# RETRAQ: MANIFOLD-CONSTRAINED CRITIC GUIDANCE FOR FLOW POLICIES IN RL

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

A learned critic can improve a pretrained flow-matching policy by steering sampled actions toward higher predicted value. However, even a critic that fits the behavior data well can provide misleading action gradients. Gradient components pointing away from the data-supported region are weakly constrained by behavior data and can lead guidance to exploit critic extrapolation errors. We introduce **RetraQ** (**Retra**ction-based **Q**-guidance), which uses geometry encoded by the pretrained flow to constrain both the *direction* and the *destination* of critic guidance. A single velocity evaluation of the frozen flow defines a denoising map. RetraQ filters the critic gradient with the map's Jacobian and applies the map itself to correct the updated action. Our analysis characterizes when these operations approximate tangent filtering and retraction onto the behavior manifold. The same update guides actions at test time and provides targets for policy distillation and online reinforcement learning. In controlled studies, RetraQ improves true return while keeping refined actions close to the data-supported region, and its distillation targets remain more stable than those of raw critic ascent. In offline-to-online training on LIBERO and OGBench, RetraQ achieves higher mean final success than the evaluated critic-guidance baselines. On a real-world plug-insertion task, test-time RetraQ improves the success rate of a behavior-cloned $\pi_{0.5}$ policy from $55.0\%$ to $76.7\%$. Online training further increases success to $98.3\%$ in only 30 minutes and $100\%$ within one hour.

**Project page:** `https://2027ICLR.github.io/retraq/`

## 1 INTRODUCTION

Generative policies can model complex, multimodal action distributions that arise in robot manipulation (Florence et al., 2022; Shafiullah et al., 2022). Flow-matching policies, in particular, learn a velocity field that transports noise into actions and can generate action chunks with only a few deterministic integration steps (Lipman et al., 2023; Liu et al., 2023b). Their efficient sampling and simple regression objective make them well suited to behavioral cloning (BC). Recent vision-language-action policies have adopted flow matching as their core action-generation mechanism (Black et al., 2025b;a; NVIDIA et al., 2025).

However, behavior cloning does not directly optimize task outcomes and can inherit suboptimal behavior from its training data. In precise manipulation, a policy may complete most of a task yet still fail at a critical stage, such as aligning a plug before insertion. Reinforcement learning (RL) provides a natural way to improve such behaviors using reward feedback (Luo et al., 2025; Physical Intelligence et al., 2025). For a pretrained flow policy, this feedback can be incorporated through a learned critic: following its action gradient $\nabla_{\mathbf{a}} Q(\mathbf{s}, \mathbf{a})$ can steer sampled actions toward higher predicted return (Silver et al., 2014; Wang et al., 2023; Psenka et al., 2024; Park et al., 2025b).

The difficulty is that a critic can predict returns well for actions in the training data, yet its gradients may still provide unreliable directions for policy improvement. (D'Oro & Jaśkowski, 2020). Consider the idealized case in which behavior actions at a fixed context lie on a smooth, lower-dimensional manifold. Two differentiable critics that agree on a local patch of the manifold must have the same tangential derivatives, while their normal derivatives can differ arbitrarily. Thus, even exact critic values on the data support leave the normal component of the action gradient undetermined. Following such a component can move an action away from the data-supported region, where



Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

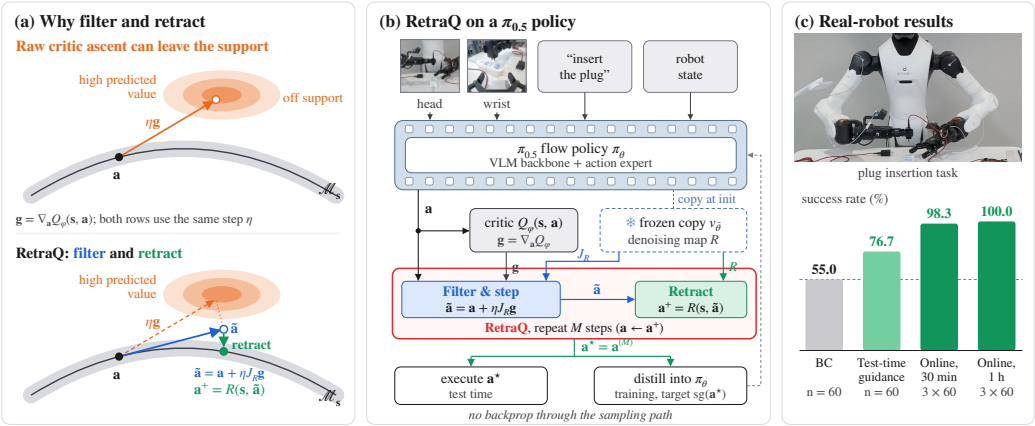



Figure 1: **RetraQ at a glance. Left:** raw critic ascent can move actions away from the behavior manifold $\mathcal{M}_{\mathbf{s}}$ by following normal gradient components. RetraQ uses the Jacobian $J_R$ of a frozen flow's denoising map $R$ to filter the gradient and uses $R$ to correct the updated action, constraining the update direction and destination. **Right:** for a $\pi_{0.5}$ policy, a frozen copy of the flow defines $R$. The refined action $\mathbf{a}^+$ is executed at test time or used as a stop-gradient training target, without backpropagating through the policy's sampling trajectory.

a high predicted value may reflect critic extrapolation error rather than true improvement (Fujimoto et al., 2019). Filtering the direction alone is not sufficient: a finite step along a tangent direction can still leave a curved manifold. Reliable critic guidance therefore requires constraining both the update *direction* and its *destination*.

