CFG-POLICY: UNIFIED DIFFUSION POLICY LEARNING FRAMEWORK VIA CLASSIFIER-FREE GUIDANCE

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion policies are increasingly used in robotic control and manipulation, vision-language-action models, and world-model-based agents. Their ability to represent multimodal action distributions offers rich opportunities for exploration in reinforcement learning (RL). However, optimizing diffusion policies with RL can reduce the behavioral diversity and further limit their exploration capability. In RL post-training, directly fine-tuning a pretrained policy may also disrupt its acquired capabilities, while action-space residuals may lack the expressive power for multimodal exploration. Together, these issues reveal a common challenge across online and offline-to-online diffusion RL: improving task performance while retaining the behavioral diversity and capabilities needed for continued adaptation. We introduce CFG-Policy, a unified, plug-and-play framework that separates exploration and exploitation into two generative policies and combines their score or velocity fields during sampling through classifier-free guidance. For online RL, a frozen exploration branch sustains behavioral diversity while an exploitation branch learns from task rewards; for offline-to-online RL, a pretrained exploitation branch remains frozen while a trainable branch steers the composed behavior. CFG-Policy preserves the underlying RL objectives and improves performance with QSM and QVPO on 5 MuJoCo tasks and with Flow-SDE+PPO on 4 RoboTwin manipulation tasks.

1 INTRODUCTION

Diffusion policies have become an expressive model class for sequential decision making. Their ability to represent complex, multimodal action distributions has supported robotic control and manipulation (Chi et al., 2025; Liu et al., 2025), vision-language-action policies (Chen et al., 2025a), and world-model-based policy adaptation (Chandra et al., 2025). Related flow-matching policies generate actions through continuous velocity fields and are increasingly used in reinforcement learning (RL) (Lipman et al., 2022; Park et al., 2025; Ding et al., 2026a; Hu et al., 2026). Beyond fitting demonstrations, these generative policies offer a rich space of plausible actions to explore when learning from rewards.

However, the diversity that makes diffusion policies attractive for exploration can decline as reward optimization concentrates the policy on currently successful behaviors. Prior work has observed mode collapse and training instability when diffusion models are fine-tuned with RL (Barceló et al., 2024). In online diffusion RL, methods such as Q-Score Matching and QVPO improve task performance by updating a single generative policy (Psenka et al., 2023; Ding et al., 2024); continued improvement then depends on that same policy still producing sufficiently diverse actions. In RL post-training, direct optimization of a pretrained diffusion or flow policy (Ren et al., 2025; Zhang et al., 2026) may disturb capabilities learned before RL. Residual policy methods can preserve a base policy, but action-space corrections may provide less expressive exploration than modifying the generative process itself (Johannink et al., 2019; Yuan et al., 2025). These settings expose the same underlying challenge: how can diffusion RL improve task performance while retaining the behavioral diversity and capabilities needed for continued adaptation?

To address these challenges, we propose a unified, plug-and-play framework called **CFG-Policy** that separates exploration and exploitation into two generative policies. Inspired by classifier-free guidance (CFG) (Ho & Salimans, 2022), it combines their score or velocity fields at each step of

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.

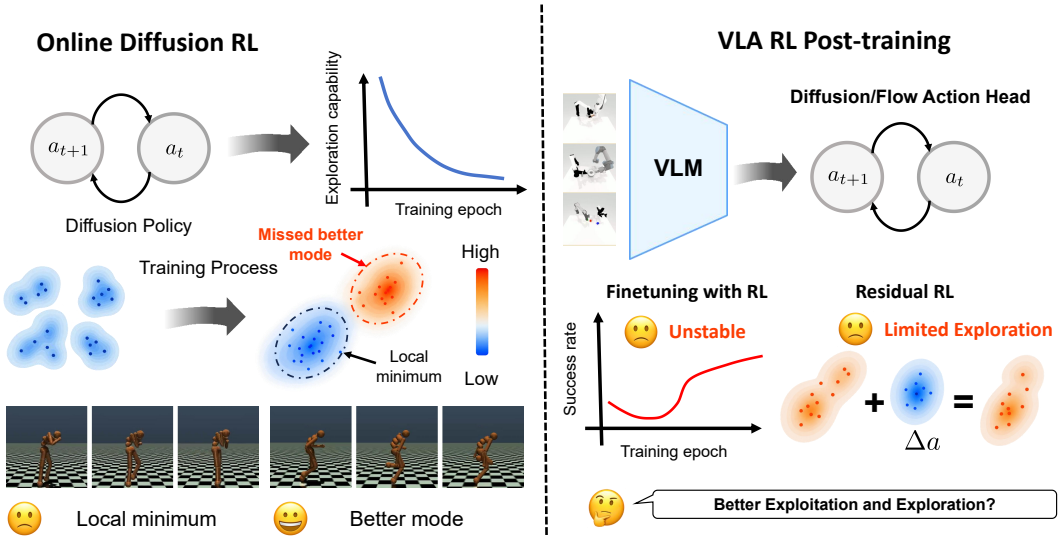


Figure 1: Exploration–exploitation challenges in diffusion RL. In online diffusion RL, optimizing a single generative policy can progressively reduce behavioral diversity and miss better modes. In VLA RL post-training, directly fine-tuning the pretrained policy may disrupt pretrained capabilities, while residual policies can provide limited exploration.

action generation. Given two fields v_A and v_B, the field of behavior policy is induced by

$$v_w = v_A + w(v_B - v_A) \tag{1}$$

where w controls their relative contribution. Both fields act on the same intermediate sample, so the combination steers the entire generative trajectory rather than interpolating completed actions. Because this composition replaces only the behavior sampler, it can be added to different diffusion RL algorithms without redefining their actor or critic objectives.

The same mechanism handles two complementary learning regimes by changing which branch is frozen. In **online RL**, a separately trained exploration policy remains fixed while the exploitation policy learns from task rewards; the composed sampler maintains an exploratory source throughout training. In **offline-to-online RL post-training**, a pretrained exploitation policy remains fixed while a second, trainable branch steers the composed policy toward higher-return behaviors. The framework thus preserves exploration while learning exploitation in the first regime, and preserves an acquired policy while learning adaptive behavior in the second. We test this formulation with QSM and QVPO on five MuJoCo tasks and with Flow-SDE+PPO on four RoboTwin manipulation tasks, finding improvements in both regimes.

Our contributions are:

- We provide a unified exploration–exploitation view of online RL and offline-to-online RL post-training: the former calls for preserving exploration while learning exploitation, whereas the latter calls for preserving an acquired policy while learning adaptive behavior;
- We introduce **CFG-Policy**, a plug-and-play framework that realizes this view by combining separate exploration and exploitation policies during generative sampling while preserving the underlying RL optimization objectives;
- We demonstrate the effectiveness and versatility of **CFG-Policy** across both regimes, improving QSM and QVPO on MuJoCo control tasks and Flow-SDE+PPO on RoboTwin manipulation tasks.

2 Related Work

Classifier-Free Guidance. Guidance was originally introduced in diffusion models to steer generation toward desired conditions. Classifier guidance uses gradients from an auxiliary classifier to modify the reverse diffusion process (Dhariwal & Nichol, 2021), while classifier-free guidance (CFG)

Table 1: Comparison of representative generative-policy RL methods. CFG-Policy is a plug-and-play module applicable to compatible generative RL backbones listed above. ✓: explicit support; △: partial or implicit support; ✗: unsupported. **E-E**: Exploration-Exploitation

Setting	Family	Method	Exploration	Adaptive E-E	E-E Decoupling	Capability Preserv.
Generative RL	Q-weighted	QVPO	✓	△	✗	✗
		SDAC	✓	✓	✗	✗
	Critic-Guided	DIPO	△	✗	✗	✗
		QSM	△	✗	✗	✗
		DACER	✓	✓	✗	✗
		DIME	✓	✓	✗	✗
RL Post-Training	Policy-Gradient Tuning	DPPO	△	✗	✗	✗
		ReinFlow	✓	△	✗	✗
		Flow-SDE	✓	✓	✗	✗
		Flow-Noise	✓	✓	✗	✗
	Residual Adaptation	Residual RL	✗	✗	✗	✓
Any compatible backbone + CFG-Policy (Ours)			✓	✓	✓	✓

instead combines conditional and unconditional model predictions during sampling (**?**). The guidance scale controls the strength of this steering and has been widely used in text-to-image systems such as GLIDE and Imagen (Nichol et al., 2021; Saharia et al., 2022). Beyond conditional generation within one model, composable diffusion models combine separately modeled concepts through their generative scores (Liu et al., 2022). Subsequent work shows that successful score composition also depends on the sampler and studies MCMC-based corrections for compositional generation (Du et al., 2023).

