Learning Multimodal One-step Flow Policy via Value-weighted Optimal Transport

Jaehun Shon* **Jinha Choi*** **Jongwook Jeon** **Jongmin Lee†**
Department of Artificial Intelligence, Yonsei University
{manfromearth, danielc174, jjwook2602, jongminlee}@yonsei.ac.kr
*Equal contribution †Corresponding author

Abstract

Offline reinforcement learning aims to learn a policy solely from fixed datasets, which often contain multimodal action distributions. Flow policies can naturally represent such multimodal behaviors, but learning an efficient one-step flow policy remains challenging: standard value guidance often leads to mode collapse or exploits overestimation bias in out-of-distribution regions. To address this, we introduce **One-step Flow policy via Optimal Transport (OptiFlow)**, a framework for one-step flow policy learning as a structured sample-allocation problem. OptiFlow jointly trains a value-aware reference flow policy and an efficient one-step policy, coupling their action samples through state-wise entropic optimal transport. For each state, critic-estimated values define the priority of distillation target actions, while the action-distance cost ensures geometrically compatible pairings. By avoiding direct critic maximization, our transport-guided approach enables in-distribution exploitation by anchoring the one-step policy to high-value, dataset-supported modes without the risk of out-of-distribution divergence. Experimental results demonstrate that OptiFlow effectively captures optimal multimodal behaviors and achieves strong performance across diverse offline RL benchmarks. Our code is available at https://github.com/Yonsei-DILLab/OptiFlow.

1 Introduction

Offline reinforcement learning (RL) aims to learn decision-making policies from fixed datasets without further environment interaction [1, 2]. This setting is crucial in domains such as robotics and control, where online exploration can be costly or unsafe. A central challenge is distribution shift: when a learned policy selects actions poorly supported by the offline data, value estimates can become unreliable and be amplified during policy optimization. Standard methods mitigate this through conservative value estimation [3, 4], behavior-constrained policy learning [5, 6], or distribution correction via dual optimization [7, 8].

Beyond distribution shift, modern offline datasets often exhibit multimodal behavior: for a given state, multiple distinct actions may be valid and lead to high return. Standard unimodal policies can struggle in such settings, often collapsing toward a mean action that is suboptimal or physically invalid [6, 9]. This issue is especially important in high-dimensional manipulation and control tasks, where multiple successful strategies may coexist. Expressive generative policies, including diffusion-based [10–12] and flow-based [13] policies, provide a natural way to model such multimodal action distributions [14–18]. Flow policies are particularly appealing because their deterministic probability paths make one-step policy learning a natural target for efficient deployment [11, 12].

However, learning an effective one-step flow policy remains challenging. Multi-step sampling is often too expensive for real-time control, but compressing a reference policy into one step can lose multimodal structure [19]. Pointwise distillation, which matches the same noise input to a target action, can force a one-step model to memorize complex paths and collapse distinct behaviors into

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

compromised averages [20]. Direct critic maximization introduces a different failure mode: the policy may exploit local optima or overestimated out-of-distribution actions instead of capturing high-return in-distribution behavior.

We address these limitations by reframing one-step flow policy learning as a structured sample-allocation problem. We propose **Flow Policy via Optimal Transport (OptiFlow)**, which couples samples from a value-aware reference flow policy and an efficient one-step policy through entropic optimal transport [21]. OptiFlow separates value from geometry: critic-estimated values determine the supervision mass by prioritizing high-return reference actions, while the transport cost encourages geometrically compatible pairings between one-step-policy samples and reference actions.

This transport-guided update avoids direct critic maximization of the deployed policy. Instead of following unreliable critic gradients into OOD regions or local optima, the one-step policy is anchored to high-value actions proposed by the behavior-regularized reference policy. At the same time, by relaxing strict pointwise matching OptiFlow lets the one-step policy flexibly map noise to actions and preserve multimodal reference-policy structure. We empirically show that OptiFlow captures multimodal behavior in controlled diagnostics and achieves strong performance across offline RL benchmarks.

2 Related Work

Expressive generative policies for offline RL. Recent work has introduced diffusion and flow-based policies as expressive alternatives to unimodal actors for modeling multimodal action distributions in offline RL [14–18]. The same expressiveness has also been leveraged in multi-agent imitation learning, where decentralized diffusion policies recover multiple coordination modes from multimodal demonstrations [22]. FQL [16] learns flow-based policies and uses critic-guided policy improvement to obtain strong offline RL performance. GFP [17] introduces value-aware behavior cloning for flow policies together with one-step distillation. OptiFlow builds on this line of work, but addresses a different question: how should value-guided supervision be allocated across multiple action samples when the reference flow policy contains several high-value modes? Rather than directly optimizing the one-step policy against the critic or applying guidance only at inference time, OptiFlow uses critic-estimated values to define a reference-policy marginal in an optimal-transport coupling.

Value-guided policy learning and behavior constraints. A central theme in offline RL is to improve policy quality while avoiding unsupported actions. Prior methods address this through conservative value estimation [3, 4], behavior regularization [5, 6], or value-weighted policy extraction [23, 24]. These approaches provide different mechanisms for balancing value improvement and support preservation. OptiFlow follows the same broad principle, but uses value information differently: critic-estimated values do not directly optimize the deployed one-step policy, but instead shape the reference-policy marginal used for transport-guided supervision.

Matching, retrieval, and optimal transport. Several offline RL and imitation learning methods use matching or retrieval mechanisms to keep learned policies close to data-supported actions. For example, retrieval-based policy constraints regularize the policy toward nearby dataset actions [25]. Optimal transport has also been used to compare or align behavior distributions in imitation learning and offline decision making [26–28]. OptiFlow differs in both purpose and construction. We do not use optimal transport merely as a distributional distance or trajectory-level alignment objective. Instead, OptiFlow solves a state-wise entropic optimal transport problem between one-step-policy and reference-policy action samples. The resulting coupling combines a value-weighted reference-policy marginal with action-space geometry and produces transport-guided regression targets for efficient one-step flow-policy learning.

3 Preliminaries

3.1 Offline Reinforcement Learning

We consider a discounted Markov decision process $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, r, \rho_0, \gamma \rangle$ [29], where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces, $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ is the transition kernel, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the

reward function, $\rho_0 \in \Delta(\mathcal{S})$ is the initial-state distribution, and $\gamma \in [0, 1)$ is the discount factor. In offline RL, the learner is given a fixed dataset $\mathcal{D} = \{(s_i, a_i, r_i, s'_i)\}_{i=1}^{|\mathcal{D}|}$ collected by potentially diverse and unknown behavior policies and cannot collect additional data from environment interactions.

For a policy π, we write $Q^\pi(s, a)$ and $V^\pi(s)$ for its action-value and value functions, $J(\pi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$ for its discounted return, and $d^\pi(s) = (1-\gamma)\sum_{t=0}^{\infty} \gamma^t \Pr_\pi(s_t = s)$ for its discounted state occupancy. OptiFlow learns a critic estimate $Q_\phi(s, a)$ from $\mathcal{D}$ and uses critic-estimated values to construct a value-weighted transport problem for updating a one-step flow policy.

3.2 Flow Policies

Flow matching [13, 30] learns a time-dependent vector field that transports samples from a simple base distribution to a target distribution. In offline RL, we apply this to conditional action generation, where the vector field $v_\theta(s, x_t, t)$ is conditioned on the state s. For behavioral cloning, we set the base sample as noise $z \sim \mathcal{N}(0, I)$ and the target action as an action $a \sim \mathcal{D}$ from the dataset. Defining the linear interpolation $x^t = (1-t)x^0 + tx^1$ for $t \in [0, 1]$, $z = x^0$, and $a = x^1$, the most basic flow-matching objective for BC is as follows:

$$\min_\theta \mathcal{L}_{\mathrm{BC}}(\theta) = \mathbb{E}_{\substack{(s,a=x^1)\sim\mathcal{D},\\ z=x^0\sim\mathcal{N}(0,I_d),\\ t\sim\mathrm{Unif}([0,1])}} \left[\left\| v_\theta(s, x^t, t) - (x^1 - x^0)\right\|_2^2\right]. \tag{1}$$

At inference time, an action is generated by sampling $z \sim \mathcal{N}(0, I_d)$ and integrating the ODE from $t = 0$ to $t = 1$: $\frac{dx^t}{dt} = v_\theta(s, x^t, t)$, starting from $z = x^0$, to obtain $a = x^1$. In practice, this integration is approximated by a finite number of Euler steps:

$$x^{k+1} = x^k + \tfrac{1}{K} v_\theta\left(s, x^k, \tfrac{k}{K}\right) \text{ for } k = 0, \ldots, K-1. \tag{2}$$

Larger K gives an expressive multi-step flow policy but requires iterative action generation. To enable efficient deployment, a one-step flow policy directly learns a mapping $a = \mu_\theta(s, z)$ from noise to action, which induces a stochastic policy $\pi_\theta(a|s)$ due to the stochasticity of the base noise z. OptiFlow uses a value-aware *multi-step* reference flow policy to model multimodal behavior and learns an efficient *one-step* flow policy $\mu_\theta(s, z)$ through transport-guided updates.

3.3 Entropic Optimal Transport

Let $p \in \Delta^N$ and $q \in \Delta^M$ be discrete probability distributions over two finite sets, and let $C \in \mathbb{R}^{N\times M}$ be a cost matrix. The discrete optimal transport problem seeks a coupling $P \in \mathbb{R}_+^{N\times M}$ whose marginals match p and q:

$$\Pi(p, q) = \left\{P \geq 0 \mid P\mathbf{1}_M = p,\ P^\top \mathbf{1}_N = q\right\}. \tag{3}$$

Exact optimal transport requires solving a linear program, which scales poorly and cannot efficiently leverage GPU parallelization. For efficient and scalable computation, we consider the entropy-regularized optimal transport problem [21]:

$$P^* = \operatorname*{arg\,min}_{P\in\Pi(p,q)} \Big[\langle P, C\rangle + \varepsilon \sum_{i,j} P_{ij}(\log P_{ij} - 1)\Big], \tag{4}$$

where $\varepsilon > 0$ controls the smoothness of the transport plan. This regularized problem admits a unique solution that can be computed efficiently using the Sinkhorn algorithm [21], an iterative matrix scaling procedure. The detailed formulation and the corresponding pseudocode are provided in Appendix C. In the following section, we detail how OptiFlow leverages this optimal transport framework state-wise to align an efficient one-step flow policy with a multimodal reference policy.

4 Flow Policy via Optimal Transport

We introduce **Flow Policy via Optimal Transport (OptiFlow)**, a framework that learns efficient one-step flow policies for offline RL through a value-weighted optimal transport formulation. While one-step flow policies are expressive enough to represent multimodal action distributions in principle,

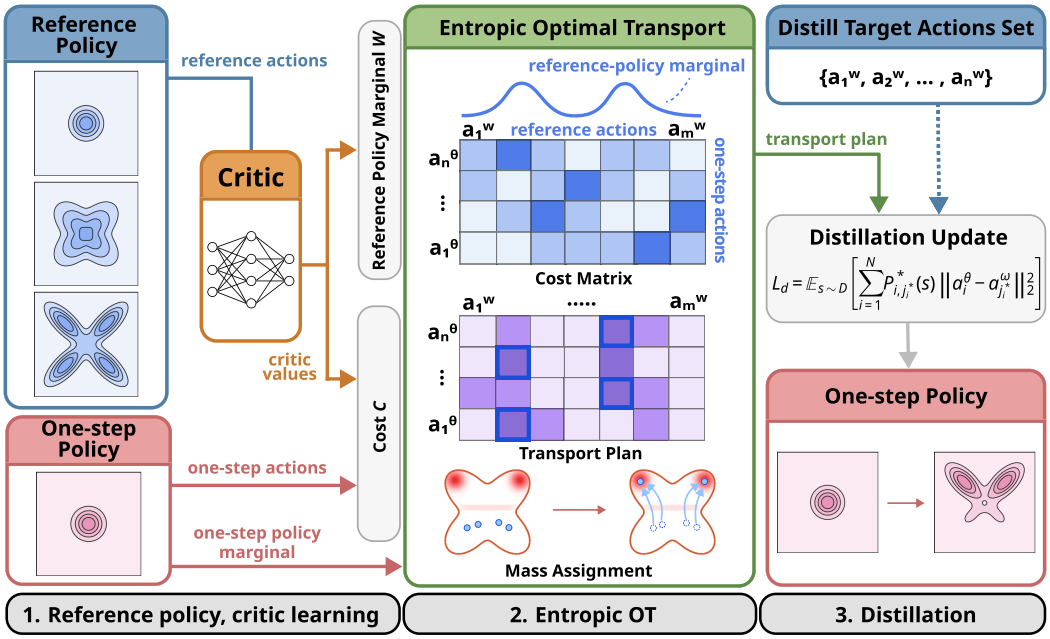


Figure 1: Overview of OptiFlow. OptiFlow learns a critic and a reference flow policy from offline data, then couples reference and one-step policy action samples with entropic optimal transport. Critic values define a reference marginal over reference actions, and the resulting transport plan provides structured distillation targets for the one-step policy.

existing approaches suffer from two distinct failure modes: (i) Mode collapse during mapping, where pointwise noise-to-action matching is prone to collapsing into compromised averages, (ii) Prone to local optima or OOD actions, where direct critic maximization can pull the one-step policy toward local optima or out-of-distribution regions. OptiFlow addresses these challenges by reframing one-step policy learning as a **structured sample-allocation problem**. We decouple the policy improvement process into three mechanisms: (i) learning a value-aware multi-step reference flow policy anchored in the dataset, (ii) constructing a value-aware geometric coupling via entropic OT, and (iii) extracting the one-step policy through transport-guided target selection. An overview is illustrated in Figure 1.

Step 1: Learning the critic. The first step is to train an action-value function $Q_\phi(s,a)$ from the offline dataset $\mathcal{D} = \{(s_i, a_i, r_i, s_i')\}_{i=1}^{|\mathcal{D}|}$ using a standard temporal-difference objective for the one-step policy μ_θ:

$$\min_\phi \mathcal{L}_Q(\phi) = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D},\, z'\sim\mathcal{N}(0,I_d)} \left[\Big(Q_\phi(s,a) - \big(r + \gamma Q_{\bar{\phi}}(s', \mu_\theta(s', z'))\big) \Big)^2 \right], \tag{5}$$

where $Q_{\bar{\phi}}$ denotes a target critic network [31]. In OptiFlow, the critic is used in two ways: it defines the priority weights for training the reference flow policy, and it constructs the reference-side marginal for the optimal transport problem. Additional implementation variants of the TD target are described in Appendix D.

