GOLDENSTART: Q-GUIDED PRIORS AND ENTROPY CONTROL FOR DISTILLING FLOW POLICIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Flow-matching policies hold great promise for reinforcement learning (RL) by capturing complex, multi-modal action distributions. However, their practical application is often hindered by prohibitive inference latency and ineffective online exploration. Although recent works have employed one-step distillation for fast inference, the structure of the initial noise distribution remains an overlooked factor that presents significant untapped potential. This overlooked factor, along with the challenge of controlling policy stochasticity, constitutes two critical areas for advancing distilled flow-matching policies. To overcome these limitations, we propose GoldenStart (GS-flow), a policy distillation method with Q-guided priors and explicit entropy control. Instead of initializing generation from uninformed noise, we introduce a Q-guided prior modeled by a conditional VAE. This state-conditioned prior repositions the starting points of the one-step generation process into high-Q regions, effectively providing a "golden start" that shortcuts the policy to promising actions. Furthermore, for effective online exploration, we enable our distilled actor to output a stochastic distribution instead of a deterministic point. This is governed by entropy regularization, allowing the policy to shift from pure exploitation to principled exploration. Our integrated framework demonstrates that by designing the generative startpoint and explicitly controlling policy entropy, it is possible to achieve efficient and exploratory policies, bridging the generative models and the practical actor-critic methods. We conduct extensive experiments on offline and online continuous control benchmarks, where our method significantly outperforms prior state-of-the-art approaches.

1 INTRODUCTION

Recent advances in policy learning have increasingly leveraged generative models to capture complex and multimodal policies (Chi et al., 2023; Ghugare & Eysenbach, 2025; Black et al., 2024a). Unlike traditional methods that assume a unimodal Gaussian distribution Schulman et al. (2015; 2017); Haarnoja et al. (2018), these approaches model the rich action distributions required for sophisticated control tasks. However, this expressive power comes at a cost: The iterative nature of the generation process, which requires multiple steps to produce a single action, leads to prohibitive inference latency. This bottleneck makes such models impractical for real-time scenarios, such as Vision-Language-Action (VLA) models (Zhai et al., 2024; Black et al., 2025).

Flow matching has recently emerged as a more efficient alternative to diffusion models (Lipman et al., 2023; Liu et al.; Albergo & Vanden-Eijnden, 2023; Geng et al., 2025). This has spurred research into the acceleration of generative policies using flow matching (Braun et al., 2024; Agrawalla et al., 2025; Espinosa-Dice et al., 2025), although these approaches often still require multiple denoising steps at the inference stage. To address this, a more aggressive solution using one-step distillation proves particularly effective by training a student network to emulate the entire multi-step transformation in a single forward pass (Park et al., 2025b). Although effective in reducing latency, these methods overlook two critical opportunities to improve policies.

First, their generative process begins from a fixed, uninformed prior, typically a standard Gaussian distribution. However, an emerging perspective in generative modeling suggests that initial noise is a critical component that can guide generation (Zhou et al., 2025; Ma et al., 2025b).

We posit that an optimized starting point (a "golden start") can create a powerful learning shortcut to high-value actions. As illustrated in Figure 1, an informed prior (yellow) strategically shifted towards high-value regions provides a more direct path to optimal actions, compared to an uninformed Gaussian distribution (gray). The second opportunity stems from the deterministic mapping inherent in the distilled policies. Given a specific prior noise, the generator learns a "point-to-point" mapping, transforming a single noise vector into a single deterministic action. This architecture inherently lacks explicit control over policy stochasticity, which is crucial for effective online exploration (Ma et al., 2025a).

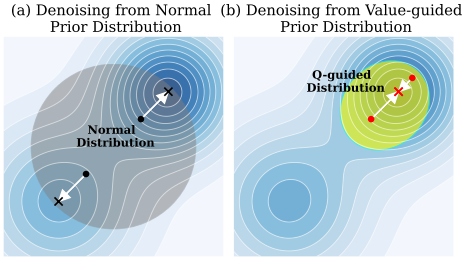


Figure 1: An illustration of denoising from an uninformed Gaussian prior (a) versus an informed, value-guided prior (b). Deeper blue indicates higher value.

To overcome these challenges, we introduce GoldenStart (GS-flow), a novel distillation framework that unifies high-speed inference with precise exploitation and adaptive exploration. Our work is built upon two key innovations: (1) First, we propose a Q-Guided Generative Prior, learned via a lightweight conditional VAE. This prior replaces the uninformative Gaussian noise with a state-aware distribution biased toward high-value actions, as identified by the critic. This provides the "golden start", effectively shortcutting the policy learning to optimal modes with negligible latency overhead. (2) Second, we introduce Entropy-Regularized Distillation, where the student policy learns a full distribution over actions, not just deterministic ones. This transforms the conventional "point-to-point" mapping into a more expressive "point-to-distribution" paradigm. During the online RL stage, an entropy regularization mechanism is activated, allowing the policy to dynamically modulate its stochasticity for robust exploration.

By co-optimizing the generative starting point and the output distribution, our framework improves the policy's ability to represent high-value actions while merging flow-based distillation models with adaptive exploration control. To this end, our approach, GS-flow, is extensively evaluated on continuous control benchmarks, including OGBench and D4RL (Park et al., 2025a; Fu et al., 2020). The results demonstrate that our method establishes a new state-of-the-art in overall performance. It particularly excels on complex tasks requiring multi-modal action representations and principled exploration, where it significantly outperforms prior methods.

2 Preliminary

2.1 Problem Definition

A reinforcement learning problem is formulated as a Markov Decision Process (MDP) (Sutton et al., 1998), defined by the tuple (S, A, P, r, γ). S is the state space, A is the action space, $P : S \times A \times S \to [0, 1]$ is the state transition probability function, $r : S \times A \to \mathbb{R}$ is the reward function, and $\gamma \in [0, 1)$ is the discount factor. A policy $\pi(a|s)$ is a distribution over actions given a state. The objective is to learn an optimal policy π^* that maximizes the expected discounted cumulative reward, $J(\pi) = \mathbb{E}_{\tau\sim\pi}\left[\sum_{t=0} \gamma^t r(s_t, a_t)\right]$, where $\tau = (s_0, a_0, s_1, a_1, \dots)$ is a trajectory sampled by executing the policy π. Offline RL involves learning from a static transition dataset $\mathcal{D} = \{(s_t, a_t, r_t, s_{t+1})\}_{t=1}^{N}$ without environmental interaction, where N is number of steps in the dataset (Levine et al., 2020). The Offline-to-Online RL setting extends this problem by introducing a subsequent online interaction phase, also with the aim of maximizing the return function $J(\pi)$.

2.2 Distillation from Flow-Matching Policy

The significant inference cost of iterative flow-matching policies has motivated researchers to distill them into single-step, fast student policies (Park et al., 2025b). This approach, named FQL, operates within an actor-critic structure and trains the student actor with a hybrid objective: concurrently minimizing a distillation loss against the flow-matching teacher while maximizing the Q value. The framework utilizes two distinct models:

Teacher Policy (π_ϕ): The flow-matching teacher policy is trained on the offline dataset $\mathcal{D}$ using a behavioral cloning (BC) objective. For a given state-action pair (s, a) sampled from the dataset and a noise sample $x_0 \sim \mathcal{N}(0, I)$, the training objective is to learn a conditional velocity field $v_\phi(x_t, s, t)$. This field is parameterized by a time variable $t \in [0, 1]$ and defines a straight path between the noise x_0 and the action a (Lipman et al., 2023; Liu et al.). Assuming t is sampled uniformly from this interval ($t \sim U(0, 1)$), the interpolated action along this path is $x_t = (1-t)x_0 + ta$. The Conditional Flow Matching (CFM) loss then trains the network to match the constant velocity of this path:

$$\mathcal{L}_{\text{CFM}}(\phi) = \mathbb{E}_{t\sim U(0,1),(s,a)\sim\mathcal{D},x_0\sim\mathcal{N}(0,I)} \left[\|v_\phi(x_t, s, t) - (a - x_0)\|^2\right] \tag{1}$$

During inference, the teacher policy π_ϕ generates a final action $a^{teacher}$ by using the trained v_ϕ to iteratively denoise an initial noise sample over multiple steps.

Student Policy (π_φ): The separate student network is trained for fast inference. It takes a state s and a noise vector x_0 as input and produces an action in a single forward pass. The student policy π_φ is trained to concurrently maximize the Q-value while staying close to the teacher's output. This is achieved by minimizing a compound loss function that combines a Q-learning objective with a distillation loss:

$$\mathcal{L}_{\text{Distill}}(\varphi) = \mathbb{E}_{s\sim\mathcal{D},x_0\sim\mathcal{N}(\mathbf{0},\mathbf{I})} \left[-Q(s, \pi_\varphi(s, x_0)) + \alpha\|\pi_\phi(s, x_0) - \pi_\varphi(s, x_0)\|^2\right], \tag{2}$$

where Q is the critic function learned within an actor-critic framework (Haarnoja et al., 2018) and the hyperparameter α controls the strength of the behavioral cloning (BC) regularization (Tarasov et al., 2023). In particular, π_φ requires no iterative denoising at inference, as it is trained to directly approximate the multi-step denoising action in a single step.