Generative Reinforcement Learning. Offline RL established diffusion policies as expressive models of multimodal behavior: Diffusion-QL combines behavior modeling with Q-value maximization (Wang et al., 2022), while IDQL reweights actions sampled from a diffusion behavior policy using critic estimates (Hansen-Estruch et al., 2023). Online diffusion RL must improve policies through interaction rather than samples from a fixed dataset (Ding et al., 2026b). QVPO uses a Q-weighted variational objective (Ding et al., 2024), and SDAC uses reweighted score matching to fit value-induced maximum-entropy policies (Ma et al., 2025). DIPO improves sampled actions with $\nabla_a Q$ before refitting the policy (Yang et al., 2023), whereas QSM aligns the policy score with the critic gradient (Psenka et al., 2023). DACER, DIME, and MaxEntDP further study actor–critic updates and entropy-aware optimization for diffusion policies (Wang et al., 2024; Celik et al., 2025; Dong et al., 2025). ExDM explores diffusion models for unsupervised reinforcement learning by modeling state visitation distributions and deriving intrinsic rewards from distributional novelty, encouraging agents to discover rarely visited states (Ying et al., 2026). Separately, CFGRL treats CFG as a controllable improvement operator for extracting a policy from offline data (Frans et al., 2025). These methods illustrate different ways to optimize or guide a generative policy, but generally leave exploration and exploitation coupled within the policy being improved.

VLA and Generative Policy RL Post-Training. Large vision-language-action (VLA) policies such as π_0 use flow matching to generate robot actions from visual and language inputs (Black et al., 2024). RL post-training seeks to improve such pretrained behavior through reward feedback. DPPO and ReinFlow develop policy-gradient methods for diffusion and flow policies (Ren et al., 2025; Zhang et al., 2026), while π_{RL} applies online RL fine-tuning to flow-based VLA models (Chen et al., 2025a). RIPT-VLA uses sparse binary success rewards and interactive rollouts to post-train pretrained VLA models (Tan et al., 2025), and Flow Policy Optimization addresses policy-ratio estimation for online fine-tuning of flow-matching VLAs (Lyu et al., 2025). In the offline setting, Adaptive Reinforced Flow Matching controls the effect of advantage-weighted RL signals on the flow loss to stabilize VLA post-training (Zhang et al., 2025). An alternative is residual adaptation, which freezes a base policy and learns an action-level correction (Johannink et al., 2019; Silver et al., 2018; Yuan et al., 2025).

However, online diffusion RL methods can lose exploratory diversity as the policy converges toward high-reward behaviors, while RL post-training needs a more stable and sample-efficient way to

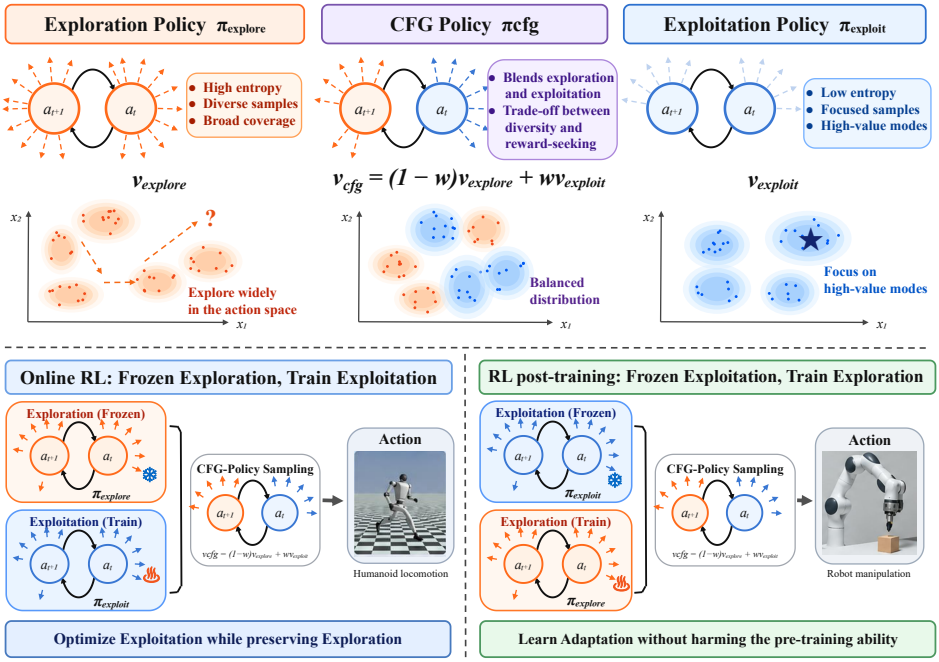


Figure 2: The sampling process of CFG-Policy. It composes two generative policies with different behavioral preferences in score space with guidance weight w.

explore beyond a pretrained policy. The proposed CFG-Policy addresses these distinct challenges by composing separate exploration and exploitation policies throughout diffusion or flow sampling. In online RL settings, a frozen exploration branch continues to supply diverse behavior as the exploitation branch improves. In offline-to-online RL post-training, the pretrained policy remains frozen while a trainable generative branch explores and steers the composed policy toward higher returns. This design preserves the selected behavioral capability, enables adaptation throughout the generative trajectory rather than only through final-action residuals, and can be plugged into compatible RL backbones without changing their optimization objectives. Notably, CFGRL uses guidance as a policy-improvement operator learned from fixed offline data (Frans et al., 2025), whereas CFG-Policy combines two behaviorally distinct policies and optimizes one through RL interaction, as summarized in Table 1.

3 Method

As shown in Figure 2, we propose CFG-Policy which aims to address the two learning regimes discussed in the Introduction through a common principle: *preserve one behavioral capability in a frozen generative policy while optimizing a complementary one*. The two policies are composed during diffusion or flow sampling, allowing the same mechanism to preserve exploration in online RL and pretrained behavior in RL post-training.

3.1 CFG-Policy: Compositional Generative Policies

Consider a diffusion or flow policy that generates an action $a = x_0$ from $x_T \sim \mathcal{N}(0, I)$. Given two generative policies π_A and π_B with generative fields $v_A(x_t, s, t)$ and $v_B(x_t, s, t)$, CFG-Policy constructs

$$v_w(x_t, s, t) = (1 - w)v_A(x_t, s, t) + wv_B(x_t, s, t), \tag{2}$$

where w controls their relative contribution. Both fields operate on the same intermediate sample x_t, so the composition steers the entire generative trajectory rather than interpolating two independently generated final actions.

Algorithm 1 CFG-POLICY for Online Reinforcement Learning

Require: Frozen π_E; trainable π_X; critic Q; replay buffer $\mathcal{B}$; base RL algorithm $\mathcal{A}$; guidance weight w

1: Initialize $\pi_X, Q, \mathcal{B}$ and freeze π_E
2: **for** each environment step **do**
3: Sample $a \sim \pi_w(\cdot|s)$ using $v_w \leftarrow (1-w)v_E + wv_X$
4: Execute a and store (s, a, r, s', d) in $\mathcal{B}$
5: Update Q and π_X using the original objectives of $\mathcal{A}$
6: **end for**
7: **Evaluation:** set $w = 1$ and execute π_X

We denote by $\pi_w^{\mathrm{sam}}(a \mid s)$ the endpoint action distribution induced by running the generative sampler with the composed field in equation 2.

Proposition 1 (Endpoint Interpretation under Path Consistency)**.** *For score-based diffusion policies, define the geometric composition at each noise level as*

$$q_{w,t}(x_t \mid s) \propto \pi_{A,t}(x_t \mid s)^{1-w} \pi_{B,t}(x_t \mid s)^w. \tag{3}$$

By equation 3, the composed score is exactly $\nabla_{x_t} \log q_{w,t}$ at every fixed t.