Step 2: Learning the multi-step reference flow policy. The next step of OptiFlow is to establish a data-supported, high-value reference policy μ_ω. To this end, we adopt the Value-aware behavior cloning (VaBC) [17] objective to train a multi-step reference flow policy. Given a dataset action a and noise $\epsilon \sim \mathcal{N}(0, I_d)$, the velocity field v_ω is trained via:

$$\min_\omega \mathcal{L}_{\text{ref}}(\omega) = \mathbb{E}_{(s,a)\sim\mathcal{D},\epsilon\sim\mathcal{N}(0,I_d),t\sim\text{Unif}([0,1])} \left[g_\eta(s,a) \left\| v_\omega(s,a_t,t) - (a-\epsilon) \right\|_2^2 \right] \tag{6}$$

$$\text{where } g_\eta(s,a) = \frac{\exp\left(\frac{\lambda}{\eta} Q_\phi(s,a)\right)}{\exp\left(\frac{\lambda}{\eta} Q_\phi(s,a)\right) + \exp\left(\frac{\lambda}{\eta} Q_\phi(s, \mu_\theta(s,z))\right)}, \quad z \sim \mathcal{N}(0, I_d). \tag{7}$$

where $\lambda = \left(\frac{1}{B}\sum_{i=1}^{B}|Q_\phi(s_i,a_i)| + \epsilon_q\right)^{-1}$ normalizes the critic scale within a minibatch of size B, with a small constant $\epsilon_q > 0$. Here, $g_\eta(s,a)$ assigns higher weights to dataset actions that outperform the current one-step policy $\mu_\theta(s,z)$ as estimated by the critic Q_ϕ. This objective clones dataset actions with high values. Consequently, μ_ω successfully captures the multi-modal structure of the data while concentrating probability density on high-return regions Importantly, the reference multi-step flow policy μ_ω serves as a *support-preserving proposal distribution* that provides a high-quality candidate action set. This ensures that the subsequent OptiFlow's sample allocation is restricted to regions that are both in-distribution and high-performing, effectively filtering out suboptimal or unsupported actions before the final one-step policy extraction.

Step 3: Value-weighted transport coupling. With the *multi-step* reference policy μ_ω providing a set of high-quality, in-distribution candidate actions, the next challenge is to allocate these targets to the *one-step* policy μ_θ without inducing mode collapse or geometric misalignment. For each state s, we sample N actions from the one-step policy and M actions from the reference flow policy:

$$a^s_{\theta,i} = \mu_\theta(s,z_i), \qquad z_i \sim \mathcal{N}(0,I_d), \qquad i = 1,\dots,N, \tag{8}$$

$$\tilde{a}^s_{\omega,j} = \mu_\omega(s,z_j), \qquad z_j \sim \mathcal{N}(0,I_d), \qquad j = 1,\dots,M. \tag{9}$$

To decouple the *value guidance* from *geometric alignment*, OptiFlow formulates the sample allocation as an optimal transport problem with **asymmetric marginal constraints**. Specifically, we impose a uniform marginal constraint $p \in \Delta^N$, with $p_i = \frac{1}{N}$, on the one-step policy samples, and a value-weighted marginal constraint $q^s \in \Delta^M$ on the reference policy actions:

$$q^s_j = \frac{\exp(Q_\phi(s,\tilde{a}^s_{\omega,j})/\tau)}{\sum_{k=1}^{M}\exp(Q_\phi(s,\tilde{a}^s_{\omega,k})/\tau)}. \tag{10}$$

where $\tau > 0$ is a temperature parameter. This asymmetric marginal design is essential for learning a valid one-step flow policy. The uniform constraint p ensures that all base noise samples $z_i \sim \mathcal{N}(0,I_d)$ are used equally, preserving the base distribution. Conversely, the value-weighted constraint q injects the critic's guidance directly into the allocation process. By assigning larger probability mass to high-return reference actions, the transport plan maps the majority of one-step samples to high-value modes.

Then, to complete this decoupling, we define the optimal transport cost solely based on geometric proximity in the action space, without any value-based terms.

$$C^s_{ij} = \frac{1}{\bar{c}^s}\|a^s_{\theta,i} - \tilde{a}^s_{\omega,j}\|_2^2, \tag{11}$$

where $\bar{c}^s$ denotes the per-state mean squared distance over all action pairs. With the asymmetric marginals (p, q^s) that determines the target weights and the cost matrix C that encodes the geometry, we compute the entropic optimal transport plan:

$$P^{s,*} = \underset{P\in\Pi(p,q^s)}{\arg\min}\left[\langle P, C^s\rangle + \varepsilon\sum_{i,j}P_{ij}(\log P_{ij} - 1)\right], \tag{12}$$

where $\Pi(p,q^s) = \{P \geq 0 \mid P\mathbf{1}_M = \frac{1}{N}\mathbf{1}_N,\ P^\top\mathbf{1}_N = q^s\}$ denotes the set of valid couplings, and $\varepsilon > 0$ is the entropic regularization hyperparameter. The optimal transport plan $P^{s,*}$ is efficiently computed by Sinkhorn algorithm (see Appendix C for more details). By keeping the critic's influence isolated within the q^s-marginal constraints and the spatial structure isolated within the cost matrix, OptiFlow ensures that probability mass is allocated proportionally to high-value action modes without compromising the geometric shape of the action distribution. This is in contrast to existing behavior-regularized offline RL methods that rely on an explicit behavioral penalty term. While strong regularization prevents out-of-distribution actions, it forces the policy to cover suboptimal dataset actions. OptiFlow circumvents this trade-off by leveraging the optimal transport coupling for one-step policy extraction. Rather than relying on a global behavioral penalty, the transport plan directly maps the one-step policy only to high-value modes, naturally discarding suboptimal candidates.

Step 4: Transport-guided distillation. Given the optimal coupling $P^{s,*}$, the final step is to learn the efficient one-step policy through a structured, transport-guided distillation. While the optimal

Algorithm 1 OptiFlow training

for each training iteration **do**
  Sample a mini-batch $\mathcal{B} = \{(s, a, r, s')\}$ from $\mathcal{D}$
  Update critic Q_ϕ using $\nabla_\phi \mathcal{L}_Q(\phi)$ (Eq. (5))
  Update reference flow policy μ_ω using $\nabla_\omega \mathcal{L}_{\text{ref}}(\omega)$ (Eq. (6))
  for each state $s \in \mathcal{B}$ **do**
    Sample one-step actions $\{a^s_{\theta,i}\}_{i=1}^N$ from $\mu_\theta(s, \cdot)$
    Sample reference actions $\{\tilde{a}^s_{\omega,j}\}_{j=1}^M$ from $\mu_\omega(s, \cdot)$
    Set $p_i = \frac{1}{N}$ and compute the value-weighted reference-policy marginal q^s_j by Eq. (10)
    Construct $C^s_{ij} = \frac{1}{\bar{c}^s} \|a^s_{\theta,i} - \tilde{a}^s_{\omega,j}\|_2^2$
    Compute $P^{s,*}$ by Sinkhorn on $(C^s, p, q, \varepsilon, T)$ (Alg. 2)
    Set $j^*_i = \arg\max_j P^{s,*}_{ij}$
  end for
  Update one-step policy μ_θ using $\nabla_\theta \mathcal{L}_{\text{distill}}(\theta)$ (Eq. (14))
  Update target critic $Q_{\bar{\phi}}$
end for

coupling provides a soft assignment, practical one-step policy learning requires explicit, unambiguous regression targets. Thus, we extract a per-row reference action index via maximum assignment:

$$j^*_i \in \arg\max_j P^{s,*}_{ij}, \qquad i = 1, \ldots, N. \tag{13}$$

The one-step policy is then updated by minimizing the mass-weighted distillation loss:

$$\min_\theta \mathcal{L}_{\text{distill}}(\theta) = \mathbb{E}_{s\sim\mathcal{D}} \left[\sum_{i=1}^N P^{s,*}_{i,j^*_i} \left\| \mu_\theta(s, z_i) - \tilde{a}^s_{\omega,j^*_i} \right\|_2^2 \right]. \tag{14}$$

During this update, the reference actions $\tilde{a}^s_{\omega,j^*_i}$ serve as fixed distillation targets, while the transport weights $P^{s,*}_{i,j^*_i}$ and hard assignments j^*_i are treated as constants; gradients flow exclusively through the one-step-policy actions $\mu_\theta(s, z_i)$. Unlike pointwise distillation that simply matches shared noise inputs and action outputs and often collapses multimodal behaviors into a compromised mean, each distillation target in OptiFlow is explicitly routed to a geometrically aligned, distinct high-value mode via optimal transport, preserving the multimodal structure.

Algorithm Summary. To sum up, the full OptiFlow (Algorithm 1) extracts an effective one-step policy that preserves the multimodal structure of high-value behaviors. It leverages a learned critic (Step 1) and a reference multi-step flow policy (Step 2) to solve per-state optimal transport problems (Step 3). By decoupling value guidance (target marginals) from behavioral geometry (cost matrix), the final transport-guided distillation (Step 4) explicitly maps noise to distinct, in-distribution high-value action modes. Notably, OptiFlow avoids direct critic maximization, which often exploits unreliable out-of-distribution regions. Instead, it anchors the policy to high-return in-distribution targets via transport plan. This structural separation mitigates the need for explicit behavior regularization while ensuring stable and mode-preserving one-step policy extraction.

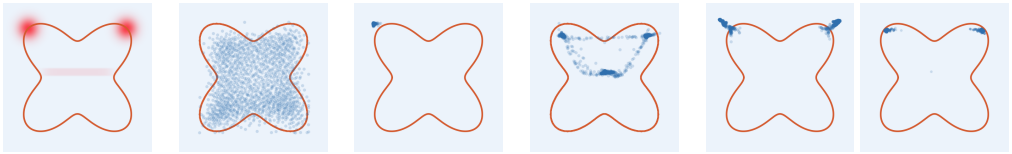
5 Experiments

Our experiments are designed to answer three questions: (i) whether OptiFlow preserves multimodal action structure in controlled settings, (ii) which design choices are responsible for the gains, including the use of a value-weighted reference-policy marginal and an OT coupling, and (iii) whether value-weighted, transport-guided distillation improves offline RL performance over conventional and flow-based baselines.

5.1 Controlled Multimodal Experiments

Bandit diagnostic. Figure 2 illustrates three requirements for offline policy distillation: preserving multimodality, avoiding suboptimal target assignments, and remaining within behavior support. Experimental results of FQL [16] highlight the trade-off between behavior matching and value

maximization: emphasizing the dataset regularization term (FQL-D in Figure 2d) biases the policy toward the data distribution but under-prioritizes high-value modes, whereas emphasizing the critic maximization term (FQL-Q in Figure 2e) can deviate from behavior support due to out-of-distribution value overestimation. Gaussian-based policies (Figure 2c) with direct critic maximization can similarly drift outside behavior support under value overestimation and fail to capture multimodality.

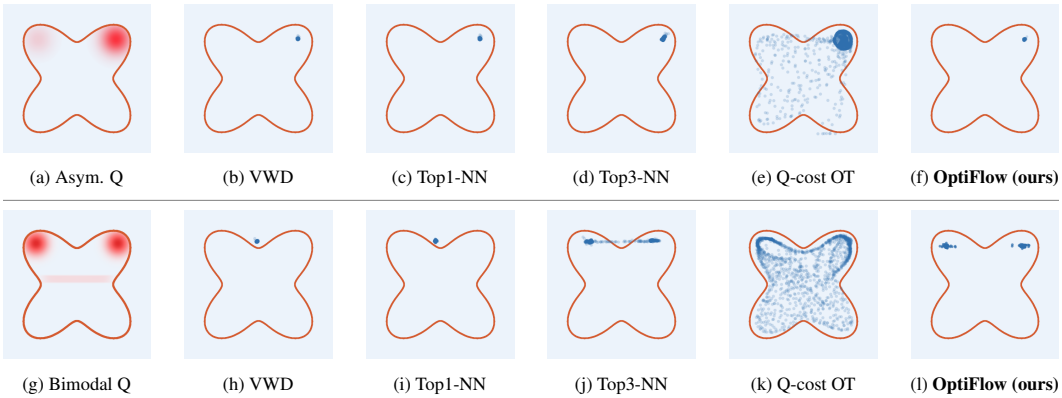


(a) True Reward (b) BC (c) ReBRAC w/ Gaussian Policy (d) FQL-D (e) FQL-Q (f) **OptiFlow (ours)**

Figure 2: **Bandit multimodal diagnostic.** A bimodal Reward landscape illustrating two high-value modes and a suboptimal region. The figure evaluates whether different methods preserve multimodality, avoid suboptimal assignments, and remain within behavior support. The experimental setup is detailed in Appendix D

In contrast, OptiFlow uses a mechanism in which the critic shapes the reference-policy marginal, while the OT coupling determines pairings according to the action-distance cost. This combination enables structured supervision that preserves multimodality (Figure 2f). Even when suboptimal actions appear among the reference samples, they are rarely selected after the row-wise maximum-mass assignment. Finally, because the targets are drawn from the behavior-regularized flow reference policy, OptiFlow remains grounded in dataset support while preserving multiple high-value modes.

Why OT coupling and value-weighted marginals? Figure 3 compares several alternatives for offline policy distillation under asymmetric and bimodal value landscapes. Value-weighted distillation (VWD) guides the one-step policy toward high-value targets, but collapses toward a single dominant mode (Figure 3b,h). Top-K nearest-neighbor distillation partially preserves multimodality by selecting multiple reference actions, but still struggles to fully capture the multimodal structure (Figure 3c,d,i,j). These results suggest that local or row-wise matching alone is insufficient for preserving the distribution over optimal modes.



(a) Asym. Q (b) VWD (c) Top1-NN (d) Top3-NN (e) Q-cost OT (f) **OptiFlow (ours)**

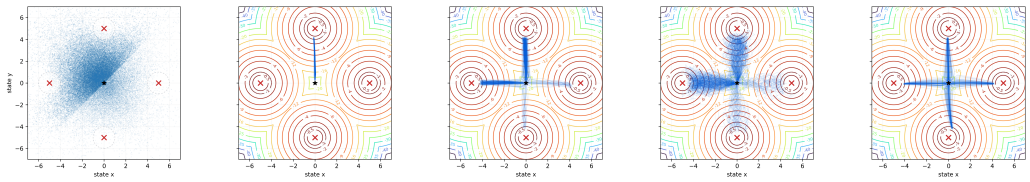
(g) Bimodal Q (h) VWD (i) Top1-NN (j) Top3-NN (k) Q-cost OT (l) **OptiFlow (ours)**

Figure 3: **Bandit diagnostic for value-aware transport.** (a,g) show asymmetric and bimodal value landscapes. The figure evaluates whether different distillation strategies can capture high-value regions while preserving the distribution over optimal modes.

A more structured alternative is to introduce OT coupling between one-step-policy and reference-policy samples. However, not all OT formulations behave similarly. One natural alternative approach of our method is to keep the reference-policy marginal uniform while injecting value directly into the transport cost, $C_{ij} = f(d_{ij}, Q(s, a^s_{\theta,i}), Q(s, \tilde{a}^s_{\omega,j}))$. Although this formulation trades off value and distance, Figure 3e,k shows that it still misallocates samples because the uniform reference marginal forces every reference action, including low-value ones, to receive equal total mass.

OptiFlow instead places value into the *reference-policy marginal* itself: high-value reference actions receive more supervision mass, while OT coupling determines geometrically compatible assignments between one-step and reference-policy samples. As shown in Figure 3f,l, this asymmetric marginal design preserves multimodal structure while concentrating supervision on high-value, dataset-supported regions.

We additionally provide a counterexample illustrating a structural limitation of uniform-marginal OT, generalize this limitation beyond the counterexample, and empirically demonstrate the importance of the reference marginal on real-world benchmarks in Appendix A.2. We describe the value-weighted, Top-K nearest-neighbor distillation formulations and more general value-, distance-aware distillation formulation in Appendix A.3, and discuss their failure modes on mode collapse to motivate the need for OptiFlow formulation.



(a) Dataset Distribution (b) FQL ($\alpha = 0.1$) (c) FQL ($\alpha = 1$) (d) FQL ($\alpha = 10$) (e) **OptiFlow-BC**

Figure 4: **Four-mode environment for evaluating multimodal behavior.** (a) Offline dataset distribution with intentionally uneven coverage across the four goal directions. (b–e) Learned policy distributions overlaid on the reward landscape. (b–d) FQL with increasing distillation strength α, exhibiting a diversity–quality trade-off. (e) OptiFlow-BC, maintaining balanced coverage across the four equally optimal modes.

