One-Step Generative Policies with Q-Learning: A Reformulation of MeanFlow

Zeyuan Wang [1], Da Li [2,3], Yulin Chen[1], Ye Shi [4], Liang Bai [1], Tianyuan Yu [1] *, Yanwei Fu [5,6],

[1]Laboratory for Big Data and Decision, National University of Defense Technology
[2] Samsung AI Center Cambridge
[3] Queen Mary University of London
[4] ShanghaiTech University
[5] Fudan University
[6] Shanghai Innovation Institute
{wzeyuan,ty.yu}@nudt.edu.cn, dali.academic@gmail.com, shiye@shanghaitech.edu.cn, yanweifu@fudan.edu.cn

Abstract

We introduce a one-step generative policy for offline reinforcement learning that maps *noise* directly to *actions* via a *residual reformulation* of MeanFlow, making it compatible with Q-learning. While one-step Gaussian policies enable fast inference, they struggle to capture complex, multimodal action distributions. Existing flow-based methods improve expressivity but typically rely on distillation and two-stage training when trained with Q-learning. To overcome these limitations, we propose to reformulate MeanFlow to enable *direct noise-to-action generation* by integrating the velocity field and noise-to-action transformation into a single policy network—eliminating the need for separate velocity estimation. We explore several reformulation variants and identify an effective *residual formulation* that supports expressive and stable policy learning. Our method offers three key advantages: 1) efficient one-step noise-to-action generation, 2) expressive modelling of multimodal action distributions, and 3) efficient and stable policy learning via Q-learning in a single-stage training setup. Extensive experiments on 73 tasks across the OGBench and D4RL benchmarks demonstrate that our method achieves strong performance in both offline and offline-to-online reinforcement learning settings.

Code — `https://github.com/HiccupRL/MeanFlowQL`
Extended version — `arxiv.org/abs/2511.13035`

Introduction

Offline reinforcement learning (RL) enables the training of decision-making agents from fixed datasets, eliminating the need for online environment interaction (Hafner and Riedmiller 2011). This is especially valuable in safety-critical domains such as robotics, autonomous driving, and healthcare, where exploration can be costly or risky. A central challenge in offline RL is learning *expressive yet efficient policies* that can capture complex, multi-modal action distributions while remaining amenable to stable value-based optimisation.

One-step Gaussian policies are widely adopted due to their fast and simple sampling procedures. However, their uni-modal nature limits expressivity, especially in tasks that involve diverse or multi-modal behaviour patterns (Pomerleau 1988; Tarasov et al. 2023a). In contrast, flow- and diffusion-based generative models (Esser et al. 2024; Jin et al. 2025; Liu et al. 2024) have shown strong potential for modelling rich, multimodal distributions and have been increasingly applied in offline RL (Park, Li, and Levine 2025; Mees et al. 2024; O'Neill et al. 2024). Most of these approaches, however, focus on *behaviour cloning* (Pomerleau 1988; Chi et al. 2023; Prasad et al. 2024; Zhang et al. 2025; Sheng et al. 2025) and do not support *value-based learning*, limiting their ability to improve policies with Q-functions.

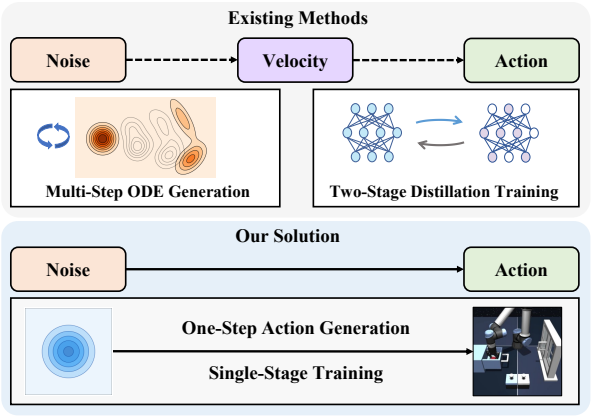

Figure 1: Existing generative policy approaches v.s. our proposed solution. Existing methods rely on velocity-based modelling, requiring either multi-step ODE integration or two-stage distillation training. In contrast, our method enables one-step action generation directly from noise, trained via a single-stage, end-to-end process—improving both inference efficiency and training stability.

Integrating expressively generative models into Q-learning is challenging. Flow- and diffusion-based policies typically require *multi-step generation*, relying on ODE or SDE solvers, which introduce *backpropagation through time (BPTT)* (Ding and Jin 2024). This leads to computational inefficiencies and instability when optimizing jointly with a critic. To address this, recent works (Chen, Wang, and Zhou

*Corresponding author

2024; Park, Li, and Levine 2025) adopt a *two-stage distillation framework*, where a multi-step generative policy is first trained via behaviour cloning, then distilled into a one-step policy together optimised with Q-values. While this avoids BPTT, it introduces additional complexity and degrades the final policy's expressivity due to the distillation bottleneck.

In this work, we propose an *expressive one-step generative policy* that can be directly optimized with Q-functions in a *single-stage* training process—without requiring distillation or iterative sampling. Our approach is inspired by the MeanFlow framework (Geng et al. 2025), which enables efficient one-step sampling by modelling the average velocity of noise-to-data transitions. Although MeanFlow was originally developed for visual generation, we repurpose its structure for learning a generative policy in offline RL.

A direct adaptation of MeanFlow involves two separate processes: first, predicting a velocity vector, and then integrating it to transform noise into actions. However, we find that this approach leads to a critical issue in Q-learning: during early training, the generated actions frequently lie outside valid bounds and require *post-hoc clipping*. This mismatch between policy outputs and the actions used for Bellman target computation destabilizes training and degrades performance.

One might attempt to simplify MeanFlow by reformulating its two-step process into a single network with direct, residual mapping—i.e., using $a = \epsilon - u(\epsilon, b, t)$, where a is predicted action, $\epsilon \sim \mathcal{N}(0, I)$ is sampled noise, u is a learnable average velocity field, and b, t denote the start and end times. While this formulation eliminates the need for explicit velocity integration, we find that such a naive reformulation suffers from poor training dynamics. In toy experiments, this model underfits and fails to capture multimodal data distributions, reintroducing the risk of generating mismatched actions—mirroring the challenges seen in the original two-step MeanFlow adaptation.

To address these issues, we propose a revised *residual MeanFlow formulation* with a carefully constructed mapping:

$$g(a_t, b, t) = a_t - u(a_t, b, t), \tag{1}$$

where a_t is a linear interpolation between an offline-dataset action and noise. Unlike the naive form, this formulation retains theoretical equivalence to the original MeanFlow under mild assumptions, with guarantees provided by the *Universal Approximation Theorem* (Tabuada and Gharesifard 2021) (see Methodology and Appendix B.3 for further discussion).