We introduce **RetraQ**, which approximates both operations using geometry already encoded by a pretrained flow policy. At a chosen flow time, a frozen flow defines a denoising map from a noisy action to an estimate of its clean counterpart. RetraQ uses the Jacobian of this map to filter the critic gradient and applies the map after the ascent step to retract the updated action toward the data-supported manifold (Figure 1). Theoretically, we show that the denoising map approximates a retraction while its Jacobian approximates tangent-space projection, and we bound how errors of a learned velocity field enter both. Unlike denoising-based Riemannian optimization of a known objective (Kharitenko et al., 2026), RetraQ follows a learned critic with unidentified normal gradients and obtains both operators from a flow policy. RetraQ requires no additional geometry model, and the filtering step is implemented with a Jacobian–vector product rather than backpropagation through the full sampling trajectory. The same refinement update applies to test-time guidance, offline and online policy distillation.

We evaluate RetraQ in controlled problems, simulation benchmarks, and real-world plug insertion. In a controlled 16-dimensional problem with a known manifold, the learned flow's Jacobian attenuates normal gradient components, and RetraQ keeps refined actions closest to the manifold among the compared critic-guided updates. In these controlled studies, RetraQ also attains higher true return and more stable distillation targets than raw critic ascent. In offline-to-online training on LIBERO and OGBench, RetraQ attains higher mean final success than the evaluated critic-guidance baselines. On real-world plug insertion, test-time RetraQ raises the success rate of a behavior-cloned $\pi_{0.5}$ policy (Black et al., 2025a) from $55.0\%$ to $76.7\%$, without updating the policy parameters. With online training, success reaches $98.3\%$ within 30 minutes and $100\%$ after one hour.

Our contributions are:

- **Geometric analysis of critic-guidance.** We analyze failure modes of critic guidance and establish conditions where flow denoising approximates tangent projection and retraction.
- **Gradient filtering with flow retraction.** We introduce RetraQ, which uses a pretrained flow's Jacobian to filter critic gradients and its denoising map to retract updated actions.

- **Sim and real-robot validation.** We validate the geometric mechanism and distillation stability in controlled studies, and demonstrate policy improvement in simulation and on a real-world contact-rich manipulation task.

Section 6 discusses connections to denoising-based manifold optimization and related flow-policy improvement methods, and Appendix A extends the comparison.

## 2 PRELIMINARIES

**Flow-matching policies.** Let $\mathbf{s}$ denote the context, including the observation, robot state, and task instruction, and let $\mathbf{a} \in \mathbb{R}^D$ be an action chunk that concatenates a sequence of control commands. Let $\mu(\cdot \mid \mathbf{s})$ denote the behavior action distribution. During training, flow matching samples $\mathbf{a} \sim \mu(\cdot \mid \mathbf{s})$ and independent noise $\mathbf{z} \sim \mathcal{N}(0, I_D)$, and forms $\mathbf{x}_t = t\,\mathbf{a} + (1-t)\,\mathbf{z}, t \in [0,1]$. The velocity field $v_\theta(\mathbf{s}, \mathbf{x}, t)$ is trained by squared-error regression to $\mathbf{a} - \mathbf{z}$ (Lipman et al., 2023; Liu et al., 2023b). The policy generates actions by numerically integrating $\mathrm{d}\mathbf{x}/\mathrm{d}t = v_\theta(\mathbf{s}, \mathbf{x}, t)$ from $\mathbf{x}(0) = \mathbf{z}$ to $t = 1$. For $t < 1$, a single velocity evaluation defines the clean-action estimate

$$D_t^\theta(\mathbf{s}, \mathbf{x}) := \mathbf{x} + (1-t)\, v_\theta(\mathbf{s}, \mathbf{x}, t). \tag{1}$$

Since $\mathbf{x}_t + (1-t)(\mathbf{a} - \mathbf{z}) = \mathbf{a}$, under the population-optimal velocity $v^\star(\mathbf{s}, \mathbf{x}, t) = \mathbb{E}[\mathbf{a} - \mathbf{z} \mid \mathbf{x}_t = \mathbf{x}, \mathbf{s}]$, this map equals the posterior mean, $D_t^\star(\mathbf{s}, \mathbf{x}) = \mathbb{E}[\mathbf{a} \mid \mathbf{x}_t = \mathbf{x}, \mathbf{s}]$.

**Behavior manifold.** For each fixed context $\mathbf{s}$, we locally model the data-supported action set as a smooth $k$-dimensional embedded manifold $\mathcal{M}_\mathbf{s} \subset \mathbb{R}^D$, with $k < D$. When $\mathbf{s}$ is fixed, we write $\mathcal{M} = \mathcal{M}_\mathbf{s}$. At $\mathbf{a} \in \mathcal{M}$, the tangent space $T_\mathbf{a}\mathcal{M}$ describes first-order variations along the manifold, while the normal space $N_\mathbf{a}\mathcal{M}$ is its orthogonal complement. Let $P_T(\mathbf{a})$ and $P_N(\mathbf{a}) = I_D - P_T(\mathbf{a})$ denote the corresponding orthogonal projectors. We use the Euclidean metric in the flow's normalized action coordinates. Let $\Pi_\mathbf{s}(\mathbf{x})$ denote the point on $\mathcal{M}_\mathbf{s}$ closest to $\mathbf{x}$ in Euclidean distance. We use this nearest-point projection locally where the closest point is unique.