Beyond the bandit setting. We further examine the preservation of multimodal behavior in a four-goal MDP introduced by [32], as illustrated in Figure 4, where the four goals constitute equally optimal modes but are represented unevenly in the offline dataset (Figure 4a). This setting tests whether the one-step policy extraction preserves multiple high-value modes despite imbalanced data coverage. To isolate the effect of transport-guided distillation from value-aware reference-policy training, we use OptiFlow-BC, a controlled variant of OptiFlow that replaces the VaBC reference policy with a standard BC reference while retaining the same value-weighted OT coupling and transport-guided distillation. OptiFlow-BC (Figure 4e) achieves balanced coverage of the four optimal modes while attaining the highest return, demonstrating that the policy extraction mechanism can preserve multimodal behavior despite imbalanced mode frequencies in the offline dataset. In contrast, FQL exhibits a clear diversity–quality trade-off (Figure 4b–d): stronger distillation improves multimodal coverage but progressively degrades policy return.

We also observe that the mode-level transport assignments remain consistent across repeated resampling of the reference actions, suggesting that the stochastic candidate set does not induce frequent assignment switches across distinct modes. The environment and reference-resampling protocols for assignment stability experiment are detailed in Appendix D.

5.2 Benchmark Experimental Setup

Benchmarks. We evaluate OptiFlow on reward-based single-task variants of OGBench [33] and on D4RL AntMaze and Adroit [34]. OGBench includes navigation and manipulation tasks with complex and often multimodal action distributions, while D4RL provides sparse-reward navigation and dexterous manipulation benchmarks. Full per-task results are provided in Appendix E.

Baselines. We compare against both conventional offline RL baselines and expressive generative-policy baselines. BC, IQL [4], ReBRAC [35], serve as standard offline RL references, while IDQL [14], IFQL, FQL [16] and GFP [17] represent the most closely related baselines based on expressive generative policies.

Table 1: **Aggregate performance across OGBench families and D4RL domains.** For state-based OGBench, each family row reports the mean ± standard error over the five single-task variants in that family. Visual OGBench rows report the corresponding visual single-task environments. D4RL rows aggregate over all tasks in the corresponding domain. Values within 95% of the best score in each row are in **bold**. Dashes denote unreported entries. Italicized entries indicate results taken from prior work [16], with complete per-task result tables in Appendix E.

Family / Domain	BC	IQL	ReBRAC	IDQL	IFQL	FQL	GFP	OptiFlow
OGBench — Navigation								
AntMaze-Large {1–5}	1.7±0.2	53.0±2.3	86.0±1.5	26.8±1.4	28.0±2.5	79.6±1.0	**93.9**±0.5	**91.9**±0.5
AntMaze-Giant {1–5}	0.0±0.0	4.3±0.8	**32.4**±3.3	0.0±0.0	0.9±0.4	9.5±2.3	18.9±2.5	26.5±1.5
Humanoid-Medium {1–5}	1.4±0.2	33.8±1.4	19.1±2.4	3.0±0.3	58.0±4.2	57.4±1.6	**71.8**±1.0	**74.4**±1.2
Humanoid-Large {1–5}	0.0±0.0	2.3±0.3	2.7±0.5	0.4±0.1	12.7±0.9	3.6±0.7	11.0±1.8	**19.1**±1.1
AntSoccer-Arena {1–5}	0.2±0.1	9.4±0.6	0.1±0.1	3.5±0.5	29.1±3.1	**61.4**±1.5	**61.1**±1.3	52.7±1.6
Navigation Avg.	0.7±0.1	20.6±0.6	28.1±0.9	6.7±0.3	25.8±1.2	42.3±0.7	**51.3**±0.7	**52.9**±0.5
OGBench — Manipulation								
Cube-Single {1–5}	2.8±1.0	50.1±2.5	87.8±1.2	91.9±0.5	83.9±0.8	91.0±1.1	**96.3**±0.5	**97.6**±0.3
Cube-Double {1–5}	0.0±0.0	2.4±0.2	8.1±1.3	2.8±0.7	10.5±0.7	25.2±1.5	43.3±1.6	**46.1**±1.3
Cube-Triple {1–5}	0.0±0.0	0.5±0.2	2.2±0.4	0.4±0.2	0.2±0.1	1.1±0.4	1.8±0.3	**3.6**±0.2
Puzzle-3×3 {1–5}	3.9±0.6	4.3±0.5	22.3±0.5	14.0±0.7	19.3±0.4	**29.5**±0.8	24.4±1.1	**28.1**±2.3
Puzzle-4×4 {1–5}	0.0±0.0	1.7±0.2	14.0±0.5	22.1±1.2	26.8±1.1	15.0±0.8	25.8±1.1	**32.1**±1.5
Scene-Play {1–5}	0.4±0.1	20.0±0.8	42.0±1.5	39.1±0.3	44.7±1.7	55.8±1.4	53.1±1.6	**59.4**±0.3
Manipulation Avg.	1.2±0.2	13.2±0.5	29.4±0.4	28.4±0.3	30.9±0.4	36.3±0.4	40.8±0.5	**44.5**±0.5
OGBench — Visual								
Visual-Cube-Single	–	*70±6*	*83±3*	–	*49±3.5*	*81±6*	72.2±3.6	**86.6**±1.3
Visual-Cube-Double	–	*34±11.5*	*4±2*	–	*8±3*	*21±5.5*	12.0±2.6	**66.7**±3.5
Visual-Scene-Play	–	***96***±1	***98***±2	–	*86±5*	***98***±1.5	**98.7**±0.3	92.8±0.8
Visual-Puzzle-3×3	–	*7±7.5*	*88±2*	–	***100***±0	*94±0.5*	86.5±1.1	93.8±1.2
Visual-Puzzle-4×4	–	*0±0*	*26±3*	–	*8±7.5*	*33±3*	19.3±1.4	**39.7**±4.8
Overall								
OGBench Visual Avg.	–	*41.4±3.0*	*59.8±1.1*	–	*50.2±2.0*	*65.4±1.8*	57.7±1.0	**75.9**±1.3
OGBench State-Based Avg.	0.9±0.1	16.5±0.4	28.8±0.5	18.5±0.2	28.6±0.6	39.0±0.4	45.6±0.4	**48.3**±0.4
AntMaze (D4RL) Avg.	*16.7*	*58.4*	*76.9*	*79.1*	61.0±2.5	**82.3**±1.3	**82.6**±1.5	**86.6**±0.7
Adroit (D4RL) Avg.	*36.7*	*53.5*	***58.6***	50.2±0.4	48.2±0.5	50.7±0.5	52.1±0.5	50.5±0.4

Training and evaluation. For state-based OGBench tasks, all methods are trained for 1M gradient steps. For D4RL and OGBench visual tasks, all methods are trained for 500K gradient steps. We report final-checkpoint performance averaged over 8 random seeds unless otherwise noted. For visual tasks, due to high computational cost, we report performance averaged over 4 random seeds and over checkpoints at 300K, 400K, and 500K steps, following the evaluation protocol of FQL [16]. Full training and evaluation details are provided in Appendix D and the result-table captions.

Method details. OptiFlow uses a value-aware reference flow policy and a one-step flow policy. For each state, we sample $N = 16$ one-step-policy actions and $M = 64$ reference-policy actions, construct a value-weighted reference-policy marginal, compute an entropic Optimal Transport plan with 30 Sinkhorn iterations, select a reference action for each one-step-policy sample via the maximum transport assignment in the corresponding row, and distill the resulting assignments into the one-step policy. Additional implementation details and full hyperparameters are given in Appendix D.

5.3 Main Results

Aggregate benchmark performance. Table 1 summarizes performance across OGBench families and D4RL domains. OptiFlow achieves the best overall state-based OGBench average, with 48.3 ± 0.4 compared to 45.6 ± 0.4 for GFP and 39.0 ± 0.4 for FQL. The gains are most pronounced on OGBench manipulation, where OptiFlow obtains the strongest aggregate performance, improving the manipulation average to 44.5 ± 0.5 compared to 40.8 ± 0.5 for GFP and 36.3 ± 0.4 for FQL. These results show that value-weighted transport-guided distillation improves over the closest flow-based

baselines in the setting most aligned with OptiFlow's motivation: learning efficient one-step policies from complex and multimodal action distributions.

OGBench manipulation and navigation. OptiFlow obtains the best aggregate score on five of the six OGBench manipulation families: Cube-Single, Cube-Double, Cube-Triple, Puzzle-4×4, and Scene-Play. On OGBench navigation, OptiFlow remains competitive: it improves over FQL on AntMaze-Giant, Humanoid-Medium, and Humanoid-Large. Overall, OptiFlow's strongest gains appear in tasks where multimodal reference-policy structure is useful for one-step policy learning.

D4RL results. On D4RL, OptiFlow achieves the best AntMaze average, improving over GFP by 4.0 points and FQL by 4.3 points. On Adroit, OptiFlow is competitive with flow-based baselines. These results suggest that OptiFlow transfers beyond OGBench, with its clearest advantage in settings where useful multimodal reference actions can be transferred to a one-step policy through structured assignment.

Sensitivity analysis. We provide additional sensitivity analyses of OptiFlow with respect to the value-weighting temperatures τ and η (Appendix F.1), the entropic regularization coefficient and the number of Sinkhorn iterations (Appendix F.2), and the numbers of one-step and reference-policy action samples (Appendix F.3).

Additional ablations and diagnostics. We first control for OptiFlow's use of multiple action samples per state and show that increased sampling alone does not explain its gains (Appendix F.4). We then disentangle the contributions of the reference policy and transport-guided policy extraction (Appendix F.5) and compare our default VaBC weighting with an AWR-style alternative (Appendix F.6). Finally, we examine whether direct critic maximization provides benefits beyond transport-based supervision (Appendix F.7) and demonstrate **strong offline-to-online fine-tuning performance** (Appendix G).

6 Conclusion

We presented **One-step Flow Policy via Optimal Transport (OptiFlow)**, a value-weighted optimal-transport method to learn efficient one-step flow policies for offline reinforcement learning. OptiFlow uses a value-aware reference flow policy to represent multimodal, data-supported behavior, and trains a one-step flow policy through transport-guided distillation. By constructing an entropic OT coupling between one-step-policy and reference-policy samples, OptiFlow separates value and geometry: critic-estimated values determine how much supervision mass each reference action receives, while action-space distance determines how one-step samples are assigned. This provides a structured alternative to independent noise-wise distillation and direct critic maximization.

Experimental results show that OptiFlow preserves multimodal action structure in controlled diagnostics and achieves strong performance across offline RL benchmarks, with especially clear gains on OGBench manipulation tasks. These results support the central claim that value-guided one-step policy learning benefits from transport-based distillation.

More broadly, OptiFlow suggests that critic information in offline RL need not be used only through direct critic maximization. It can instead shape structured supervision for policy learning, opening a path toward value-guided generative policy learning methods that preserve multimodal behavior while retaining efficient one-step policy deployment.

OptiFlow also has several limitations. Compared to standard offline RL methods, especially existing flow-based policies, it introduces additional computational overhead from repeated policy sampling and Sinkhorn-based optimal transport updates. However, these operations mainly consist of matrix-based computations that can be efficiently parallelized on modern GPUs (see Appendix A.4 for detailed analysis of computational overhead of OptiFlow).

Acknowledgments and Disclosure of Funding

This work was supported by the Institute of Information & communications Technology Planning & Evaluation (IITP) grants funded by the Korea government (MSIT) (No. RS-2020-II201361, Artificial Intelligence Graduate School Program (Yonsei University), RS-2024-00457882, AI Research Hub Project) and the National Research Foundation of Korea (NRF) grants funded by the Korea government(MSIT) (RS-2026-25505230, RS-2026-25619452). This research was also supported by a grant from the Institute for AI and Social Innovation at Yonsei University and the Yonsei University Research Fund of 2026-22-0204.

References

[1] Sascha Lange, Thomas Gabel, and Martin Riedmiller. *Batch Reinforcement Learning*, pages 45–73. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-27645-3. doi: 10.1007/978-3-642-27645-3_2.

[2] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *CoRR*, abs/2005.01643, 2020.

[3] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in neural information processing systems*, 2020.

[4] Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022.

[5] Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

[6] Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[7] Jongmin Lee, Wonseok Jeon, Byung-Jun Lee, Joelle Pineau, and Kee-Eung Kim. Optidice: Offline policy optimization via stationary distribution correction estimation, 2021. URL `https://arxiv.org/abs/2106.10783`.

[8] Ofir Nachum, Bo Dai, Ilya Kostrikov, Yinlam Chow, Lihong Li, and Dale Schuurmans. Algaedice: Policy gradient from arbitrary experience, 2019. URL `https://arxiv.org/abs/1912.02074`.

[9] Seohong Park, Kevin Frans, Sergey Levine, and Aviral Kumar. Is value learning really the main bottleneck in offline rl? *Advances in Neural Information Processing Systems*, 37:79029–79056, 2024.

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

[11] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024.

[12] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

[13] 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.

[14] 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, 2023.

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

[16] Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. In *International Conference on Machine Learning (ICML)*, 2025.

[17] Franki Nguimatsia Tiofack, Theotime Le Hellard, Fabian Schramm, Nicolas Perrin-Gilbert, and Justin Carpentier. Guided flow policy: Learning from high-value actions in offline reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026.

[18] Yejun Jang, Hong Chul Nam, Jeong Min Park, GIMIN BAE, and Hyun Kwon. Q-guided flow q-learning. In *CoRL 2025 Workshop RemembeRL*, 2025.

[19] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *arXiv preprint arXiv:2202.00512*, 2022.

[20] Kevin Frans, Danijar Hafner, Sergey Levine, and Pieter Abbeel. One step diffusion via shortcut models. *arXiv preprint arXiv:2410.12557*, 2024.

[21] Marco Cuturi. Sinkhorn distances: Lightspeed computation of optimal transport. *Advances in neural information processing systems*, 26, 2013.

[22] Dayi Dong, Maulik Bhatt, Seoyeon Choi, and Negar Mehr. Mimic-d: Multi-modal imitation for multi-agent coordination with decentralized diffusion policies, 2026. URL `https://arxiv.org/abs/2509.14159`.

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

[24] Ziyu Wang, Alexander Novikov, Konrad Zolna, Josh S Merel, Jost Tobias Springenberg, Scott E Reed, Bobak Shahriari, Noah Siegel, Caglar Gulcehre, Nicolas Heess, et al. Critic regularized regression. *Advances in Neural Information Processing Systems*, 33:7768–7778, 2020.

[25] Yuhang Ran, Yi-Chen Li, Fuxiang Zhang, Zongzhang Zhang, and Yang Yu. Policy regularization with dataset constraint for offline reinforcement learning. In *International conference on machine learning*, pages 28701–28717. PMLR, 2023.

[26] Yicheng Luo, Zhengyao Jiang, Samuel Cohen, Edward Grefenstette, and Marc Peter Deisenroth. Optimal transport for offline imitation learning. *arXiv preprint arXiv:2303.13971*, 2023.

[27] Wei-Di Chang, Scott Fujimoto, David Meger, and Gregory Dudek. Imitation learning from observation through optimal transport. *arXiv preprint arXiv:2310.01632*, 2023.

[28] Arip Asadulaev, Rostislav Korst, Alexander Korotin, Vage Egiazarian, Andrey Filchenkov, and Evgeny Burnaev. Rethinking optimal transport in offline reinforcement learning. *Advances in Neural Information Processing Systems*, 37:123592–123607, 2024.

[29] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, second edition, 2018.

[30] Michael S Albergo and Eric Vanden-Eijnden. Building normalizing flows with stochastic interpolants. *arXiv preprint arXiv:2209.15571*, 2022.