Our method offers three key advantages: i) Efficient one-step noise-to-action generation without iterative inference; ii) Expressive modelling of complex, multimodal action distributions via reformulated MeanFlow-based policies; iii) Stable joint optimisation with Q-functions through direct, single-stage training—without distillation. In addition, we introduce two practical enhancements to improve the efficacy of our method, including a value-guided rejection sampling and an adaptive behaviour cloning regularisation (See Methodology section for details). Collectively, they contribute to the robustness and performance of our one-step policy, leading to strong results across 73 tasks on the OGBench (Park et al. 2025) and D4RL (Fu et al. 2020) benchmarks in both the offline and offline-to-online settings (see Experiments).

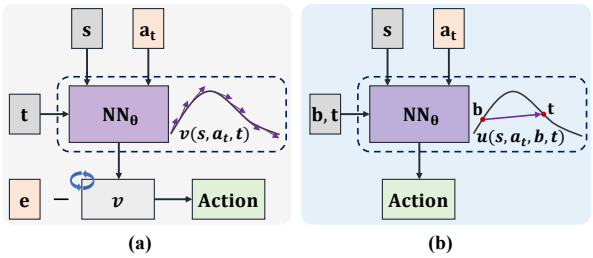

Figure 2: Comparison between traditional and proposed flow-based policy network schemes. (a) Given state s and noisy input a_t, traditional flow-based methods estimate the *instantaneous velocity* $v(s, a_t, t)$ at time t, as a one-step approximation of multi-step ODE integration. (b) Our method reformulate MeanFlow to estimate the *average velocity* over the interval $[b, t]$, enabling direct one-step action generation.

Preliminary

Offline Reinforcement Learning

In this work, we consider a Markov Decision Process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, \rho, p, \gamma)$ (Sutton, Barto et al. 1998; Puterman 2014), where $\mathcal{S}$ denotes the state space, $\mathcal{A} \subseteq \mathbb{R}^d$ is the d-dimensional continuous action space, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, $\rho \in \Delta(\mathcal{S})$ is the initial state distribution, $p(s'|s, a)$ defines the transition dynamics, and $\gamma \in [0, 1)$ is the discount factor. In offline RL, the agent does not interact with the environment during training. Instead, it learns from a fixed dataset $\mathcal{D} = \{\tau^{(n)}\}_{n=1}^{N}$ composed of trajectories $\tau = (s_0, a_0, r_0, \ldots, s_H, a_H, r_H)$, collected by some unknown behaviour policy. The typical objective is to learn a policy $\pi_\theta(a|s)$ that maximizes the expected discounted return with some regularization:

$$\mathcal{L}_\pi(\theta) = -J(\pi_\theta) + \alpha R(\pi_\theta), \tag{2}$$

where $J(\pi_\theta) = \mathbb{E}_{\tau \sim p^{\pi_\theta}(\tau)} \left[\sum_{t=0}^{H} \gamma^t r(s_t, a_t) \right]$, $p^{\pi_\theta}(\tau)$ is the trajectory distribution induced by π_θ under the dynamics p and initial state ρ, and α controls the regularisation strength.

In the behaviour cloning (BC) regularised actor-critic learning framework, the objective in Eq 2 becomes

$$\mathcal{L}_\pi(\theta) = \mathbb{E}_{s,a \sim \mathcal{D},\, a^\pi \sim \pi_\theta} \left[-Q_\phi(s, a^\pi) - \alpha \log \pi_\theta(a|s) \right], \tag{3}$$

where the second term is a behaviour cloning regularizer, $\mathcal{D}$ denotes the offline dataset and $Q_\phi(s, a)$ is the critic parameterised by ϕ, which is trained to minimise the Bellman error as follows

$$\mathcal{L}_Q = \mathbb{E}_{(s,a,r,s') \sim \mathcal{D}} \Big[Q_\phi(s, a) - (r + \gamma\, Q_{\bar{\phi}}(s', \pi_\theta(s'))) \Big]^2, \tag{4}$$

where $\bar{\phi}$ corresponds to the target network.

Flow Matching and MeanFlow

Flow Matching learns to transform noise into data via time-dependent velocities, whereas MeanFlow enables one-step sampling by modelling the time-averaged velocity.

Flow matching. Flow matching (Lipman et al. 2023; Liu, Gong, and Liu 2023) is a generative modelling framework that learns a continuous time-dependent velocity field bridging a simple base distribution and a complex data distribution. In our context, let $e \sim p_{\text{prior}}$ be sampled from a simple prior (e.g. a standard Gaussian) and $a \sim p_{\text{data}}$ from the empirical action data distribution. We consider an interpolation $a_t = (1-t)\,a + t\,e$ for $t \in [0,1]$. Flow matching trains a velocity field $v_\theta(a_t, t)$ to predict the ground-truth displacement between a and e. Specifically, one trains v_θ by minimizing:

$$\min_\theta \mathop{\mathbb{E}}_{\substack{a\sim p_{\text{data}},\\ e\sim p_{\text{prior}},\, t\sim\mathcal{U}(0,1)}} \left[\|v_\theta(a_t,t) - (e-a)\|_2^2\right], \tag{5}$$

where $a_t = (1-t)a + te$ as above. By learning to predict $e - a$, the model captures the time-dependent instantaneous velocity needed to transform a into e (or vice versa).

MeanFlow. MeanFlow (Geng et al. 2025) builds on flow matching by introducing an *average velocity* field $u(a_t, b, t)$. For any $0 \le b < t \le 1$, the average velocity is defined as the time-average of the instantaneous velocity v over the interval $[b, t]$:

$$u(a_t, b, t) \;\coloneqq\; \frac{1}{t-b}\int_b^t v(a_\tau, \tau)\, d\tau. \tag{6}$$

Differentiating both sides with respect to t (treating b as constant) yields the *MeanFlow Identity* as introduced in (Geng et al. 2025):

$$u(a_t, b, t) \;=\; v(a_t, t) \;-\; (t-b)\,\frac{\partial}{\partial t} u(a_t, b, t). \tag{7}$$

MeanFlow establishes a principled relationship between the average velocity u and the instantaneous velocity v. Crucially, it enables *one-step* generation: by choosing $b = 0$ and $t = 1$ in (6), one can generate a sample in a single step:

$$\hat{a} \;=\; e \;-\; u_\theta(e,\; b=0,\; t=1), \tag{8}$$

where $e \sim \mathcal{N}(0, I)$ and u_θ is the learned average velocity function. This formula generates $\hat{a}$ via one-step velocity estimation using the learned u_θ, without integrating over time.

Flow policies in RL. Recent works have proposed using flow matching to learn policies in RL. In such *flow policies*, the policy π_θ is implicitly defined by a state-conditional velocity field $v_\theta(s, a_t, t)$. For example, one offline training objective for a flow policy is:

$$\mathcal{L}_{\text{Flow}}(\theta) = \mathop{\mathbb{E}}_{\substack{(s,a)\sim D,\\ e\sim\mathcal{N}(0,I),\, t\sim\mathcal{U}(0,1)}} \left[\|v_\theta(s,a_t,t) - (e-a)\|_2^2\right], \tag{9}$$

with $a_t = (1-t)a + te$. The flow model v_θ here plays the role of a generative policy that, given state s and noise e, outputs a velocity which can be used to produce an action.