**Critic-guided refinement.** Let $Q(\mathbf{s}, \mathbf{a})$ be a learned estimate of expected return, differentiable in the action. Holding $\mathbf{s}$ fixed, we write $Q(\mathbf{a}) := Q(\mathbf{s}, \mathbf{a})$ and $\mathbf{g}(\mathbf{a}) := \nabla_\mathbf{a} Q(\mathbf{a})$. We study refinements of an action sampled from the pretrained flow policy:

$$\mathbf{a}^+ = \mathcal{C}(\mathbf{a} + \eta F(\mathbf{a})\mathbf{g}(\mathbf{a}))\,, \qquad \eta > 0, \tag{2}$$

where $F(\mathbf{a}) \in \mathbb{R}^{D\times D}$ transforms the critic-gradient direction and $\mathcal{C}$ corrects the resulting endpoint. Their dependence on $\mathbf{s}$ is left implicit. Raw critic ascent corresponds to $F = I_D$ and $\mathcal{C} = \mathrm{id}$. The refined action can be executed directly or used as a regression target for the flow policy.

Appendix A addresses existing critic-guided flow-policy improvement methods. Section 3 motivates tangent filtering and retraction, and Section 4 approximates both operations with the pretrained flow.

## 3 GEOMETRIC CONSTRAINTS ON CRITIC GUIDANCE

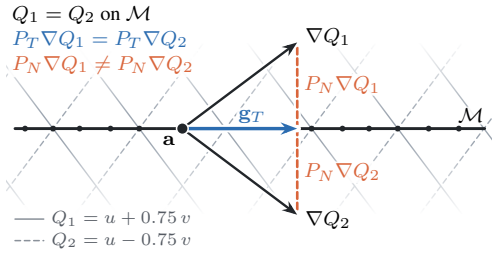
Fix the context $\mathbf{s}$, write $\mathcal{M} = \mathcal{M}_\mathbf{s}$ and $\Pi = \Pi_\mathbf{s}$, and consider an action $\mathbf{a} \in \mathcal{M}$. Suppose two continuously differentiable critics agree on a local patch of $\mathcal{M}$ around $\mathbf{a}$. Because their values change identically along every curve within this patch, their tangential gradients at $\mathbf{a}$ must agree. Their normal gradients, however, are unconstrained by these on-manifold values and can differ arbitrarily (Figure 2a). Thus, even exact critic values on the behavior manifold do not determine the full action gradient. Appendix B formalizes this observation and extends it to data concentrated in a thin neighborhood of the manifold.

**Direction: filter the critic gradient.** Let $\mathbf{g} = \nabla_\mathbf{a} Q(\mathbf{a})$ and decompose

$$\mathbf{g} = \mathbf{g}_T + \mathbf{g}_N, \qquad \mathbf{g}_T = P_T(\mathbf{a})\mathbf{g}, \quad \mathbf{g}_N = P_N(\mathbf{a})\mathbf{g}.$$

The tangential component $\mathbf{g}_T$ is determined by how the critic varies along the behavior manifold, whereas $\mathbf{g}_N$ depends on how the critic extends away from it. Following $\mathbf{g}_N$ can move the action into regions weakly supported by the behavior data, where high predicted value may reflect critic extrapolation error rather than true improvement (Fujimoto et al., 2019). We therefore retain only the tangential component, corresponding to $F(\mathbf{a}) = P_T(\mathbf{a})$ in Eq. (2).

**(a) Direction.** Values on $\mathcal{M}$ fix the tangential gradient, but not the normal one.




**(b) Destination.** A finite tangent step leaves a curved $\mathcal{M}$; a retraction returns it.

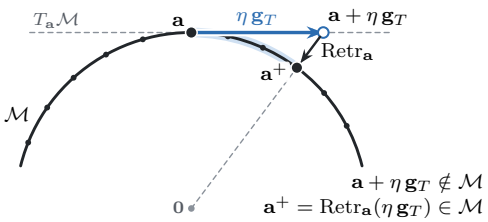



Figure 2: **Constraints on the update direction and destination. (a)** The critics $Q_1(u, v) = u + 0.75v$ and $Q_2(u, v) = u - 0.75v$ agree on the behavior manifold $\mathcal{M} = \{(u, 0) : u \in \mathbb{R}\}$. Their tangential gradients agree, while their normal components have opposite signs. **(b)** A finite tangent step from $\mathbf{a}$ leaves the circular manifold. Projecting the endpoint back onto the circle yields $\mathbf{a}^+$ and locally defines a retraction.

**Destination: correct the endpoint.** Filtering the direction alone is insufficient. Because the tangent space is only a local linear approximation, a finite step $\mathbf{a} + \eta\mathbf{g}_T$ can leave a curved manifold even when $\mathbf{g}_T$ is exactly tangent (Figure 2b). For a sufficiently small tangent displacement $\boldsymbol{\xi} \in T_{\mathbf{a}}\mathcal{M}$, nearest-point projection defines the local retraction

$$\mathrm{Retr}_{\mathbf{a}}(\boldsymbol{\xi}) := \Pi(\mathbf{a} + \boldsymbol{\xi}),$$

which returns the stepped action to $\mathcal{M}$ while preserving the tangent displacement to first order (Boumal, 2023). Choosing $F(\mathbf{a}) = P_T(\mathbf{a})$ and $\mathcal{C} = \Pi$ therefore gives the geometric update

$$\mathbf{a}^+ = \Pi\big(\mathbf{a} + \eta\mathbf{g}_T\big) = \mathrm{Retr}_{\mathbf{a}}(\eta\mathbf{g}_T). \tag{3}$$

Because $\mathbf{g}_T$ is determined entirely by the critic values on the local manifold patch, this update is unchanged by arbitrary differences in normal critic derivatives.