[31] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[32] Tuomas Haarnoja, Haoran Tang, Pieter Abbeel, and Sergey Levine. Reinforcement learning with deep energy-based policies. In *International conference on machine learning*, pages 1352–1361. PMLR, 2017.

[33] Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu, editors, *International Conference on Learning Representations*, volume 2025, pages 94937–94982, 2025.

[34] 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.

[35] Denis Tarasov, Vladislav Kurenkov, Alexander Nikulin, and Sergey Kolesnikov. Revisiting the minimalist approach to offline reinforcement learning. *Advances in Neural Information Processing Systems*, 36:11592–11620, 2023.

[36] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

[37] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[38] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Yash Katariya, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL `http://github.com/jax-ml/jax`.

[39] Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Volodymyr Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. In *International Conference on Machine Learning (ICML)*, 2018.

[40] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Appendix Contents

A Discussion **14**

A.1 Limitations 14

A.2 Why use a value-aware reference marginal in optimal transport? 15

A.3 Alternative distillation methods and why OptiFlow? 16

A.4 Computational cost 18

B Broader Impacts **19**

C Sinkhorn Algorithm **20**

C.1 Algorithm 20

D Implementation Details **20**

E Complete Experimental Results **24**

F Ablation Studies **26**

F.1 Sensitivity to value-weighting temperatures 26

F.2 Sensitivity to Sinkhorn hyperparameters 26

F.3 Reference and one-step action sampling 27

F.4 Controlling for the action-sampling budget 27

F.5 Disentangling reference policy quality and distillation 28

F.6 AWR-style weighting for value-aware behavior cloning 28

F.7 Direct Q-maximization 29

G Offline-to-Online Fine-tuning **30**

A Discussion

In this section, we provide additional analysis and discussion of OptiFlow. We first discuss its limitations and then examine two key aspects of its transport-based formulation. We show why value information should be encoded in the reference marginal rather than solely in the transport cost, and analyze alternative distillation strategies to clarify the role of global transport-based matching in preserving multimodality. Finally, we characterize the computational cost of OptiFlow, including its scaling with the number of Sinkhorn iterations and policy samples.

A.1 Limitations

OptiFlow introduces additional training-time computation because it solves a state-wise entropic optimal transport problem over sampled one-step-policy and reference-policy actions. This requires Sinkhorn iterations during training, making OptiFlow more expensive than purely pointwise distillation or direct actor updates. However, this overhead does not affect deployment: after training, OptiFlow uses only the one-step flow policy for action generation, so inference remains as efficient as other one-step policies. Appendix A.4 discusses the trade-off between Sinkhorn iterations, training cost, and policy performance.

OptiFlow is most beneficial when the reference flow policy provides multiple useful candidate actions and the supervision mass get allocated to advantage. When the reference action distribution is effectively simple, or when a single local target is already sufficient for policy extraction, the advantage over simpler one-step distillation methods may be smaller.

A.2 Why use a value-aware reference marginal in optimal transport?

We construct a simple 2×2 example showing that, under a uniform reference marginal, incorporating value information solely through the transport cost is insufficient: regardless of the cost design, the induced transport plan cannot guarantee policy improvement.

Counterexample. Consider a 2×2 transport problem with uniform student and teacher marginals in certain state $s \in \mathcal{S}$,

$$p = \left(\frac{1}{2}, \frac{1}{2}\right), \qquad q = \left(\frac{1}{2}, \frac{1}{2}\right).$$

The transport plan P must satisfy

$$\sum_j P_{ij} = p_i, \qquad \sum_i P_{ij} = q_j.$$

Suppose both target policy's actions have the same value,

$$Q^\pi(s, a^s_{\theta,1}) = Q^\pi(s, a^s_{\theta,2}) = 7,$$

while the reference policy's actions have highly asymmetric values,

$$Q^\pi(s, \tilde{a}^s_{\omega,1}) = 10, \qquad Q^\pi(s, \tilde{a}^s_{\omega,2}) = 1.$$

Even if the cost function is designed so that both target policy actions prefer the first reference policy's action, e.g.,

$$c_{11} < c_{12}, \qquad c_{21} < c_{22},$$

the uniform reference-side marginal constraint still imposes

$$P_{11} + P_{21} = \frac{1}{2}, \qquad P_{12} + P_{22} = \frac{1}{2}.$$

Therefore, at least half of the total mass must be transported to the low-value reference action a^ω_2. This is independent of how the cost is constructed, as long as the reference-side marginal follows uniform distribution.

Indeed, the expected value induced by any feasible transport plan is fixed by the reference-side marginal:

$$\sum_{i,j} P_{ij} Q^\pi(s, \tilde{a}^s_{\omega,j}) = \sum_j q_j Q^\pi(s, \tilde{a}^s_{\omega,j}) = \frac{1}{2} \cdot 10 + \frac{1}{2} \cdot 1 = 5.5.$$

Thus, any cost formulation cannot improve the target policy in this example, unless the uniform marginal constraint can be mitigated or changed.

In contrast, our softmax-marginal formulation sets the reference-side marginal according to the value,

$$w_j = \frac{\exp(Q^\pi(s, \tilde{a}^s_{\omega,j})/\tau)}{\sum_k \exp(Q^\pi(s, \tilde{a}^s_{\omega,k})/\tau)}.$$

When τ is small enough, the marginal concentrates on the high-value reference action:

$$q^{\tau^*} \approx (1, 0).$$

Then the transport plan

$$P \approx \begin{pmatrix} \frac{1}{2} & 0 \\ \frac{1}{2} & 0 \end{pmatrix}$$

is feasible, and both student actions can be guided toward the optimal teacher action. The induced teacher-side value becomes approximately

$$\sum_j q^{\tau^*}_j Q^\pi(s, \tilde{a}^s_{\omega,j}) \approx 10 > 7,$$

rather than the uniform-marginal value 5.5. This counterexample supports why the value information should be encoded in the reference-side marginal, not in the transport cost.

Generalization beyond the 2×2 example. The observation above extends to an arbitrary number of sampled actions. For any state s, let $\{\tilde{a}^s_{\omega,j}\}_{j=1}^M$ denote the reference actions and q^s their prescribed marginal. For any feasible coupling $P \in \Pi(p^s, q^s)$, we have

$$\sum_{i,j} P_{ij} Q^\pi(s, \tilde{a}^s_{\omega,j}) = \sum_j q^s_j Q^\pi(s, \tilde{a}^s_{\omega,j}),$$

since $\sum_i P_{ij} = q^s_j$. Thus, the expected value of the reference actions assigned through the coupling is determined solely by the reference-side marginal q^s, regardless of the transport cost. Under a uniform reference marginal, **this quantity reduces to the average value of the sampled reference actions, regardless of the cost matrix.**

In contrast, the value-weighted marginal assigns more mass to higher-value reference actions, where the policy improvement is controlled by τ: the expected assigned value increases monotonically as τ decreases, from the candidate average at $\tau \to \infty$ to the best candidate value at $\tau \to 0$, so a sufficiently small τ guarantees improvement whenever some candidiates beats the current policy. We use softmax rater than an argmax for q^s since the latter concentrates all supervision on one candidate and collapses the multimodal structure, so τ trades assigned value against mode coverage.

Empirical effect of the reference marginal. We further evaluate whether the choice of reference marginal has a corresponding empirical effect. We compare full OptiFlow, which combines VaBC with the value-weighted marginal, against (i) a variant that retains VaBC but replaces the value-weighted marginal with a uniform marginal, and (ii) OptiFlow-BC, which replaces VaBC with standard BC while retaining the value-weighted marginal.

Table 2: **Effect of the reference marginal.** We compare full OptiFlow against using a uniform reference marginal and against replacing VaBC with standard BC while retaining the value-weighted marginal. We follow the same evaluation protocol as Table 9.

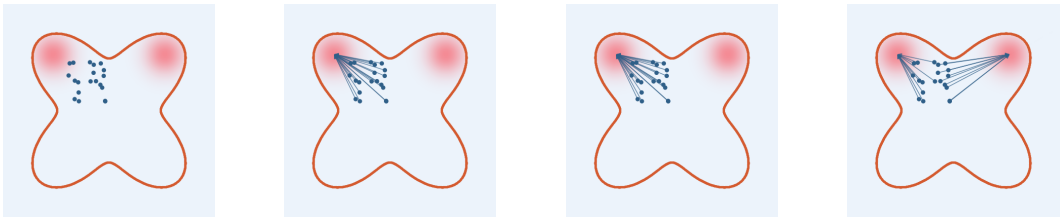
Task	OptiFlow	VaBC + Uniform	BC + Value-weighted
antmaze-large-navigate{1}	**94.0**$\pm$0.9	0.2$\pm$0.2	**91.0**$\pm$2.0
antmaze-giant-navigate{1}	**5.4**$\pm$2.4	0.0$\pm$0.0	0.9$\pm$0.3
humanoidmaze-medium-navigate{1}	**82.4**$\pm$1.2	3.6$\pm$0.5	50.5$\pm$8.3
humanoidmaze-large-navigate{1}	**48.3**$\pm$2.2	0.0$\pm$0.0	43.4$\pm$4.1
antsoccer-arena-navigate{4}	**46.8**$\pm$4.1	11.6$\pm$2.2	43.6$\pm$3.4
cube-single-play{2}	**98.5**$\pm$0.5	25.9$\pm$3.6	**97.5**$\pm$1.3
cube-double-play{2}	**52.4**$\pm$2.9	0.6$\pm$0.2	39.1$\pm$3.5
scene-play{2}	**97.5**$\pm$1.1	47.4$\pm$6.3	**98.2**$\pm$0.8
puzzle-3x3-play{4}	**25.3**$\pm$6.8	1.8$\pm$0.4	1.1$\pm$0.7
puzzle-4x4-play{4}	**22.8**$\pm$2.9	5.5$\pm$0.9	20.9$\pm$4.2
Average	**57.3**$\pm$1.0	9.7$\pm$0.7	48.6$\pm$1.3

Table 2 shows that the value-weighted marginal is critical to the performance of OptiFlow. Replacing it with a uniform marginal while retaining VaBC reduces the average performance from 57.3 to 9.7. In contrast, replacing VaBC with standard BC while retaining the value-weighted marginal results in a substantially smaller decrease, from 57.3 to 48.6. These results are consistent with the analysis above: the reference marginal directly controls how much transport mass is allocated to high-value reference actions, whereas the transport cost alone cannot alter the total mass assigned to each reference action.

A.3 Alternative distillation methods and why OptiFlow?

Here, we analyze alternative distillation strategies such as value-weighted and nearest-neighbor matching, and discuss their failure modes illustrated in Figure 5.

Our method can be viewed as combining ideas from value-weighted distillation (VWD) and Top-K nearest-neighbor (Top-K NN) matching, while addressing their limitations. Define reference-policy actions $\{\tilde{a}^s_{\omega,j}\}_{j=1}^N$, where $\tilde{a}^s_{\omega,j} = \mu_\omega(s, z_j)$, and one-step policy actions $\{a^s_{\theta,i}\}_{i=1}^N$, where



(a) Initialization (b) VDWM (c) Top-2 NN (d) **OptiFlow (ours)**

Figure 5: Initialization-dependent matching in an equal-value bimodal bandit. The reference set contains two equal-value optimal anchors, while all generated particles are initialized in the left half of the support. Value-distance weighted matching (VDWM) selects an anchor independently for each particle and therefore assigns all particles to the closer left mode. Top-2 nearest-neighbor matching still assigns each particle independently to its nearest anchor, again collapsing to the left mode. In contrast, OptiFlow preserves multimodality since the marginal constraint assigns equal mass to both modes, preserving the bimodal allocation.

$a^s_{\theta,i} = \mu_\theta(s, z_i)$. We use the same noise bank for both policies, i.e., $z_i = z_j$ when $i = j$, with $z_i \sim \mathcal{N}(0, I)$. VWD assigns a value-based weight to each reference action,

$$w_j = \frac{f(Q(s, \tilde{a}^s_{\omega,j}))}{\sum_{\ell=1}^N f(Q(s, \tilde{a}^s_{\omega,\ell}))}.$$

It then minimizes

$$\mathcal{L}_{\text{VWD}} = \sum_{j=1}^N w_j \left\| a^\theta_j - \tilde{a}^s_{\omega,j} \right\|_2^2.$$

In practice, we consider w_j as:

$$w_j = \frac{\exp(Q(s, \tilde{a}^s_{\omega,j})/\tau)}{\sum_{\ell=1}^N \exp(Q(s, \tilde{a}^s_{\omega,\ell})/\tau)}.$$

Top-K NN first selects the K highest-value reference samples,

$$\mathcal{T}_K(s) = \text{TopK}_{j\in\{1,\dots,M\}}\, Q(s, \tilde{a}^s_{\omega,j}),$$

and then assigns each one-step policy's action to its nearest reference action within this set,

$$j^*_i = \arg\min_{j\in\mathcal{T}_K(s)} \left\| a^s_{\theta,i} - \tilde{a}^s_{\omega,j} \right\|_2^2.$$

The corresponding distillation loss is

$$\mathcal{L}_{\text{TopK}} = \sum_{i=1}^N \left\| a^s_{\theta,i} - \tilde{a}^s_{\omega,j^*_i} \right\|_2^2.$$

In the bandit example in Figure 2, VWD collapses to the middle point, a well-known failure mode when distilling multi-step flow models into a one-step policy with naive ℓ_2 regression [20]. Top-K NN is also sensitive to the choice of K. When $K = 1$, the policy collapses to a single selected mode. When $K = 3$, it can preserve some multimodality, but it still cannot suppress probability mass in the intermediate region between modes. Although Top-K NN shows potential for capturing multimodality, it relies on hard selection and does not specify how much mass each selected reference action should receive, making it overly restrictive and sensitive to the choice of K.

Now let's consider Value-Distance Weighted Matching(VDWM) distillation, more general formulation which is closer to our algorithm. Define a score

$$\xi_{ij} = f(d_{ij}, Q(s, a^s_{\theta,i}), Q(s, \tilde{a}^s_{\omega,j})),$$

and perform weighted distillation:

$$\mathcal{L}_{\text{VDWM}} = \sum_i \xi_{ij^*_i} \left\| a^s_{\theta,i} - \tilde{a}^s_{\omega,j^*_i} \right\|^2,$$

where $j_i^* = \arg\max_j s_{ij}$. For instance, one may choose

$$\xi_{ij} = \frac{\exp\big(-d_{ij} + \alpha(Q(s, \tilde{a}^s_{\omega,j}) - Q(s, a^s_{\theta,i}))\big)}{\sum_{k=1}^{M} \exp\big(-d_{ik} + \alpha(Q(s, \tilde{a}^s_{\omega,k}) - Q(s, a^s_{\theta,i}))\big)}$$

However, this approach performs independent selection for each i. As a result, it does not impose any global mass-allocation constraint across reference actions, and therefore has probability of collapse onto a subset of modes. In particular, when multiple target modes have comparable value, small differences in initialization, distance, or estimated value can break the symmetry arbitrarily, causing many samples to select the same mode.

In contrast, our method constructs a global coupling with value-conditioned marginals. This induces a structured allocation of mass across reference actions, so multiple high-value modes can be represented proportionally rather than selected independently. As a result, our method mitigates mode collapse and better preserves multimodality. We illustrate this behavior with the bandit example in Figure 5.

A.4 Computational cost

OptiFlow introduces additional computational cost from three sources: (i) per-state Sinkhorn updates for optimal transport, (ii) action generation from the K-step reference flow model, and (iii) action generation from the one-step policy. We analyze how these factors contribute to the computation cost.