2.3 Multi-Crescent Task

To demonstrate our insight, we design the Multi-Crescent environment, shown in Figure 2. The environment consists of six separate, nonconvex, crescent-shaped regions of high reward, designed to challenge agents that are prone to Q-value overestimation. The reward is structured into three levels: the top-left/bottom-right crescents provide a moderate reward, the middle-left/middle-right crescents provide a higher reward, and the globally optimal top-right/bottom-left crescents offer the maximum reward. All other areas yield zero reward. This setup emulates a complex environment with multiple levels of local optima.

When constructing the offline dataset, we deliberately excluded all samples from the two highest-reward crescent regions (top-left/bottom-right), as shown by the blue scatter points in Figure 5a. This environment poses two challenges to the algorithms: 1) During the offline learning phase: The algorithm needs to identify and converge to the higher-reward mode present within the dataset (middle-left/middle-right) while suppressing Q-value overestimation for unseen regions. 2) During the online exploration phase: The algorithm must demonstrate efficient exploration to discover and exploit the globally optimal modes (top-left/bottom-right) that were never present in the initial dataset. This environment allows us to assess whether an algorithm can escape the pull of a suboptimal data distribution to find the globally optimal policy. More details can be found in the Appendix D.

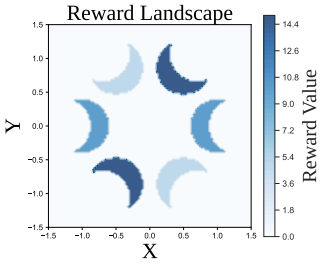


Figure 2: The visualization of the multi-crescent task.

3 Methodology

3.1 Overview of the Algorithm

Our method, GS-flow, is designed to mitigate the two challenges of imprecise exploitation and ineffective exploration common in existing distilled policies through a two-phase training process, as illustrated in Figure 3. The first phase, Q-Guided Prior Learning, focuses on solving the suboptimal starting point problem. Instead of beginning the generation process from a standard, uninformed Gaussian noise, we use the Advantage Noise Selection module to actively identify advantage initial noises, which lead to high-value actions. We then train a conditional Variational Autoencoder

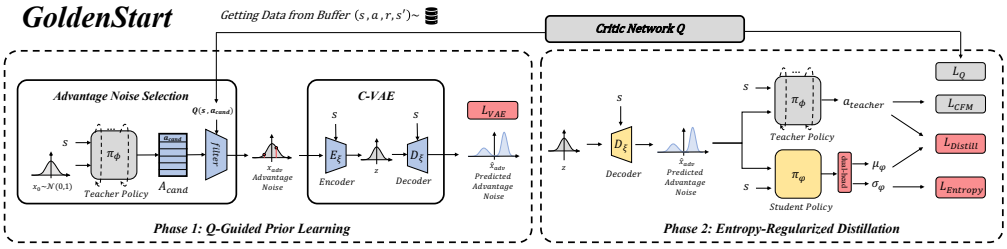


Figure 3: Overview of our algorithm. During training, we first learn a structured prior for the initial noise, which is then used to distill the teacher policy. For online exploration, actions are sampled from the student's entropy-regularized distribution. During evaluation, the deterministic mean of the policy's output is used. The critic update steps are omitted for clarity, detailed in Appendix B.

(CVAE) to model the distribution of these advantage noises, effectively learning an informed, state-conditioned prior. The second phase, Entropy-Regularized Distillation, uses the learned prior to train a highly capable student policy. Both the teacher and student policies are provided with an initial noise sampled from our learned prior. Furthermore, the student model is designed as a stochastic policy and trained with a hybrid objective that combines distillation with an entropy regularization term. This endows the final actor with controllable stochasticity, allowing it to explore intelligently during online fine-tuning. The complete training pipeline, which integrates these two phases with standard actor-critic updates, is detailed in Algorithm 1.

At inference time, GS-flow operates with high efficiency using only the VAE decoder and the student policy, which are highlighted in yellow in Figure 3. Given the current state, the VAE decoder generates an advantage prior. This prior is then fed into the student actor to produce an action distribution. For online exploration, an action is sampled from this distribution with its learned mean and variance. For evaluation, we only use the mean to maximize exploitation.

Algorithm 1 GS-flow

Initialize: Critic Q_θ, VAE (E_{ξ_1}, D_{ξ_2}), Teacher Policy π_ϕ, Student Policy π_φ.
for each training step **do**
  Sample a batch $\{(s, a, r, s')\}$ from dataset $\mathcal{D}$.
  # — 1. Update Critic —
  Update critic parameters θ using Temporal Difference (TD) learning.
  # — 2. Update Prior Learning Network —
  For each state s, generate N_{cand} candidate actions $A_{\text{cand}} = \{a_j\}_{j=1}^{N_{\text{cand}}}$ using π_ϕ.
  Find the prior noise x_{adv} corresponding to the highest-Q action (Eq. 4).
  Update VAE parameters ξ_1, ξ_2 by minimizing the CVAE loss (Eq. 5).
  # — 3. Update Student Policy —
  Update teacher policy parameters ϕ using the flow matching loss (Eq. 1).
  Generate a sampled prior for the current state: $\hat{x}_{\text{adv}} \leftarrow D_{\xi_2}(s, \mathcal{N}(\mathbf{0}, \mathbf{I}))$.
  Generate the teacher's target action: $a_{\text{teacher}} \leftarrow \pi_\phi(s, \hat{x}_{\text{adv}})$.
  Update student policy parameters φ by minimizing the actor loss $\mathcal{L}_{\text{Actor}}$ (Eq. 9).
end for
return Trained student policy π_φ.

3.2 Q-Guided Prior Learning

To realize our first insight of initiating the denoising process from golden starting points, we propose learning a Q-Guided Prior to model the distribution of what we named "advantage noises", denoted as x_{adv}. For this purpose, we employ a conditional Variational Autoencoder (CVAE) due to its flexibility in learning arbitrary multi-modal distributions. To achieve this, we first need to construct samples $\mathcal{B}_{\text{adv}}$ of these advantage noises for model training.

Advantage Noise Selection. We introduce a data collection module named Advantage Noise Selection, shown in Phase 1 of Figure 3. Given the state s, we first collect a set of N_{cand} candidate

actions, denoted as $a_{\text{cand}} \in A_{\text{cand}}$, generated by the teacher policy π_ϕ with N_{cand} different initial noises x_0, which is sampled from a normal distribution:

$$A_{\text{cand}} = \{a_j = \pi_\phi(s, x_j) \mid x_j \sim \mathcal{N}(\mathbf{0}, \mathbf{I})\}_{j=1}^{N_{\text{cand}}}. \tag{3}$$

Although these candidate actions are all feasible behaviors learned from the dataset, they are not necessarily optimal. To identify the most promising starting point, we leverage the critic Q to evaluate all candidate actions. The initial noise that generates the action with the highest Q-value is designated as the advantage noise for s:

$$x_{\text{adv}}(s) = \arg\max_{x_j} Q(s, \pi_\phi(s, x_j)). \tag{4}$$

This selection process is applied on-the-fly within each training step, using the most up-to-date teacher policy to generate a new batch of pairings, $\mathcal{B}_{\text{adv}} = \{(s, x_{\text{adv}}(s))\}$. This batch then serves as the target distribution for the CVAE update.

State Conditional VAE. With the data collected before, we then train a Conditional Variational Autoencoder (CVAE) (Kingma & Welling, 2013) to model the state-conditioned distribution $p_{\xi_2}(x_{\text{adv}}|s)$. The CVAE consists of a conditional encoder $E_{\xi_1}(x, s)$ and a conditional decoder $D_{\xi_2}(z, s)$, where z is the latent vector. The encoder maps a prior-state pair to a latent distribution, while the decoder reconstructs the prior from a latent sample. The model is trained by minimizing the weighted sum of a reconstruction loss and a KL-divergence regularization term:

$$\mathcal{L}_{\text{VAE}}(\xi_1, \xi_2) = \mathcal{L}_{\text{recon}} + \lambda_{\text{KL}} \mathcal{L}_{\text{KL}}, \tag{5}$$

where λ_{KL} is the scalar weight. The KL-divergence term $\mathcal{L}_{\text{KL}}$ regularizes the latent space by encouraging the encoded distribution to be close to a standard normal distribution $\mathcal{N}(0, I)$:

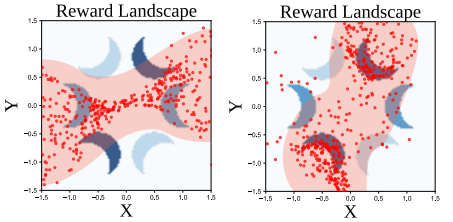
$$\mathcal{L}_{\text{KL}} = \mathbb{E}_{(s, x_{\text{adv}}) \sim \mathcal{B}_{\text{adv}}} \left[D_{KL} \left(q_{\xi_1}(z \mid x_{\text{adv}}, s) \parallel \mathcal{N}(0, I) \right) \right], \tag{6}$$

where q_{ξ_1} is the approximate posterior distribution, a diagonal Gaussian parameterized by the encoder E_{ξ_1}: $q_{\xi_1}(z \mid x_{\text{adv}}, s) = \mathcal{N}\left(\mu_{\xi_1}(x_{\text{adv}}, s), \Sigma_{\xi_1}(x_{\text{adv}}, s)\right)$. Assuming $\hat{x}_{\text{adv}}$ denotes the prior predicted by $D_{\xi_2}(z, s)$, the loss of reconstruction $\mathcal{L}_{\text{recon}}$ can be calculated as follows:

$$\mathcal{L}_{\text{recon}} = \mathbb{E}_{(s, x_{\text{adv}}) \sim \mathcal{B}_{\text{adv}}, z \sim q_{\xi_1}(z | x_{\text{adv}}, s)} \left[\| \hat{x}_{\text{adv}} - x_{\text{adv}} \|^2 \right]. \tag{7}$$

Notably, CVAE is capable of approximating an arbitrarily potentially multimodal prior distribution, offering an advantage over methods that learn a Gaussian distribution.

