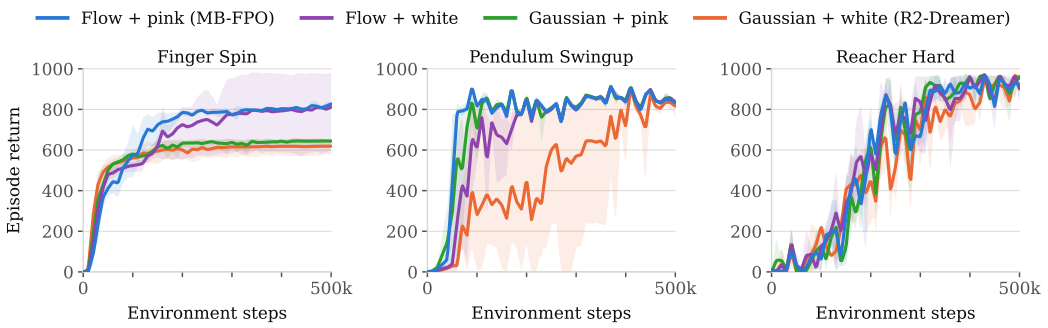


Figure 7: Flow policy vs. pink noise. Each panel crosses the policy class (flow or Gaussian) with the colour of the exploration noise used when acting in the environment (pink or white). Blue (MB-FPO) and orange (R2-Dreamer) are the main-figure runs; purple and green are the two added cells. Episode return on the three DMC tasks where MB-FPO outperforms R2-Dreamer.

environment step, and the flow can represent non-Gaussian action distributions. To test whether this amortization loses anything relative to online planning, we hold the world model fixed and change only how actions are selected. Comparing with TD-MPC2 itself would change both the world model and the decision-making mechanism at once, so we instead run TD-MPC2's MPPI planner in action space, at horizon one, on the same R2-Dreamer world model, scoring candidates with the critic trained alongside R2-Dreamer's Gaussian actor. This isolates the decision-making mechanism, but it does not reproduce TD-MPC2's full planner, which uses a multi-step horizon and a policy prior. Scoring with a critic learned for a network policy is not specific to our setup: TD-MPC2's planner likewise scores candidates with a critic trained for its policy prior. Figure 6 shows that MB-FPO matches MPPI on walker_walk and learns faster on the other three tasks. We attribute MPPI's lag mainly to its lack of amortization: every search starts from scratch, and its one-step score is dominated by a critic learned for a different policy, so candidates are likely separated mostly by critic error, whereas MB-FPO accumulates its weighted updates across states and training.

4.5 Ablation: flow policy vs. pink noise

MB-FPO differs from the Gaussian actor of R2-Dreamer in two ways: the policy class, and the exploration noise used when acting in the environment, which is pink (Eq. 6) rather than white. We ablate the design choices by running a Gaussian actor with pink noise and a flow actor with white noise on the three DMC tasks where MB-FPO outperforms R2-Dreamer (Figure 7). We see that the flow policy maintains its strong performance also with white noise, while the Gaussian policy benefits from pink noise on pendulum_swingup, in line with what was reported in (Eberhard et al., 2023), but not on finger_spin, where only the flow policy reaches the higher return. The flow policy is more expressive than a Gaussian, which helps in DMC tasks with sparse rewards.

5 Related work

Model based RL. Our method follows the Dreamer line (Hafner et al., 2025), which learns behaviors from imagined latent rollouts, and builds on R2-Dreamer (Morihira et al., 2026), which replaces pixel reconstruction with a redundancy-reduction loss (Zbontar et al., 2021) (Section 2).

Flow and diffusion policies in RL. GCPs are common in imitation learning (Chi et al., 2023). For online RL, DDPO-style methods differentiate through the denoising chain's per-step likelihoods (Black et al., 2024), DPPO (Ren et al., 2025) applies PPO to the chain treated as a multi-step MDP, and ReinFlow (Zhang et al., 2025) makes likelihoods tractable by injecting learnable noise into sampling. Off-policy, DIPO (Yang et al., 2023) fits the policy to actions moved along the critic's action gradient, and QVPO (Ding et al., 2024) weights the diffusion loss by non-negative transformations of Q-values, close in spirit to our advantage weighting. FPO (McAllister et al., 2025) avoids likelihoods with a CFM-loss ratio, which FPO++ (Yi et al., 2026) refines for robot control

with per-sample ratios and an asymmetric trust region. Because a world model provides a fresh rollout at every gradient step, we need neither multi-epoch batch reuse nor the machinery that stabilizes it (Engstrom et al., 2020). MB-FPO is thus closer to advantage-weighted regression (Peng et al., 2019) than to PPO (Schulman et al., 2017), with a CFM residual in place of a log-density. The off-policy flow method FMER (Gao et al., 2026) derives the same objective and contributes the latent-action parameterization we adopt (Section 3).