First, we identify the training-time bottleneck by varying the number of Sinkhorn iterations and analyzing the computational overhead introduced by OT updates.

Second, we analyze how the computational cost scales with the number of one-step flow policy and reference K-step flow policy samples (N, M), which affect reference action generation and transport matching. The corresponding experiments are reported in Appendix F.3.

Finally, we compare OptiFlow against alternative baselines in terms of both training and inference runtime.

Effect of Sinkhorn iterations on runtime. We use 30 Sinkhorn iterations as the default setting. To assess whether this choice introduces unnecessary computational overhead, we vary the number of iterations and measure the average computation time per training step. As shown in Table 3, reducing the number of iterations from 30 to 20 or 10 yields only marginal runtime improvements, suggesting that Sinkhorn updates are not a dominant training-time bottleneck. We analyze the corresponding performance behavior in Table 12b.

Sinkhorn Iterations	10	20	**30** (default)	40
ms/step	$5.14_{\pm 0.04}$	$5.26_{\pm 0.04}$	$5.37_{\pm 0.04}$	$5.47_{\pm 0.04}$

Table 3: **Effect of the number of Sinkhorn iterations on runtime.** We report the average computation time per training step (ms/step) over 15 tasks measured on a single NVIDIA H100 GPU.

Computational scaling with the number of samples (N, M). We analyze how the computational cost scales with the number of one-step flow policy samples and reference K-step flow policy samples (N, M), which jointly affect both reference action generation and transport matching. As shown in Table 4, the computational cost is more sensitive to the number of reference-policy samples M than to the number of one-step-policy samples N. In particular, increasing M introduces substantially higher runtime overhead due to the increased cost of reference action generation and transport matching, whereas increasing N results in comparatively moderate overhead. We analyze the corresponding performance behavior in Appendix F.3.

$N \backslash M$	4	16	**64** (default)	256	Avg(M)
4	3.73±0.13	3.53±0.01	4.99±0.04	16.74±0.10	7.25±0.93
8	3.39±0.07	3.65±0.01	5.10±0.04	16.57±0.09	7.18±0.92
16 (default)	3.42±0.01	3.74±0.02	5.36±0.04	16.83±0.09	7.34±0.94
32	3.50±0.02	3.85±0.01	5.76±0.04	17.42±0.09	7.63±0.97
Avg(N)	3.51±0.04	3.69±0.02	5.30±0.05	16.89±0.07	

Table 4: **Effect of the number of samples (N, M) on runtime.** We report the average computation time per training step (ms/step) over 15 tasks measured on a single NVIDIA H100 GPU.

Training and inference runtime comparison. Finally, we compare the training and inference runtime of OptiFlow with alternative baselines. As shown in Figure 6, OptiFlow introduces additional training-time overhead compared to standard one-step methods due to the need to generate and match multiple samples (N, M) from the one-step and reference flow policies. In contrast, inference remains efficient since the learned policy requires only a single forward pass at test time. Combined with our Sinkhorn iteration analysis, these results suggest that the dominant computational cost arises from sample generation and matching rather than from the OT updates themselves. Reducing the sampling cost required for effective transport-based distillation is an important direction for future work.

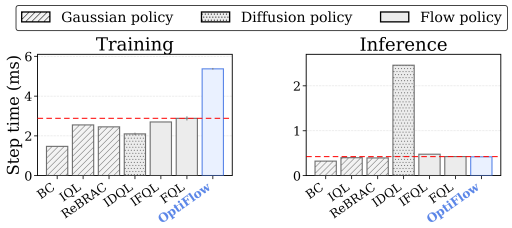


Figure 6: **Comparison of training and inference time across methods.** We report the average training time per gradient step (ms/step), averaged over 15 tasks and measured on a single NVIDIA H100 GPU.

B Broader Impacts

This work studies offline reinforcement learning in simulated control benchmarks. The main potential benefit is improved policy learning from fixed datasets, which may reduce the need for unsafe or expensive online exploration when applying RL to physical systems. OptiFlow also produces a one-step policy at deployment time, which can make generative-policy methods more practical in latency-sensitive control settings.

The main risk is the usual risk of offline RL: a learned policy may behave poorly when deployed outside the coverage of the dataset, especially if the critic assigns inaccurate values to unsupported actions. While OptiFlow is designed to reduce direct critic exploitation by using value information through transport-based target construction, it does not remove the need for careful validation before deployment. In safety-critical applications, the method should be combined with dataset auditing, environment-specific safety constraints, and out-of-distribution monitoring.

C Sinkhorn Algorithm

C.1 Algorithm

Algorithm 2 Sinkhorn Algorithm

Initialize $K = \exp(-C/\varepsilon)$
Initialize $u = \mathbf{1}_N$, $v = \mathbf{1}_M$
for $t = 1, \ldots, T$ **do**
    $u_i \leftarrow p_i/(Kv)_i$
    $v_j \leftarrow q_j/(K^\top u)_j$
end for
$P = \mathrm{diag}(u)K\mathrm{diag}(v)$

For completeness, we provide the Sinkhorn procedure [21] used to solve the entropy-regularized optimal transport problem in Step 3 of our method. Given the cost matrix C, reference-side marginal p, target-side marginal q, and entropic regularization parameter ε, the algorithm alternates row and column scaling updates to compute the transport plan.

In all experiments, we use a fixed number of $T = 30$ Sinkhorn iterations.

D Implementation Details

Architectures. The critic, reference flow policy, and one-step policy all use $[512, 512, 512, 512]$ MLPs with GeLU [36] activations. LayerNorm [37] is applied to the critic only. The critic is a 2-ensemble; we aggregate with the mean by default, unless otherwise stated as minimum. The one-step policy takes concat(s, z) through the MLP and a linear head (init scale 10^{-2}); policy outputs are clipped to $[-1, 1]$ at use time. All methods are implemented in JAX [38].

Image processing. For pixel-based OGBench environments, we use a smaller variant of the IMPALA encoder [39] and apply random-shift augmentation with probability 0.5, following the official implementation of [16]. We use 3-frame stacking, prestacked at dataset load time.

Flow teacher. We use linear-interpolant flow matching with uniform time sampling and integrate with $K = 10$ Euler steps. Time t is concatenated directly into the MLP input.

Value-aware BC. The reference action in the VaBC weight g_η is the stop-graded one-step policy action, and the minibatch scale $\lambda = (B^{-1} \sum_i |Q_\phi(s_i, a_i)|)^{-1}$ is also stop-graded. In practice, g_η is implemented as a 2-class softmax between $Q_\phi(s, a)$ and $Q_\phi(s, \mu_\theta(s, z))$. Gradients from this loss flow only into the reference policy.

Transport and distillation. Per state, we sample $N = 16$ one-step actions and $M = 64$ reference actions. The cost is squared Euclidean distance normalized by its batchwise mean. The one-step marginal is uniform, and the reference marginal is a softmax with temperature τ applied to reference critic values. We run balanced Sinkhorn in the log domain for $T = 30$ iterations, with entropic regularization fixed at $\varepsilon = 0.05$ across all tasks. We use hard row-wise anchors $j_i^* = \arg\max_j P_{ij}^*$, with distillation weights $P_{i,j_i^*}^*$. The actor objective combines the VaBC and distillation losses with equal weights.

Training and evaluation. We train for 1M gradient steps on state-based OGBench tasks and 500K steps on D4RL. Every 100K steps we evaluate with 100 episodes and reported scores are taken at the final training step and averaged over 8 seeds unless otherwise stated. For OGBench visual tasks, we report the average success rates across the last three evaluation epochs (300K, 400K, 500K), each evaluated over 50 episodes, following the protocol of FQL [16] due to computational constraints.

Bandit multimodal diagnostic. For Figure 2, FQL-D uses $\alpha = 15$ and FQL-Q uses $\alpha = 1$, where larger α places greater relative weight on the distillation term. For reference, at $\alpha = 1000$, the

resulting policy becomes nearly indistinguishable from BC. The reward landscape contains two high-value modes with maximum reward $+20$ and a suboptimal band across the middle with reward $+10$ (the faint red region).

Four-mode environment diagnostic We consider a two-dimensional point-mass environment adapted from the multi-goal environment of [32]. The state and action spaces are bounded by $[-7, 7]^2$ and $[-1, 1]^2$, respectively, and the dynamics are given by $s' = \mathrm{clip}(s + a)$. Four symmetric goals are located at $\mathcal{G} = \{(5, 0), (-5, 0), (0, 5), (0, -5)\}$. The transition reward is

$$r(s, a, s') = -30\|a\|_2^2 - \min_{g\in\mathcal{G}}\|s' - g\|_2^2 + 10 \cdot \mathbb{I}\left[\min_{g\in\mathcal{G}}\|s' - g\|_2 < 1\right],$$

and an episode terminates upon reaching any goal or after 30 steps. The reward contours in Figure 4 visualize the state-dependent term $-\min_{g\in\mathcal{G}}\|s-g\|_2^2$; consequently, all four goals are equally optimal from the symmetric initial region.

We first collect 100,000 transitions using a uniform-random behavior policy. To introduce asymmetric coverage while retaining all four directional modes, we independently remove 80% of transition tuples (s, a, s') for which either s or s' lies below the diagonal $y = x$. This leaves 55,044 transitions and changes the initial action proportions from approximately uniform to $\mathrm{R/L/T/B} = 16.2/35.4/34.7/13.7\%$. For FQL, we sweep $\alpha \in \{0.01, 0.1, 0.3, 0.5, 1, 3, 5, 10\}$, where larger α places greater relative weight on flow-policy distillation. The figure presents representative results for $\alpha \in \{0.1, 1, 10\}$. For OptiFlow-BC, we use a standard flow-BC reference policy and set the value-weighted marginal temperature to $\tau = 0.1$; no VaBC temperature η is used.

OT assignment stability diagnostic Using the same four-mode environment, we examine whether OT assignments remain directionally consistent when the reference actions are resampled. With all networks frozen, we fix four one-step actions at a given state, with one action pointing toward each of the right, left, top, and bottom goals. We then independently resample $M = 64$ reference actions 100 times and recompute the value-weighted marginal, Sinkhorn transport plan, and hard assignments at every round. Each assigned reference action is labeled according to the goal direction toward which it moves. We perform this diagnostic at the symmetric origin, where all four directions are equally valuable, and at nearby states where the values are no longer tied.

Online fine-tuning. For offline-to-online RL, we extend offline OptiFlow training with 1M additional gradient steps starting from the 1M-step offline checkpoint, adding online transitions to a unified replay buffer that already contains the offline dataset. Online data are collected using the deterministic one-step policy with no exploration noise, and we continue to train all components of OptiFlow with the same objective as in offline training.

Hyperparameters. We provide complete lists of hyperparameters used across all tasks and methods in Tables 5 to 8. We use the hyperparameters setup shown in [16] for the baseline experiments and those presented in [17].

Modified TD target. Standard TD target takes the form $y = r + \gamma Q(s', a')$. [17] presents a variant of the TD target $y^{\text{VaBC}}(s, r, s') = r + \frac{\gamma}{2}\left(Q_{\bar{\phi}}(s', \mu_\theta(s', z)) + Q_{\bar{\phi}}(s', \mu_\omega(s', z))\right)$ where $z \sim \mathcal{N}(0, I_d)$. Here $\mu_\theta(s', z)$ denotes the action from the one-step policy and $\mu_\omega(s', z)$ denotes the action from the reference policy. This TD target is formulated as the average of the two estimates of Q-value. This is a more conservative variant of the TD target, and we report results of OptiFlow and GFP on the specified tasks in Table 6

Table 5: **Shared hyperparameters used across all tasks and methods** unless otherwise noted in Table 6. We evaluated across all tasks and methods equally according to Table 5 and 6 where relevant.

Hyperparameter	**Value**
MLP hidden dims	$[512, 512, 512, 512]$
Activation	GeLU [36]
LayerNorm [37]	Critic only
Clipped double Q-learning	False (default), True (antmaze-{large, giant}-navigate, adroit)
Optimizer	Adam [40]
Learning rate	3×10^{-4}
Discount γ	0.99 (default), 0.995 (antmaze-giant, humanoidmaze, antsoccer)
TD target	Standard
Target network smoothing coefficient	0.005
Batch size	256
Gradient steps	1M (OGBench), 500k (OGBench visual tasks, D4RL)
Critic update	Every step (default), every 5 steps (antsoccer for OptiFlow only)
Euler integration steps K	10
Student samples N	16
Teacher samples M	64
Sinkhorn iterations T	30
Entropic regularization ε	0.05

Table 6: **Per-task overrides to the shared defaults** in Table 5 for OptiFlow and GFP only. We bring these overrides exactly from GFP setup. All other baselines and tasks not listed use the shared defaults. We make these overrides in order to maintain consistency with all baselines following the convention, and to provide accurate comparison against GFP which makes the following exact overrides due to apparently significant performance improvements. OptiFlow also demonstrates slightly improved performance with this setting. We report OptiFlow and GFP results on the following tasks with these overrides and all other tasks not listed follow the shared defaults in Table 5.

Task	**Discount γ**	**TD target**
antmaze-large-navigate-singletask-task{1-5}-v0	0.995	–
humanoidmaze-large-navigate-singletask-task{1-5}-v0	0.999	–
humanoidmaze-medium-navigate-singletask-task{1-5}-v0	–	y^{VaBC}
cube-single-play-singletask-task{1-5}-v0	–	y^{VaBC}
cube-double-play-singletask-task{1-5}-v0	–	y^{VaBC}
cube-triple-play-singletask-task{1-5}-v0	–	y^{VaBC}

Table 7: **Task-specific hyperparameters for offline RL on OGBench.** We use the hyperparameters used by For OptiFlow, τ is the teacher-marginal softmax temperature and η is the value-aware BC temperature. "-" indicates that the experimental result is taken from prior work (or does not exist).

Task	BC	IQL α	ReBRAC (α_1, α_2)	IDQL N	IFQL N	FQL α	GFP (α, η)	**OptiFlow** (τ, η)
antmaze-large-navigate-singletask{1-5}-v0	–	10	(1e-3, 1e-2)	32	32	10	(0.3,1e-4)	(2.0, 1e-2)
antmaze-giant-navigate-singletask{1-5}-v0	–	10	(1e-3, 1e-2)	32	32	10	(0.1,1e-1)	(1.0, 1e-3)
humanoidmaze-medium-navigate-singletask{1-5}-v0	–	10	(1e-2,1e-2)	32	32	30	(0.3,1e-3)	(1.0, 1e-3)
humanoidmaze-large-navigate-singletask{1-5}-v0	–	10	(1e-2,1e-2)	32	32	30	(0.3,1e-4)	(2.0, 1e-1)
antsoccer-arena-navigate-singletask{1-5}-v0	–	1	(1e-2,1e-2)	32	64	10	(0.1,1e-2)	(1.0, 1e-2)
cube-single-play-singletask{1-5}-v0	–	1	(1,0)	32	32	300	(10,1e-1)	(2.0, 1e-1)
cube-double-play-singletask{1-5}-v0	–	0.3	(1e-1,0)	32	32	300	(1.0,1e-2)	(2.0, 1e-2)
cube-triple-play-singletask{1-5}-v0	–	1	(1e-1,0)	64	128	300	(0.1,1e-5)	(1.0, 1e-5)
scene-play-singletask{1-5}-v0	–	10	(1e-1,1e-2)	32	32	300	(10,1e-3)	(1.0, 1e-2)
puzzle-3x3-play-singletask{1-5}-v0	–	10	(3e-1,1e-2)	32	32	1000	(3.0,1e-3)	(1.0, 1e-3)
puzzle-4x4-play-singletask{1-5}-v0	–	3	(3e-1,1e-2)	32	32	1000	(3.0,1e-5)	(2.0,1e-2)
visual-cube-single-play-singletask1-v0	–	–	–	–	–	–	(10,1e-1)	(1.0,1e-1)
visual-cube-double-play-singletask1-v0	–	–	–	–	–	–	(0.3,1e-2)	(2.0,1e-2)
visual-scene-play-singletask1-v0	–	–	–	–	–	–	(10,1e-3)	(1.0,1e-1)
visual-puzzle-3x3-play-singletask1-v0	–	–	–	–	–	–	(3.0, 1e-2)	(1.0,1e-2)
visual-puzzle-4x4-play-singletask1-v0	–	–	–	–	–	–	(1.0, 1e-4)	(1.0,1e-1)

Table 8: **Task-specific hyperparameters for offline RL on D4RL.** For OptiFlow, τ is the teacher-marginal softmax temperature and η is the value-aware BC temperature. "-" indicates that the experimental result is taken from prior work (or does not exist).