Validation. We validate the effectiveness of our Q-guided prior in the MultiCrescent environment. Figure 4 visualizes the distribution generated by the VAE decoder during inference. The red points represent $\hat{x}_{\text{adv}}$, and the red region generated via KDE (Silverman, 2018) represents the predicted prior distribution. After the offline phase (left panel), the prior captures the high-value modes (middle-left/middle-right) within the static dataset. After online fine-tuning (right panel), the prior adapts its density to focus on the newly discovered, globally optimal action modes (top-left/bottom-right). This demonstrates that our learned prior captures the distribution of advantage noises. Furthermore, Figure 5c shows that the actions generated from $\hat{x}_{\text{adv}}$ yield higher Q values compared to the baseline shown in Figure 5b.

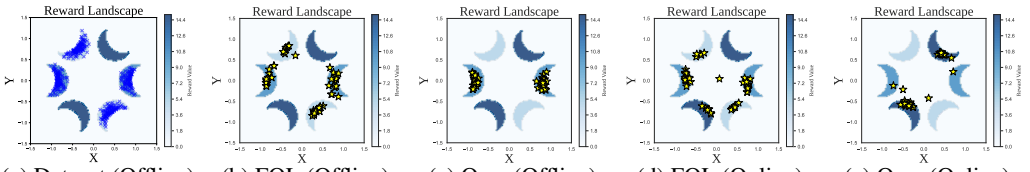


(a) Learned Prior (Offline) (b) Learned Prior (Online)

Figure 4: Visualization of the learned prior distribution after different training stages.

3.3 Entropy-Regularized Distillation

Previous flow-matching policy distillation methods produce a deterministic actor. Although efficient for exploitation, it is ill-suited for online exploration due to its lack of inherent stochasticity. This can be viewed as a point-to-point generation process, where a starting noise is mapped to a single target action. Inspired by recent approaches that augment generative models with distributional models (Dong et al., 2025), we propose an entropy-regularized distillation method. This transforms the



(a) Dataset (Offline) (b) FQL (Offline) (c) Ours (Offline) (d) FQL (Online) (e) Ours (Online)

Figure 5: Results on the multi-crescent task. Blue crosses denote samples from the offline dataset, while yellow stars represent the actions produced by the policies. **(a):** shows the offline dataset, which excludes the two globally optimal modes. **(b, c):** shows action distributions after the offline phase. Our method captures the higher-value modes within the dataset, while the baseline shows a less focused distribution. **(d, e):** shows action distributions after the online fine-tuning phase. Our method quickly discovers and converges to both highest-reward modes. In contrast, the baseline only finds one. More results on this task can be found in Appendix 8.

distillation from a point-to-point mapping into a point-to-adaptive-distribution process, providing the agent with a principled method for balancing the exploration-exploitation trade-off.

To achieve this, we parameterize the student policy $\pi_\varphi(a|s, \hat{x}_{\text{adv}})$ as a Gaussian distribution using a dual-headed architecture that outputs both a mean $\mu_\varphi(s, \hat{x}_{\text{adv}})$ and a standard deviation $\sigma_\varphi(s, \hat{x}_{\text{adv}})$. The action a_φ for exploration is computed as:

$$a_\varphi(s, \hat{x}_{\text{adv}}, \epsilon) = \mu_\varphi(s, \hat{x}_{\text{adv}}) + \sigma_\varphi(s, \hat{x}_{\text{adv}}) \odot \epsilon, \quad \text{where} \quad \epsilon \sim \mathcal{N}(0, I). \tag{8}$$

The actor policy is trained by minimizing a composite objective that balances three key components: imitation of the teacher, value maximization, and entropy regularization. The training objective for our entropy-regularized actor is a composite loss function designed to balance three key objectives: (1) imitating the high-quality teacher policy, (2) maximizing expected return according to the critic, and (3) maintaining sufficient policy entropy to encourage exploration. Therefore, with the advantage noise $\hat{x}_{\text{adv}} = D_{\xi_2}(z, s)$, the total actor loss is defined as follows:

$$\mathcal{L}_{\text{Actor}} = \mathbb{E}_{z\sim\mathcal{N}(0,I), s\sim\mathcal{D}} \left[\alpha_1 \mathcal{L}_{\text{Distill}} + \mathcal{L}_{\text{Q}} - \alpha_2 \mathcal{H}(\pi_\varphi(\cdot|s, \hat{x}_{\text{adv}}))\right]. \tag{9}$$

The distillation term $\mathcal{L}_{\text{Distill}}$ anchors the mean behavior of the student policy to the high-quality teacher actions, and α_1 is the scalar weight to control BC behavior. Two details are critical to ensure that this process has a low-variance and stable training signal. First, both teacher and student policies are conditioned on identical advantage noise $\hat{x}_{\text{adv}}$. Second, the loss is computed using only the student's deterministic mean $\mu_\varphi(s, \hat{x}_{\text{adv}})$ rather than a stochastic sample. These design choices reduce the variance of the loss signal and improve training stability. The loss is defined as follows:

$$\mathcal{L}_{\text{Distill}} = \|\mu_\varphi(s, \hat{x}_{\text{adv}}) - a_{\text{teacher}} = \pi_\phi(s, \hat{x}_{\text{adv}})\|^2. \tag{10}$$

The value maximization term $\mathcal{L}_{\text{Q}}$ encourages the policy to seek actions that the critic evaluates as having a high value (Fujimoto & Gu, 2021). Following the standard approach in Soft Actor-Critic (SAC) (Haarnoja et al., 2018), we use a sampled action a_φ from the policy: $a_\varphi \sim \pi_\varphi(\cdot|s, \hat{x}_{\text{adv}})$. The loss is then calculated as its negative Q-value:

$$\mathcal{L}_{\text{Q}} = -Q(s, a_\varphi). \tag{11}$$

The third entropy bonus term $\mathcal{H}(\pi_\varphi(\cdot|s, \hat{x}_{\text{adv}}))$ is the entropy of the policy under $\hat{x}_{\text{adv}}$. To automate the trade-off between reward and entropy, the temperature parameter α_2 is learned by minimizing a separate loss function that aims to match the entropy to a predefined target entropy $\mathcal{H}_{\text{target}}$. This allows the agent to dynamically adjust its stochasticity, exploring more when the entropy is below the target and exploiting more when it is sufficient. These two components are calculated as follows:

$$\mathcal{H}(\pi_\varphi(\cdot|s, \hat{x}_{\text{adv}})) = -\mathbb{E}_{a_\varphi\sim\pi_\varphi} \left[\log \pi_\varphi(a_\varphi|s, \hat{x}_{\text{adv}})\right], \tag{12}$$

$$\mathcal{L}_{\alpha_2} = \mathbb{E}_{s\sim\mathcal{D}} \left[\alpha_2(\mathcal{H}(\pi_\varphi(\cdot|s, \hat{x}_{\text{adv}}) - \mathcal{H}_{\text{target}}))\right]. \tag{13}$$

Validation. The online fine-tuning results, shown in Figures 5d and 5e, highlight the superiority of our exploration mechanism. Our method effectively explores the action space and identifies both of the highest Q-value peaks (top-left/bottom-right). In contrast, the baseline lacks an exploration strategy and finds just one of the peaks. Even better, our method finds both modes using significantly fewer samples. This demonstrates the clear efficiency of its entropy-regularized exploration.

Table 1: Offline performance on OGBench and D4RL benchmarks, averaged over 5 seeds (3 for Visual Environments due to computational cost). Best results are in **bold**. The performance of baseline methods is reported from Park et al. (2025b).