A straightforward way to integrate flow policies into offline RL is to replace the BC loss in the actor objective (3) with the flow matching loss (9). Compared to Gaussian or diffusion-based policies, flow policies offer a powerful yet efficient framework for modelling multi-modal action distributions.

[1]Note that *state* s is an important input of our model. But for simplicity, and to save space in our equations, we will continue to denote the pair (s, a_t) simply by a_t

Methodology

Standard Gaussian policies generate actions directly from noise and offer efficient inference, but they lack the expressivity needed to model complex, multimodal action distributions. Flow-based generative policies improve expressivity through iterative sampling, yet their integration with Q-learning requires backpropagation through time (BPTT), making joint training unstable and inefficient. Distillation-based approaches bypass BPTT by employing a two-stage training process, which sacrifices model expressivity in the final one-step policy.

This raises a key question: *Can we design a generative policy that enables one-step action sampling while preserving the expressivity of flow-based models and supporting stable joint training with Q-functions?* To answer this question, we introduce our idea of reformulating MeanFlow into an end-to-end policy function.

Naive MeanFlow Policies

A straightforward approach to enable one-step action generation is to directly adopt the original MeanFlow formulation, which estimates the average velocity and uses it to transform noise into actions. Specifically, MeanFlow generates an action as follows:

$$\begin{aligned} v_{\text{ave}} &= u(e, b{=}0, t{=}1), \,//\text{vel. est.} \\ \hat{a} &= e - v_{\text{ave}}, \,//\text{vel. integ.} \end{aligned} \tag{10}$$

where $e \sim \mathcal{N}(0, I)$ is Gaussian noise, and u denotes the learned average velocity field.

Although this formulation avoids iterative velocity integration over time and thus enables efficient inference, it suffers from a key limitation: the decoupled processes reduce controllability. The sampled noise e often induces the generated actions to fall outside the valid action bounds (e.g., $[-1, 1]$), especially in early training when u is inaccurate. Consequently, the generated actions frequently require post-hoc clipping, which breaks the alignment between the policy output and the actions used in the Bellman target. This mismatch destabilises Q-learning and can degrade overall performance, as shown in the results in Experiments and analysis in the Appendix.

Reformulated MeanFlow: Direct Noise-to-Action Mapping

The core limitation of the original MeanFlow formulation lies in its inference pipeline: noise $\rightarrow$ velocity $\rightarrow$ action. This decoupled design, which first predicts a velocity and then transforms noise into an action, often leads to instability—particularly during early training when the estimated velocity field is still inaccurate.

A natural workaround is to reformulate MeanFlow into a single-step residual form, such as $a = \epsilon - u(\epsilon, b, t)$, effectively merging velocity estimation and action generation into one forward pass. However, this naive reformulation performs poorly in practice: it fails to capture multimodal distributions, as shown in the toy experiments in the Appendix.

We propose a revised *residual MeanFlow formulation* with a carefully constructed mapping

$$g(a_t, b, t) = a_t - u(a_t, b, t), \tag{11}$$

which reduces to the original MeanFlow output when $b = 0$, $t = 1$, and $a_1 = e$:

$$g(e, b = 0, t = 1) = e - u(e, b = 0, t = 1). \quad (12)$$

This revised residual transformation satisfies two key desiderata: 1) The mapping g_θ retains the representational capacity of the original MeanFlow velocity function u_θ, as guaranteed by the *Universal Approximation Theorem (UAT)* (Leshno et al. 1993; Tabuada and Gharesifard 2021). 2) For inputs of the form given in Equation 10, the model enables well-bounded one-step outputs (typically within the [-1,1] range) even in the early stages of training through appropriate initialisation strategies, such as zero initialisation or small-variance Kaiming initialisation. A more detailed derivation of this formulation is provided in Appendix B.

This proposed formulation enables the network to learn a complete noise-to-action transformation directly, without depending on intermediate velocity estimation. Our method alleviates the early-stage instability caused by post-hoc clipping, while preserving the expressivity and efficiency inherent in the MeanFlow framework.

Training Objective of Our One-Step Policy

To train our residual mapping $g_\theta(a_t, b, t)$, we leverage the *MeanFlow Identity*, which defines the average velocity field $u(a_t, b, t)$ as the difference between the instantaneous velocity $v(a_t, t)$ and its time derivative $\frac{d}{dt}u(z_t, b, t)$, as shown in Eq 7. Specifically, we follow (Geng et al. 2025) to optimise g_θ to satisfy this identity by minimising the following objective:

$$\mathcal{L}_{\text{MFI}}(\theta) = \mathbb{E}\left\| g_\theta(a_t, b, t) - \text{sg}(g_{\text{tgt}}) \right\|_2^2, \quad (13)$$

where $\mathcal{L}_{\text{MFI}}$ represents loss of MeanFlow Identity, $\text{sg}(\cdot)$ denotes the stop-gradient operation. The core challenge lies in constructing an accurate regression target g_{tgt}, which we derive as follows.

Recall from our reformulation that:

$$g(a_t, b, t) = a_t - u(a_t, b, t). \quad (14)$$

We substitute $u(a_t, b, t)$ using the velocity-based formulation from the MeanFlow Identity in Eq 7 which leads to:

$$\begin{aligned} g(a_t, b, t) &= a_t - [v(a_t, t) - (t - b)\frac{d}{dt}(a_t - g(a_t, b, t))] \\ &= a_t - [v(a_t, t) - (t - b)(v(a_t, t) - \frac{d}{dt}g(a_t, b, t))]. \end{aligned} \quad (15)$$

To compute the total time derivative $\frac{d}{dt}g(a_t, b, t)$, we apply the chain rule:

$$\frac{d}{dt}g(a_t, b, t) = \frac{da_t}{dt} \cdot \partial_{a_t} g + \frac{db}{dt} \cdot \partial_b g + \frac{dt}{dt} \cdot \partial_t g. \quad (16)$$

Since $\frac{da_t}{dt} = v(a_t, t)$, $\frac{db}{dt} = 0$, and $\frac{dt}{dt} = 1$, we simplify to:

$$\frac{d}{dt}g(a_t, b, t) = v(a_t, t) \cdot \partial_{a_t} g + \partial_t g. \quad (17)$$

Substituting Equation 17 back into Equation 15, finally get:

$$\begin{aligned} g_{tgt} = a_t &+ (t - b - 1) \cdot v(a_t, t) \\ &- (t - b) \cdot [v(a_t, t) \cdot \partial_{a_t} g + \partial_t g]. \end{aligned} \quad (18)$$