**Local improvement guarantee.** Let $Q^\pi$ denote the true action-value function when the refined action chunk is followed by a fixed policy $\pi$. If the learned critic $Q$ agrees with $Q^\pi$ on the local manifold patch, then their tangential gradients also agree there. Consequently, under the smoothness conditions of Proposition 2, the ideal update in Eq. (3) strictly increases $Q^\pi$ for all sufficiently small positive step sizes whenever $\mathbf{g}_T \neq \mathbf{0}$. Formal statements and proofs are given in Appendix B.

In practice, neither the behavior manifold nor the operators $P_T$ and $\Pi$ are available explicitly. Section 4 shows how the pretrained flow policy provides approximations to both.

## 4 RetraQ: Retraction-based Q-guidance

Section 3 motivates two operations for critic guidance: tangent filtering with $P_T$ and endpoint correction with $\Pi_{\mathbf{s}}$. RetraQ approximates both using a frozen behavior-cloned (BC) flow. In Eq. (2), it sets $F = J_R$ and $\mathcal{C} = R$, where $R$ is a denoising map of the flow and $J_R$ its Jacobian (Figure 3).

### 4.1 The Flow's Denoising Map as Filter and Retraction

**The denoising map.** Let $\bar{\theta}$ denote the frozen parameters of the pretrained BC flow. Dividing Eq. (2) by $t$ gives $\mathbf{x}_t/t = \mathbf{a} + \sigma_t\mathbf{z}$ with $\sigma_t := (1 - t)/t$. Thus, after rescaling, a flow state can be viewed as a clean action corrupted by Gaussian noise. At a fixed *guidance level* $t_g \in (0, 1)$, let $\sigma_g := \sigma_{t_g}$. We interpret an input action $\mathbf{a}$ as the rescaled noisy observation $\mathbf{x}_{t_g}/t_g$ and apply the clean-action estimate of Eq. (1), which takes one forward pass:

$$R(\mathbf{s}, \mathbf{a}) := D^{\bar{\theta}}_{t_g}(\mathbf{s}, t_g\mathbf{a}) = t_g\mathbf{a} + (1 - t_g)\, v_{\bar{\theta}}(\mathbf{s}, t_g\mathbf{a}, t_g), \qquad J_R(\mathbf{s}, \mathbf{a}) := \frac{\partial R(\mathbf{s}, \mathbf{a})}{\partial \mathbf{a}} \in \mathbb{R}^{D\times D}. \tag{4}$$

To see why $R$ and $J_R$ approximate these two operators, consider the population-optimal velocity, for which the clean-action map is the posterior mean of the clean action (Section 2). Let $\mathbf{A} \sim \mu(\cdot \mid \mathbf{s})$

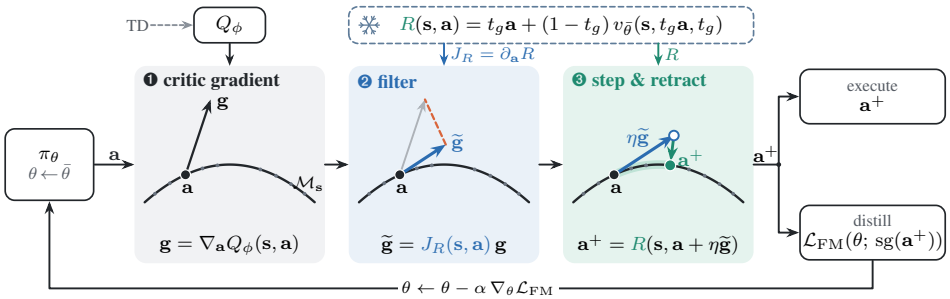



Figure 3: **One RetraQ update.** ❶ At an action $\mathbf{a}$, the critic gradient $\mathbf{g}$ may contain components normal to the behavior manifold. ❷ The Jacobian $J_R$ of the frozen flow's denoising map keeps tangent directions and suppresses normal ones. ❸ After a step along the filtered direction $\widetilde{\mathbf{g}}$, the same map moves the action back toward the behavior manifold, giving $\mathbf{a}^+$. Repeating the update gives the refined action $\mathbf{a}^\star$, which is executed or distilled into the actor $\pi_\theta$ as a stop-gradient target.

and $\mathbf{Z} \sim \mathcal{N}(0, I_D)$ be independent. Then (Manor & Michaeli, 2024)

$$R^{\mathrm{opt}}(\mathbf{s},\mathbf{a}) = \mathbb{E}\big[\mathbf{A} \mid \mathbf{A} + \sigma_g \mathbf{Z} = \mathbf{a}\big], \qquad J_R^{\mathrm{opt}}(\mathbf{s},\mathbf{a}) = \sigma_g^{-2}\,\mathrm{Cov}\big[\mathbf{A} \mid \mathbf{A} + \sigma_g \mathbf{Z} = \mathbf{a}\big]. \tag{5}$$

Near the behavior manifold, this posterior concentrates around $\Pi_\mathbf{s}(\mathbf{a})$ and varies primarily along tangent directions, suggesting $R \approx \Pi_\mathbf{s}$ and $J_R \approx P_T$. Proposition 3 (Appendix B.3) makes this precise. For a compact $\mathcal{M}_\mathbf{s}$ with smooth positive density and for points inside a tubular neighborhood of radius smaller than its reach,

$$\underbrace{R^{\mathrm{opt}}(\mathbf{s},\mathbf{a}) = \Pi_\mathbf{s}(\mathbf{a}) + O(\sigma_g^2)}_{\text{❸ retraction, } \mathcal{C}=\Pi_\mathbf{s}} \qquad \text{and, for } \mathbf{a} \in \mathcal{M}_\mathbf{s}, \qquad \underbrace{J_R^{\mathrm{opt}}(\mathbf{s},\mathbf{a}) = P_T(\mathbf{a}) + O(\sigma_g^2)}_{\text{❷ tangent filter, } F=P_T}. \tag{6}$$

For a learned flow, these approximations incur additional error from the velocity field and its input derivatives (Proposition 4). We use a fixed guidance level $t_g = 0.9$ across all platforms. Appendix C.1 discusses this choice.