If the family $\{q_{w,t}\}_{t=0}^T$ is approximately consistent with the shared forward diffusion and its terminal marginal is close to the common diffusion prior, then the endpoint distribution induced by the composed reverse sampler satisfies

$$\pi_w^{\mathrm{sam}}(a \mid s) \approx q_{w,0}(a \mid s) \propto \pi_A(a \mid s)^{1-w} \pi_B(a \mid s)^w. \tag{4}$$

The geometric composition is exact at each fixed noise level, but need not form an exact diffusion marginal path across time. Appendix B.1 analyzes this path-consistency gap and provides the conditions underlying the endpoint approximation.

3.2 ONLINE RL: PRESERVE EXPLORATION, LEARN EXPLOITATION

In online RL, reward optimization may progressively reduce policy diversity, hence harming exploration. CFG-Policy maintains a frozen exploration policy π_E to keep a stable ability of exploration and learns an exploitation policy π_X from task rewards. During environment interaction, we use

$$v_w = (1-w)v_E + wv_X. \tag{5}$$

The composed policy π_w serves as the behavior policy for data collection, while the critic and π_X are updated using the original objectives of the base RL algorithm. The exploration branch remains fixed throughout training, providing a persistent source of exploratory behavior.

The guidance coefficient directly controls this trade-off. Under a simple sharpening assumption, its effect can be characterized through policy entropy.

Theorem 1 (Entropy of the Geometric Composition)**.** *For a fixed state s, define the endpoint geometric-composition surrogate*

$$q_w(a \mid s) \propto \pi_E(a \mid s)^{1-w} \pi_X(a \mid s)^w. \tag{6}$$

Assume that the exploitation policy is a sharpened version of the exploration policy,

$$\pi_X(a \mid s) = \frac{\pi_E(a \mid s)^\beta}{Z_\beta(s)}, \qquad \beta > 1. \tag{7}$$

Then, for $w_1 < w_2$,

$$H(q_{w_1}(\cdot \mid s)) \geq H(q_{w_2}(\cdot \mid s)). \tag{8}$$

Thus, increasing w makes the geometric-composition surrogate more concentrated toward the exploitation policy. Under the path-consistency conditions of Proposition 1, the sampler-induced endpoint policy π_w^{sam} is approximated by q_w, providing an idealized explanation of how guidance controls exploration and exploitation. The algorithm for online RL is summarized in Algorithm 1 and the proof is provided in Appendix B.3.

Algorithm 2 CFG-POLICY for RL Post-Training

Require: Frozen pretrained π_X^0; trainable steering policy π_E^θ; base RL algorithm $\mathcal{A}$; guidance weight w

1: Freeze π_X^0 and initialize π_E^θ
2: **for** each RL iteration **do**
3: Construct $v_w^\theta \leftarrow (1-w)v_X^0 + wv_E^\theta$
4: Collect rollouts with the composed policy
5: Compute the original RL objective of $\mathcal{A}$
6: Stop gradients through π_X^0 and update only π_E^θ
7: **end for**
8: **return** $(\pi_X^0, \pi_E^\theta, w)$

3.3 RL POST-TRAINING: PRESERVE PRETRAINED BEHAVIOR, LEARN ADAPTATION

RL post-training presents the complementary setting: a pretrained generative policy already provides useful behavior, while direct RL updates may alter its parameters. CFG-Policy therefore freezes the pretrained policy π_X^0 and optimizes an additional steering policy π_E^θ:

$$v_w^\theta = (1-w)v_E^\theta + wv_X^0. \tag{9}$$

The composed policy is used for rollout and, when required, for likelihood recomputation. The base RL objective is kept unchanged, while gradients are stopped through v_X^0 and only π_E^θ is optimized. Unlike action-space residuals, the trainable branch steers the full generative trajectory while leaving the pretrained action generator unchanged.

The two regimes are therefore symmetric: online RL preserves exploration while learning exploitation, whereas RL post-training preserves a pretrained policy while learning adaptive behavior. The composition mechanism itself remains unchanged. The algorithm for RL post-training is summarized in Algorithm 2.

3.4 ADAPTIVE COMPOSITION

A fixed guidance weight may not provide the appropriate exploration–exploitation balance throughout training. We therefore adapt w by maximizing the critic value of the composed policy while maintaining a minimum behavioral deviation:

$$\max_w J_Q(w) \qquad \text{s.t.} \qquad D_{\text{path}}(w) \geq \delta, \tag{10}$$

where $J_Q(w)$ denotes the critic value under guidance weight w, and $D_{\text{path}}(w)$ prevents premature collapse toward the exploitation policy.

We define $D_{\text{path}}(w)$ as a pathwise discrepancy between the composed policy and the pure exploitation policy. Given trajectories sampled from π_w, we compare their reverse-transition means with those of π_1 at the same intermediate states:

$$\widehat{D}_{\text{path}}(w) = \text{median}_{s\in\mathcal{S}}\left[\frac{1}{d_a R}\sum_{r=1}^{R}\sum_{t=0}^{T-1}\frac{\left\|\mu_t^w(x_t^{(r)}, s) - \mu_t^1(x_t^{(r)}, s)\right\|_2^2}{2\sigma_t^2}\right]. \tag{11}$$

Here, $x_t^{(r)}$ is sampled from the composed trajectory under w, and both transition means are evaluated at the same $x_t^{(r)}$. Thus, D_{path} measures deviation from the pure exploitation sampler along the generated trajectory, rather than the KL divergence between the final action distributions.

We optimize the corresponding Lagrangian $\mathcal{L}(w, \lambda) = J_Q(w) + \lambda\left(D_{\text{path}}(w) - \delta\right)$, using finite-difference estimates for w and dual updates for λ.

Table 2: Results across five environments. Values are mean ± sample standard deviation over 5 runs. The maximum value for each task is shown in bold.

Method	HalfCheetah-v4	Walker2d-v4	Ant-v4	Humanoid-v4	Hopper-v4
QSM Original	5020 ± 455	2041 ± 477	-24 ± 3	5922 ± 262	2753 ± 307
QSM CFG-Policy	6218 ± 533	2349 ± 583	1425 ± 308	6094 ± 169	3101 ± 461
Improvement	$+23.9\%$	$+15.1\%$	$+6141.3\%$	$+2.90\%$	$+12.7\%$
QVPO Original	11166 ± 5076	5220 ± 689	7129 ± 61	6612 ± 214	3727 ± 176
QVPO CFG-Policy	$\mathbf{12990 \pm 1993}$	$\mathbf{5692 \pm 413}$	$\mathbf{7374 \pm 45}$	$\mathbf{7041 \pm 385}$	$\mathbf{3871 \pm 192}$
Improvement	$+16.3\%$	$+9.0\%$	$+3.4\%$	$+6.49\%$	$+3.8\%$

Under idealized conditions, this adaptive objective admits a simple non-degradation guarantee. If the original coefficient w_0 is feasible, $J_Q(w) = J(\pi_w)$ equals the true return, and the constrained problem is solved globally, then its solution $w^\star$ satisfies

$$J(\pi_{w^\star}) \geq J(\pi_{w_0}). \tag{12}$$

Further deviation, details of pathwise discrepancy and calibration of δ are provided in Appendix B.2.

4 Experiments

We evaluate CFG-Policy in the two complementary regimes introduced in the Method: online RL and RL post-training. Our experiments study whether (i) CFG-Policy can be applied to different generative RL algorithms without changing their original objectives, (ii) the same compositional mechanism is effective in both learning regimes, and (iii) adaptive generative composition is preferable to fixed guidance or simple action-space blending.

4.1 Experimental Setup

Online RL. We evaluate on five MuJoCo tasks: HalfCheetah-v4, Walker2d-v4, Ant-v4, Humanoid-v4, and Hopper-v4. CFG-Policy is instantiated with QSM (Psenka et al., 2023) and QVPO (Ding et al., 2024). The original actor and critic objectives are unchanged. The exploitation branch is inherited from the base algorithm, while a separately pretrained RND exploration policy (Burda et al., 2018), which we use it for the exploration branch, is kept frozen during RL training. We report mean performance and sample standard deviation over 5 runs.

RL post-training. We evaluate on four RoboTwin tasks (Chen et al., 2025b): Adjust Bottle, Pick Dual Bottles, Rotate QR Code, and Grab Roller. For both π_0 and $\pi_{0.5}$ (Black et al., 2024; Intelligence et al., 2025), the pretrained policy is frozen as the exploitation branch, while a second branch initialized from the same SFT checkpoint is optimized with RL. We compare against direct Flow-SDE+PPO fine-tuning and report task success rates.