Algorithm 1 One-Step Generative Policy with Q-learning

while not converged **do**
  Sample batch $\{(s, a, r, s')\} \sim \mathbb{D}$
  ▷ Train Critic Q_ϕ
  $e \sim \mathbb{N}(0, I_d)$
  $\{a_1, \ldots, a_K\} = \texttt{vmap}(g_\theta)(s', e_{1:K}, b = 0, t = 1)$
  $a' \leftarrow \arg\max_{a_i} Q(s', a_i), \quad a_i \in \{a_1, \ldots, a_K\}$
  Update ϕ to minimize $\mathbb{E}[(Q_\phi(s, a) - r - \gamma Q_{\bar{\phi}}(s', a'))^2]$
  ▷ Train One-Step MeanFlow Policy π_θ
  $t, b \sim \mathbb{U}(0, 1)$
  $a_t = (1 - t)a + te$
  $v = e - a$
  $g, dgdt = \texttt{jvp}(g_\theta, (s, a_t, b, t), (s, v, 0, 1))$
  $g_{\text{tgt}} = a_t + (t - b - 1)v - (t - b)dgdt$
  $err = g - [g_{\text{tgt}}]_{\text{sg}}$
  $\mathcal{L}_{\text{MFI}} = \text{metric}(err)$ ▷ Behaviour Cloning Loss
  $a^\pi \leftarrow g_\theta(s, e)$
  $\mathcal{L}_Q = -Q_\phi(s, a^\pi)$ ▷ Q-learning Loss
  Update θ to minimize $\mathcal{L}_Q + \alpha\mathcal{L}_{\text{MFI}}$
return One-step policy g_θ
function POLICY(s) ▷ One-Step Policy
  $\{a_1, \ldots, a_K\} = \texttt{vmap}(g_\theta)(s, e_{1:K}, b = 0, t = 1)$
  $a' = \arg\max_{a_i} Q(s, a_i), \quad a_i \in \{a_1, \ldots, a_K\}$
  return a'

It is worth noting that the formulation remains agnostic of any network parameterisation. The target relies solely on the instantaneous velocity v and the noisy data a_t as ground-truth signals, without requiring any integral computation. Specifically, recall that $a_t = (1 - t)a + te$; by default, the instantaneous velocity $v(a_t, t)$ simplifies to $e - a$. The gradient terms $\partial_{a_t} g$ and $\partial_t g$ are readily computed using the current policy network g_θ. Full algorithmic details are provided in Algorithm 1.

Value-Guided Action Selection in Bellman Iteration

While performing behaviour cloning, we also need to update the critic function through the Bellman equation iteratively. However, despite the efficiency gains brought by our one-step generative policy, its stochastic sampling of the input noise induces some action-sampling variance, which destabilises Bellman updates and may lead to divergence during critic training. To address this issue, we adopt a value-guided rejection sampling scheme for action selection.

Furthermore, by utilising the `vmap` function in the `JAX` framework, we can sample these candidates in parallel and evaluate them with negligible computational overhead. Formally, we define the value-guided behaviour policy sampling as

$$\pi_\theta^K(a|s) \triangleq \underset{a \in \{a_1, \ldots, a_K \sim g_\theta(s, e, b=0, t=1)\}}{\text{argmax}} Q(s, a),$$

where K actions are sampled simultaneously from the learned policy. The action with the highest Q-value is selected for the Bellman iteration as follows:

$$\mathcal{L}_Q = \mathbb{E}_{(s,a,r,s') \sim \mathcal{D}}\Big[\big(Q_\phi(s, a) - \big(r + \gamma\, Q_{\bar{\phi}}(x', a')\big)\big)^2\Big], \quad (19)$$

where $a' \sim \pi_\theta^K(a|s')$, $\bar{\phi}$ denotes a delayed target network to promote training stability. We incorporate value-guided action sampling into the Bellman update without modifying

the underlying policy model, effectively filtering out low-quality samples and focusing optimisation on more promising regions of the action space. Consequently, it enhances both training stability and final policy performance.

Adaptive Behaviour-Cloning Coefficient

Overall, the training objective for our one-step generative policy with Q-learning can be formalised as follows:

$$\theta^\star \;=\; \arg\max_\theta \mathbb{E}_{\substack{(s,a)\sim\mathcal{D}\\ a^\pi\sim\pi_\theta(\cdot|s)}}\Big[\underbrace{Q_\phi\big(s,a^\pi\big)}_{\text{Q loss}} \;-\; \underbrace{\alpha\,\mathcal{L}_{\text{MFI}}}_{\text{BC loss}}\Big]. \quad (20)$$

Empirically, the choice of α is crucial, as it directly governs the trade-off between behaviour cloning and Q-learning. To adaptively balance their contributions, we dynamically adjust the behaviour cloning coefficient α according to historic training dynamics. Concretely, we maintain a moving average $\overline{\mathcal{L}_Q}$ of the behaviour cloning loss over a sliding window and update α as follows:

$$\alpha \leftarrow \begin{cases} 1.2\times\alpha, & \text{if } \mathcal{L}_Q > 5\cdot\overline{\mathcal{L}_Q},\\ 0.8\times\alpha, & \text{if } \mathcal{L}_Q < 0.2\cdot\overline{\mathcal{L}_Q},\\ \alpha, & \text{otherwise.}\end{cases} \quad (21)$$

Despite differences, this strategy is inspired by the adaptive KL penalty introduced in PPO (Schulman et al. 2017) and ensures that the behaviour cloning regularisation remains neither dominant nor negligible during training. Thus, this stabilises training and results in more balanced policy updates. Empirical evaluations decided the introduced scaling factors.

Experiments

In this section, we present experimental evaluations of our one-step generative policy with Q-learning, comparing its performance against 10 baselines across 73 tasks. Additionally, we conduct analytical studies to help understand our method.

Experimental Setup

Benchmarks. We evaluate our method on a diverse set of offline RL tasks spanning locomotion and manipulation domains. The primary benchmark is the OGBench suite (Park, Li, and Levine 2025), which includes 50 state-based tasks and 5 visual-based tasks across 10 environments: antmaze-large, antmaze-giant, humanoidmaze-medium, humanoidmaze-large, antsoccer-arena, cube-single, cube-double, scene-play, puzzle-3×3, and puzzle-4×4. Additionally, we include 18 challenging tasks from the D4RL benchmark (Fu et al. 2020), covering antmaze and adroit manipulation scenarios. OGBench tasks feature semi-sparse rewards and complex behavioural distributions, while D4RL tasks are known for their high-dimensional action spaces.