### 4.2 The RetraQ Update

Replacing $P_T$ by $J_R$ and $\Pi_\mathbf{s}$ by $R$ in the ideal update of Eq. (3) gives RetraQ. For a context $\mathbf{s}$, an action $\mathbf{a}$, and a differentiable critic $Q$,

$$\begin{aligned} &❶ \quad \mathbf{g} = \nabla_\mathbf{a} Q(\mathbf{s},\mathbf{a}) && \text{critic ascent direction,} \\ &❷ \quad \widetilde{\mathbf{g}} = J_R(\mathbf{s},\mathbf{a})\,\mathbf{g} && \text{filter the direction,} \\ &❸ \quad \mathbf{a}^+ = R\big(\mathbf{s}, \mathbf{a} + \eta\,\widetilde{\mathbf{g}}\big) && \text{step and retract.} \end{aligned} \tag{7}$$

The update needs no manifold estimate or backpropagation through the sampling chain: step ❷ is one Jacobian–vector product through $R$. Applying Eq. (7) $M$ times to an initial action $\mathbf{a}^{(0)}$ gives the *refined action* $\mathbf{a}^\star = \textsc{Refine}(\mathbf{s}, \mathbf{a}^{(0)})$ (Algorithm 1); all experiments use $M = 1$. The step size $\eta$ is calibrated once so that the critic-induced displacement of the first step matches a target $d_{\mathrm{target}}$ (Appendix C.1).

### 4.3 Critics and Policy Improvement

RetraQ needs only the action gradient of a critic. The refined action $\mathbf{a}^\star$ is a constant ($\mathrm{sg}[\cdot]$) that is either executed or used as a regression target, in three settings that share Refine (Table 5).

**Test-time guidance.** All parameters stay fixed, and $\mathbf{a}^\star$ is executed. Standard rollouts record one action per state and so say little about how a slightly different action would fare, which is what $\nabla_\mathbf{a} Q$ must capture. We therefore collect *forked rollouts*: at a few task-critical *fork states*, we replay the same prefix, perturb the chunk that the frozen policy predicts there (the reference chunk $\mathbf{a}_c^{\mathrm{ref}}$ of fork group $c$), and let the frozen policy finish the episode. Since branches share the state, outcome differences can be attributed to the action. The fork critic scores the deviation from the reference,

$$Q_\phi(\mathbf{s},\mathbf{a}) = \mathrm{sigmoid}\big(B_\psi(\mathbf{s}) + A_\omega(\mathbf{a} - \mathbf{a}_c^{\mathrm{ref}})\big), \qquad A_\omega(\mathbf{0}) = 0, \tag{8}$$

**Algorithm 1** Offline-to-online RetraQ. REFINE (lines 1–8) is shared by all settings: test-time guidance runs only REFINE and executes $\mathbf{a}^\star$, and offline distillation skips line 12. The full version is Algorithm 2.

**Require:** frozen BC flow $v_{\bar{\theta}}$; demonstrations $\mathcal{D}_{\text{demo}}$; replay buffer $\mathcal{D}$; guidance level $t_g$; target displacement $d_{\text{target}}$; refinement steps $M$; loss weights $\lambda_{\text{imp}}, \lambda_{\text{BC}}$

1: **function** REFINE($\mathbf{s}, \mathbf{a}^{(0)}$; $\hat{Q}, \eta$)
2: **for** $\ell = 0, \dots, M-1$ **do**
3: $\mathbf{g} \leftarrow \nabla_{\mathbf{a}} \hat{Q}(\mathbf{s}, \mathbf{a}^{(\ell)})$ ▷ ❶ critic ascent direction
4: $\widetilde{\mathbf{g}} \leftarrow J_R(\mathbf{s}, \mathbf{a}^{(\ell)})\, \mathbf{g}$ ▷ ❷ damp normal part; one JVP
5: $\mathbf{a}^{(\ell+1)} \leftarrow \text{sg}\big[R(\mathbf{s}, \mathbf{a}^{(\ell)} + \eta\, \widetilde{\mathbf{g}})\big]$ ▷ ❸ step, then retract
6: **end for**
7: **return** $\mathbf{a}^\star = \mathbf{a}^{(M)}$
8: **end function**