Task	Gaussian Policies			Diffusion Policies			Flow Policies				
	BC	IQL	ReBRAC	IDQL	SRPO	CAC	FAWAC	FBRAC	IFQL	FQL	Ours
OGBench											
AntMaze Large Navigate	0 ± 0	48 ± 9	91 ± 10	0 ± 0	0 ± 0	42 ± 7	1 ± 1	70 ± 20	24 ± 17	80 ± 8	**88.4** ± 2.7
AntMaze Giant Navigate	0 ± 0	0 ± 0	**27** ± 22	0 ± 0	0 ± 0	0 ± 0	0 ± 0	0 ± 1	0 ± 0	4 ± 5	10.4 ± 5.9
HumanoidMaze Medium	1 ± 0	32 ± 7	16 ± 9	1 ± 1	0 ± 0	38 ± 19	6 ± 2	25 ± 8	**69** ± 19	19 ± 12	45.0 ± 19.7
HumanoidMaze Large	0 ± 0	3 ± 1	2 ± 1	0 ± 0	0 ± 0	1 ± 1	0 ± 0	0 ± 1	6 ± 2	**7** ± 6	4.6 ± 4.4
AntSoccer Arena	1 ± 0	3 ± 2	0 ± 0	0 ± 1	0 ± 0	0 ± 0	12 ± 3	24 ± 4	16 ± 9	39 ± 6	**46.0** ± 10.5
Cube Single Play	3 ± 1	85 ± 8	92 ± 4	96 ± 2	82 ± 16	80 ± 30	81 ± 9	83 ± 13	73 ± 3	**97** ± 2	95.6 ± 4.1
Cube Double Play	0 ± 0	1 ± 1	7 ± 3	16 ± 10	0 ± 0	2 ± 2	2 ± 1	22 ± 12	9 ± 5	36 ± 6	**51.3** ± 6.2
Scene Play	1 ± 1	12 ± 3	50 ± 13	33 ± 14	2 ± 2	50 ± 40	18 ± 8	46 ± 10	0 ± 0	76 ± 9	**88.0** ± 8.6
Puzzle-3x3 Play	1 ± 1	2 ± 1	2 ± 1	0 ± 0	0 ± 0	0 ± 0	1 ± 1	2 ± 2	0 ± 0	16 ± 5	**25.2** ± 10.7
Puzzle-4x4 Play	0 ± 0	5 ± 2	10 ± 3	**26** ± 6	7 ± 4	1 ± 1	0 ± 0	5 ± 1	21 ± 11	11 ± 3	16.7 ± 4.1
Average	0.7	19.1	29.7	17.2	9.1	21.4	12.1	27.7	21.8	38.5	**47.1**
D4RL AntMaze											
AntMaze U-Maze	55	77	98	94	97	66 ± 5	90 ± 6	94 ± 3	92 ± 6	96 ± 2	**99.6** ± 0.8
AntMaze U-Maze Diverse	47	54	84	80	82	66 ± 11	55 ± 7	82 ± 9	62 ± 12	89 ± 5	**93.2** ± 7.1
AntMaze Medium Play	0	66	**90**	84	81	49 ± 24	52 ± 12	77 ± 7	56 ± 15	78 ± 7	77.2 ± 9.0
AntMaze Medium Diverse	1	74	84	**85**	75	0 ± 1	44 ± 15	77 ± 6	60 ± 25	71 ± 13	75.5 ± 11.0
AntMaze Large Play	0	42	52	64	54	0 ± 0	10 ± 6	32 ± 21	55 ± 9	84 ± 7	**86.5** ± 5.2
AntMaze Large Diverse	0	30	64	68	54	0 ± 0	16 ± 10	20 ± 17	64 ± 8	83 ± 4	**84.8** ± 5.2
Average	17.2	57.2	78.7	79.2	73.8	30.2	44.5	63.7	64.8	83.5	**86.1**
Visual Environments											
Visual Cube Single Play	–	70 ± 12	83 ± 6	–	–	–	–	55 ± 8	49 ± 7	81 ± 12	**92.7** ± 4.1
Visual Cube Double Play	–	34 ± 23	4 ± 4	–	–	–	–	6 ± 2	8 ± 6	21 ± 11	**42.0** ± 11.8
Visual Scene Play	–	97 ± 2	98 ± 4	–	–	–	–	46 ± 4	86 ± 10	98 ± 3	**100.0** ± 0.0
Visual Puzzle-3x3	–	7 ± 15	88 ± 4	–	–	–	–	7 ± 2	**100** ± 0	94 ± 1	88.67 ± 9.0
Visual Puzzle-4x4	–	0 ± 0	26 ± 6	–	–	–	–	0 ± 0	8 ± 15	**33** ± 6	31.00 ± 7.1
Average	–	41.6	59.8	–	–	–	–	22.8	50.2	65.4	**70.9**

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

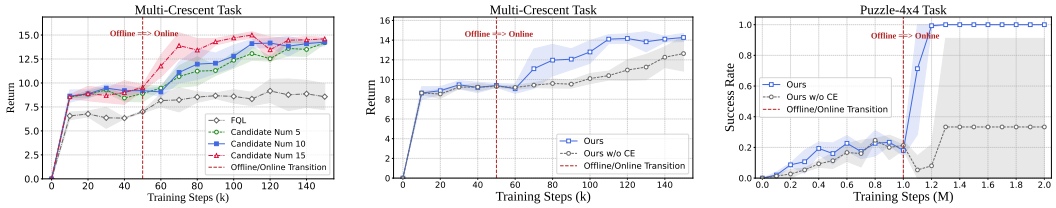
We test our method across the OGBench (Park et al., 2025a) tasks, the standard D4RL AntMaze (Fu et al., 2020) tasks, and a set of challenging Visual Environments (Park et al., 2025a). The baselines range from standard Gaussian policies (BC, IQL, ReBRAC) (Kostrikov et al., 2022; Tarasov et al., 2023), to more expressive Diffusion Policies (IDQL, SRPO, CAC) (Hansen-Estruch et al., 2023; Chen et al., 2024; Ding & Jin, 2024), and finally to Flow Policies (FAWAC, FBRAC, IFQL) (Nair et al., 2021; Wang et al., 2023; Park et al., 2025b). And state-of-the-art flow-matching distillation model FQL (Park et al., 2025b). For offline-to-online, we also compared with Cal-QL and RLPD (Nakamoto et al., 2023; Ball et al., 2023). We validate our model in both offline and offline-to-online fine-tuning settings to demonstrate the effectiveness of our two contributions. More details on environments and baselines are shown in Appendices E and F.

4.2 RESULTS AND ANALYSIS

The Impact of the Learned Prior on Offline Performance. The results in Table 1 demonstrate the effectiveness of our proposed method, GS-flow. GS-flowachieves new state-of-the-art performance on average, outperforming all baselines. This advantage is particularly pronounced on several tasks with multi-modal action spaces, where our method shows significant gains over the strong FQL baseline. According to the results in OGBench, the strength can be illustrated by the contrasting results on the Cube tasks. On Cube Single Play, a task with a relatively unimodal optimal policy, GS-flow performs comparably to the strong FQL baseline. In contrast, on the more complex Cube Double Play, which requires coordinating two objects and therefore presents a significantly more multimodal Q landscape, the offline score of GS-flow (51.3%) dramatically outperforms all competing methods. The contrast underscores our algorithm's specialized capability in multi-modal challenges. The advantage is further substantiated in other complex manipulation tasks such as Puzzle-3x3 and Puzzle-4x4, and in challenging locomotion environments such as HumanoidMaze Medium Navigate, where GS-flowachieves more than double the score of FQL. Furthermore, GS-flowachieves the highest average scores on the remaining two benchmarks, D4RL AntMaze and Visual Environments, demonstrating the effectiveness of our algorithm in the offline setting.

Table 2: Offline-to-online performance comparison. Similar to Table 1, we report the results over 5 seeds. The best online results are highlighted in **bold**.