Generative policies and world models. Prior combinations of flow or diffusion policies with world models start from an imitation-trained policy. FlowMPC (Hamel, 2026) and SAILOR (Jain et al., 2025) use MPPI in a world model at inference to refine or correct an imitation policy. DiWA (Chandra et al., 2025) and World4RL (Jiang et al., 2025) fine-tune pretrained diffusion policies fully offline inside learned world models, and WAM (Han & Yilmaz, 2026) fine-tunes a behavior-cloned diffusion policy with PPO in a DreamerV2-based world model. MODIP (El Asri et al., 2026) distills MPC-planned trajectories into a behavior-cloned diffusion policy during offline-to-online fine-tuning. MB-FPO instead trains a flow policy from scratch with online model-based RL, learning the world model and the policy jointly. Concurrent work, MBDPO (Cheng et al., 2026), also trains a diffusion policy from scratch online, using a TD-MPC2-style latent world model and weighting a score-matching target by imagined returns; MB-FPO instead uses advantage-weighted CFM in Dreamer-style imagination.

6 Conclusion and Limitations

We present MB-FPO, a model-based reinforcement learning algorithm that trains flow policies in imagination with advantage-weighted flow matching and explores with time-correlated noise. It is more sample-efficient than model-free methods such as FPO++, DrQ-v2, DACER, and SAC on pixel-based and state-based control tasks. Compared with R2-Dreamer's Gaussian actor on the same world model, it learns more consistently across seeds and reaches higher returns. Our ablation attributes these gains to combining the flow policy with correlated noise. At states with exact action symmetry, MB-FPO places mass on both optimal actions during training, which the Gaussian actor does not. On the same world model, it also learns faster than one-step action-space MPPI on three of four tasks.

Limitations. We train the flow policy with advantage-weighted flow matching for simplicity. However, the world model is differentiable, which enables more sophisticated approaches, such as computing policy gradients by backpropagating through imagined rollouts. Furthermore, we leverage pink noise in the action space to enable exploration; however, more sophisticated approaches, such as time-correlated noise in the base distribution, can improve exploration. Another interesting research direction is to use the model prediction to guide the flow during inference.

Ethics statement

This work involves no human subjects and no personal data: every result is on simulated RL benchmarks, built on public, freely available datasets and simulators.

AI use statement

In this work we used generative AI tools (Anthropic Claude) for code implementation assistance, experiment orchestration and infrastructure scripting, analysis and plotting scripts, and drafting and editing of text. The research questions, method design decisions, experimental scope, and conclusions are the authors'. All AI-assisted code was reviewed by the authors, all reported results were produced by the released implementation and verified against logged training data, and all AI-drafted text was reviewed and revised by the authors. We take responsibility for the final content of this work.

Reproducibility statement

We will release our implementation, configurations, and launch commands publicly upon publication. Every figure and table in the paper regenerates directly from logged training metrics via plotting

scripts in that repository. Each training run's log records its complete command line, making any reported run's exact configuration recoverable.

REFERENCES

Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron Courville, and Marc G. Bellemare. Deep reinforcement learning at the edge of the statistical precipice. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, 2024.

Akshay L Chandra, Iman Nematollahi, Chenguang Huang, Tim Welschehold, Wolfram Burgard, and Abhinav Valada. DiWA: Diffusion policy adaptation with world models. In *Conference on Robot Learning (CoRL)*, 2025.

Xiaoyuan Cheng, Wenxuan Yuan, Zhancun Mu, Yuanzhao Zhang, Yiming Yang, Hai Wang, Zhuo Sun, and Che Liu. Scaling world-model reinforcement learning through diffusion policy optimization. *arXiv preprint arXiv:2605.26282*, 2026.

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*, 2023.

Shutong Ding, Ke Hu, Zhenhao Zhang, Kan Ren, Weinan Zhang, Jingyi Yu, Jingya Wang, and Ye Shi. Diffusion-based reinforcement learning via Q-weighted variational policy optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Gabriel Dulac-Arnold, Nir Levine, Daniel J. Mankowitz, Jerry Li, Cosmin Paduraru, Sven Gowal, and Todd Hester. Challenges of real-world reinforcement learning: Definitions, benchmarks and analysis. *Machine Learning*, 110(9):2419–2468, 2021.