Baselines. We compare our method against ten representative baselines spanning three policy classes. *Gaussian policies* include behavioural Cloning (BC) (Pomerleau 1988), Implicit Q-Learning (IQL) (Kostrikov, Nair, and Levine 2021), and ReBRAC (Tarasov et al. 2023a). *Diffusion-based policies* include IDQL (Hansen-Estruch et al. 2023), SRPO (Chen et al. 2023a), and Consistency Actor-Critic (CAC) (Ding and Jin 2024). *Flow-based policies* include FAWAC (Nair et al. 2020), FBRAC (Zhang, Zhang, and Gu 2025), IFQL (Wang, Hunt, and Zhou 2022), and Flow Q-Learning (FQL) (Park, Li, and Levine 2025). *Specifically designed for online RL* include Cal-QL (Nakamoto et al. 2023) and PLPD (Ball et al. 2023).

Evaluation metrics. For offline RL, we assess the performance of the methods after a fixed number of gradient steps. Specifically, following (Tarasov et al. 2023b) we avoid reporting the best performance across different evaluation epochs to prevent potential bias in the results. In our experiments, we employ 8 random seeds for state-based tasks and 4 seeds for pixel-based tasks. For the OGBench locomotion and manipulation tasks, performance is measured as the percentage of episodes that achieve the goal. For the D4RL Adroit tasks, we measure binary task success rates (in percentage) for antmaze and normalized returns for adroit, following the original evaluation scheme (Fu et al. 2020). We refer to Appendix D for the full training and evaluation details.

Comparisons with SOTA

Offline RL performance. Following FQL (Park, Li, and Levine 2025), we evaluate our method on the offline subset of the OGBench and D4RL benchmarks, which include diverse tasks with varying dynamics and action dimensionalities. Table shows that our approach achieves strong performance across a wide range of tasks, particularly in environments such as OGBench antmaze, humanoidmaze, puzzle, and visual manipulation, which likely demand expressive and multimodal policy behaviours due to complex dynamics.

Compared to flow-based baselines such as FQL, IFQL, and FBRAC, our method either matches or surpasses their performance while maintaining a simpler one-stage training pipeline and a one-step inference mechanism. Although performance on particularly sparse environments remains modest—such as in giant maze navigation—the results are consistent with broader trends observed across baselines, indicating inherent task difficulty rather than model-specific limitations. (We additionally add an online-finetuning extension on this task as shown in Table and show that although initial offline performance was limited, our method achieved strong improvements after online finetuning, suggesting the bottleneck may lie in the offline dataset.) Overall, the results indicate that our one-step generative policy serves as a viable and competitive approach for offline reinforcement learning.

Offline-to-online transfer. We further evaluate our method in offline-to-online settings following (Park, Li, and Levine 2025) (while adding antmaze-giant-navigate), where policies are first pre-trained on static offline datasets and then fine-tuned through online interaction. As shown in Table , our method consistently improves after fine-tuning and achieves strong final performance across a diverse set of tasks. More notably, although the offline performances are not ideal on antmaze-giant-navigate and cube-double-play, our method gains a significant performance boost in the online adaptation.

These results highlight the robustness and sample efficiency of our approach when transitioning from offline training to online adaptation. Notably, these improvements are obtained without any modifications to the model architecture or

Task Category	Gaussian Policies			Diffusion Policies			Flow Policies				
	BC	IQL	ReBRAC	IDQL	SRPO	CAC	FAWAC	FBRAC	IFQL	FQL	Ours
OGBench antmaze-large-singletask (**5** tasks)	11 ±1	53 ±3	**81** ±5	21 ±5	11 ±4	33 ±4	6 ±1	60 ±6	28 ±5	79 ±3	**81** ±3
OGBench antmaze-giant-singletask (**5** tasks)	0 ±0	4 ±1	**26** ±8	0 ±0	0 ±0	0 ±0	0 ±0	4 ±4	3 ±2	9 ±6	0 ±0
OGBench humanoidmaze-medium-singletask (**5** tasks)	2 ±1	33 ±2	22 ±8	1 ±0	1 ±1	53 ±8	19 ±1	38 ±5	60 ±14	58 ±5	**62** ±1
OGBench humanoidmaze-large-singletask (**5** tasks)	1 ±0	2 ±1	2 ±1	1 ±0	0 ±0	0 ±0	0 ±0	2 ±0	11 ±2	4 ±2	**20** ±3
OGBench antsoccer-arena-singletask (**5** tasks)	1 ±0	8 ±2	0 ±0	12 ±4	1 ±0	2 ±4	12 ±0	16 ±1	33 ±6	60 ±2	**62** ±3
OGBench cube-single-singletask (**5** tasks)	5 ±1	83 ±3	91 ±2	**95** ±2	80 ±5	85 ±9	81 ±4	79 ±7	79 ±2	**96** ±1	**95** ±2
OGBench cube-double-singletask (**5** tasks)	2 ±1	7 ±1	12 ±1	15 ±6	2 ±1	6 ±2	5 ±2	15 ±3	14 ±3	**29** ±2	3 ±2
OGBench scene-singletask (**5** tasks)	5 ±1	28 ±1	41 ±3	46 ±3	20 ±1	40 ±7	30 ±3	45 ±5	30 ±3	56 ±2	**60** ±1
OGBench puzzle-3x3-singletask (**5** tasks)	2 ±0	9 ±1	21 ±1	10 ±2	18 ±1	19 ±0	6 ±2	14 ±4	19 ±1	30 ±1	**66** ±8
OGBench puzzle-4x4-singletask (**5** tasks)	0 ±0	7 ±1	14 ±1	29 ±3	10 ±3	15 ±3	1 ±0	13 ±1	25 ±5	17 ±2	**40** ±6
D4RL antmaze (**6** tasks)	17	57	78	79	74	30 ±3	44 ±3	64 ±7	65 ±7	**84** ±3	**83** ±2
D4RL adroit (**12** tasks)	48	53	**59**	52 ±1	51 ±1	43 ±2	48 ±1	50 ±2	52 ±1	52 ±1	54 ±3
Visual manipulation (**5** tasks)	-	42 ±4	60 ±2	-	-	-	-	22 ±2	50 ±5	**65** ±2	55 ±2

[1] Due to the high computational cost of pixel-based tasks, we selectively benchmark 5 methods that achieve strong performance on state-based OGBench tasks.

Table 1: Offline RL results. Our method achieves the best or near-best performance on most of the 73 diverse, challenging benchmark tasks. The performances are averaged over 8 seeds (4 seeds for pixel-based tasks), but the cells without the ± sign indicate that the numbers are taken from prior works (Tarasov et al. 2023b; Hansen-Estruch et al. 2023; Chen et al. 2023a).