9: $\theta \leftarrow \bar{\theta}$; $\eta \leftarrow$ unset ▷ actor starts at BC; $\bar{\theta}$ stays frozen
10: Warm up the critic on $\mathcal{D}_{\text{demo}}$; actor fixed ▷ no targets from an untrained critic
11: **while** training budget remains **do**
12: Add labeled rollouts of $\pi_\theta$ to $\mathcal{D}$ ▷ new data from current actor
13: **for** each update cycle **do**
14: Update the critic on $\mathcal{D}$; freeze $\hat{Q} = \min_i Q_{\phi_i}$ ▷ fixed critic for this cycle
15: **for** each actor step of the cycle **do**
16: Sample $\{(\mathbf{s}_j, \mathbf{a}_j^{\text{exec}})\}_j$ from $\mathcal{D}$ ▷ executed chunks
17: **if** $\eta$ is unset **then** calibrate $\eta$ on this batch ▷ then held fixed
18: $\mathbf{a}_j^\star \leftarrow$ REFINE($\mathbf{s}_j, \mathbf{a}_j^{\text{exec}}$; $\hat{Q}, \eta$); $\mathbf{a}_j^{\text{BC}} \sim \pi_{\bar{\theta}}(\cdot \mid \mathbf{s}_j)$ ▷ target, anchor
19: Take a gradient step on $\mathcal{L}_{\text{actor}}(\theta)$ ▷ Eq. (9)
20: **end for**
21: **end for**
22: **end while**

so its action gradient comes entirely from $A_\omega$. Guidance is applied at detected fork states; elsewhere the BC policy runs unchanged (Appendix C.2.1).

**Distillation.** In the other two settings, an actor $\pi_\theta$, initialized at $\bar{\theta}$, fits refined actions with the flow-matching loss $\mathcal{L}_{\text{FM}}(\theta; \mathbf{s}, \mathbf{a}) := \mathbb{E}_{t,\mathbf{z}} \|v_\theta(\mathbf{s}, \mathbf{x}_t, t) - (\mathbf{a} - \mathbf{z})\|_2^2$ and is anchored to the frozen BC policy:

$$\mathcal{L}_{\text{actor}}(\theta) = \mathbb{E}_{\mathbf{s}}\big[\lambda_{\text{imp}}\, \mathcal{L}_{\text{FM}}(\theta; \mathbf{s}, \mathbf{a}^\star) + \lambda_{\text{BC}}\, \mathcal{L}_{\text{FM}}(\theta; \mathbf{s}, \mathbf{a}^{\text{BC}})\big], \qquad \mathbf{a}^{\text{BC}} \sim \pi_{\bar{\theta}}(\cdot \mid \mathbf{s}). \tag{9}$$

*Offline distillation* refines the current actor's actions on a fixed dataset. *Offline-to-online learning* (Algorithm 1) interleaves collecting rollouts into a replay buffer $\mathcal{D}$ with critic and actor updates. Its twin TD critics (IQL (Kostrikov et al., 2022) on the robot) are regressed only at executed chunks, so their values are supported by data while their normal gradients are not, exactly the situation of Section 3; refinement therefore starts from executed chunks.

## 5 EXPERIMENTS

We evaluate RetraQ at three levels. Controlled problems with a known manifold test the mechanism directly: whether the flow's Jacobian filters normal directions, whether updates exploit the critic, and where the learned geometry begins to fail (Section 5.2). Simulation benchmarks with component ablations test whether the two constraints improve policies, and which one carries the gain (Section 5.3). A real-robot task tests the same update for test-time guidance, offline distillation, and offline-to-online learning (Section 5.4).

### 5.1 EXPERIMENTAL SETUP

We use controlled problems with a known manifold; three simulation benchmarks, LIBERO (Liu et al., 2023a) (ten Goal and ten Object tasks, and the six Spatial tasks with the lowest BC success, since the remaining Spatial tasks leave little room for improvement), RoboMimic Can (Mandlekar et al., 2022), and OGBench cube-single (Park et al., 2025a); and a real-robot insertion task with a

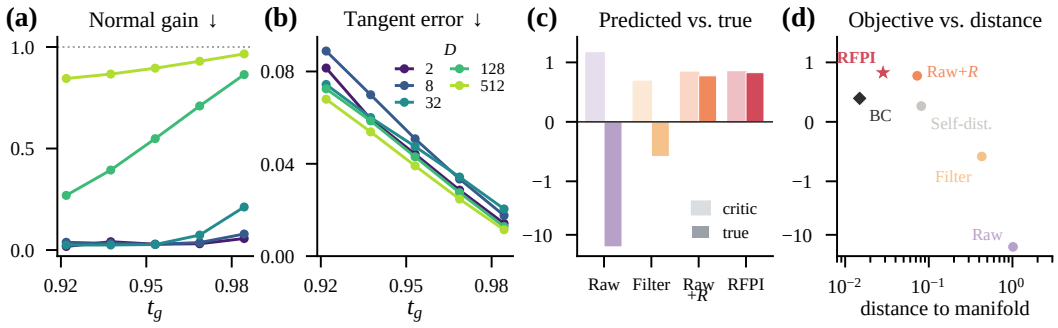



Figure 4: **Controlled problems with a known manifold. (a, b)** Median normal gain (0: ideal filter; 1: no filtering) and tangent error of the learned $J_R$ versus the guidance level, for flat manifolds of increasing dimension at a fixed flow-training budget. **(c)** $D = 16$ problem: objective predicted by the critic (light) and true objective (solid) after refinement. **(d)** True objective versus the 95th-percentile distance to the manifold. Symmetric-log scales in (c, d); details in Appendix D.2.

$\pi_{0.5}$ policy (Black et al., 2025a). Baselines are our implementations of AWR (Peng et al., 2019), FlowDPG (Shi et al., 2026), Q-VGM (Wang et al., 2026), QAM (Li & Levine, 2026), and, on the robot, QGF (Zhou et al., 2026). Component ablations isolate the two constraints: *raw ascent* $\mathbf{a}+\eta\mathbf{g}$, *filter-only* $\mathbf{a}+\eta J_R\mathbf{g}$, *Raw+R* $R(\mathbf{a}+\eta\mathbf{g})$, and RetraQ $R(\mathbf{a}+\eta J_R\mathbf{g})$; a *self-distillation* control refits the actor to its own actions. On LIBERO, we report the final checkpoint, averaged over three actor seeds and 30 episodes per task and seed. Appendix D gives the full protocols.