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*, 2023.

Zakariae El Asri, Philippe Gratias-Quiquandon, Nicolas Thome, and Olivier Sigaud. MODIP: Efficient model-based optimization for diffusion policies. *arXiv preprint arXiv:2606.10825*, 2026.

Logan Engstrom, Andrew Ilyas, Shibani Santurkar, Dimitris Tsipras, Firdaus Janoos, Larry Rudolph, and Aleksander Madry. Implementation matters in deep RL: A case study on PPO and TRPO. In *International Conference on Learning Representations (ICLR)*, 2020.

Ting Gao, Stavros Orfanoudakis, Nan Lin, Elvin Isufi, Winnie Daamen, and Serge Hoogendoorn. Flow matching policy with entropy regularization. *arXiv preprint arXiv:2603.17685*, 2026.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning (ICML)*, 2018.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning (ICML)*, 2019.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations (ICLR)*, 2020.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, 2025.

Chandon Hamel. FlowMPC: Improving flow matching policies with world models. *arXiv preprint arXiv:2606.16286*, 2026.

Yuci Han and Alper Yilmaz. Enhancing policy learning with world-action model. *arXiv preprint arXiv:2603.28955*, 2026.

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

Arnav Kumar Jain, Vibhakar Mohta, Subin Kim, Atiksh Bhardwaj, Juntao Ren, Yunhai Feng, Sanjiban Choudhury, and Gokul Swamy. A smooth sea never made a skilled SAILOR: Robust imitation via learning to search. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Zhennan Jiang, Kai Liu, Yuxin Qin, Shuai Tian, Yupeng Zheng, Mingcai Zhou, Chao Yu, Haoran Li, and Dongbin Zhao. World4RL: Diffusion world models for policy refinement with reinforcement learning for robotic manipulation. *arXiv preprint arXiv:2509.19080*, 2025.

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*, 2023.

David McAllister, Songwei Ge, Brent Yi, Chung Min Kim, Ethan Weber, Hongsuk Choi, Haiwen Feng, and Angjoo Kanazawa. Flow matching policy gradients. *arXiv preprint arXiv:2507.21053*, 2025.

Naoki Morihira, Amal Nahar, Kartik Bharadwaj, Yasuhiro Kato, Akinobu Hayashi, and Tatsuya Harada. R2-Dreamer: Redundancy-reduced world models without decoders or augmentation. In *International Conference on Learning Representations (ICLR)*, 2026.

Chaoyi Pan, Giridharan Anantharaman, Nai-Chieh Huang, Claire Jin, Daniel Pfrommer, Chenyang Yuan, Frank Permenter, Guannan Qu, Nicholas Boffi, Guanya Shi, et al. Much ado about noising: Dispelling the myths of generative robotic control. In *International Conference on Learning Representations*, 2026.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Michael Psenka, Alejandro Escontrela, Pieter Abbeel, and Yi Ma. Learning a diffusion model policy from rewards via Q-score matching. In *International Conference on Machine Learning (ICML)*, 2024.

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.

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

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. DeepMind control suite. *arXiv preprint arXiv:1801.00690*, 2018.

Yinuo Wang, Likun Wang, Yuxuan Jiang, Wenjun Zou, Tong Liu, Xujie Song, Wenxuan Wang, Liming Xiao, Jiang Wu, Jingliang Duan, and Shengbo Eben Li. Diffusion actor-critic with entropy regulator. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Long Yang, Zhixiong Huang, Fenghao Lei, Yucun Zhong, Yiming Yang, Cong Fang, Shiting Wen, Binbin Zhou, and Zhouchen Lin. Policy representation via diffusion probability model for reinforcement learning. *arXiv preprint arXiv:2305.13122*, 2023.

Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. In *International Conference on Learning Representations*, 2022.

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.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-World: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2019.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow Twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, 2021.

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 Advantage and critic

Returns. On an imagined rollout of H steps we compute the λ-return R_t of Eq. 1 from the predicted rewards $\hat{r}_t$, continuation probabilities $\hat{c}_t$ and critic values, bootstrapping from the critic at the last step, with $\gamma = 1 - 1/333$ and $\lambda = 0.95$.