Task	IQL	ReBRAC	Cal-QL	RLPD	IFQL	FQL	Ours
antmaze-giant-navigate	2 ±1 → 3 ±1	6 ±5 → 34 ±4	0 ±0 → 0 ±0	0 ±0 → 48 ±13	0 ±0 → 69 ±12	4 ±2 → 9 ±3	0 ±0 → **82** ±5
humanoidmaze-medium-navigate	21 ±13 → 16 ±8	16 ±20 → 1 ±1	0 ±0 → 0 ±0	0 ±0 → 8 ±10	56 ±35 → 82 ±20	12 ±7 → 22 ±12	62 ±1 → **100** ±1
antsoccer-arena-navigate	2 ±1 → 0 ±0	0 ±0 → 0 ±0	0 ±0 → 0 ±0	0 ±0 → 0 ±0	26 ±15 → 39 ±10	28 ±8 → **86** ±5	62 ±3 → **87** ±5
cube-double-play	0 ±1 → 0 ±0	6 ±5 → 28 ±28	0 ±0 → 0 ±0	0 ±0 → 0 ±0	12 ±9 → 40 ±5	40 ±11 → **92** ±3	3 ±2 → **95** ±2
scene-play	14 ±11 → 10 ±9	55 ±10 → **100** ±0	1 ±2 → 50 ±53	0 ±0 → **100** ±0	0 ±1 → 60 ±39	82 ±11 → **100** ±1	60 ±1 → **100** ±1
puzzle-4x4-play	5 ±2 → 1 ±1	8 ±4 → 14 ±35	0 ±0 → 0 ±0	0 ±0 → **100** ±1	23 ±6 → 19 ±33	8 ±3 → 38 ±52	40 ±6 → **100** ±1
antmaze-umaze-v2	77 → **96**	98 → 75	77 → **100**	0 ±0 → **98** ±3	94 ±5 → **96** ±2	97 ±2 → **99** ±1	98 ±1 → **99** ±1
antmaze-umaze-diverse-v2	60 → 64	74 → **98**	32 → **98**	0 ±0 → 94 ±5	69 ±20 → 93 ±5	79 ±16 → **100** ±1	79 ±2 → **100** ±1
antmaze-medium-play-v2	72 → 90	88 → **98**	72 → **99**	0 ±0 → **98** ±2	52 ±19 → 93 ±2	77 ±7 → **97** ±2	86 ±2 → **99** ±1
antmaze-medium-diverse-v2	64 → 92	85 → **99**	62 → **98**	0 ±0 → **97** ±2	44 ±26 → 89 ±4	55 ±19 → **97** ±3	78 ±2 → **99** ±2
antmaze-large-play-v2	38 → 64	68 → 32	32 → **97**	0 ±0 → **93** ±5	64 ±14 → 80 ±5	66 ±40 → 84 ±30	80 ±3 → **94** ±4
antmaze-large-diverse-v2	27 → 64	67 → 72	44 → **92**	0 ±0 → **94** ±3	69 ±6 → 86 ±5	75 ±24 → **94** ±3	76 ±1 → **96** ±3
pen-cloned-v1	84 → 102	74 → 138	−3 → −3	3 ±2 → 120 ±10	77 ±7 → 107 ±10	53 ±14 → **149** ±6	79 ±3 → **151** ±7
door-cloned-v1	1 → 20	0 → **102**	0 → 0	0 ±0 → **102** ±7	3 ±2 → 50 ±15	0 ±0 → **102** ±5	3 ±1 → 96 ±4
hammer-cloned-v1	1 → 57	7 → 125	0 → 0	0 ±0 → 128 ±29	4 ±2 → 60 ±14	0 ±0 → 127 ±17	10 ±5 → **132** ±11
relocate-cloned-v1	0 → 0	1 → 7	0 → 0	0 ±0 → 2 ±2	0 ±0 → 5 ±3	0 ±1 → **62** ±8	1 ±1 → 19 ±8

Table 2: Offline-to-online RL results. Following FQL (Park, Li, and Levine 2025), the results are obtained by first training the policy offline for 1M steps and subsequently performing online fine-tuning.

training objective, allowing the policy to interact with the environment in a more reward-sensitive and direct manner. This unified design facilitates more effective exploration, which is critical for rapid adaptation from offline to online. Overall, the strong performance of our generative policy in both static and interactive settings highlights its potential for real-world deployment in evolving environments.

Further Analysis

Our method vs naive MeanFlow. To evaluate the effectiveness of our proposed reformulation strategy, we compare it against the original MeanFlow framework under the same environment and with well-tuned hyperparameters. As shown in Fig. 3, the naive MeanFlow approach demonstrates performance (left) and flow loss (mid.) metrics that are nearly indistinguishable from ours during the initial stages of training. However, the bound loss (right) metric—defined as the mean absolute magnitude of action values outside $[-1, 1]$—is substantially higher for the naive MeanFlow approach. This indicates it predicts actions not in the valid range. Thus, its performance then degrades significantly due to inaccurate Q-value estimates from the critic, which impair the training of the generative flow model and result in a low success rate.

Analysis of coefficient α in policy learning. The hyperparameter α controls the relative strength of the behaviour cloning term versus the Q-function objective in the training loss. As shown in Fig. 4, overly small values of α lead to unstable updates due to over-reliance on potentially noisy Q estimates, while overly large values lead to simply behaviour cloning learning. Specifically, although we employ an adaptive α tuning mechanism in our main experiments, the model's final performance still exhibits sensitivity to initial α values, especially in sparse-reward tasks, e.g. Puzzle3x3. This highlights the importance of a balanced trade-off, and motivates future research on more robust adaptive schemes.

To further examine the sensitivity of our method to the coefficient α, we analyze its evolution during training under the proposed adaptive adjustment strategy. As shown in Fig. 5, we can see α evolves smoothly and remains well-regulated throughout learning based on our adaptive mechanism. This dynamic adaptation effectively balances the trade-off between behaviour cloning and value maximisation, thereby enhancing the robustness of our method across different train-

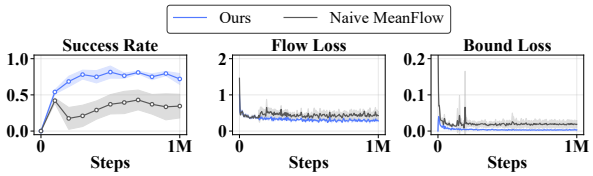

Figure 3: Comparison of learning curves. Naive MeanFlow exhibits significantly larger bound losses in the early training stages. This instability directly undermines convergence of its flow loss, leading to a substantial performance gap compared to ours on the antsoccer-arena-singletask.

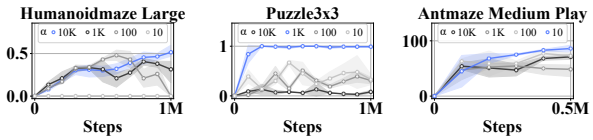

Figure 4: Sensitivity to the behaviour cloning coefficient α. Effect of different values of α on offline RL performance.

ing stages and contributing to our superior performance over FQL (Park, Li, and Levine 2025).