Task	BC	IQL α	ReBRAC (α_1, α_2)	IDQL N	IFQL N	FQL α	GFP (α, η)	**OptiFlow** (τ, η)
antmaze-umaze	–	–	–	–	32	10	(0.1,1e-3)	(1.0, 1e-3)
antmaze-umaze-diverse	–	–	–	–	32	10	(0.1,1e-3)	(1.5, 1e-6)
antmaze-medium-play	–	–	–	–	32	10	(0.03,1e-3)	(0.8, 1e-3)
antmaze-medium-diverse	–	–	–	–	32	10	(0.03,1e-3)	(2.0, 1e-4)
antmaze-large-play	–	–	–	–	32	3	(0.03,1e-5)	(0.5, 1e-5)
antmaze-large-diverse	–	–	–	–	32	3	(0.03,1e-5)	(0.8, 1e-4)
pen-human-v1	–	–	–	32	32	10000	(3.0,1e-4)	(5.0, 1e-3)
pen-cloned-v1	–	–	–	32	32	10000	(3.0,1e-5)	(5.0, 1e-3)
pen-expert-v1	–	–	–	32	32	3000	(1.0,1e-3)	(3.0, 1e-3)
door-human-v1	–	–	–	32	32	30000	(10,1e-2)	(3.0, 1e-5)
door-cloned-v1	–	–	–	32	128	30000	(10,1e-2)	(1.0, 1e-6)
door-expert-v1	–	–	–	32	32	30000	(10,1e-2)	(10, 1e-3)
hammer-human-v1	–	–	–	128	32	30000	(10,1e-5)	(5.0, 1e-5)
hammer-cloned-v1	–	–	–	32	32	10000	(10,1e-5)	(50, 1e-3)
hammer-expert-v1	–	–	–	32	32	30000	(10,1e-2)	(5.0, 1e-2)
relocate-human-v1	–	–	–	32	128	10000	(10,1e-4)	(10, 1e-3)
relocate-cloned-v1	–	–	–	64	32	30000	(10,1e-4)	(5.0, 1e-4)
relocate-expert-v1	–	–	–	32	32	30000	(10,1e-4)	(5.0, 1e-4)

E Complete Experimental Results

This section presents experimental results spanning diverse OGBench (Table 9, 10) and D4RL (Table 11) benchmarks tasks. Table 5 and Table 6 indicate the hyperparameters used across all tasks and methods. All results are averaged over 8 random seeds except for the visual tasks which are averaged over 4 random seeds due to high computational cost.

Table 9: **OGBench results part 1/2.** All methods are trained for 1M gradient steps (500K for visual tasks) and evaluated at the final step over 100 episodes. We report success rate as mean ± standard error across 8 random seeds for state-based tasks and 4 random seeds for visual tasks. **Bold** indicates results within 95% of the best score on each task. *Italics* indicate results taken from prior work [16].

Task	BC	IQL	ReBRAC	IDQL	IFQL	FQL	GFP	**OptiFlow**
antmaze-large-navigate-singletask								
task1-v0	0.0±0.0	51.4±4.3	**93.3**±1.5	58.1±3.1	31.9±6.0	79.5±2.1	**94.5**±1.4	**94.0**±0.9
task2-v0	0.0±0.0	40.3±5.3	**87.7**±1.1	0.0±0.0	8±1.9	61.4±3.6	**91.1**±1.4	**87.6**±1.1
task3-v0	0.9±0.4	71.9±3.1	75±6.4	27.5±4.9	34.5±8.4	**94.6**±1.2	**97.2**±0.7	**93.8**±1.0
task4-v0	6.3±0.7	48.5±3.9	**87.4**±1.6	0.0±0.0	18.9±5.2	81.4±1.9	**91.3**±0.8	**88.8**±1.1
task5-v0	1.3±0.5	53.1±7.9	86.5±2.4	48.5±3.8	46.6±4.6	81.3±1.2	**95.6**±0.9	**95.1**±1.0
antmaze-giant-navigate-singletask								
task1-v0	0.0±0.0	0.0±0.0	**39.4**±9.9	0.0±0.0	0.0±0.0	10.5±2.9	8.6±2.5	5.4±2.4
task2-v0	0.0±0.0	0.1±0.1	16.6±9.0	0.0±0.0	0.0±0.0	16.4±4.9	**36.6**±4.6	25.9±3.4
task3-v0	0.0±0.0	0.1±0.1	**53.5**±8.8	0.0±0.0	0.0±0.0	0.0±0.0	3.3±0.7	0.4±0.2
task4-v0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	28.1±7.8	**43.0**±4.3
task5-v0	0.1±0.1	21.4±4.1	52.6±4.8	0.0±0.0	4.5±2.2	20.6±9.8	17.9±8.3	**57.9**±4.6
humanoidmaze-medium-navigate-singletask								
task1-v0	0.1±0.1	32±3.9	13.9±6.5	2.8±0.5	61.9±8.4	19±4.2	**84.8**±2.1	**82.4**±1.2
task2-v0	0.4±0.3	39.5±2.9	10.6±3.2	3.9±0.4	84.3±8.8	**94**±1.1	**92.3**±1.6	**92.9**±1.7
task3-v0	4.5±0.3	30.1±1.9	47.6±6.2	1.6±0.7	45.9±17.0	74±6.4	**85.1**±4.0	**82.1**±1.8
task4-v0	0.6±0.2	0.3±0.2	10.5±4.4	0.6±0.3	0.0±0.0	3±1.4	0.1±0.1	**19.1**±5.4
task5-v0	1.4±0.6	66.9±4.7	12.8±5.6	5.9±0.9	**98.1**±0.4	**97**±0.7	**96.5**±1.4	**95.5**±0.7
humanoidmaze-large-navigate-singletask								
task1-v0	0.0±0.0	2.3±0.6	0.8±0.4	0.1±0.1	12±2.3	1±0.5	36±5.8	**48.3**±2.2
task2-v0	0.1±0.1	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	**0.6**±0.6	0.0±0.0
task3-v0	0.1±0.1	6.5±1.5	9.9±1.6	1.9±0.5	**51.1**±3.7	15.4±3.0	8.1±2.3	20.9±1.8
task4-v0	0.0±0.0	1±0.3	1.8±1.5	0.0±0.0	0.3±0.2	0.1±0.1	**3.8**±3.7	0.1±0.1
task5-v0	0.0±0.0	1.9±0.5	0.9±0.5	0.0±0.0	0.0±0.0	1.5±1.2	6.7±4.9	**26.3**±4.4
antsoccer-arena-navigate-singletask								
task1-v0	0.3±0.2	15.6±0.8	0.1±0.1	15.8±2.6	66.9±7.3	**79.3**±2.0	**79.1**±2.5	67.9±3.3
task2-v0	0.5±0.2	17.3±1.6	0.3±0.3	0.0±0.0	46.3±12.7	**92.8**±1.5	**94.1**±1.9	81.5±1.6
task3-v0	0.0±0.0	5.8±1.9	0.0±0.0	1.8±0.6	5.8±3.7	**54.8**±2.8	**52.5**±2.4	43.3±3.9
task4-v0	0.0±0.0	3.1±0.9	0.3±0.3	0.0±0.0	26.4±2.8	38.1±4.4	42.5±3.8	**46.8**±4.1
task5-v0	0.0±0.0	5.4±1.4	0.0±0.0	0.0±0.0	0.3±0.3	**42.0**±5.1	37.2±3.0	23.8±4.2
cube-single-play-singletask								
task1-v0	2.5±1.3	56.3±4.3	92.1±1.3	92±0.8	79.1±1.5	93.5±1.8	**96**±0.9	**99.1**±0.3
task2-v0	6.1±4.3	53.5±6.1	86.7±4.2	**98.5**±0.5	89.5±0.8	**96.1**±0.6	**97.7**±1.1	**98.5**±0.5
task3-v0	4.9±1.9	57.9±6.2	89.7±2.1	**99.6**±0.2	89.3±0.9	**95.1**±1.8	**98.8**±0.6	**99.0**±0.6
task4-v0	0.4±0.4	42±6.1	87.3±2.5	86.5±0.5	79±2.1	84.5±3.4	**95.7**±1.0	**95.8**±0.9
task5-v0	0.0±0.0	40.8±5.5	83.1±2.7	82.9±2.1	82.5±2.7	85.8±3.4	**93.5**±1.7	**95.4**±1.2
cube-double-play-singletask								
task1-v0	0.0±0.0	11.6±1.1	27.2±6.3	3.9±1.2	27.5±2.8	50.1±4.1	73.5±3.6	**80.4**±2.3
task2-v0	0.0±0.0	0.0±0.0	6.4±1.2	4.4±1.9	9.8±1.1	33.8±5.1	48.3±3.7	**52.4**±2.9
task3-v0	0.0±0.0	0.0±0.0	3.3±0.8	0.5±0.2	5.3±0.8	21.1±2.2	40.6±3.7	**46.3**±3.3
task4-v0	0.0±0.0	0.0±0.0	1.6±0.7	0.0±0.0	0.8±0.3	4.6±0.9	5.2±1.2	**16.8**±1.3
task5-v0	0.0±0.0	0.4±0.3	1.9±0.6	5.4±2.5	8.9±1.0	16.6±3.1	**48.7**±4.3	34.6±3.9
cube-triple-play-singletask								
task1-v0	0.2±0.1	2.7±1.0	10.8±1.8	2.1±1.0	1.2±0.6	5.7±2.1	9.0±1.6	**17.9**±0.8
task2-v0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
task3-v0	0.0±0.0	0.0±0.0	**0.3**±0.1	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
task4-v0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
task5-v0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0

Table 10: **OGBench results 2/2.** We follow the same evaluation protocol as Table 9. Dashes denote unreported entries.

Task	BC	IQL	ReBRAC	IDQL	IFQL	FQL	GFP	**OptiFlow**
puzzle-3x3-play-singletask								
task1-v0	19.1±2.9	12.1±2.4	**96.9**±0.9	69.9±3.7	92±1.3	90.4±1.4	**96.1**±1.2	**95.3**±1.5
task2-v0	0.1±0.1	3.3±0.6	1.8±0.9	0.0±0.0	2.4±1.1	**16.9**±2.3	0.2±0.1	0.0±0.0
task3-v0	0.0±0.0	1.4±0.4	1.9±0.6	0.0±0.0	1±0.3	**12.0**±1.3	0.7±0.2	0.0±0.0
task4-v0	0.0±0.0	2.5±0.5	5.0±1.8	0.0±0.0	0.1±0.1	11.1±1.8	5.3±1.2	**25.3**±6.8
task5-v0	0.3±0.3	2.3±0.7	6.1±1.3	0.0±0.0	0.9±0.3	17.3±1.8	**19.6**±5.3	**20.1**±8.9
puzzle-4x4-play-singletask								
task1-v0	0.0±0.0	3.3±0.9	23.8±1.1	26.1±2.5	43.9±2.5	28.9±2.7	**53.3**±3.1	**51.8**±3.6
task2-v0	0.0±0.0	1.6±0.4	10.0±0.6	3.5±1.4	**16.9**±3.5	13.3±1.3	7.2±1.2	11.6±3.7
task3-v0	0.0±0.0	1.3±0.3	16.8±1.1	42.9±3.2	40±2.2	16.3±1.6	46.0±3.9	**63.1**±3.6
task4-v0	0.0±0.0	1.0±0.5	11.0±1.6	**33.3**±4.2	22.9±2.5	8.8±1.1	16.2±1.7	22.8±2.9
task5-v0	0.0±0.0	1.3±0.4	8.6±0.7	4.5±0.4	10.5±1.1	7.8±1.2	6.3±1.2	**11.1**±2.7
scene-play-singletask								
task1-v0	1.8±0.7	65±2.6	**95.6**±1.4	**99.8**±0.2	**99.4**±0.4	**100**±0	**99.6**±0.2	**100**±0
task2-v0	0.0±0.0	10.9±1.9	46.1±5.5	0.3±0.2	43.4±8.0	78.1±6.4	91.0±2.9	**97.5**±1.1
task3-v0	0.0±0.0	23±2.1	67.5±5.2	**95.3**±1.3	80.8±2.4	**95.1**±0.8	75.0±7.6	**98.0**±0.6
task4-v0	0.0±0.0	1.1±0.5	0.9±0.4	0.0±0.0	0.1±0.1	**6.0**±2.3	0.0±0.0	1.5±0.8
task5-v0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0	0.0±0.0
visual-{–}-play-singletask-task1-v0								
cube-single	-	*70±6*	***83***±3	-	*49±3.5*	*81±6*	72.2±3.6	**86.6**±1.3
cube-double	-	*34±11.5*	*4±2*	-	*8±3*	*21±5.5*	12.0±2.6	**66.7**±3.5
scene	-	***96***±1	***98***±2	-	*86±5*	***98***±1.5	**98.7**±0.3	92.8±0.8
puzzle-3x3	-	*7±7.5*	*88±2*	-	***100***±0	*94±0.5*	86.5±1.1	93.8±1.2
puzzle-4x4	-	*0.0±0.0*	*26±3*	-	*8±7.5*	*33±3*	19.3±1.4	**39.7**±4.8

Table 11: **D4RL results** across AntMaze navigation and Adroit dexterous manipulation tasks. All methods are trained for 500K gradient steps and evaluated at the final step over 100 episodes. We report scores as mean ± standard error across 8 random seeds, following D4RL conventions: success rate for AntMaze and normalized return for Adroit. **Bold** indicates results within 95% of the best score on each task. *Italics* indicate results taken from prior work [14, 35].