Advantage. The baseline is the critic's value at the same state, and the difference is divided by DreamerV3's return scale S: an exponential moving average (decay 0.01) of the range between the 5th and 95th percentile of R over the imagined batch, floored at 1,

$$\tilde{A}_t = \frac{R_t - V_\psi(\boldsymbol{z}_t)}{S}.$$

On our tasks the standard deviation of $\tilde{A}$ sits around 0.02–0.06, far below the unit scale that the temperature β in Eq. 4 assumes, so we standardize it over the imagined batch and clamp,

$$\hat{A}_t = \mathrm{clip}\Big(\frac{\tilde{A}_t}{\mathrm{std}(\tilde{A}) + 10^{-8}},\ -100,\ 100\Big),$$

with $\mathrm{std}(\tilde{A})$ taken over every step of the batch. The exponent $\hat{A}/\beta$ in w is additionally capped at 5.

Critic. The critic is DreamerV3's, unchanged by the actor: it is trained by maximum likelihood on the λ-returns, regularized toward a slow-moving average of itself, on the imagined rollout (weighted by ω_t) and on the replayed sequences.

B Advantage-weighted flow matching as AWR

We derive Eq. 4 as advantage-weighted regression (AWR) (Peng et al., 2019) with the CFM loss standing in for the negative log-likelihood, following FPO (McAllister et al., 2025) and FMER (Gao et al., 2026).

Policy improvement step. Let π_k be the current policy and $A_k(\boldsymbol{z}, \boldsymbol{a})$ its advantage. AWR improves π_k by solving, at each state,

$$\max_{\pi}\ \mathbb{E}_{\boldsymbol{a}\sim\pi}\big[A_k(\boldsymbol{z}, \boldsymbol{a})\big] - \beta\,\mathrm{KL}\big(\pi \,\|\, \pi_k\big),$$

whose solution is $\pi^*(\boldsymbol{a} \mid \boldsymbol{z}) \propto \pi_k(\boldsymbol{a} \mid \boldsymbol{z})\,\exp\big(A_k(\boldsymbol{z}, \boldsymbol{a})/\beta\big)$. Projecting π^* onto the parametric family by $\min_\theta \mathrm{KL}(\pi^* \,\|\, \pi_\theta)$ gives, up to terms independent of θ,

$$\max_{\theta}\ \mathbb{E}_{\boldsymbol{a}\sim\pi_k}\Big[\exp\big(A_k(\boldsymbol{z}, \boldsymbol{a})/\beta\big)\ \log \pi_\theta(\boldsymbol{a} \mid \boldsymbol{z})\Big].$$

The expectation is over actions from the current policy, which is what an imagined rollout provides: with a fresh rollout for every gradient step, π_k is the policy at the start of that step. In practice the regressed actions include the exploration noise of Eq. 5, so they are drawn from π_k perturbed by that noise rather than from π_k itself, and the fitted target is correspondingly broader; this is what keeps the flow from contracting.

CFM loss in place of the log-likelihood. A flow's $\log \pi_\theta$ requires integrating the divergence of v_θ along the sampling path. As in FPO and FMER, we instead treat the CFM loss of a sample as a surrogate for its negative log-likelihood, $-\log \pi_\theta(\boldsymbol{a} \mid \boldsymbol{z}) \approx \mathcal{L}_{\mathrm{CFM}}(\theta; \boldsymbol{z}, \boldsymbol{a}) + C$ with C independent

of θ; the CFM loss matches an evidence lower bound only under a τ-dependent weighting, which we, like FPO, drop. Substituting the Monte Carlo estimate $\hat{\mathcal{L}}_{\mathrm{CFM}}$ turns the maximization above into

$$\min_{\theta} \mathbb{E}_{\boldsymbol{a}\sim\pi_k}\Big[\exp\big(A_k(\boldsymbol{z},\boldsymbol{a})/\beta\big)\,\hat{\mathcal{L}}_{\mathrm{CFM}}(\theta;\boldsymbol{z},\boldsymbol{a})\Big],$$

which is Eq. 4 without the indicator, summed over imagined steps with the discount weights ω_t.