Task	IQL	ReBRAC	Cal-QL	RLPD	IFQL	FQL	Ours
HumanoidMaze Medium	$21 \pm 13 \rightarrow 16 \pm 8$	$16 \pm 20 \rightarrow 1 \pm 1$	$0 \pm 0 \rightarrow 0 \pm 0$	$0 \pm 0 \rightarrow 8 \pm 10$	$56 \pm 35 \rightarrow \mathbf{82} \pm 20$	$12 \pm 7 \rightarrow 22 \pm 12$	$45 \pm 20 \rightarrow 67 \pm 6$
AntSoccer Arena	$2 \pm 1 \rightarrow 0 \pm 0$	$0 \pm 0 \rightarrow 0 \pm 0$	$0 \pm 0 \rightarrow 0 \pm 0$	$0 \pm 0 \rightarrow 0 \pm 0$	$26 \pm 15 \rightarrow 39 \pm 10$	$28 \pm 8 \rightarrow \mathbf{86} \pm 5$	$46 \pm 10 \rightarrow 77 \pm 9$
Cube Double Play	$0 \pm 1 \rightarrow 0 \pm 0$	$6 \pm 5 \rightarrow 28 \pm 28$	$0 \pm 0 \rightarrow 0 \pm 0$	$0 \pm 0 \rightarrow 0 \pm 0$	$12 \pm 9 \rightarrow 40 \pm 5$	$40 \pm 11 \rightarrow 92 \pm 3$	$51 \pm 6 \rightarrow \mathbf{99} \pm 1$
Scene Play	$14 \pm 11 \rightarrow 10 \pm 9$	$55 \pm 10 \rightarrow \mathbf{100} \pm 0$	$1 \pm 2 \rightarrow 50 \pm 53$	$0 \pm 0 \rightarrow \mathbf{100} \pm 0$	$0 \pm 1 \rightarrow 60 \pm 39$	$82 \pm 11 \rightarrow \mathbf{100} \pm 1$	$88 \pm 9 \rightarrow \mathbf{100} \pm 0$
Puzzle-4x4 Play	$5 \pm 2 \rightarrow 1 \pm 1$	$8 \pm 4 \rightarrow 14 \pm 35$	$0 \pm 0 \rightarrow 0 \pm 0$	$0 \pm 0 \rightarrow \mathbf{100} \pm 1$	$23 \pm 6 \rightarrow 19 \pm 33$	$8 \pm 3 \rightarrow 38 \pm 52$	$17 \pm 4 \rightarrow \mathbf{100} \pm 0$
Average	$8.4 \rightarrow 5.4$	$17.0 \rightarrow 28.6$	$0.2 \rightarrow 10.0$	$0.0 \rightarrow 41.6$	$23.4 \rightarrow 48.0$	$34.0 \rightarrow 67.6$	$49.4 \rightarrow \mathbf{88.6}$
AntMaze U-Maze	$77 \rightarrow 96$	$98 \rightarrow 75$	$77 \rightarrow \mathbf{100}$	$0 \pm 0 \rightarrow 98 \pm 3$	$94 \pm 5 \rightarrow 96 \pm 2$	$97 \pm 2 \rightarrow 99 \pm 1$	$100 \pm 1 \rightarrow \mathbf{100} \pm 1$
AntMaze U-Maze Diverse	$60 \rightarrow 64$	$74 \rightarrow 98$	$32 \rightarrow 98$	$0 \pm 0 \rightarrow 94 \pm 5$	$69 \pm 20 \rightarrow 93 \pm 5$	$79 \pm 16 \rightarrow \mathbf{100} \pm 1$	$93 \pm 7 \rightarrow 98 \pm 3$
AntMaze Medium Play	$72 \rightarrow 90$	$88 \rightarrow 98$	$72 \rightarrow \mathbf{99}$	$0 \pm 0 \rightarrow 98 \pm 2$	$52 \pm 19 \rightarrow 93 \pm 2$	$77 \pm 7 \rightarrow 97 \pm 2$	$77 \pm 9 \rightarrow 98 \pm 1$
AntMaze Medium Diverse	$64 \rightarrow 92$	$85 \rightarrow \mathbf{99}$	$62 \rightarrow 98$	$0 \pm 0 \rightarrow 97 \pm 2$	$44 \pm 26 \rightarrow 89 \pm 4$	$55 \pm 19 \rightarrow 97 \pm 3$	$76 \pm 11 \rightarrow 98 \pm 2$
AntMaze Large Play	$38 \rightarrow 64$	$68 \rightarrow 32$	$32 \rightarrow \mathbf{97}$	$0 \pm 0 \rightarrow 93 \pm 5$	$64 \pm 14 \rightarrow 80 \pm 5$	$66 \pm 40 \rightarrow 84 \pm 30$	$86 \pm 5 \rightarrow 91 \pm 10$
AntMaze Large Diverse	$27 \rightarrow 64$	$67 \rightarrow 72$	$44 \rightarrow 92$	$0 \pm 0 \rightarrow 94 \pm 3$	$69 \pm 6 \rightarrow 86 \pm 5$	$75 \pm 24 \rightarrow 94 \pm 3$	$85 \pm 5 \rightarrow \mathbf{96} \pm 4$
Average	$56.3 \rightarrow 78.3$	$80.0 \rightarrow 79.0$	$53.2 \rightarrow \mathbf{97.3}$	$0.0 \rightarrow 95.7$	$65.3 \rightarrow 89.5$	$74.8 \rightarrow 95.2$	$86.2 \rightarrow 96.8$
Pen Cloned	$84 \rightarrow 102$	$74 \rightarrow 138$	$-3 \rightarrow -3$	$3 \pm 2 \rightarrow 120 \pm 10$	$77 \pm 7 \rightarrow 107 \pm 10$	$53 \pm 14 \rightarrow \mathbf{149} \pm 6$	$71 \pm 6 \rightarrow 146 \pm 6$
Door Cloned	$1 \rightarrow 20$	$0 \rightarrow 102$	$-0 \rightarrow -0$	$0 \pm 0 \rightarrow 102 \pm 7$	$3 \pm 2 \rightarrow 50 \pm 15$	$0 \pm 0 \rightarrow 102 \pm 5$	$1 \pm 1 \rightarrow \mathbf{105} \pm 4$
Hammer Cloned	$1 \rightarrow 57$	$7 \rightarrow 125$	$0 \rightarrow 0$	$0 \pm 0 \rightarrow 128 \pm 29$	$4 \pm 2 \rightarrow 60 \pm 14$	$0 \pm 0 \rightarrow 127 \pm 17$	$10 \pm 3 \rightarrow \mathbf{132} \pm 5$
Relocate Cloned	$0 \rightarrow 0$	$1 \rightarrow 7$	$-0 \rightarrow -0$	$0 \pm 0 \rightarrow 2 \pm 2$	$-0 \pm 0 \rightarrow 5 \pm 3$	$0 \pm 1 \rightarrow 62 \pm 8$	$0 \pm 0 \rightarrow \mathbf{63} \pm 12$
Average	$21.5 \rightarrow 44.8$	$20.5 \rightarrow 93.0$	$-0.8 \rightarrow -0.8$	$0.8 \rightarrow 88.0$	$21.0 \rightarrow 55.5$	$13.2 \rightarrow 110.0$	$20.5 \rightarrow \mathbf{111.5}$



(a) Effect of the candidate number. (b) Ablation on Multi-Crescent. (c) Ablation on Puzzle-4x4.

Figure 6: Ablation studies on the offline-to-online transition. **(a):** The plot analyzes the impact of the candidate number on learning efficiency. **(b, c):** The plots demonstrate the effectiveness of our controllable entropy, showing significant performance gains of our full method over a deterministic variant in both the Multi-Crescent environment and the Puzzle-4x4 task.

Effective Online Exploration via Controllable Entropy. The second key advantage of GSflow, its capacity for effective online exploration, is enabled by its controllable entropy mechanism based on the output of the distribution actor. The performance improvements on online finesetuning shown in Table 2 are comparable, especially in tasks that require extensive exploration. The Puzzle-4x4 environment serves as a powerful case study. As noted by the authors of FQL (Park et al., 2025b), this task is particularly challenging for methods with limited exploration capabilities. The baseline FQL reflects this, improving from 8% to 38% after online training. In the contrast, GSflow leverages its entropy-regularized stochastic policy to achieve a score from 17% to 100%, matching the performance of specialized online methods like RLPD (Ball et al., 2023). Furthermore, our method significantly outperforms RLPD on other complex tasks such as AntSoccer and Cube Double. These results demonstrate that our entropy-regularized distillation successfully combines the high performance of the teacher model with the advantage of principled, controllable exploration found in traditional Gaussian policies (Haarnoja et al., 2018). We believe the idea of moving beyond a "point-to-point" mapping to a "point-to-distribution" process is simple yet valuable, allowing GSflow to effectively balance exploitation and exploration.

4.3 Further Analysis

The Importance of the Learned Prior. To analyze the impact of our proposed prior learning mechanism, we evaluate its performance while varying the number of candidate actions, N_{cand}, used in the Advantage Noise Selection module. As depicted in Figure 6a, there is a clear trend: increasing the number of candidates improves both sample efficiency and final performance. The red curve ($N_{\text{cand}} = 15$) achieves the highest return, while the green curve ($N_{\text{cand}} = 5$) learns more slowly. However, even with only five candidates, our method significantly outperforms the FQL (the gray curve), which can be viewed as a degenerate case of our approach without the Q-guided prior learning module($N_{\text{cand}} = 0$). This strongly validates the effectiveness of learning a structured prior. Given the trade-off between performance and computational overhead of the selection module, we chose $N_{\text{cand}} = 10$ (the blue curve) as a balanced setting for all main experiments.

The Importance of the Controllable Entropy. To isolate the contribution of our controllable entropy, we conduct ablation studies in the Multi-Crescent and Puzzle-4x4 environments. As shown in Figures 6b and 6c, our full method (the blue curve), which uses a dual-headed architecture with an entropy-regularized loss, demonstrates significantly higher learning efficiency during the online phase compared to a deterministic variant that uses only our learned prior module (the gray curve, denoted "Ours w/o CE"). In particular, the online performance of the gray curve in the Puzzle-4x4 task is similar to that of the FQL (as seen in Table 1). This similarity in performance strongly suggests that our controllable entropy mechanism is the key component that provides superior online exploration, effectively addressing this known limitation in prior work.

Computational Cost Analysis. We demonstrate that significant performance gains of our method do not come at a prohibitive computational cost, particularly during the critical inference phase. We compare the wall-clock time for a single training step and a single inference step against FQL and IFQL. As presented in Figure 7, the inference time of GS-flow(0.51 ms) is only marginally higher than that of FQL (0.42 ms), which is caused by the VAE Decoder model (D_{ξ_2}) and remains significantly faster than the multi-step IFQL (0.97 ms). This confirms that our method preserves the single-step efficiency of the distillation paradigm. Although the training time for GS-flow(3.10 ms) is higher due to the additional inference in the Advantage Noise Selection module under candidate number $N_{\text{cand}} = 10$, this one-time training cost is a well-justified trade-off for the substantial improvements in both policy quality and online adaptability.

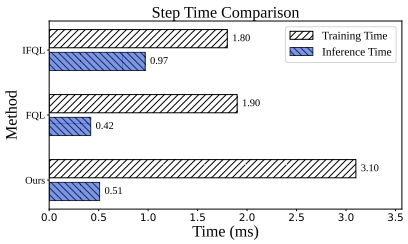


Figure 7: Average step time required on cube-double task.