Effect of candidate size K in value-guided selection. In our value-guided rejection sampling, hyperparameter K controls the number of candidate actions generated and scored by the Q-function for each state. As shown in Fig. 6, we observe that increasing K from 1 to 5 leads to a substantial performance gain across various tasks. However, setting K too large (e.g., 20) may impair performance by reducing stochasticity and exploration diversity, as overly greedy selection can cause the critic function to focus on a narrow set of high-value actions, thereby limiting behavioural variability. In practice, we find that $K = 5$ offers the best trade-off, preserving policy exploration and maintaining computational efficiency.

Related Works

Generalised behaviour cloning. Behaviour cloning (BC) casts policy learning as a supervised task over expert state-action pairs (Hussein et al. 2017; Block et al. 2023). Generalised variants, such as advantage-weighted imitation (Peters and Schaal 2007), steer learning toward high-return trajectories by weighting actions by their cumulative rewards. Recent methods extend this idea in different ways: SfBC (Chen et al. 2023b) and IDQL (Hansen-Estruch et al. 2023) pretrain a behaviour policy and critic to support importance sampling; QGPO (Lu et al. 2023) distills the critic into an energy-based model; GMPO and GMPG (Zhang et al. 2024) apply weighted policy gradient updates directly without pretraining; and QIPO (Zhang, Zhang, and Gu 2025) incorporates Q-value-derived energy into flow-matching updates. Despite these advances, recent studies highlight their inefficiency and limited performance in complex tasks (Park et al. 2024).

Generative policy with Q-learning. Incorporating Q-functions into generative policies, enabling direct reward-driven optimisation, offers a more principled alternative to behaviour cloning. These works often optimise policies with Q-learning by differentiating through the generative

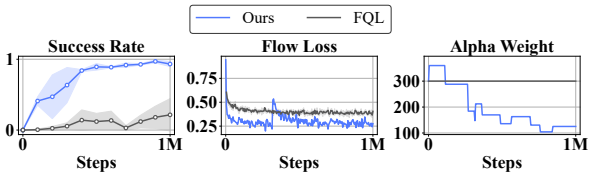

Figure 5: Training dynamics under our framework in humanoidmaze-large-navigate-singletask. Despite potential sensitivity, the adaptive coefficient α adjusts dynamically and evolves smoothly, contributing to stable and robust policy learning throughout the training process.

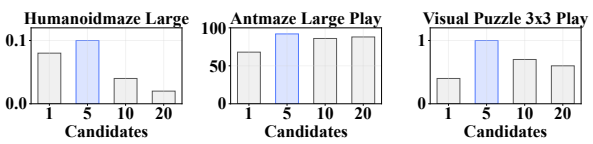

Figure 6: Performance effect of candidates number K used in value-guided rejection sampling on three representative tasks.

model's reverse process. Methods such as DQL (Wang, Hunt, and Zhou 2022), Consistency-AC (Ding and Jin 2024), and DACER (Wang et al. 2024) perform policy improvement via reverse diffusion, but suffer from high computational cost due to repeated denoising and unrolled optimisation. To alleviate this, one-step generative policies have recently been proposed to avoid iterative sampling (Chen et al. 2024; Prasad et al. 2024; Koirala and Fleming 2025). Existing approaches follow two main directions: distillation-based methods that compress multi-step teachers into one-step student policies (Chen et al. 2023a; Park, Li, and Levine 2025), and algorithmic innovations that enable direct action generation, such as consistency (Song et al. 2023) and shortcut (Frans et al. 2025) models. Among them, FQL (Park, Li, and Levine 2025) distils a flow-based policy into a one-step model optimised with Q-learning, while SORL (Espinosa-Dice et al. 2025) leverages shortcut frameworks (Frans et al. 2025) to flexibly scale training and inference. However, both approaches necessitate multi-stage pipelines or inference-time scaling, undermining their potential as effective one-step policy frameworks.

Conclusion

We propose a one-step generative framework for offline RL that enables expressive and stable policy learning without relying on multi-step integration or policy distillation. Through a residual reformulation of MeanFlow, our method supports direct Q-learning with multimodal policies via a single-stage training pipeline. To further improve performance and robustness, we incorporate value-guided rejection sampling and adaptive behaviour cloning regularisation. Experiment results show our method outperforming or matching prior work on most of the 73 tasks and reaching state-of-the-art performance on nearly all offline-to-online transfers, highlighting robust generalisation and adaptation. Overall, this work bridges the gap between one-step generative modelling and value-based reinforcement learning, advancing the development of more expressive and scalable offline policy learning.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant No. 62206305. Zeyuan Wang completed this work during the author's visiting study at the Shanghai Innovation Institute, whose support are gratefully acknowledged. Prof. Yanwei Fu was partly supported by the project from the Ministry of Science and Higher Education of the Republic of Kazakhstan, No. BR24992975, 'Development of a digital twin of a food processing enterprise using artificial intelligence and IIoT technologies' (2024–2026).

References

Ball, P. J.; Smith, L.; Kostrikov, I.; and Levine, S. 2023. Efficient online reinforcement learning with offline data. In *International Conference on Machine Learning (ICML)*.

Block, A.; Jadbabaie, A.; Pfrommer, D.; Simchowitz, M.; and Tedrake, R. 2023. Provable guarantees for generative behavior cloning: Bridging low-level stability and high-level behavior. *Advances in Neural Information Processing Systems (NeurIPS)*.

Chen, H.; Lu, C.; Wang, Z.; Su, H.; and Zhu, J. 2023a. Score regularized policy optimization through diffusion behavior. *arXiv preprint arXiv:2310.07297*.

Chen, H.; Lu, C.; Ying, C.; Su, H.; and Zhu, J. 2023b. Offline Reinforcement Learning via High-Fidelity Generative Behavior Modeling. In *The Eleventh International Conference on Learning Representations (ICLR)*.

Chen, J.; Ganguly, B.; Xu, Y.; Mei, Y.; Lan, T.; and Aggarwal, V. 2024. Deep generative models for offline policy learning: Tutorial, survey, and perspectives on future directions. *arXiv preprint arXiv:2402.13777*.

Chen, T.; Wang, Z.; and Zhou, M. 2024. Diffusion policies creating a trust region for offline reinforcement learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Chi, C.; Xu, Z.; Feng, S.; Cousineau, E.; Du, Y.; Burchfiel, B.; Tedrake, R.; and Song, S. 2023. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*.

Ding, Z.; and Jin, C. 2024. Consistency models as a rich and efficient policy class for reinforcement learning. In *12th International Conference on Learning Representations (ICLR)*.

Espinosa-Dice, N.; Zhang, Y.; Chen, Y.; Guo, B.; Oertell, O.; Swamy, G.; Brantley, K.; and Sun, W. 2025. Scaling Offline RL via Efficient and Expressive Shortcut Models. *arXiv preprint arXiv:2505.22866*.