### 5.2 Does the Flow Provide the Geometry, and Does It Prevent Exploitation?

A known manifold lets us measure what success rates cannot: whether $J_R$ filters normal directions, whether the critic is exploited, and whether refined actions stay on the manifold (Figure 4). **The flow's Jacobian acts as a tangent filter, with the trade-off predicted by the theory.** For $D \leq 32$, the median normal gain stays below $0.1$ at all but the highest guidance level, so $J_R$ removes over $90\%$ of the normal component while keeping tangent directions (tangent error below $0.09$). Raising $t_g$ lowers the tangent error but raises the normal gain toward 1, that is, $J_R \to I_D$, as Proposition 4 predicts. The effect is strong at $D = 128$, and at $D = 512$ filtering is largely lost under the fixed training budget (normal gain $0.84$–$0.97$), where the flow also fits worst. **The retraction closes the gap between predicted and true improvement.** The critic predicts an improvement for every update. Raw ascent, however, turns a predicted objective of $1.50$ into a true objective of $-16.8$; filtering narrows this gap, but filter-only still turns $0.68$ into $-0.58$. For both retracted updates, predicted and true objectives differ by at most $0.07$ (Figure 4c). **RetraQ attains the highest true objective and stays closest to the manifold.** Its refined actions are $2.6\times$ closer to the manifold than those of Raw$+R$ and $36\times$ closer than those of raw ascent (Figure 4d), so with the retraction held fixed, the filter further reduces the distance to the manifold.

### 5.3 Policy Improvement in Simulation

On LIBERO, actors start from BC and train online with a frozen geometry reference (Table 1). **RetraQ improves BC on every suite and attains the highest mean success**, $11$–$15$ points above BC and $5.5$–$7.8$ points above the strongest baseline on each suite (Q-VGM on Spatial and Goal, QAM on Object), with three seeds per arm. **The retraction carries most of the gain, and filtering adds to it.** Without the retraction, filter-only trails RetraQ by $8.6$–$14.6$ points; adding the retraction to raw ascent (Raw$+R$) recovers $5.6$–$16.1$ points, and filtering adds another $1.9$–$4.3$ points on every suite, consistent in sign but within the variation across three seeds. Self-distillation falls below BC, so refitting the actor alone does not explain the improvement; Appendix D.7 argues that behavior regularization does not explain it either.

**What the filter changes differs across tasks.** On OGBench cube-single, the filtered gradient stays almost parallel to the raw one (median cosine $0.99$ throughout training) at $0.42$–$0.48$ of its norm. Since the step is calibrated to a fixed displacement, the filter then mainly rescales the step, and

Table 1: **Final online success (%) on LIBERO**, mean ± standard deviation over three actor seeds; actors train online from BC, and baselines are our implementations. Best in bold.

| Method | Spatial (6 tasks) | Goal (10 tasks) | Object (10 tasks) |
|---|---|---|---|
| BC | 63.3 | 69.0 | 55.3 |
| AWR | $64.8 \pm 0.6$ | $71.9 \pm 3.3$ | $58.4 \pm 0.7$ |
| FlowDPG | $60.4 \pm 3.6$ | $68.1 \pm 3.2$ | $49.3 \pm 8.0$ |
| Q-VGM | $70.6 \pm 0.6$ | $73.6 \pm 0.4$ | $59.9 \pm 2.0$ |
| QAM | $61.5 \pm 2.7$ | $73.1 \pm 0.7$ | $60.9 \pm 0.7$ |
| Self-distillation | $35.7 \pm 12.9$ | $59.6 \pm 11.9$ | $42.2 \pm 5.3$ |
| Raw ascent | $59.3 \pm 7.3$ | $67.9 \pm 5.8$ | $56.6 \pm 6.4$ |
| Filter-only | $63.5 \pm 10.1$ | $70.9 \pm 5.6$ | $57.8 \pm 6.0$ |
| Raw+$R$ | $75.4 \pm 1.7$ | $79.4 \pm 1.4$ | $62.1 \pm 6.7$ |
| **RetraQ** | $\mathbf{78.1 \pm 4.2}$ | $\mathbf{81.3 \pm 3.7}$ | $\mathbf{66.4 \pm 5.2}$ |