5 Related Works

Efficient Inference for Generative Policies. Generative policies, including diffusion (Ho et al., 2020) and flow-matching models (Liu et al.; Albergo & Vanden-Eijnden, 2023; Geng et al., 2025), excel in representing multimodal action distributions in RL (Chi et al., 2023; Hansen-Estruch et al., 2023; Ding & Jin, 2024; Nair et al., 2021). However, their practical adoption is hindered by high inference latency (Shi & Zhang; Zhai et al., 2024). While one-step distillation methods (Park et al., 2025b) have improved inference speed, they often overlook the impact of the noise prior on optimization, a factor shown to be promising in the image generation field (Zhou et al., 2025). DSRL (Wagenmaker et al., 2025) takes advantage of this idea by learning a Gaussian prior distribution for online adaptation, without optimizing for inference latency. In contrast, our method integrates a more flexible prior while introducing negligible inference overhead.

Online Exploration for Generative Policies. Another key challenge for generative policies is principled online exploration (Fan et al., 2025). One line of research focuses on introducing stochasticity into the inference denoising process (Yang et al., 2023; Black et al., 2024b; Chen et al., 2025). Another line focuses on the training phase, using techniques such as reweighted score matching (Ma et al., 2025a) and entropy estimation with Gaussian Mixture Models, which can be computationally expensive (Wang et al., 2024). Recently, EXPO (Dong et al., 2025) enhances sample efficiency by training an additional Gaussian edit policy with entropy regularization. In contrast to these methods, our approach is more lightweight, integrating entropy control directly into the distillation process.

6 Conclusion

In this work, we introduced GS-flow, a novel framework for distilling flow-matching policies. Our method makes two key contributions: it learns a Q-Guided Generative Prior to provide a "golden start" that shortcuts the policy to high-value actions, and it uses Entropy-Regularized Distillation to endow the policy with controllable, principled exploration. Extensive experiments show that GS-flowestablishes a new state-of-the-art in overall performance on challenging benchmarks, particularly excelling on complex tasks that require multi-modal actions and effective exploration. Our framework successfully bridges the gap between expressive generative models and practical actor-critic methods, delivering a potent combination of inference speed, precision, and exploratory power.

REFERENCES

Bhavya Agrawalla, Michal Nauman, Khush Agarwal, and Aviral Kumar. floq: Training critics via flow-matching for scaling compute in value-based rl. *arXiv preprint arXiv:2509.06863*, 2025.

Michael Samuel Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://arxiv.org/abs/2209.15571.

Philip J Ball, Laura Smith, Ilya Kostrikov, and Sergey Levine. Efficient online reinforcement learning with offline data. In *International Conference on Machine Learning*, pp. 1577–1594. PMLR, 2023.

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

Kevin Black, Michael Janner, Yilun Du, Ilya Kostrikov, and Sergey Levine. Training diffusion models with reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024b. URL https://openreview.net/forum?id=YCWjhGrJFD.

Kevin Black, Manuel Y Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. *arXiv preprint arXiv:2506.07339*, 2025.

Max Braun, Noémie Jaquier, Leonel Rozo, and Tamim Asfour. Riemannian flow matching policy for robot motion learning. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5144–5151. IEEE, 2024.

Huayu Chen, Cheng Lu, Zhengyi Wang, Hang Su, and Jun Zhu. Score regularized policy optimization through diffusion behavior. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=xCRr9DrolJ.

Tianyi Chen, Haitong Ma, Na Li, Kai Wang, and Bo Dai. One-step flow policy mirror descent. *arXiv preprint arXiv:2507.23675*, 2025.

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*, pp. 02783649241273668, 2023.

Zihan Ding and Chi Jin. Consistency models as a rich and efficient policy class for reinforcement learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=v8jdwkUNXb.

Perry Dong, Qiyang Li, Dorsa Sadigh, and Chelsea Finn. Expo: Stable reinforcement learning with expressive policies. *arXiv preprint arXiv:2507.07986*, 2025.

Nicolas Espinosa-Dice, Yiyi Zhang, Yiding Chen, Bradley Guo, Owen Oertell, Gokul Swamy, Kiante Brantley, and Wen Sun. Scaling offline rl via efficient and expressive shortcut models. *arXiv preprint arXiv:2505.22866*, 2025.

Jiajun Fan, Shuaike Shen, Chaoran Cheng, Yuxin Chen, Chumeng Liang, and Ge Liu. Online reward-weighted fine-tuning of flow matching with wasserstein regularization. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=2IoFFexvuw.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. *Advances in neural information processing systems*, 34:20132–20145, 2021.

Scott Fujimoto, Herke van Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning (ICML)*, pp. 1587–1596. PMLR, 2018.

Zhengyang Geng, Mingyang Deng, Xingjian Bai, J Zico Kolter, and Kaiming He. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447*, 2025.

Raj Ghugare and Benjamin Eysenbach. Normalizing flows are capable models for rl, 2025. URL https://arxiv.org/abs/2505.23527.

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*, 2018.

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, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=68n2s9ZJWF8.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

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

Haitong Ma, Tianyi Chen, Kai Wang, Na Li, and Bo Dai. Efficient online reinforcement learning for diffusion policy, 2025a. URL https://arxiv.org/abs/2502.00361.

Nanye Ma, Shangyuan Tong, Haolin Jia, Hexiang Hu, Yu-Chuan Su, Mingda Zhang, Xuan Yang, Yandong Li, Tommi Jaakkola, Xuhui Jia, et al. Scaling inference time compute for diffusion models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pp. 2523–2534, 2025b.

Ashvin Nair, Abhishek Gupta, Murtaza Dalal, and Sergey Levine. Awac: Accelerating online reinforcement learning with offline datasets, 2021. URL https://arxiv.org/abs/2006.09359.

Mitsuhiko Nakamoto, Simon Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning. *Advances in Neural Information Processing Systems*, 36:62244–62269, 2023.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. OGBench: Benchmarking offline goal-conditioned RL. In *The Thirteenth International Conference on Learning Representations*, 2025a. URL https://openreview.net/forum?id=M992mjgKzI.

Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. In *Forty-second International Conference on Machine Learning*, 2025b. URL https://openreview.net/forum?id=KVf2SFL1pi.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. PMLR, 2015.

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

Chang Shi and Amy Zhang. Fastdp: Deployable diffusion policy for fast inference speed. In *RLC 2025 Workshop on Practical Insights into Reinforcement Learning for Real Systems*.

Bernard W Silverman. *Density estimation for statistics and data analysis*. Routledge, 2018.

Richard S Sutton, Andrew G Barto, et al. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998.

Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 2023.

Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. *arXiv preprint arXiv:2506.15799*, 2025.

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. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=AHvFDPi-FA.

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.

Yuexiang Zhai, Hao Bai, Zipeng Lin, Jiayi Pan, Shengbang Tong, Yifei Zhou, Alane Suhr, Saining Xie, Yann LeCun, Yi Ma, and Sergey Levine. Fine-tuning large vision-language models as decision-making agents via reinforcement learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=nBjmMF2IZU.

Zikai Zhou, Shitong Shao, Lichen Bai, Shufei Zhang, Zhiqiang Xu, Bo Han, and Zeke Xie. Golden noise for diffusion models: A learning framework. In *International Conference on Computer Vision*, 2025.

A Use of Large Language Models

We utilized Large Language Models as a tool to assist in the preparation of this paper. Specifically, LLMs were used for polishing the language, correcting grammar, and providing suggestions for LaTeX formatting to improve the manuscript's presentation. Our use of LLMs is in full compliance with the ICLR 2026 policy. We reviewed all LLM-generated outputs and take full responsibility for the scientific claims and all content in this work.

B Critic Update Details

Our critic network, denoted as $Q_\theta(s, a)$, is trained to estimate the expected discounted cumulative reward (the Q-value) for taking action a in state s. To improve training stability and mitigate the overestimation of Q-values, we employ standard techniques from modern actor-critic methods (Fujimoto et al., 2018; Haarnoja et al., 2018). Specifically, we use a twin-critic architecture, maintaining two separate Q-networks ($Q_{\theta_1}, Q_{\theta_2}$), and use slowly-updated target networks ($Q_{\theta'_1}, Q_{\theta'_2}$) to construct the Bellman target. The critic parameters are optimized by minimizing the Mean Squared Bellman Error (MSBE). For a given transition (s, a, r, s') from the replay buffer, we first compute the target value, y. The next action, a', is sampled from our stochastic student policy, π_φ, and the target value includes an entropy term to maintain consistency with the actor's objective:

$$y = r + \gamma \left(\min_{i=1,2} Q_{\theta'_i}(s', a') - \alpha_2 \log \pi_\varphi(a'|s') \right), \quad \text{where} \quad a' \sim \pi_\varphi(\cdot|s'). \tag{14}$$

The total loss for the critic networks is the sum of the MSBE for each critic with respect to this common target value:

$$\mathcal{L}_{\text{Critic}}(\theta_1, \theta_2) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}} \left[\sum_{i=1,2} \left(Q_{\theta_i}(s,a) - y \right)^2 \right]. \tag{15}$$

The target network parameters θ' are updated via Polyak averaging with the main critic parameters θ at each training step: $\theta' \leftarrow \tau\theta + (1-\tau)\theta'$, where τ is a small interpolation factor.