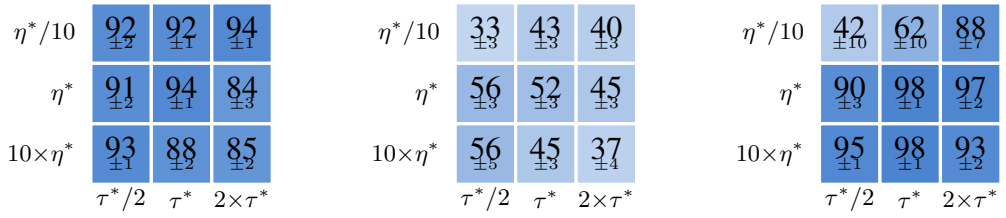
Task	BC	IQL	ReBRAC	IDQL	IFQL	FQL	GFP	**OptiFlow**
antmaze-umaze	*54.6*	*83.3*	***97.8***	***94.0***	86.5±2.1	**95.0**±1.8	**96.5**±0.5	**96.0**±0.9
antmaze-umaze-diverse	*45.6*	*70.6*	***88.3***	*80.2*	58.5±5.3	83.3±4.8	**91.8**±1.6	84.9±1.6
antmaze-medium-play	*0.0*	*64.6*	***84.0***	***84.2***	54.8±6.5	76.5±2.7	83.2±3.2	**88.0**±1.4
antmaze-medium-diverse	*0.0*	*61.7*	*76.3*	***84.8***	60.3±8.8	70.0±4.2	57.5±8.2	**87.4**±1.6
antmaze-large-play	*0.0*	*42.5*	*60.4*	*63.5*	46.3±4.8	**87.0**±1.7	82.1±1.8	82.6±2.4
antmaze-large-diverse	*0.0*	*27.6*	*54.4*	*67.9*	59.5±7.4	**81.75**±1.4	**84.5**±1.2	**80.6**±1.8
pen-human-v1	*34.4*	*81.5*	***103.5***	56.1±2.3	69.8±2.4	40.0±3.5	59.1±3.6	61.1±2.3
pen-cloned-v1	*56.9*	*77.2*	***91.8***	74.2±1.9	76.9±2.8	74.9±4.5	78.7±2.7	67.4±2.5
pen-expert-v1	*85.1*	*133.6*	***154.1***	133.7±1.2	135.7±1.9	144.8±1.2	141.2±2.0	127.6±2.2
door-human-v1	*0.5*	*3.1*	*0.0*	**3.7**±0.6	2.9±0.6	0.1±0.1	0.1±0.0	2.0±0.4
door-cloned-v1	*−0.1*	*0.8*	*1.1*	**6.1**±1.3	0.2±0.2	1.7±0.5	0.3±0.2	3.0±1.6
door-expert-v1	*34.9*	***105.3***	***104.6***	**103.9**±0.3	97.9±2.6	**104.0**±0.3	**104.2**±0.2	**104.3**±0.4
hammer-human-v1	*1.5*	*2.5*	*0.2*	1.4±0.3	1.5±0.3	2.3±0.6	**5.7**±2.2	5.1±1.0
hammer-cloned-v1	*0.8*	*1.1*	*6.7*	1.8±0.3	0.2±0.2	**7.5**±2.7	**7.4**±1.5	2.1±0.5
hammer-expert-v1	*125.6*	***129.6***	***133.8***	113.7±3.4	97.9±2.6	125.7±0.4	122.4±1.2	**128.8**±0.2
relocate-human-v1	*0.0*	*0.1*	*0.0*	0.0±0.0	0.2±0.1	0.0±0.0	**0.6**±0.2	0.0±0.0
relocate-cloned-v1	*−0.1*	*0.2*	*0.9*	0.0±0.0	−0.0±0.0	0.2±0.1	**1.5**±0.3	0.0±0.0
relocate-expert-v1	*101.3*	***106.5***	***106.6***	**107.5**±0.3	95.6±2.8	**106.8**±0.8	**104.2**±2.1	**104.4**±1.3

F Ablation Studies

We conduct a comprehensive set of ablation studies to validate the design choices and probe the behavior of OptiFlow. We first analyze sensitivity to the task-specific temperatures τ and η, as well as to the Sinkhorn regularization coefficient and iteration count. We then study the effect of the reference and one-step action sample counts (M, N) and show that the gains of OptiFlow cannot be explained by a larger action-sampling budget alone. Next, we disentangle the contributions of reference-policy quality and transport-based policy extraction, and compare our default value-aware behavior cloning objective with an AWR-style weighting alternative. Finally, we examine whether direct Q-maximization provides additional benefits beyond transport-based supervision and evaluate OptiFlow under offline-to-online fine-tuning.

F.1 Sensitivity to value-weighting temperatures

OptiFlow introduces two task-specific hyperparameters: the reference-marginal softmax temperature τ, which controls how sharply the transport plan is biased toward high-value reference actions, and the value-aware BC temperature η, which controls the strength of value weighting in reference policy training.



(a) OptiFlow antmaze-large **(b) OptiFlow cube-double** **(c) OptiFlow scene**

Figure 7: **Sensitivity analysis to the reference-marginal temperature τ, and to the value-aware BC temperature η.** We evaluate the sensitivity by testing variations around task-specific hyperparameters (τ^*, η^*) from Table 7, using 8 random seeds. Figures (a), (b), and (c) report OptiFlow's sensitivity to τ^* and η^* on *antmaze-large-navigate-task1*, *cube-double-play-task2*, and *scene-play-task2*, respectively.

Result figures show that OptiFlow's performance varies smoothly around the tuned settings (τ^*, η^*), with no single hyperparameter dominating the result. It is simple to find appropriate task-specific (τ^*, η^*) values via minor tuning with an understanding of how these parameters affect the learning curve depending on the difficulty and the nature of tasks.

F.2 Sensitivity to Sinkhorn hyperparameters

We normalize each per-state cost matrix by its mean (Eq. 11), so the entropic regularization coefficient ϵ has a comparable scale across tasks. We use $\epsilon = 0.05$ and 30 Sinkhorn iterations by default, and sweep both choices on the three tasks from Figure 7.

(a) Regularization coefficient ϵ

Task	.01	**.05**	.1	.5
Antmaze	91.6±2.7	**94.0**±0.9	93.8±1.2	93.9±0.9
Cube	**62.1**±3.7	52.4±2.9	57.0±5.0	49.9±2.9
Scene	93.7±2.4	**97.5**±1.1	96.0±1.2	82.9±3.6

(b) Sinkhorn iterations

Task	10	20	**30**	40
Antmaze	91.8±1.9	92.9±1.9	**94.0**±0.9	93.3±1.9
Cube	50.3±3.7	48.9±3.2	**52.4**±2.9	51.8±3.9
Scene	96.4±1.5	94.8±2.2	**97.5**±1.1	95.9±1.4

Table 12: **Sensitivity to Sinkhorn hyperparameters.** Antmaze, Cube, and Scene denote antmaze-large-navigate-task1, cube-double-play-task2, and scene-play-task2. Evaluation protocol follows Table 9.

The sweeps show a mild task-dependent trade-off. The fixed $\epsilon = 0.05$ is a robust intermediate point on the normalized cost scale, while 30 iterations performs best on all three tasks and adds little overhead (Appendix A.4).

F.3 Reference and one-step action sampling

The transport plan in OptiFlow is constructed over N one-step actions and M reference actions sampled per state, and the quality of the resulting supervision depends on whether these sample sets adequately cover the conditional action distribution. We sweep (N, M) over $\{4, 8, 16, 32\} \times \{4, 16, 64, 256\}$, holding all other hyperparameters fixed, to identify how many samples are needed for stable transport-based distillation and show that we only need an adequate number of samples.

Table 13: **Antmaze-large-navigate-task1.** We follow the same evaluation protocol as Table 9, varying one-step samples N and reference samples M.

$N \backslash M$	4	16	**64** (default)	256
4	2.8±0.8	80±2.3	92.2±2.2	**97.2**±0.3
8	7.8±2.8	38.7±4.0	90.7±0.7	**96.7**±0.6
16 (default)	18.5±4.3	39.8±3.3	**94.0**±0.9	**94.2**±0.6
32	4.8±0.7	29.8±3.1	81.8±3.0	83.0±2.2

It can seen that performance depends mainly on M and is largely insensitive to N. The default $(N, M) = (16, 64)$ matches or achieves the best result on each task: smaller M degrades performance sharply, while $M = 256$ gives consistent gain as can be expected. Sampling larger number of reference actions may be advantageous for distill target actions selection. Due to performance vs computation trade off, we use $(N, M) = (16, 64)$ by default, concluding that OptiFlow only needs sufficient number of one-step and reference action samples, and excessive number of samples are not critical.

F.4 Controlling for the action-sampling budget

OptiFlow samples multiple actions per state to construct the transport coupling, raising the possibility that its gains arise simply from a larger action-sampling budget rather than from transport-based supervision. To control for this factor, we modify FQL and GFP to draw either $N = 16$ or $N = 64$ one-step actions per state and average their actor gradients over the sampled actions, while leaving the remaining training procedure unchanged.

Table 14: **Controlling for the action-sampling budget.** FQL and GFP with $N = 16$ or $N = 64$ draw multiple actions per state and average the actor gradient over the sampled actions. OptiFlow uses $N = 16$ one-step policy samples per state. We follow the same evaluation protocol as Table 9.

Task	**FQL (original)**	**FQL** ($N = 16$)	**FQL** ($N = 64$)	**GFP (original)**	**GFP** ($N = 16$)	**GFP** ($N = 64$)	**OptiFlow**
antmaze-large-navigate{1}	79.5±2.1	79.5±1.8	76.3±4.0	**94.5**±1.4	**95.2**±0.6	**94.4**±0.7	**94.0**±0.9
antmaze-giant-navigate{1}	10.5±2.9	8.0±2.5	12.3±3.5	8.6±2.5	15.9±7.0	**17.5**±6.8	5.4±2.4
humanoidmaze-medium-navigate{1}	19.0±4.2	25.0±10.6	10.3±4.1	**84.8**±2.1	**83.0**±1.8	**84.7**±1.0	**82.4**±1.2
humanoidmaze-large-navigate{1}	1.0±0.5	2.0±0.7	3.3±1.8	36.0±5.8	22.5±7.4	17.5±7.4	**48.3**±2.2
antsoccer-arena-navigate{4}	38.1±4.4	34.8±1.7	43.5±2.4	42.5±3.8	42.5±1.8	**44.6**±1.4	**46.8**±4.1
cube-single-play{2}	**96.1**±0.6	**97.8**±1.0	**98.0**±0.8	**97.7**±1.1	**99.3**±0.2	**98.8**±0.3	**98.5**±0.5
cube-double-play{2}	33.8±5.1	36.0±5.2	36.3±4.0	48.3±3.7	40.8±2.0	41.8±1.9	**52.4**±2.9
scene-play{2}	78.1±6.4	79.8±3.5	78.8±5.6	91.0±2.9	90.6±2.2	89.7±1.5	**97.5**±1.1
puzzle-3x3-play{4}	11.1±1.8	12.5±1.2	13.3±1.9	5.3±1.2	6.2±0.8	7.9±1.6	**25.3**±6.8
puzzle-4x4-play{4}	8.8±1.1	9.8±1.2	9.3±1.2	16.2±1.7	17.1±1.4	15.4±1.2	**22.8**±2.9
Average	37.6±1.1	38.5±1.5	38.1±1.3	52.5±0.9	51.3±0.8	51.2±1.2	**57.3**±1.0

Increasing the number of sampled actions does not yield a consistent improvement for either baseline. FQL obtains average scores of 37.6, 38.5, and 38.1 with $N = 1$, 16, and 64, respectively, while GFP obtains 52.5, 51.3, and 51.2. All variants remain below OptiFlow's 57.3. Notably, the $N = 64$ baseline variants use more one-step action samples per state than OptiFlow ($N = 16$), yet do not close the performance gap. Thus, the performance improvement cannot be explained by the number of one-step actions sampled during training alone.

F.5 Disentangling reference policy quality and distillation

To disentangle whether OptiFlow's performance gains arise from a stronger reference policy or from the distillation procedure itself, we separately evaluate the reference and one-step policies and introduce OptiFlow-BC, which replaces the VaBC reference policy with a standard BC reference while keeping the OT distillation procedure unchanged.

Table 15: **Reference and one-step policy performance.** We report the performance of the reference (Ref.) and distilled one-step policies for FQL, OptiFlow-BC, GFP, and OptiFlow. We follow the same evaluation protocol as Table 9.

	FQL		OptiFlow-BC		GFP		OptiFlow	
Task	**BC Ref.**	**one-step**	**BC Ref.**	**one-step**	**VaBC Ref.**	**one-step**	**VaBC Ref.**	**one-step**
antmaze-large-navigate{1}	1.5±1.0	79.5±2.1	1.1±0.5	**91.0**±2.0	88.5±2.3	**94.5**±1.4	18.4±3.9	**94.0**±0.9
antmaze-giant-navigate{1}	0.0±0.0	**10.5**±2.9	0.0±0.0	0.9±0.3	0.3±0.2	8.6±2.5	0.0±0.0	5.4±2.4
humanoidmaze-medium-navigate{1}	0.5±0.3	19.0±4.2	0.8±0.3	50.5±8.3	28.8±2.4	**84.8**±2.1	46.1±1.1	**82.4**±1.2
humanoidmaze-large-navigate{1}	0.0±0.0	1.0±0.5	0.0±0.0	43.4±4.1	2.1±1.1	36.0±5.8	0.0±0.0	**48.3**±2.2
antsoccer-arena-navigate{4}	1.4±0.2	38.1±4.4	1.4±0.5	43.6±3.4	7.8±1.6	42.5±3.8	11.8±2.2	**46.8**±4.1
cube-single-play{2}	10.4±2.6	**96.1**±0.6	8.6±1.5	**97.5**±1.3	44.4±6.0	**97.7**±1.1	45.1±7.1	**98.5**±0.5
cube-double-play{2}	0.3±0.2	33.8±5.1	0.0±0.0	39.1±3.5	2.3±0.6	48.3±3.7	1.4±0.4	**52.4**±2.9
scene-play{2}	2.4±0.8	78.1±6.4	2.8±0.9	**98.2**±0.8	94.0±2.5	91.0±2.9	69.8±3.7	**97.5**±1.1
puzzle-3x3-play{4}	0.4±0.2	11.1±1.8	0.1±0.1	1.1±0.7	10.3±1.6	5.3±1.2	4.3±0.5	**25.3**±6.8
puzzle-4x4-play{4}	0.3±0.2	8.8±1.1	0.0±0.0	20.9±4.2	5.1±1.6	16.2±1.7	1.5±0.3	**22.8**±2.9
Average	1.7±0.3	37.6±1.1	1.5±0.2	48.6±1.3	28.4±3.4	52.5±0.9	19.8±2.8	**57.3**±1.0

As shown in the Table 15, FQL and OptiFlow-BC show comparable reference performance, yet OptiFlow-BC yields a better one-step policy on 8 of 10 tasks, improving the average from 37.6 to 48.6. Since the reference policy is matched by construction, this indicates that a large part of OptiFlow's gain comes from the distillation process itself. Comparing OptiFlow-BC with full OptiFlow demonstrates that the Value-aware BC reference policy and the OT distillation are complementary: replacing the BC reference policy with a VaBC one improves the one-step average further, from 48.6 to 57.3. VaBC would become unnecessary only with a very large M, since the marginal can only select within the sampled pool. However, larget M increases compute cost, so VaBC is what makes a moderate M sufficient, by improving the quality of the candidate pool.

Notably, OptiFlow's reference policy is weaker than GFP's on average (19.8 vs 28.4) while its one-step policy is stronger (57.3 vs 52.5), which further indicates that reference performance alone does not determine one-step performance.

F.6 AWR-style weighting for value-aware behavior cloning

Our reference flow policy is trained using the value-aware behavior cloning (VaBC) objective in Eq. (6), where the weight g_η is implemented as a two-class softmax between the value of the dataset action and that of the current one-step policy action. As an alternative, we consider the AWR-style exponential weighting used in GFP [17]. Specifically, OptiFlow-AWR replaces the default weight g_η with

$$g_\eta^{\mathrm{AWR}}(s, a, z) = \exp\left(\frac{\lambda}{\eta}\left[Q_\phi(s, a) - Q_\phi(s, \mu_\theta(s, z))\right]\right), \tag{15}$$

while leaving all other components of OptiFlow unchanged.

Table 16: **AWR-style weighting for value-aware behavior cloning.** OptiFlow-AWR replaces the default softmax weight g_η in the reference-policy objective with an AWR-style exponential advantage weight, while leaving all other components unchanged. We follow the same evaluation protocol as Table 9.