Dropping non-positive advantages. Multiplying the weight by $\mathbb{1}[\hat{A} > 0]$ replaces the target with $\pi^*(\boldsymbol{a} \mid \boldsymbol{z}) \propto \pi_k(\boldsymbol{a} \mid \boldsymbol{z}) \exp(A_k/\beta)\,\mathbb{1}[A_k > 0]$, the current policy restricted to actions better than the critic's baseline and reweighted toward the best of them. Because this target is a normalized distribution, fitting it lowers the probability of the excluded actions implicitly, without the explicit push-away of FPO's signed gradient $-\hat{A}\,\nabla_\theta \hat{\mathcal{L}}_{\mathrm{CFM}}$, which for $\hat{A} < 0$ increases a loss that is unbounded above. The weight is therefore non-negative and, with the cap on $\hat{A}/\beta$ (Appendix A), bounded.

C Hyperparameters

Table 1 lists the hyperparameters of MB-FPO. The flow actor's Euler steps and CFM draws are the same as in our FPO++ baseline. The shared settings are those of R2-Dreamer and are identical for MB-FPO, the Gaussian R2-Dreamer actor, the MPPI variant and every ablation cell.

Table 1: Hyperparameters of MB-FPO. Budgets are counted in simulator frames.

Flow actor	
Euler steps N	10 (as FPO++)
CFM draws per action n_{mc}	16 (as FPO++)
Flow time τ	$\mathrm{U}[0.005, 0.995]$
Velocity network	MLP, 3 layers of 256, SiLU, 32-d sinusoidal time embedding
Velocity output scale	0.25
CFM reduction over action dimensions	sum $/\sqrt{\dim \boldsymbol{a}}$
Squash	$\boldsymbol{a} = \tanh(\boldsymbol{x}_1)$; CFM target $\operatorname{arctanh}(\boldsymbol{a})$, $\boldsymbol{a}$ clipped to $\pm(1 - 10^{-3})$
Temperature β	1
Advantage	standardized over batch, clipped to ± 100; $\hat{A}/\beta$ capped at 5
Exploration noise scale σ	0.3
Exploration noise colour	pink ($\alpha = 1$) in the environment, white in imagination
Shared (all model-based methods)	
Imagination horizon H	15
Discount γ, return λ	$1 - 1/333$, 0.95
Critic	DreamerV3 two-hot (255 bins), slow target (fraction 0.02)
World model	R2-Dreamer, 12M configuration; RSSM deterministic 2048, stochastic 32×16
Barlow Twins λ_{bt}	5×10^{-4}
Optimizer	LaProp, lr 4×10^{-5}, $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-20}$
Gradient clipping, warmup	AGC 0.3, 1000 steps
Batch	16 sequences $\times$ 64 steps
Train ratio	512 replayed steps per agent step
Parallel environments	16
Action repeat	2
Budget	500k frames (cartpole_swingup 300k; Meta-World reach 250k)

D Algorithm

Algorithm 1 One MB-FPO actor–critic update

Require: world model, actor v_θ, critic V_ψ, horizon H, Euler steps N, MC draws n_{mc}, temperature β, noise scale σ

1: Sample a batch of sequences from replay; infer posterior latent states $\boldsymbol{z}_0 = (h_0, z_0)$
2: **for** $t = 0, \dots, H-1$ **do**
3: $\quad \boldsymbol{x}_0 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$; integrate N Euler steps of $v_\theta(\cdot, \cdot, \boldsymbol{z}_t)$ to obtain $\boldsymbol{x}_1$
4: $\quad \boldsymbol{a}_t \leftarrow \mathrm{clip}\big(\tanh(\boldsymbol{x}_1) + \sigma\boldsymbol{\xi}_t,\ -1,\ 1\big),\ \boldsymbol{\xi}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ ▷ white noise in imagination
5: $\quad \boldsymbol{z}_{t+1} \sim p_\phi(\cdot \mid \boldsymbol{z}_t, \boldsymbol{a}_t)$ ▷ world-model transition
6: **end for**
7: Predict $\hat{r}_t, \hat{c}_t$ from the world model; compute λ-returns and advantages $\hat{A}_t$ (Appendix A)
8: $w_t \leftarrow \exp(\hat{A}_t/\beta)\, \mathbb{1}[\hat{A}_t > 0]$
9: Draw $(\tau, \boldsymbol{x}_0)$ n_{mc} times per $(\boldsymbol{z}_t, \boldsymbol{a}_t)$; form $\hat{\mathcal{L}}_{\text{CFM},t}$ with endpoint $\boldsymbol{x}_1 = \mathrm{arctanh}(\boldsymbol{a}_t)$
10: $\theta \leftarrow \theta - \eta \nabla_\theta \sum_t \omega_t\, w_t\, \hat{\mathcal{L}}_{\text{CFM},t}$; update V_ψ on the λ-returns
11: Discard the rollout