Esser, P.; Kulal, S.; Blattmann, A.; Entezari, R.; Müller, J.; Saini, H.; Levi, Y.; Lorenz, D.; Sauer, A.; Boesel, F.; et al. 2024. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first International conference on machine learning (ICML)*.

Frans, K.; Hafner, D.; Levine, S.; and Abbeel, P. 2025. One Step Diffusion via Shortcut Models. In *The Thirteenth International Conference on Learning Representations (ICLR)*.

Fu, J.; Kumar, A.; Nachum, O.; Tucker, G.; and Levine, S. 2020. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*.

Geng, Z.; Deng, M.; Bai, X.; Kolter, J. Z.; and He, K. 2025. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447*.

Hafner, R.; and Riedmiller, M. 2011. Reinforcement learning in feedback control. *Machine Learning*.

Hansen-Estruch, P.; Kostrikov, I.; Janner, M.; Kuba, J. G.; and Levine, S. 2023. Idql: Implicit q-learning as an actor-critic method with diffusion policies. *arXiv preprint arXiv:2304.10573*.

Hussein, A.; Gaber, M. M.; Elyan, E.; and Jayne, C. 2017. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*.

Jin, Y.; Sun, Z.; Li, N.; Xu, K.; Xu, K.; Jiang, H.; Zhuang, N.; Huang, Q.; Song, Y.; MU, Y.; and Lin, Z. 2025. Pyramidal Flow Matching for Efficient Video Generative Modeling. In *The Thirteenth International Conference on Learning Representations (ICLR)*.

Koirala, P.; and Fleming, C. 2025. Flow-Based Single-Step Completion for Efficient and Expressive Policy Learning. *arXiv preprint arXiv:2506.21427*.

Kostrikov, I.; Nair, A.; and Levine, S. 2021. Offline Reinforcement Learning with Implicit Q-Learning. In *Deep RL Workshop NeurIPS 2021*.

Leshno, M.; Lin, V. Y.; Pinkus, A.; and Schocken, S. 1993. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks*.

Lipman, Y.; Chen, R. T.; Ben-Hamu, H.; Nickel, M.; and Le, M. 2023. Flow Matching for Generative Modeling. In *11th International Conference on Learning Representations (ICLR)*.

Liu, A. H.; Le, M.; Vyas, A.; Shi, B.; Tjandra, A.; and Hsu, W.-N. 2024. Generative Pre-training for Speech with Flow Matching. In *The Twelfth International Conference on Learning Representations (ICLR)*.

Liu, X.; Gong, C.; and Liu, Q. 2023. Flow Straight and Fast: Learning to Generate and Transfer Data with Rectified Flow. In *The Eleventh International Conference on Learning Representations (ICLR)*.

Lu, C.; Chen, H.; Chen, J.; Su, H.; Li, C.; and Zhu, J. 2023. Contrastive energy prediction for exact energy-guided diffusion sampling in offline reinforcement learning. In *Forty-first International conference on machine learning (ICML)*.

Mees, O.; Ghosh, D.; Pertsch, K.; Black, K.; Walke, H. R.; Dasari, S.; Hejna, J.; Kreiman, T.; Xu, C.; Luo, J.; et al. 2024. Octo: An open-source generalist robot policy. In *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA*.

Nair, A.; Gupta, A.; Dalal, M.; and Levine, S. 2020. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*.

Nakamoto, M.; Zhai, S.; Singh, A.; Sobol Mark, M.; Ma, Y.; Finn, C.; Kumar, A.; and Levine, S. 2023. Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning. *Advances in Neural Information Processing Systems (NeurIPS)*.

O'Neill, A.; Rehman, A.; Maddukuri, A.; Gupta, A.; Padalkar, A.; Lee, A.; Pooley, A.; Gupta, A.; Mandlekar, A.; Jain, A.; et al. 2024. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*.

Park, S.; Frans, K.; Eysenbach, B.; and Levine, S. 2025. OGBench: Benchmarking Offline Goal-Conditioned RL. In *The Thirteenth International Conference on Learning Representations (ICLR)*.

Park, S.; Frans, K.; Levine, S.; and Kumar, A. 2024. Is value learning really the main bottleneck in offline RL? *Advances in Neural Information Processing Systems (NeurIPS)*.

Park, S.; Li, Q.; and Levine, S. 2025. Flow Q-Learning. In *Forty-second International conference on machine learning (ICML)*.

Peters, J.; and Schaal, S. 2007. Reinforcement learning by reward-weighted regression for operational space control. In *Proceedings of the 24th International conference on machine learning (ICML)*.

Pomerleau, D. A. 1988. Alvinn: An autonomous land vehicle in a neural network. *Advances in Neural Information Processing Systems (NeurIPS)*.

Prasad, A.; Lin, K.; Wu, J.; Zhou, L.; and Bohg, J. 2024. Consistency Policy: Accelerated Visuomotor Policies via Consistency Distillation. *CoRR*.

Puterman, M. L. 2014. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Sheng, J.; Wang, Z.; Li, P.; and Liu, M. 2025. MP1: Mean Flow Tames Policy Learning in 1-step for Robotic Manipulation. *arXiv preprint arXiv:2507.10543*.

Song, Y.; Dhariwal, P.; Chen, M.; and Sutskever, I. 2023. Consistency Models. In *International conference on machine learning (ICML)*.

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

Tabuada, P.; and Gharesifard, B. 2021. Universal approximation power of deep residual neural networks via nonlinear control theory. In *International Conference on Learning Representations (ICLR)*.

Tarasov, D.; Kurenkov, V.; Nikulin, A.; and Kolesnikov, S. 2023a. Revisiting the minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems (NeurIPS)*.

Tarasov, D.; Nikulin, A.; Akimov, D.; Kurenkov, V.; and Kolesnikov, S. 2023b. CORL: Research-oriented deep offline reinforcement learning library. *Advances in Neural Information Processing Systems (NeurIPS)*.

Wang, Y.; Wang, L.; Jiang, Y.; Zou, W.; Liu, T.; Song, X.; Wang, W.; Xiao, L.; Wu, J.; Duan, J.; et al. 2024. Diffusion actor-critic with entropy regulator. *Advances in Neural Information Processing Systems (NeurIPS)*.

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

Zhang, J.; Xue, R.; Niu, Y.; Chen, Y.; Yang, J.; Li, H.; and Liu, Y. 2024. Revisiting Generative Policies: A Simpler Reinforcement Learning Algorithmic Perspective. *arXiv preprint arXiv:2412.01245*.

Zhang, Q.; Liu, Z.; Fan, H.; Liu, G.; Zeng, B.; and Liu, S. 2025. Flowpolicy: Enabling fast and robust 3d flow-based policy via consistency flow matching for robot manipulation. In *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

Zhang, S.; Zhang, W.; and Gu, Q. 2025. Energy-weighted flow matching for offline reinforcement learning. *arXiv preprint arXiv:2503.04975*.