We provide introduction of the tasks of benchmark and further implementation details in Appendix A and B.4.

4.2 Online Reinforcement Learning

As shown in Table 2, CFG-Policy consistently improves both QSM and QVPO on all evaluated tasks. For QSM, the improvement is particularly significant on Ant-v4, where CFG-Policy increases the return from -24 to 1425 ± 308. It also improves HalfCheetah-v4, Walker2d-v4, and Hopper-v4, demonstrating that preserving a separate exploration branch can provide additional exploration capability during reward optimization.

The benefit is not limited to weaker baselines. Starting from the stronger QVPO baseline, CFG-Policy further improves performance across all five environments, including HalfCheetah-v4 and Humanoid-v4. Figure 3 shows consistent improvements throughout training, demonstrating that compositional sampling enhances existing RL objectives without modification. Despite introducing an additional

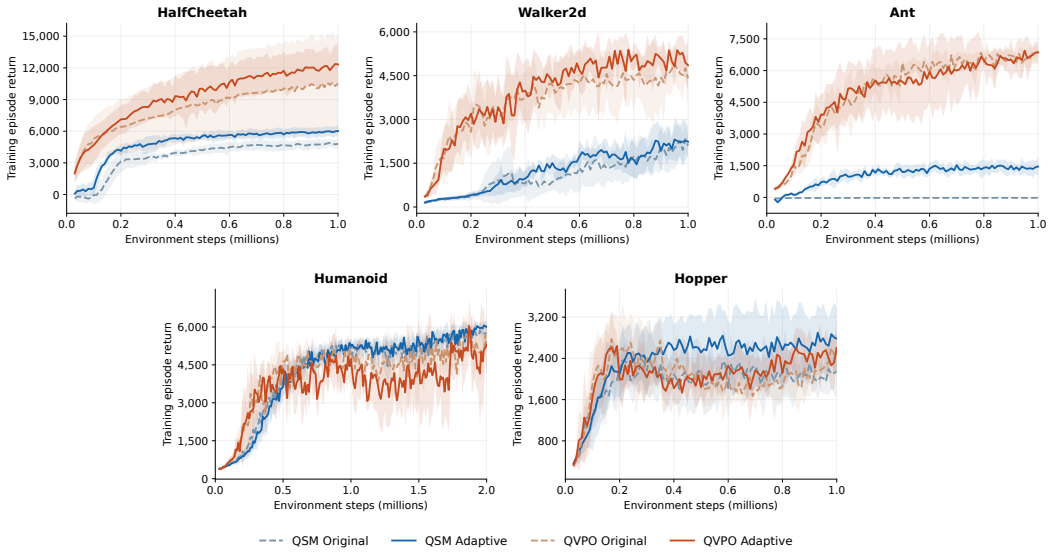


Figure 3: Learning curve of QSM, QVPO before and after using CFG-Policy on five MuJoCo tasks. The x-axis is the environment steps and the y-axis shows average episodic return with one standard deviation shaded.

Table 3: Success rates of the original Flow-SDE + PPO and CFG-Policy.

Model	Method	Adjust Bottle	Pick Dual Bottles	Rotate QR Code	Grab Roller
π_0	FlowSDE+PPO	82.81%	43.75%	34.38%	92.19%
	CFG-Policy	**98.44%**	**57.81%**	**42.19%**	**93.75%**
$\pi_{0.5}$	FlowSDE+PPO	89.06%	12.50%	7.81%	**100.00%**
	CFG-Policy	**92.19%**	**15.62%**	**10.94%**	**100.00%**

generative branch, CFG-Policy incurs only marginal computational overhead in memory and training time. We provide the cost details in Appendix B.5.

4.3 RL Post-Training

We next evaluate CFG-Policy for RL post-training with pretrained generative policies. Unlike online RL, where the exploration branch is preserved, here the pretrained exploitation policy is frozen and a separate steering policy is optimized.

Table 3 compares CFG-Policy with direct Flow-SDE+PPO fine-tuning on four RoboTwin manipulation tasks using both π_0 and $\pi_{0.5}$. CFG-Policy improves non-saturated tasks while maintaining saturated performance.

For π_0, CFG-Policy increases the success rate on Adjust Bottle from 82.81% to 98.44%, and on Pick Dual Bottles from 43.75% to 57.81%. Similar improvements are observed for $\pi_{0.5}$ on the non-saturated tasks, while both methods reach full success on Grab Roller.

These results show that generative composition enables policy adaptation through an additional trainable branch while preserving the original pretrained action generator.

4.4 Ablation Study

We investigate two design choices of CFG-Policy: (1) adaptive guidance coefficient selection and (2) generative-space composition compared with action-space interpolation.

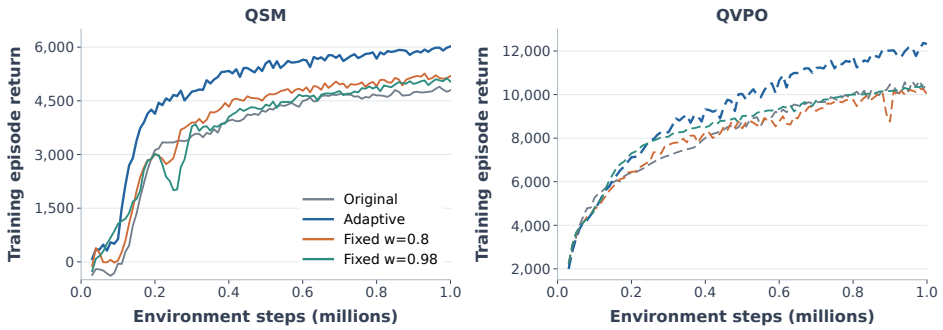


Figure 4: Comparison of adaptive guidance with two fixed guidance coefficients. The dashed line corresponds to QVPO and the solid line corresponds to QSM.

Table 4: Action-space blending ablation on HalfCheetah-v4. Results are the mean and sample standard deviation across five seeds, with ten evaluation episodes per seed.

Blend coefficient a	Return	Change vs. original QVPO
0	-599 ± 171	-105.44%
0.2	-362 ± 127	-103.29%
0.5	883 ± 608	-91.98%
0.8	4078 ± 1132	-62.96%
0.98	10642 ± 4787	-3.33%
1.0	11166 ± 5076	–

Adaptive versus fixed guidance. Figure 4 compares adaptive guidance with fixed coefficients $w \in \{0.8, 0.98\}$ on HalfCheetah-v4. For both QSM and QVPO, adaptive guidance achieves higher final training returns than the tested fixed coefficients. This indicates that dynamically adjusting the exploration–exploitation trade-off can provide a more flexible alternative than using a single manually selected guidance coefficient throughout training.

Generative composition versus action-space blending. We further compare CFG-Policy with directly interpolating actions sampled from the exploration and exploitation policies. As shown in Table 4, action-space blending consistently underperforms the original QVPO policy, even when using a large exploitation coefficient.

In contrast, CFG-Policy combines generative fields during the sampling trajectory, allowing both policies to influence the denoising process. The results highlight the advantage of composing policies in the generative space rather than interpolating independently generated actions.

5 Conclusion and Limitation

In this paper, we introduced **CFG-Policy** to address exploration–exploitation challenges across online diffusion RL and offline-to-online RL post-training. By composing separate generative policies during diffusion or flow sampling, it preserves one behavioral capability while the other is optimized using the original RL objective. In online RL, a frozen exploration branch sustains behavioral diversity as exploitation improves; in post-training, a frozen pretrained policy retains its acquired capabilities while a trainable branch steers behavior. With QSM and QVPO on 5 MuJoCo tasks, CFG-Policy achieves higher final returns and faster convergence, improving sample efficiency. With Flow-SDE+PPO on 4 RoboTwin manipulation tasks, it improves or matches task success rates. These results support CFG-Policy as a plug-and-play sampling interface for both learning regimes. Nevertheless, the preferred guidance weight w varies substantially across tasks. Although our adaptive strategy reduces the need for manual selection, task-specific tuning can still find weights that outperform it. Developing more effective task-aware guidance adaptation is therefore an important direction for future work.