C Theoretical Analysis

C.1 Preliminaries and Notation

Let us formalize the key concepts:

- **Optimal noise set**: $\mathcal{X}^*(s) = \{x_0 : Q(s, \pi_\phi(s, x_0)) = \max_a Q(s,a)\}$
- **Optimal prior**: $p^*(x_0|s)$ - uniform over $\mathcal{X}^*(s)$
- **Learned prior**: $p_{\text{adv}}(x_0|s)$ - our CVAE-based prior
- **Baseline prior**: $p_0(x_0) = \mathcal{N}(0, I)$
- **Value function**: $J(\pi; p, s) = \mathbb{E}_{x_0\sim p(\cdot|s)}[Q(s, \pi(s, x_0))]$

Assumption 1 (Lipschitz Continuity): $Q(s,a)$ is L_Q-Lipschitz in a, and $\pi_\phi(s, x_0)$ is L_π-Lipschitz in x_0.

C.2 Value Bound via Wasserstein Distance

Lemma 1 (Value Sensitivity)**.** *Under Assumption 1, for any priors p, p':*

$$|J(\pi; p, s) - J(\pi; p', s)| \leq L_Q L_\pi \cdot W_1(p(\cdot|s), p'(\cdot|s))$$

where W_1 is the 1-Wasserstein distance.

Proof. By Lipschitz continuity, $f(x_0) = Q(s, \pi(s, x_0))$ is L_f-Lipschitz with $L_f = L_Q L_\pi$. The result follows from the dual formulation of W_1. □

Corollary 1. *For any prior p:*

$$J(\pi; p, s) \geq J(\pi; p^*, s) - L_f \cdot W_1(p, p^*)$$

C.3 Superiority of the Learned Prior

Lemma 2 (Prior Improvement)**.** *Under our training procedure:*

$$W_1(p_{adv}, p^*) \leq W_1(p_0, p^*) - \Delta(s)$$

where $\Delta(s) > 0$ quantifies the improvement.

Proof. Let $\hat{p}^*_{N_{\text{cand}}}$ be the empirical distribution of the N advantage noise samples drawn from the true advantage distribution p^*(Eq. 4). According to the triangle inequality, the distance $W_1(p_{\text{adv}}, p^*)$ is bounded as:

$$W_1(p_{\text{adv}}, p^*) \leq \underbrace{W_1(p_{\text{adv}}, \hat{p}^*_{N_{\text{cand}}})}_{\text{Optimization Error}} + \underbrace{W_1(\hat{p}^*_{N_{\text{cand}}}, p^*)}_{\text{Statistical Error}}$$

For the **optimization error**, the VAE's training objective, $\mathcal{L}_{\text{VAE}}$, is designed to minimize the divergence between p_{adv} and $\hat{p}^*_{N_{\text{cand}}}$. For a sufficiently expressive and well-trained VAE, this error can be made small Kingma & Welling (2013).

For the **statistical error**, which measures how well N finite samples represent the true distribution p^*. This error converges to zero as the number of samples N increases ($\mathbb{E}[W_1(\hat{p}^*_{N_{\text{cand}}}, p^*)] \propto 1/\sqrt{N}$).

In contrast, the prior p_0 is fixed, and its distance to the optimal prior, $W_1(p_0, p^*)$, is a fixed positive constant $C_0 > 0$. Since the sum of the optimization and statistical errors for our method can be made smaller than C_0, there exists a positive $\Delta(s)$ such that:

$$W_1(p_{\text{adv}}, p^*) \leq W_1(p_0, p^*) - \Delta(s)$$

□

C.4 Main Theoretical Result

Theorem 1 (Value Lower Bound Improvement). *Let LB_{adv} and LB_0 be the respective performance lower bounds for our method (using prior p_{adv}) and the baseline (using prior p_0). Under Assumption 1, with high probability:*

$$LB_{adv} \geq LB_0 + L_f \cdot \Delta(s)$$

where $\Delta(s) = W_1(p_0, p^*) - W_1(p_{adv}, p^*) > 0$, and $L_f = L_Q L_\pi$.

Proof. From the corollary, we have the performance lower bounds for our method and the baseline:

$$J(\pi; p_{\text{adv}}, s) \geq J(\pi; p^*, s) - L_f \cdot W_1(p_{\text{adv}}, p^*) \tag{16}$$

$$J(\pi; p_0, s) \geq J(\pi; p^*, s) - L_f \cdot W_1(p_0, p^*) \tag{17}$$

We define the right-hand side of inequalities equation 16 and equation 17 as the lower bounds LB_{adv} and LB_0, respectively. Then we have:

$$\begin{aligned} LB_{\text{adv}} - LB_0 &= (J(\pi; p^*, s) - L_f \cdot W_1(p_{\text{adv}}, p^*)) - (J(\pi; p^*, s) - L_f \cdot W_1(p_0, p^*)) \\ &= L_f \cdot (W_1(p_0, p^*) - W_1(p_{\text{adv}}, p^*)) \end{aligned} \tag{18}$$

A successfully trained VAE ensures that $W_1(p_{\text{adv}}, p^*) < W_1(p_0, p^*)$. Therefore, the term $\Delta(s) = W_1(p_0, p^*) - W_1(p_{\text{adv}}, p^*)$ is strictly positive, leading to the conclusion:

$$LB_{\text{adv}} - LB_0 = L_f \cdot \Delta(s) > 0.$$

□

D Additional Studies in the Multi-Crescent Environment

To further validate the effectiveness of our custom Multi-Crescent Environment at highlighting key algorithmic challenges, we conducted a broader set of experiments with different baseline settings, as shown in Figure 8 in the main text. In this analysis, the gray bars represent the final offline training performance, while the blue bars show the performance after the subsequent online fine-tuning phase. The hyperparameter α corresponds to the weight of the behavioral cloning (BC) term in the FQL loss function (Equation 2). A smaller α places a relatively larger emphasis on the Q-maximization term. The primary results in the main body compare our method against FQL with a high BC weight ($\alpha = 100$).

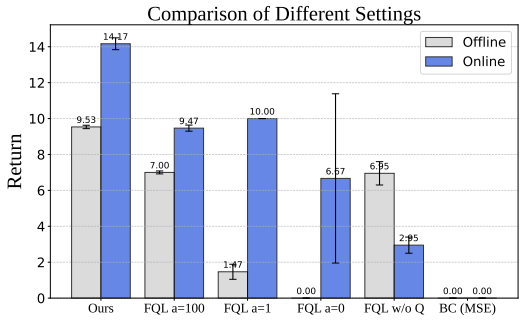


Figure 8: Additional Studies in the Multi-Crescent Environment

By lowering α, we test the hypothesis that our non-convex environment can induce Q-value over-estimation in the baseline. The experimental results confirm this hypothesis. When $\alpha = 1$, FQL's

offline performance drops sharply. The policy learns to target points overestimated by the critic—locations between the tips of the crescent shapes but outside the actual high-reward regions—leading to a significant decrease in return. In the extreme case where $\alpha = 0$ (i.e., pure Q-maximization), the offline return predictably falls to zero, further demonstrating our environment's ability to challenge methods susceptible to Q-value overestimation. The final two columns in the figure are designed to evaluate the performance of pure imitation learning. The "FQL w/o Q" baseline isolates the effect of imitation learning via flow-matching, while "BC (MSE)" represents a standard behavioral cloning approach. Our method outperforms both of these baselines. Notably, the standard FQL provides only a marginal improvement over "FQL w/o Q." This is because the reward modes in our environment are disconnected; when the Q-guidance is too weak (high α), the policy struggles to jump from one mode to another, and when it is too strong (low α), the policy is misled by critic overestimation. In contrast, our algorithm learns an initial noise distribution that directly fits the inherently high-Q actions from the dataset, making it significantly more robust to the effects of Q-value overestimation.

E Benchmark Descriptions

E.1 OGBench

OGBench (Offline Goal-Conditioned RL Benchmark) Park et al. (2025a) is a high-quality benchmark designed for offline goal-conditioned reinforcement learning. It aims to systematically evaluate the capabilities of algorithms across several key dimensions, such as trajectory stitching, long-horizon reasoning, handling high-dimensional inputs (e.g., pixels), and coping with environmental stochasticity. We utilize a variety of environments from OGBench in our experiments, spanning locomotion, manipulation, and visual tasks. Notably, we evaluate on the default task for each environment. For instance, in the `cube-double-play` environment, we exclusively use the `cube-double-play-singletask-task2-v0` task, which can be found in Park et al. (2025a;b).

The specific environments used in our work include:

- **AntMaze and HumanoidMaze**: These are maze navigation tasks requiring an agent to control a complex quadruped robot (Ant) or a 21-DoF humanoid robot (Humanoid), respectively, to reach a target location. We employ various maze layouts, including "Large" and "Giant", with the "Navigate" dataset type to test long-horizon planning and hierarchical control capabilities.
- **AntSoccer**: This is a more challenging locomotion task that requires the Ant agent to dribble a soccer ball while navigating. We use the "Arena" (open-field) version of this environment.
- **Cube**: This is a robotic manipulation task involving multi-block pick-and-place operations. The agent must move, stack, or swap single or multiple cubes according to a goal configuration. We use the "Single Play" and "Double Play" versions to test the agent's ability to learn generalizable multi-object manipulation skills from unstructured, random trajectories.
- **Scene**: This is a complex sequential manipulation task requiring the robot arm to interact with various household objects, including a drawer, a window, button locks, and a cube. It is designed to challenge the agent's sequential and long-horizon reasoning abilities.
- **Puzzle**: In this task, a robot arm must solve a "Lights Out" puzzle. The agent presses buttons on a grid to toggle the color of the pressed button and its neighbors to match a goal configuration. We use the 3×3 and 4×4 grid versions to specifically test for combinatorial generalization.
- **Visual Environments**: Many tasks in OGBench, particularly the manipulation suite, support both state-based and pixel-based inputs. We evaluate our methods in the corresponding visual environments (e.g., Visual Cube, Visual Scene, Visual Puzzle), which require the agent to learn control policies directly from 64×64 RGB images.