Task	OptiFlow (default)	OptiFlow-AWR
antmaze-large-navigate{1}	94.0±0.9	97.6±0.8
antmaze-giant-navigate{1}	5.4±2.4	4.4±1.3
humanoidmaze-medium-navigate{1}	82.4±1.2	88.6±2.4
humanoidmaze-large-navigate{1}	48.3±2.2	34.2±9.2
antsoccer-arena-navigate{4}	46.8±4.1	36.8±3.9
cube-single-play{2}	98.5±0.5	98.8±0.6
cube-double-play{2}	52.4±2.9	48.8±6.3
scene-play{2}	97.5±1.1	93.4±3.4
puzzle-3x3-play{4}	25.3±6.8	1.8±1.8
puzzle-4x4-play{4}	22.8±2.9	20.6±3.9
Average	**57.3**±1.0	52.5±1.4

As shown in Table 16, AWR-style weighting improves performance on some tasks, including antmaze-large and humanoidmaze-medium, but does not provide a consistent improvement across tasks. Averaged over the 10 evaluated tasks, OptiFlow-AWR achieves 52.5 ± 1.4, compared with 57.3 ± 1.0 for the default weighting. We therefore retain the default two-class softmax weighting for the reference-policy objective.

F.7 Direct Q-maximization

A standard alternative to transport-based supervision is to optimize the one-step policy directly against the critic. We add a Q-maximization term weighted by λ_q (default $\lambda_q = 0$, recovering pure OptiFlow) to the actor objective and sweep its weight to test whether direct critic guidance complements or substitutes for transport-based distillation.

Table 17: **Direct Q-maximization ablation.** We follow the same evaluation protocol as Table 9. $\lambda_q = 0$ is the OptiFlow default reported in our main results.

Task	$\lambda_q = 0$ (default)	$\lambda_q = 10^{-3}$	$\lambda_q = 10^{-2}$	$\lambda_q = 10^{-1}$
humanoidmaze-medium-navigate-task1	**84.4**±1.2	81.6±1.7	81.4±2.4	**88.8**±1.2
antsoccer-arena-navigate-task1	67.9±2.4	42.0±3.0	45.3±2.1	**76.4**±3.3
cube-double-play-task2	**52.4**±2.9	**54.3**±3.3	**52.9**±3.9	7.5±1.2
puzzle-4x4-play-task1	**51.8**±3.6	39.4±5.2	10.4±2.1	0.9±0.4

The effect of direct critic maximization is mixed and task-dependent, likely because λ_q interacts with τ and η, which control the sharpness of value weighting in the transport plan and reference-policy training. Although $\lambda_q = 0.1$ improves humanoidmaze-medium and antsoccer-arena under our setting, it sharply degrades cube-double and puzzle-4×4, and larger values can collapse performance. Jointly tuning λ_q with τ and η may improve some tasks, but it introduces another sensitive hyperparameter.

OptiFlow already achieves strong performance across diverse benchmarks with $\lambda_q = 0$, showing that value-aware transport-based distillation is sufficient in our setting. Together with the diagnostics in Figure. 2 from the main experiment section, this supports using the critic primarily for transport-based

target construction rather than direct one-step policy maximization. Accordingly, all main OptiFlow results use $\lambda_q = 0$, isolating the effect of value-weighted OT distillation and avoiding an additional task-sensitive tuning parameter. This matches our design choice of avoiding direct critic maximization to address mode collapse and overestimation bias in out-of-distribution regions.

G Offline-to-Online Fine-tuning

While OptiFlow is designed for the offline setting, the structure of OT coupling between the reference policy and one-step target policy is naturally compatible with continued learning from newly collected data. We initialize from an offline OptiFlow checkpoint at 1M training steps and continue training with online interaction, retaining the same reference policy, transport-based target construction, and one-step policy distillation. This study examines whether OptiFlow can convert offline gains into further improvements online without algorithmic modification.

Table 18: **Task-specific OptiFlow hyperparameters for offline-to-online RL.** We individually tune these hyperparameters for each task. "–" indicates that the default values from Table 5 were used.

Task	**Q-agg**	$(\tau,\ \eta)$
humanoidmaze-medium-navigate-singletask-task1-v0	–	–
antsoccer-arena-navigate-singletask-task4-v0	–	–
cube-double-play-singletask-task2-v0	–	–
scene-play-singletask-task2-v0	–	–
puzzle-4x4-play-singletask-task4-v0	–	–
antmaze-umaze-v2	–	–
antmaze-umaze-diverse-v2	–	–
antmaze-medium-play-v2	–	–
antmaze-medium-diverse-v2	–	–
antmaze-large-play-v2	–	–
antmaze-large-diverse-v2	–	–
pen-cloned-v1	qmean	(1.0,1e-2)
door-cloned-v1	qmean	(1.0,1e-5)
hammer-cloned-v1	qmean	(2.0,1e-4)
relocate-cloned-v1	qmean	(1.0,1e-4)

Table 19: **Offline-to-online RL results.** Each cell reports the offline $\rightarrow$ online performance, and we follow the same evaluation protocol as Table 9. **Bold** indicates online results within 95% of the best online score on each task. *Italics* indicate results taken from FQL [16].

Task	IQL	ReBRAC	IFQL	FQL	**OptiFlow**
OGBench					
humanoidmaze-medium-navigate{1}	$21_{\pm 13} \rightarrow 16_{\pm 8}$	$16_{\pm 20} \rightarrow 1_{\pm 1}$	$70.5_{\pm 2.3} \rightarrow 43.0_{\pm 3.4}$	$18.1_{\pm 1.4} \rightarrow 24.6_{\pm 3.6}$	$79.0_{\pm 2.0} \rightarrow \mathbf{97.1}_{\pm 0.7}$
antsoccer-arena-navigate{4}	$2_{\pm 1} \rightarrow 0_{\pm 0}$	$0_{\pm 0} \rightarrow 0_{\pm 0}$	$40.8_{\pm 3.5} \rightarrow 36.0_{\pm 5.3}$	$33.8_{\pm 2.8} \rightarrow \mathbf{82.8}_{\pm 2.1}$	$41.0_{\pm 5.1} \rightarrow 71.5_{\pm 3.5}$
cube-double-play{2}	$0_{\pm 1} \rightarrow 0_{\pm 0}$	$6_{\pm 5} \rightarrow 28_{\pm 28}$	$11.8_{\pm 1.7} \rightarrow 55.2_{\pm 5.1}$	$35.1_{\pm 3.2} \rightarrow \mathbf{94.0}_{\pm 1.1}$	$49.1_{\pm 5.1} \rightarrow \mathbf{94.9}_{\pm 1.3}$
scene-play{2}	$14_{\pm 11} \rightarrow 10_{\pm 9}$	$55_{\pm 10} \rightarrow \mathbf{100}_{\pm 0}$	$29.0_{\pm 4.8} \rightarrow 94.2_{\pm 1.7}$	$79.5_{\pm 4.2} \rightarrow \mathbf{99.8}_{\pm 0.2}$	$94.8_{\pm 1.8} \rightarrow \mathbf{100.0}_{\pm 0.0}$
puzzle-4x4-play{4}	$5_{\pm 2} \rightarrow 1_{\pm 1}$	$8_{\pm 4} \rightarrow 14_{\pm 35}$	$23.5_{\pm 2.3} \rightarrow 25.0_{\pm 16.4}$	$9.2_{\pm 1.0} \rightarrow \mathbf{100.0}_{\pm 0.0}$	$13.0_{\pm 2.5} \rightarrow 64.1_{\pm 17.5}$
AntMaze (D4RL)					
antmaze-umaze	$77 \rightarrow \mathbf{96}$	$98 \rightarrow 75$	$91.5_{\pm 1.5} \rightarrow \mathbf{96.0}_{\pm 0.8}$	$96.8_{\pm 0.6} \rightarrow \mathbf{99.5}_{\pm 0.3}$	$95.8_{\pm 1.0} \rightarrow \mathbf{99.5}_{\pm 0.3}$
antmaze-umaze-diverse	$60 \rightarrow 64$	$74 \rightarrow \mathbf{98}$	$61.0_{\pm 9.8} \rightarrow \mathbf{95.2}_{\pm 1.0}$	$89.0_{\pm 1.9} \rightarrow \mathbf{99.4}_{\pm 0.3}$	$80.1_{\pm 2.3} \rightarrow \mathbf{99.0}_{\pm 0.4}$
antmaze-medium-play	$72 \rightarrow 90$	$88 \rightarrow \mathbf{98}$	$36.2_{\pm 9.8} \rightarrow 90.5_{\pm 1.9}$	$75.8_{\pm 2.5} \rightarrow \mathbf{97.5}_{\pm 0.3}$	$82.8_{\pm 2.0} \rightarrow \mathbf{98.1}_{\pm 0.5}$
antmaze-medium-diverse	$64 \rightarrow 92$	$85 \rightarrow \mathbf{99}$	$59.5_{\pm 8.9} \rightarrow 92.5_{\pm 1.3}$	$65.6_{\pm 5.7} \rightarrow \mathbf{96.0}_{\pm 0.7}$	$76.5_{\pm 4.5} \rightarrow \mathbf{97.6}_{\pm 0.7}$
antmaze-large-play	$38 \rightarrow 64$	$68 \rightarrow 32$	$67.5_{\pm 2.5} \rightarrow 86.5_{\pm 1.6}$	$84.0_{\pm 2.4} \rightarrow \mathbf{93.2}_{\pm 0.6}$	$83.0_{\pm 1.6} \rightarrow \mathbf{94.2}_{\pm 0.6}$
antmaze-large-diverse	$27 \rightarrow 64$	$67 \rightarrow 72$	$66.5_{\pm 3.0} \rightarrow 83.8_{\pm 2.1}$	$87.0_{\pm 1.9} \rightarrow \mathbf{95.1}_{\pm 0.7}$	$83.0_{\pm 1.7} \rightarrow \mathbf{95.8}_{\pm 0.9}$
Adroit (D4RL)					
pen-cloned	$84 \rightarrow 102$	$74 \rightarrow \mathbf{138}$	$71.6_{\pm 2.6} \rightarrow 106.0_{\pm 3.4}$	$54.4_{\pm 4.9} \rightarrow 124.0_{\pm 1.9}$	$45.2_{\pm 2.7} \rightarrow \mathbf{132.0}_{\pm 1.9}$
door-cloned	$1 \rightarrow 20$	$0 \rightarrow \mathbf{102}$	$0.3_{\pm 0.2} \rightarrow 35.2_{\pm 3.1}$	$0.3_{\pm 0.3} \rightarrow 72.7_{\pm 4.4}$	$-0.1_{\pm 0.1} \rightarrow 78.0_{\pm 3.2}$
hammer-cloned	$1 \rightarrow 57$	$7 \rightarrow \mathbf{125}$	$2.0_{\pm 0.7} \rightarrow 52.1_{\pm 5.3}$	$1.2_{\pm 0.5} \rightarrow 110.0_{\pm 9.8}$	$2.2_{\pm 1.0} \rightarrow 97.6_{\pm 9.9}$
relocate-cloned	$0 \rightarrow 0$	$1 \rightarrow 7$	$-0.1_{\pm 0.0} \rightarrow 3.2_{\pm 2.1}$	$0.5_{\pm 0.2} \rightarrow \mathbf{73.6}_{\pm 3.8}$	$-0.1_{\pm 0.0} \rightarrow -0.0_{\pm 0.1}$

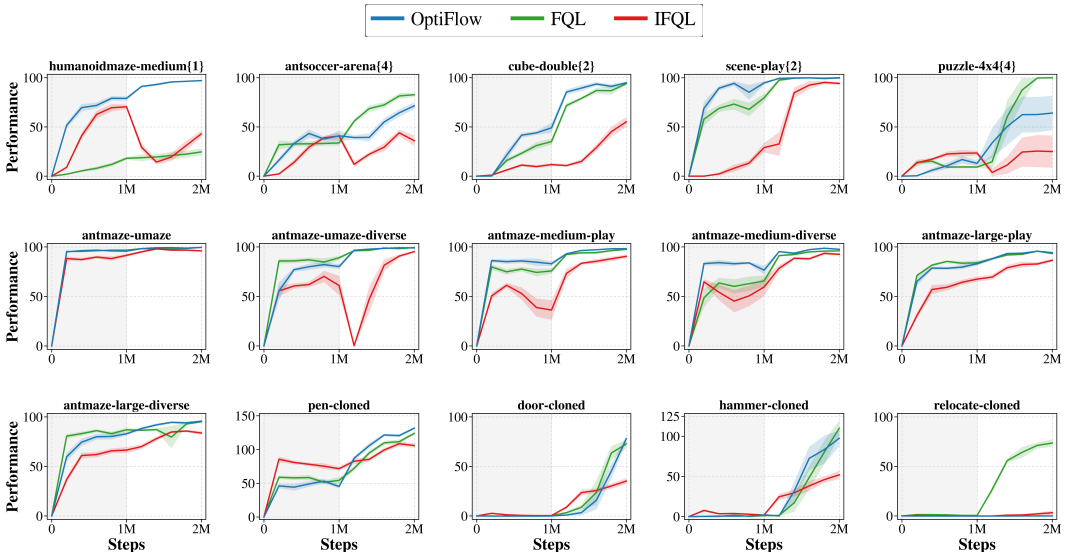


Figure 8: **Offline-to-online RL results.** Online fine-tuning starts at 1M steps. The results are averaged over 8 random seeds and plotted every 200K steps.

We report these 15 tasks to match the offline-to-online task set of FQL [16]. Across the evaluated OGBench and D4RL tasks, online fine-tuning improves over the offline checkpoint without modifying the OptiFlow training procedure. Newly collected transitions are incorporated directly into critic updates, VaBC reference-policy training, and transport-based target construction. These results suggest that the OT coupling structure is compatible with online fine-tuning and can extend beyond the strictly offline setting.

Some exceptions are puzzle-4$\times$4-play-singletask-task4-v0, which exhibits high seed variance under the default offline hyperparameters: five of eight seeds reach a score of 100, while three remain at 0; and adroit tasks that require hyperparameter tuning. Relocate-cloned task can be further tuned to show improving performance. We report this result without task-specific online tuning unless otherwise stated in Table 18. Overall, OptiFlow shows robust offline-to-online fine-tuning behavior across the evaluated tasks.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: We clearly state the contribution points and the scope in the abstract and introduction that are discussed and demonstrated throughout the paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: We provide a limitations section before the conclusion of the main paper in section A.1.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not present formal theoretical results such as theorems, propositions, or convergence guarantees.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: We provide the full code-base needed to exactly replicate the benchmark experiments. We include all hyperparameters and protocols in the Table 5. The provided code-base can reproduce the experiments with simple guideline in the README.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We include a git repository of the code-base which contains all experiment setups needed to run the experiments as we denote in the details within the Table 5 in the appendix.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.

- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: We specify every eval protocol, setup, and hyperparameters in cleanly organized tables starting with the Table 5.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: We report errors for all experimental results where appropriate. We state for each experimental table that the results are reported in standard error.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide the details on compute in the discussion section A.4.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.

- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, we understand and abide by rules and conform with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: We include our statement as a separate section in Appendix B.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

Answer: [N/A]

Justification: We release our code-base, but it is not subject to safeguards requirement.

Guidelines:

- The answer [N/A] means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: We properly cite all calls or uses of existing assets whenever appropriate.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: We provide our code-base as supplement materials as a link to anonymized repository with self-contained, explanatory README.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: Our work does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: Our work does not involve crowdsourcing nor research with human subjects.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: We do not use LLM for any important, original, or non-standard components.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.