AI USE STATEMENT

In this work, we used generative AI tools to assist with drafting, reorganizing, and polishing portions of the manuscript. This assistance included improving the clarity and flow of the exposition, suggesting terminology and mathematical notation, and identifying potentially relevant references for the literature discussion. AI-generated text and reference suggestions were treated as draft material, not as independent evidence for the paper's scientific claims. The theoretical results and experimental findings are supported by the derivations and experiments reported in the paper. The authors reviewed and revised the AI-assisted text and take responsibility for the final content of this work, including its claims, citations, and any material produced with the aid of generative AI.

REFERENCES

Roberto Barceló, Cristóbal Alcázar, and Felipe Tobar. Avoiding mode collapse in diffusion models fine-tuned with reinforcement learning. *arXiv preprint arXiv:2410.08315*, 2024.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

Onur Celik, Zechu Li, Denis Blessing, Ge Li, Daniel Palenicek, Jan Peters, Georgia Chalvatzaki, and Gerhard Neumann. Dime: Diffusion-based maximum entropy reinforcement learning. *arXiv preprint arXiv:2502.02316*, 2025.

Akshay L Chandra, Iman Nematollahi, Chenguang Huang, Tim Welschehold, Wolfram Burgard, and Abhinav Valada. Diwa: Diffusion policy adaptation with world models. *arXiv preprint arXiv:2508.03645*, 2025.

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, Quanlu Zhang, Zhaofei Yu, Guoliang Fan, et al. πrl: Online rl fine-tuning for flow-based vision-language-action models. *arXiv preprint arXiv:2510.25889*, 8, 2025a.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025b.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

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

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. *Advances in Neural Information Processing Systems*, 37:53945–53968, 2024.

Shutong Ding, Ke Hu, Shan Zhong, Haoyang Luo, Weinan Zhang, Jingya Wang, Jun Wang, and Ye Shi. Genpo: Generative diffusion models meet on-policy reinforcement learning. *Advances in neural information processing systems*, 38:130443–130474, 2026a.

Shutong Ding, Zejia Zhong, Zhongyi Wang, Ke Hu, Bikang Pan, Jingya Wang, and Ye Shi. Sample-efficient diffusion-based reinforcement learning with critic guidance. *arXiv preprint arXiv:2605.30056*, 2026b.

Xiaoyi Dong, Jian Cheng, and Xi Sheryl Zhang. Maximum entropy reinforcement learning with diffusion policy, 2025. URL https://arxiv.org/abs/2502.11612.

Yilun Du, Conor Durkan, Robin Strudel, Joshua B. Tenenbaum, Sander Dieleman, Rob Fergus, Jascha Sohl-Dickstein, Arnaud Doucet, and Will Grathwohl. Reduce, reuse, recycle: Compositional generation with energy-based diffusion models and mcmc, 2023. URL https://arxiv.org/abs/2302.11552.

Kevin Frans, Seohong Park, Pieter Abbeel, and Sergey Levine. Diffusion guidance is a controllable policy improvement operator. *arXiv preprint arXiv:2505.23458*, 2025.

Philippe Hansen-Estruch, Ilya Kostrikov, Michael Janner, Jakub Grudzien Kuba, and Sergey Levine. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*, 2023.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Ke Hu, Shutong Ding, Panxin Tao, Jingya Wang, and Ye Shi. Genpo++: Generative policy optimization with jacobian-free likelihood ratios. *arXiv preprint arXiv:2606.06967*, 2026.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $_pi0.5$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Residual reinforcement learning for robot control. In *2019 international conference on robotics and automation (ICRA)*, pp. 6023–6029. IEEE, 2019.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Nan Liu, Shuang Li, Yilun Du, Antonio Torralba, and Joshua B. Tenenbaum. Compositional visual generation with composable diffusion models, 2022. URL https://arxiv.org/abs/2206.01714.

Zhaoyang Liu, Mokai Pan, Zhongyi Wang, Kaizhen Zhu, Haotao Lu, Haipeng Zhang, Jingya Wang, and Ye Shi. Sample from what you see: Visuomotor policy learning via diffusion bridge with observation-embedded stochastic differential equation. *arXiv preprint arXiv:2512.07212*, 2025.

Mingyang Lyu, Yinqian Sun, Erliang Lin, Huangrui Li, Ruolin Chen, Feifei Zhao, and Yi Zeng. Reinforcement fine-tuning of flow-matching policies for vision-language-action models, 2025. URL https://arxiv.org/abs/2510.09976.

Haitong Ma, Tianyi Chen, Kai Wang, Na Li, and Bo Dai. Efficient online reinforcement learning for diffusion policy. *arXiv preprint arXiv:2502.00361*, 2025.

Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. *arXiv preprint arXiv:2502.02538*, 2025.

Michael Psenka, Alejandro Escontrela, Pieter Abbeel, and Yi Ma. Learning a diffusion model policy from rewards via q-score matching. *arXiv preprint arXiv:2312.11752*, 2023.

Allen Ren, Justin Lidard, Lars Ankile, Anthony Simeonov, Pulkit Agrawal, Anirudha Majumdar, Benjamin Burchfiel, Hongkai Dai, and Max Simchowitz. Diffusion policy policy optimization. In *International Conference on Learning Representations*, volume 2025, pp. 77288–77329, 2025.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022.

Tom Silver, Kelsey Allen, Josh Tenenbaum, and Leslie Kaelbling. Residual policy learning. *arXiv preprint arXiv:1812.06298*, 2018.

Shuhan Tan, Kairan Dou, Yue Zhao, and Philipp Krähenbühl. Interactive post-training for vision-language-action models, 2025. URL https://arxiv.org/abs/2505.17016.

Yinuo Wang, Likun Wang, Yuxuan Jiang, Wenjun Zou, Tong Liu, Xujie Song, Wenxuan Wang, Liming Xiao, Jiang Wu, Jingliang Duan, et al. Diffusion actor-critic with entropy regulator. *Advances in Neural Information Processing Systems*, 37:54183–54204, 2024.

Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

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.

Chengyang Ying, Huayu Chen, Xinning Zhou, Zhongkai Hao, Hang Su, and Jun Zhu. Exploratory diffusion model for unsupervised reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 2251–2281, 2026.

Xiu Yuan, Tongzhou Mu, Stone Tao, Yunhao Fang, Mengke Zhang, and Hao Su. Policy decorator: Model-agnostic online refinement for large policy model. In *The Thirteenth International Conference on Learning Representations*, 2025.

Hongyin Zhang, Shiyuan Zhang, Junxi Jin, Qixin Zeng, Yifan Qiao, Hongchao Lu, and Donglin Wang. Balancing signal and variance: Adaptive offline rl post-training for vla flow models, 2025. URL https://arxiv.org/abs/2509.04063.

Tonghe Zhang, Chao Yu, Sichang Su, and Yu Wang. Reinflow: Fine-tuning flow matching policy with online reinforcement learning. *Advances in Neural Information Processing Systems*, 38: 106282–106319, 2026.

A Environmental Details

We evaluate on nine continuous-control tasks spanning two benchmarks: five MuJoCo locomotion environments and four RoboTwin manipulation environments (Figure 5). The two domains probe complementary aspects of policy learning: online interaction in locomotion and reinforcement-learning post-training in object manipulation.

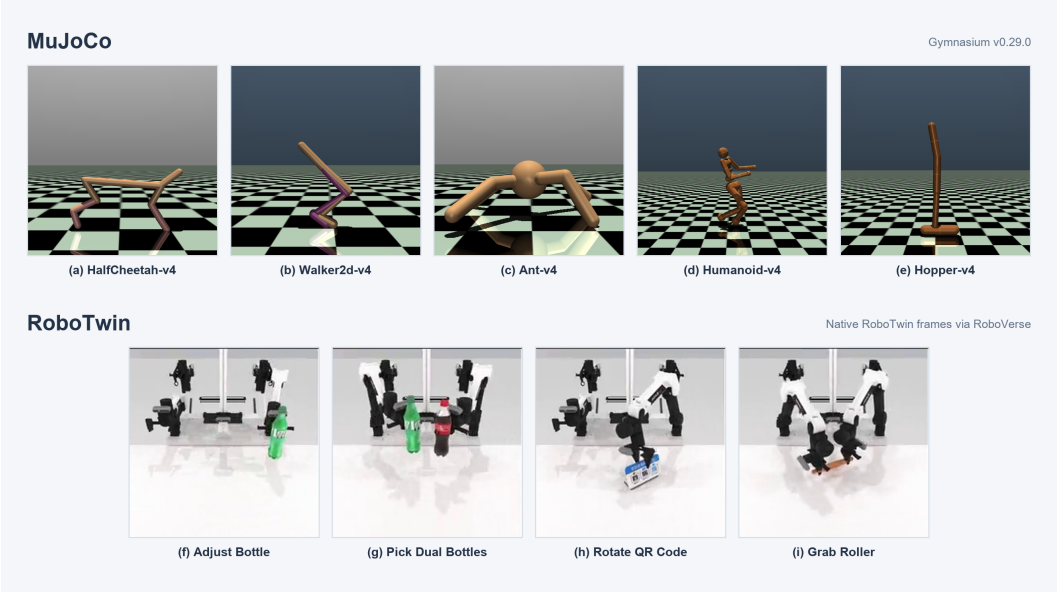


Figure 5: Evaluation environments. (a) HalfCheetah-v4, (b) Walker2d-v4, (c) Ant-v4, (d) Humanoid-v4, and (e) Hopper-v4 are MuJoCo locomotion tasks. (f) Adjust Bottle, (g) Pick Dual Bottles, (h) Rotate QR Code, and (i) Grab Roller are RoboTwin manipulation tasks.