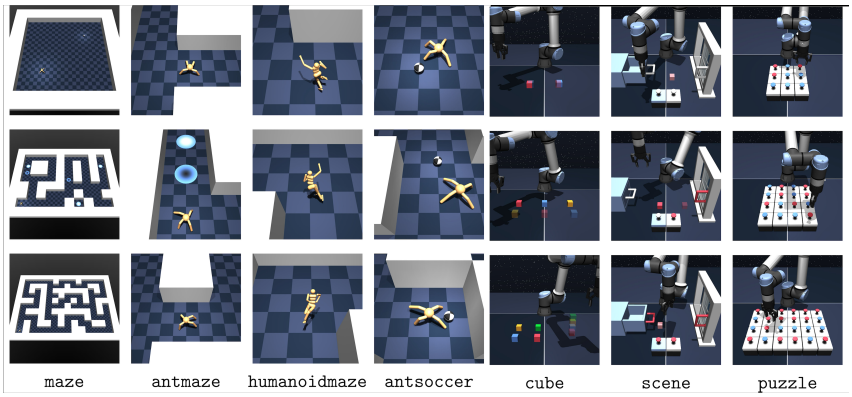


Figure 9: Visualization of the OGBench tasks.

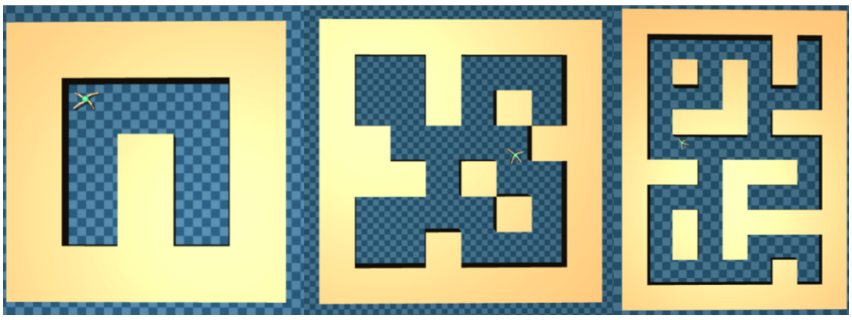


Figure 10: Visualization of the D4RL tasks.

E.2 D4RL

D4RL (Datasets for Deep Data-Driven Reinforcement Learning) Fu et al. (2020) is a public benchmark focused on offline reinforcement learning. It is designed to provide datasets that reflect challenges present in real-world applications, such as narrow data distributions, undirected multi-task data, sparse rewards, and suboptimal data. These characteristics make D4RL an essential benchmark for evaluating the robustness and generalization of offline RL algorithms.

Our experiments primarily make use of the **AntMaze** environment from D4RL. This is a popular navigation task that requires an 8-DoF 'Ant' quadruped robot to reach a specified goal in a maze. The task features sparse rewards (a reward is only given upon reaching the goal), and the datasets are generated by a non-Markovian controller. This setup is designed to test an algorithm's ability to stitch effective trajectories from undirected data to solve long-horizon, sparse-reward tasks. We also implement on the Adroit domain, which involves controlling a 24-DoF robotic hand. Task examples are shown in Figure 10

F Details on Additional Baseline Methods

In this section, we provide additional details on the baseline methods used in the paper (except FQL, which is introduced in Preliminary) , categorized by their underlying policy structure and learning paradigm. The settings for all baseline methods are adopted directly from the original paper (Park et al., 2025b) for comparison. You can find more implement details in their paper.

F.1 Offline RL Baselines

For the offline RL experiments, we compare with 10 recent and representative methods to demonstrate our contributions.

Gaussian Policies. For standard offline RL methods that use Gaussian policies, we consider **BC**, **IQL** (Kostrikov et al., 2022), and **ReBRAC** (Tarasov et al., 2023). In particular, ReBRAC is known to perform well on many D4RL tasks (Fu et al., 2020), which are based on a behavior-regularized actor-critic framework.

Diffusion Policies. For methods based on diffusion policies, we compare against **IDQL** (Hansen-Estruch et al., 2023), **SRPO** (Chen et al., 2024), and Consistency-AC (**CAC**) (Ding & Jin, 2024). These methods employ different policy extraction techniques: IDQL is based on rejection sampling, whereas SRPO and CAC utilize policy distillation. CAC trains the distillation policy within the behavior-regularized actor-critic framework and is based on consistency models.

Flow Policies. We also consider several flow-based variants of existing algorithms to cover different policy extraction schemes. Flow Advantage-Weighted Actor-Critic (**FAWAC**) is a flow-based variant of AWAC (Nair et al., 2021), which uses the Advantage-Weighted Regression (AWR) objective for policy learning. Flow Behavior-Regularized Actor-Critic (**FBRAC**) is the flow counterpart to Diffusion-QL (DQL) (Wang et al., 2023), which is based on the original Q-loss with backpropagation through time. Implicit Flow Q-Learning (**IFQL**) is the flow counterpart to IDQL, based on a rejection sampling scheme.

F.2 Offline-to-Online Finetuning Baselines

The offline methods include **IQL**, which learns a policy implicitly through advantage-weighted regression over learned Q and Value functions; **ReBRAC**, a stable behavior-regularized actor-critic algorithm; and **IFQL**, a flow-based policy utilizing rejection sampling. We also include two methods designed for data-driven online RL: **Cal-QL** (Nakamoto et al., 2023), which calibrates the Q-function with the offline dataset to enable safer online exploration, and **RLPD** (Ball et al., 2023), which employs a balanced sampling strategy from both offline and online data buffers to accelerate fine-tuning.

G Hyperparameters

G.1 Hyperparameters Settings

Table 4 lists the hyperparameters used for the cube-double experiment, based on the provided execution command.

The 'Offline Alpha' and 'Online Alpha' refer to the pre-set values of the hyperparameter α_1 in Equation 9 for the offline-to-online transition. This distinction is made because the confidence in the critic's estimates differs between the offline and online phases. It is a common phenomenon in offline RL that the critic often overestimates Q-values, necessitating regulation of the Behavior Cloning (BC) weight. However, during the online phase, excessive reliance on the BC term can stifle exploration, which is why these values are set in advance.

Additionally, when running the online phase for the puzzle environment, we utilized the balanced sampling technique from Ball et al. (2023). To ensure a fair comparison, we also applied this technique to our FQL agent. We found that only our method showed performance improvements with this technique.

G.2 The Effect of Latent Dimension.

We investigate the sensitivity of our learned prior to the VAE's latent dimension. Figure 11 illustrates the results. We observe that a low-dimensional, compact latent space is optimal for this task, with performance peaking at a dimension of 1 or 2 for both offline and online settings. As the latent dimension increases, performance gradually degrades, particularly during the online phase. This suggests that a higher-dimensional space may increase the difficulty of learning a meaningful prior, potentially introducing noise or leading to overfitting. However, our method still outperforms the FQL across different tested dimensions in both settings. This demonstrates the fundamental robustness and benefit of our learned prior, even when its key hyperparameter is not perfectly tuned.

Table 3: Hyperparameters for the cube-double experiment.

Hyperparameter	Value
Offline Steps	1,000,000
Online Steps	1,000,000
Seed	0,2,4,8,16
Latent Dimension	8 (default)
KL Weight	0.1 (default)
Reconstruction Weight	1 (default)
Number of Candidates	10 (default)
Offline Alpha1	300
Online Alpha1	50
Offline Temperature	0 (default)
Target Entropy Multiplier	0.5 (default)

Table 4: Hyperparameters for the Puzzle 3x3 experiment.

Hyperparameter	Value
Offline Steps	1,000,000
Online Steps	1,000,000
Seed	0,2,4,8,16
Latent Dimension	8 (default)
KL Weight	0.1 (default)
Reconstruction Weight	1 (default)
Number of Candidates	10 (default)
Offline Alpha1	1000
Online Alpha1	10
Offline Temperature	0 (default)
Target Entropy Multiplier	0.5 (default)
Balanced Sampling(Ball et al. (2023))	True

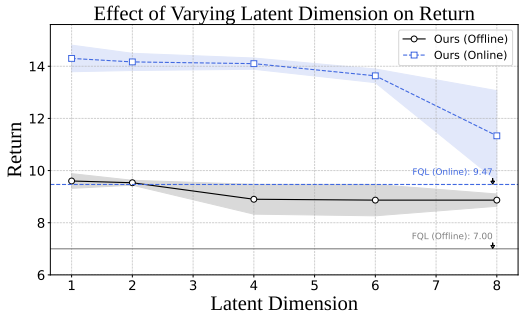


Figure 11: The effect of the VAE's latent dimension on the final return in both offline (black) and online (blue) settings.