MuJoCo locomotion. We use the `v4` versions of five Gymnasium MuJoCo tasks **?**. Each agent receives a continuous observation and controls its joints through continuous actions. The tasks reward forward movement while imposing task-dependent control and stability requirements:

- **HalfCheetah-v4** ($d_o = 17$, $d_a = 6$): coordinate the front and rear legs of a planar cheetah-like robot to run forward rapidly while limiting control effort.
- **Walker2d-v4** ($d_o = 17$, $d_a = 6$): generate a stable alternating gait for a planar biped and maintain an upright torso.
- **Ant-v4** ($d_o = 27$, $d_a = 8$): coordinate four articulated legs to move a three-dimensional quadruped forward while keeping its body stable.
- **Humanoid-v4** ($d_o = 376$, $d_a = 17$): control a three-dimensional humanoid to move forward without falling, requiring coordinated whole-body motion.
- **Hopper-v4** ($d_o = 11$, $d_a = 3$): propel a planar single-legged robot forward through repeated balanced hops.

Here d_o and d_a denote the default observation and action dimensions, respectively.

RoboTwin manipulation. We select four RoboTwin tasks that cover single-arm object reorientation and coordinated dual-arm grasping Chen et al. (2025b):

- **Adjust Bottle**: select the appropriate arm and pick up a bottle with its cap facing upward.
- **Pick Dual Bottles**: pick up two bottles, using one arm for each bottle.
- **Rotate QR Code**: grasp a QR-code payment sign and rotate it until the code faces the robot.
- **Grab Roller**: use both arms to grasp the same roller on the tabletop.

Evaluation protocol. For online reinforcement learning in MuJoCo, we report the mean episodic return and sample standard deviation over five independent runs. For RoboTwin post-training, we evaluate the optimized policy using the task success rate. These metrics are reported separately because episodic return and manipulation success measure different outcomes.

A.1 Algorithm Details

CFG-Policy for Online Reinforcement Learning. We consider a fixed exploration policy π_E, a trainable exploitation policy π_X, a critic Q, a replay buffer $\mathcal{B}$, an underlying RL optimizer $\mathcal{A}$, and a guidance coefficient w. We first initialize π_X, Q, and $\mathcal{B}$, while keeping the parameters of π_E frozen throughout training. At each environment step, both policies evaluate the same intermediate generative state conditioned on the current observation s_t, and their generative fields are composed as

$$v_w = (1-w)v_E + wv_X.$$

The original sampler is otherwise left unchanged, and the resulting composed policy π_w is used to generate the behavior action a_t. After executing a_t, the transition (s_t, a_t, r_t, s_{t+1}) is stored in $\mathcal{B}$. The optimizer $\mathcal{A}$ then updates the critic and the exploitation policy using its original RL objective, with gradients blocked through π_E. Thus, CFG-Policy only modifies the sampling interface while preserving the optimization rule of the underlying RL algorithm. After training, the learned exploitation policy π_X is returned.

Algorithm 3 CFG-Policy for Online Reinforcement Learning

Require: Frozen exploration policy π_E, trainable exploitation policy π_X, critic Q, replay buffer $\mathcal{B}$, RL optimizer $\mathcal{A}$, guidance coefficient w.

```
Initialize π_X, Q, and B.
Freeze parameters of π_E.
for each environment interaction step do
    Observe state s_t.
    Construct the composed generative field v_w = (1 − w)v_E + wv_X.
    Sample action a_t ∼ π_w(·|s_t) using the original sampler.
    Execute a_t and store (s_t, a_t, r_t, s_{t+1}) in B.
    Update Q and π_X using the original RL objective.
end for
return exploitation policy π_X.
```

CFG-Policy for RL Post-Training. We consider a frozen pretrained exploitation policy π_X^0, a trainable steering policy π_E^θ, an underlying RL optimizer $\mathcal{A}$, and a guidance coefficient w. The base policy π_X^0 is loaded from the pretrained checkpoint and kept fixed throughout RL post-training, while π_E^θ is initialized from the pretrained action-generation module so that the two branches start from compatible generative dynamics. At each RL iteration, both policies evaluate the same intermediate generative state under the same observation context, and their fields are composed as

$$v_w = (1-w)v_X^0 + wv_E^\theta.$$

The resulting composed policy π_w is then used to generate trajectories with the original generative sampler. The optimizer $\mathcal{A}$ computes its original RL objective from the collected trajectories and updates only the steering policy, with gradients blocked through the frozen base policy. In our implementation, we instantiate this procedure with Flow-SDE sampling and PPO with generalized advantage estimation, while the compositional mechanism itself is independent of these choices. After training, inference uses the frozen base policy together with the learned steering policy and the guidance coefficient w.

B Theoretical Analysis of CFG Policy Composition

B.1 CFG Composition as Distribution Multiplication and Path Consistency

We provide the distributional interpretation of CFG composition for score-based diffusion policies. The key distinction is that geometric composition is exact at every fixed noise level, while the corresponding family of densities is not necessarily an exact diffusion marginal path.

Algorithm 4 CFG-Policy for RL Post-Training

Require: Frozen pretrained policy π_X^0, trainable steering policy π_E^θ, RL optimizer $\mathcal{A}$, guidance coefficient w.

1: Freeze parameters of π_X^0.
2: Initialize π_E^θ from the pretrained action-generation module.
3: **for** each RL iteration **do**
4: Construct the composed generative field $v_w = (1-w)v_X^0 + wv_E^\theta$.
5: Generate trajectories using the composed policy π_w.
6: Compute the original RL objective $\mathcal{L}_{\mathrm{RL}}(\pi_w)$.
7: Stop gradients through π_X^0.
8: Update only π_E^θ.
9: **end for**
10: **return** composed policy $(\pi_X^0, \pi_E^\theta, w)$.

Fixed-noise-level composition. For a fixed state s, let $p_{A,t}(x_t \mid s)$ and $p_{B,t}(x_t \mid s)$ denote the noisy marginals of two diffusion policies at noise level t. Their scores are

$$s_{A,t}(x_t, s) = \nabla_{x_t} \log p_{A,t}(x_t \mid s), \qquad s_{B,t}(x_t, s) = \nabla_{x_t} \log p_{B,t}(x_t \mid s). \tag{13}$$

CFG-Policy composes the two scores as

$$s_{w,t}(x_t, s) = (1-w)s_{A,t}(x_t, s) + ws_{B,t}(x_t, s). \tag{14}$$

Define the geometric composition

$$q_{w,t}(x_t \mid s) = \frac{p_{A,t}(x_t \mid s)^{1-w} p_{B,t}(x_t \mid s)^w}{Z_{w,t}(s)}, \tag{15}$$

where

$$Z_{w,t}(s) = \int p_{A,t}(x_t \mid s)^{1-w} p_{B,t}(x_t \mid s)^w \, dx_t. \tag{16}$$

Since the normalization term is independent of x_t,

$$\nabla_{x_t} \log q_{w,t}(x_t \mid s) = (1-w)\nabla_{x_t} \log p_{A,t}(x_t \mid s) + w\nabla_{x_t} \log p_{B,t}(x_t \mid s) \tag{17}$$

$$= s_{w,t}(x_t, s). \tag{18}$$

Therefore, the CFG-composed score is exactly the score of the geometric composition $q_{w,t}$ at every fixed noise level.

Why fixed-level composition does not imply an exact endpoint product? The above identity does not imply that the family $\{q_{w,t}\}_{t=0}^T$ is itself generated by applying the original forward diffusion to the endpoint density $q_{w,0}$.

For the VP diffusion used in our score-based implementations, the forward process can be written as

$$x_t = \alpha_t x_0 + \sigma_t \epsilon, \qquad \epsilon \sim \mathcal{N}(0, I). \tag{19}$$

Let $\mathcal{T}_t$ denote the corresponding forward noising operator,

$$\mathcal{T}_t[p](x_t) = \int \mathcal{N}(x_t; \alpha_t x_0, \sigma_t^2 I) p(x_0) \, dx_0. \tag{20}$$

Then

$$p_{A,t} = \mathcal{T}_t[p_A], \qquad p_{B,t} = \mathcal{T}_t[p_B], \tag{21}$$

and hence

$$q_{w,t} \propto (\mathcal{T}_t[p_A])^{1-w} \, (\mathcal{T}_t[p_B])^w \, . \tag{22}$$

In contrast, directly diffusing the endpoint geometric composition gives

$$\bar{q}_{w,t} = \mathcal{T}_t[q_{w,0}] \propto \mathcal{T}_t \left[p_A^{1-w} p_B^w\right] . \tag{23}$$

In general,

$$\mathcal{T}_t\left[p_A^{1-w}p_B^w\right] \not\propto \left(\mathcal{T}_t[p_A]\right)^{1-w}\left(\mathcal{T}_t[p_B]\right)^w. \tag{24}$$

Thus, forward diffusion and geometric composition do not generally commute. Consequently, the exact fixed-noise-level identity in equation 18 does not by itself imply an exact product form for the endpoint distribution generated by the composed reverse sampler.

Path-consistency residual. We next characterize the discrepancy locally in diffusion time. Consider the shared forward SDE

$$dX_t = f_t(X_t)\,dt + g_t\,dW_t, \tag{25}$$

with Fokker–Planck operator

$$\mathcal{L}_t^* p = -\nabla_x \cdot (f_t p) + \frac{g_t^2}{2}\Delta_x p. \tag{26}$$

Both component marginals satisfy

$$\partial_t p_{A,t} = \mathcal{L}_t^* p_{A,t}, \qquad \partial_t p_{B,t} = \mathcal{L}_t^* p_{B,t}. \tag{27}$$

For the geometric composition in equation 15, direct differentiation gives

$$\partial_t q_{w,t} = \mathcal{L}_t^* q_{w,t} + q_{w,t}\left(R_{w,t} - \mathbb{E}_{q_{w,t}}[R_{w,t}]\right), \tag{28}$$

where

$$R_{w,t}(x_t, s) = \frac{g_t^2}{2} w(1-w)\left\|s_{B,t}(x_t, s) - s_{A,t}(x_t, s)\right\|_2^2. \tag{29}$$

Equation equation 28 explicitly quantifies the path-consistency gap. In particular, only the state-dependent variation of $R_{w,t}$ affects the diffusion path: if $R_{w,t}$ is constant in x_t, the residual is canceled by the normalization term $\mathbb{E}_{q_{w,t}}[R_{w,t}]$.

We therefore define the accumulated path-consistency error

$$\epsilon_{\mathrm{pc}}(s) = \int_0^T \mathbb{E}_{x_t \sim q_{w,t}(\cdot|s)}\left[\left|R_{w,t}(x_t, s) - \mathbb{E}_{q_{w,t}}[R_{w,t}]\right|\right] dt. \tag{30}$$

When ϵ_{pc} is small, the geometric family $\{q_{w,t}\}$ approximately follows the same diffusion dynamics as the two component policies.

Endpoint interpretation. At sufficiently large noise levels, the two component marginals approach the same diffusion prior, so that $q_{w,T}$ is also close to the sampler initialization distribution. Under this terminal-matching condition and small path-consistency error, the endpoint distribution π_w^{sam} induced by the composed reverse sampler is therefore well approximated by

$$\pi_w^{\mathrm{sam}}(a \mid s) \approx q_{w,0}(a \mid s) \propto \pi_A(a \mid s)^{1-w}\pi_B(a \mid s)^w. \tag{31}$$

Hence, distribution multiplication should be interpreted as an exact fixed-noise-level property of CFG and an approximate endpoint interpretation of the complete diffusion sampling process.

B.2 Adaptive Composition

Pathwise divergence estimator. The adaptive guidance objective in Section 3.4 constrains the composed sampler to maintain sufficient deviation from the pure exploitation branch. We measure this deviation along trajectories generated by the composed policy.

For a state s, let $x_t^{(r)}$ denote the intermediate sample at denoising step t along the r-th trajectory generated using guidance weight w. At the same intermediate sample $x_t^{(r)}$, we evaluate the reverse-transition means of the composed sampler and the pure exploitation sampler, denoted by $\mu_t^w(x_t^{(r)}, s)$ and $\mu_t^1(x_t^{(r)}, s)$, respectively. When the two reverse transitions share variance $\sigma_t^2 I$, their conditional Gaussian KL is

$$D_t^{(r)}(w;s) = \frac{\left\| \mu_t^w(x_t^{(r)}, s) - \mu_t^1(x_t^{(r)}, s) \right\|_2^2}{2\sigma_t^2}. \tag{32}$$

In practice, we estimate the pathwise discrepancy as

$$\widehat{D}_{\text{path}}(w) = \text{median}_{s\in\mathcal{S}} \left[\frac{1}{d_a R} \sum_{r=1}^{R} \sum_{t=0}^{T-1} \frac{\left\| \mu_t^w(x_t^{(r)}, s) - \mu_t^1(x_t^{(r)}, s) \right\|_2^2}{2\sigma_t^2} \right], \tag{33}$$

where $\mathcal{S}$ is a set of states sampled from the replay buffer, d_a is the action dimension, and R is the number of sampled generative trajectories per state. For QVPO, when multiple candidate actions are sampled for each state, the corresponding estimates are additionally averaged over the candidates.

Importantly, $x_t^{(r)}$ is sampled from the trajectory generated by the composed policy with guidance weight w. Both μ_t^w and μ_t^1 are evaluated at this same $x_t^{(r)}$; we do not independently generate trajectories from the two policies and compare their resulting samples. The reference μ_t^1 corresponds to the $w = 1$ pure exploitation branch. For QSM, it retains the exploitation-strength scaling used by the original sampler.

The final reverse step is deterministic in the actual sampling procedure. For the sole purpose of evaluating Equation equation 33, we assign this step an auxiliary comparison variance. This variance is used only in the discrepancy estimator and does not introduce additional noise into the generated action.

Therefore, $\widehat{D}_{\text{path}}$ should be interpreted as an auxiliary KL-based discrepancy accumulated along the generative trajectory, rather than as the exact endpoint divergence $D_{\text{KL}}(\pi_w \| \pi_1)$.

Adaptive guidance optimization. Using the above discrepancy, we optimize the guidance weight through

$$\max_w J_Q(w) \qquad \text{s.t.} \qquad D_{\text{path}}(w) \geq \delta, \tag{34}$$

where $J_Q(w)$ is the estimated value of actions generated under guidance weight w. The corresponding Lagrangian is

$$\mathcal{L}(w, \lambda) = J_Q(w) + \lambda \left(D_{\text{path}}(w) - \delta \right). \tag{35}$$

We update w using finite-difference estimates and update the dual variable λ according to the constraint violation. In implementation, $D_{\text{path}}(w)$ is replaced by the empirical estimator $\widehat{D}_{\text{path}}(w)$ in Equation equation 33.

Non-degradation under exact optimization. We finally justify the observation stated in Section 3.4. Define the feasible set

$$\mathcal{W}_\delta = \{ w : D_{\text{path}}(w) \geq \delta \}. \tag{36}$$

Let w_0 be the original guidance coefficient and assume $w_0 \in \mathcal{W}_\delta$. Let

$$w^\star = \arg\max_{w \in \mathcal{W}_\delta} J_Q(w). \tag{37}$$

If the constrained optimization is solved globally, then by optimality,

$$J_Q(w^\star) \geq J_Q(w_0). \tag{38}$$

Under the additional assumption that the critic objective equals the true return of the corresponding composed policy,

$$J_Q(w) = J(\pi_w), \tag{39}$$

we obtain

$$J(\pi_{w^\star}) \geq J(\pi_{w_0}). \tag{40}$$

Thus, under these idealized assumptions, the adaptive solution cannot perform worse than the original feasible guidance coefficient.

Calibration of δ. When CFG is enabled, we sample a set of states $\mathcal{S}_{\mathrm{cal}}$ from the replay buffer and evaluate the normalized trajectory divergence at a reference guidance coefficient w_{ref}. We set

$$\delta = \kappa\, \widehat{D}_{\mathrm{path}}(w_{\mathrm{ref}}; \mathcal{S}_{\mathrm{cal}}), \tag{41}$$

where $\widehat{D}_{\mathrm{path}}$ averages over sampled trajectories for each state and takes the median across states. We use $\kappa = 1$ for QVPO; QSM optionally adjusts the target through κ. The resulting δ is fixed after calibration and is not recomputed as the policies evolve.

B.3 Proof of Theorem 1

Proof. For a fixed state s, assume

$$\pi_X(a \mid s) = \frac{\pi_E(a \mid s)^\beta}{Z_\beta(s)}, \qquad \beta > 1. \tag{42}$$

Substituting this relationship into the geometric-composition surrogate,

$$q_w(a \mid s) \propto \pi_E(a \mid s)^{1-w} \pi_X(a \mid s)^w \tag{43}$$

$$\propto \pi_E(a \mid s)^{1-w} \left(\pi_E(a \mid s)^\beta\right)^w \tag{44}$$

$$= \pi_E(a \mid s)^{1+w(\beta-1)}. \tag{45}$$

Define

$$\alpha(w) = 1 + w(\beta - 1). \tag{46}$$

Then

$$q_w(a \mid s) = \frac{\pi_E(a \mid s)^{\alpha(w)}}{Z(\alpha(w), s)}, \tag{47}$$

where

$$Z(\alpha, s) = \int \pi_E(a \mid s)^\alpha \, da. \tag{48}$$

For brevity, we omit the conditioning on s below. The entropy is

$$H(q_w) = -\mathbb{E}_{q_w}\left[\log q_w(a)\right] \tag{49}$$

$$= -\alpha \mathbb{E}_{q_w}\left[\log \pi_E(a)\right] + \log Z(\alpha). \tag{50}$$

Since

$$\frac{\partial \log Z(\alpha)}{\partial \alpha} = \mathbb{E}_{q_w}\left[\log \pi_E(a)\right], \tag{51}$$

and

$$\frac{\partial}{\partial \alpha} \mathbb{E}_{q_w}\left[\log \pi_E(a)\right] = \mathrm{Var}_{q_w}\left[\log \pi_E(a)\right], \tag{52}$$

we obtain

$$\frac{\partial H(q_w)}{\partial \alpha} = -\alpha\, \mathrm{Var}_{q_w}\left[\log \pi_E(a)\right] \leq 0. \tag{53}$$

Moreover,

$$\frac{\partial \alpha(w)}{\partial w} = \beta - 1 > 0. \tag{54}$$

Therefore,

$$\frac{\partial H(q_w)}{\partial w} = \frac{\partial H(q_w)}{\partial \alpha} \frac{\partial \alpha}{\partial w} \leq 0. \tag{55}$$

Hence, the entropy of the geometric-composition surrogate decreases monotonically with the guidance coefficient w, which proves the theorem.

□

B.4 Policy Implementation details

B.4.1 Online RL Policy Implementation

Exploration branch pre-training. We pretrain a separate exploration policy using the intrinsic reward produced by Random Network Distillation (RND). The RND prediction error provides a novelty signal, while the exploration policy generates actions. This pretraining is conducted separately from reward-based training, and its environment steps are excluded from the formal training budget. In the frozen-branch setting, the pretrained exploration policy remains fixed during formal training.

Classifier-free guidance sampling. At each reverse diffusion step, the exploitation and exploration policies predict noise from the same noisy action x_t. We combine their predictions as

$$\epsilon_{\text{cfg}}(x_t, t) = w\, \epsilon_{\text{exploit}}(x_t, t) + (1 - w)\, \epsilon_{\text{explore}}(x_t, t), \tag{56}$$

and use the combined prediction in the original DDPM reverse update. Guidance therefore acts throughout the generative trajectory, rather than interpolating two completed actions.

Guidance coefficient. The coefficient $w \in [0, 1]$ is a global scalar controlling the relative influence of the two branches: larger values favor exploitation, whereas smaller values favor RND-driven exploration. In the adaptive variant, w is updated during training to improve estimated action value while maintaining a target divergence from the pure exploitation branch.

For detailed implementation parameters, see Table 5.

Table 5: Network, optimization, and diffusion settings for Online RL.

Parameter	QVPO	QSM
Policy hidden layers	3×256, Mish	2×512, Mish
Time embedding	32-dim sinusoidal embedding	64-dimensional learnable Fourier embedding with a 2×128 conditioning MLP
Critic	3×256 hidden layers and 25 quantiles	2×512 hidden layers
RND hidden dimensions	$128/128$	$128/128$
RND pre-training steps	100k	100k
Policy learning rate	1×10^{-4}	3×10^{-4} initially, with cosine decay
Critic learning rate	3×10^{-4}	3×10^{-4}
Batch size	256	256
Discount factor γ	0.99	0.99
Target update coefficient τ	0.005	0.005
Diffusion formulation	Discrete-time DDPM with noise prediction	Discrete-time DDPM with noise prediction
Reverse steps	20	5
Noise schedule	Cosine	VP ($b_{\min} = 0.1$, $b_{\max} = 10$)
Sampling noise scale	1.0	1.0

B.4.2 RL POST-TRAINING POLICY IMPLEMENTATION

We initialize the base policy from a task-specific SFT checkpoint and keep it frozen throughout RL post-training. After loading the SFT weights, we copy its action decoder and action projections to create an independently trainable exploration expert. Both experts share the frozen vision–language backbone and its observation features, but maintain separate action-expert parameters. Consequently, they produce the same velocity before RL updates. At each denoising step, we combine their predictions as

$$v_{\mathrm{CFG}} = (1 - w)v_{\mathrm{base}} + w\, v_{\mathrm{exp}}, \tag{57}$$

where w is the guidance scale. Only the exploration expert and value head receive gradient updates.

We use the standard Flow-SDE and clipped PPO training procedure. During action generation, one of the five denoising steps is randomly selected for a stochastic Flow-SDE transition, while the remaining steps use ODE transitions. PPO updates use the recorded transition log probability and GAE advantages. The policy predicts 50 action steps, executes the first 25, and then observes the environment again. Each run trains for 50 rounds and is evaluated every 10 rounds on 64 fixed scenes. We test guidance scales of 0.5, 0.8, and 0.98; the original single-policy Flow-SDE + PPO method is trained separately as the baseline.

For detailed implementation parameters, see Table 6.

Table 6: Network and flow-sampling hyperparameters for CFG post-training.

Parameter	π_0	$\pi_{0.5}$
Network and optimization		
Visual inputs	3 images	3 images
Action dimension	14	14
Trainable actor	copied action expert	copied action expert
Critic input	action features (1024-D)	VLM features (2048-D)
Critic hidden widths	512, 256, 128	512, 256, 128
Actor learning rate	5×10^{-6}	5×10^{-6}
Critic learning rate	10^{-4}	10^{-4}
Global / micro batch size	256 / 8	256 / 8
Flow formulation and sampling		
Action prediction horizon	50	50
Executed action chunk	25	25
Denoising steps	5	5
Flow-SDE noise level	0.5	0.3
Training transitions	1 SDE + 4 ODE	1 SDE + 4 ODE
Evaluation transitions	5 ODE	5 ODE

B.5 COST COMPARISON

The runtime results are summarized in Table 7. Training time is measured over 1,000 environment steps. Evaluation time is averaged over five 1,000-step episodes.

Table 7: Runtime comparison on HalfCheetah-v4.

Algorithm	Policy	Training (s)	GPU memory (MiB)
QSM	Original	2.42	507
QSM	CFG-Policy	2.94	577
QVPO	Original	8.87	831
QVPO	CFG-Policy	9.37	867