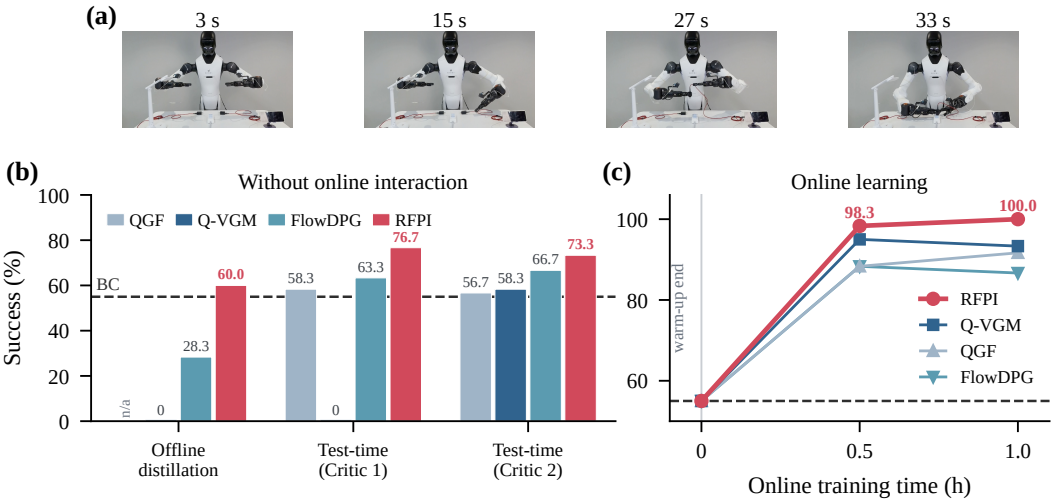



Figure 5: **Real-robot plug insertion. (a)** A successful rollout of the online RetraQ policy. **(b)** Without online interaction: offline distillation, and test-time guidance of the frozen BC policy with two fork critics (n/a: not applicable). **(c)** Offline-to-online learning from a shared BC policy and critic warm-up; means over three training runs. Numbers in Appendix D.8.

RetraQ and Raw+$R$ both approach the ceiling (98.8% and 99.0%). On RoboMimic Can, the filter also rotates the gradient (cosine 0.67–0.71), and RetraQ reaches the highest held-out success, 70.3%, against 53.0% for BC, 69.0% for Q-VGM, and 67.0% for Raw+$R$ (paired 95% interval for RetraQ minus Raw+$R$: $[-2.0, 8.7]$ points). An exploratory Can study that varies the critic's training set, with rounds selected on the evaluation episodes, suggests that RetraQ benefits more from a stronger critic (Appendix D.5).

### 5.4 Real-Robot Guidance and Online Improvement

We evaluate phone-charging plug insertion on an AgiBot G02 dual-arm robot with a behavior-cloned $\pi_{0.5}$ policy that succeeds in 55% of trials; every evaluation uses 60 trials (Appendix D.8).

**Test-time guidance.** Without any parameter update, RetraQ raises success to 76.7% and 73.3% with two fork critics, linear and nonlinear in the action deviation, each trained on 270 forked rollouts. All guidance methods take steps of the same median size, so they differ only in direction and destination; the best baseline, FlowDPG, reaches 63.3% and 66.7%, and Q-VGM falls to 0% with the linear critic (Figure 5b). With 60 trials per arm, RetraQ's gain over BC is significant with Critic 1 (Fisher's exact test, $p = 0.02$; $p = 0.06$ with Critic 2), whereas its margin over FlowDPG is not ($p \geq 0.16$). Appendix D.8 analyzes Q-VGM's failure.

**Offline distillation.** Distilling a fixed set of refined targets under one shared, large displacement budget separates the methods further. Q-VGM targets drive success to $0\%$, with the policy repeatedly hitting the table, and FlowDPG targets to $28.3\%$, with frequent misaligned insertions, whereas the RetraQ-distilled policy reaches $60.0\%$, on par with BC ($p = 0.71$).

**Offline-to-online learning.** All methods start from the same BC policy and warmed-up IQL critic, and then train their own critics under the same interaction budget; results are means over three training runs per method, each checkpoint evaluated with 60 trials per run. RetraQ reaches the highest success at every checkpoint (Figure 5c): $98.3\%$ after 30 minutes and $100\%$ after one hour, against at most $95.0\%$ and $93.3\%$ for the baselines.

## 6 Related Work

**Critic-guided improvement of generative policies.** A learned critic can improve a diffusion or flow policy through its values alone (Peng et al., 2019; Kostrikov et al., 2022; Zhang et al., 2025), by backpropagating $\nabla_{\mathbf{a}} Q$ through the generation process (Wang et al., 2023; Park et al., 2025b; Li & Levine, 2026), or by applying $\nabla_{\mathbf{a}} Q$ locally at one action without differentiating through sampling (Yang et al., 2023; Shi et al., 2026; Wang et al., 2026; Zhou et al., 2026). The local methods, closest to RetraQ, follow the gradient unfiltered and correct the endpoint at most by clipping or best-iterate selection, that is, $F = I_D$ and $\mathcal{C} \approx \mathrm{id}$ in Eq. (2). RetraQ instead uses the frozen flow's denoising map as both operators, $F = J_R$ and $\mathcal{C} = R$, and so acts on exactly the normal gradient component that the data leave undetermined (Section 3). Offline RL counters the same extrapolation by constraining the policy or its values (Fujimoto et al., 2019; Kumar et al., 2019); RetraQ constrains each update geometrically, with the behavior model already at hand.

**Riemannian optimization through denoising.** The ideal update of Eq. (3) is Riemannian gradient ascent with the projection retraction (Absil & Malick, 2012; Boumal, 2023), and a denoiser's Jacobian is a scaled posterior covariance (Efron, 2011; Manor & Michaeli, 2024). Landing with the Score (Kharitenko et al., 2026) combines the two into denoising Riemannian gradient descent (DRGD) for a known objective and a pretrained diffusion denoiser. RetraQ shares this filter–step–denoise structure but differs in each ingredient: it follows a *learned* critic, for which filtering is motivated by the non-identifiability of normal gradients (Section 3); it obtains $R$ and $J_R$ from one velocity evaluation of a context-conditioned flow policy by rescaling flow states into Gaussian-corrupted actions (Section 4.1); it bounds how errors of a learned velocity and its input derivatives enter both operators (Proposition 4); and it uses refined actions as distillation and online-learning targets. Manifold-preserving diffusion guidance (He et al., 2024) and latent-noise steering (Wagenmaker et al., 2025) are compared in Appendix A.

## 7 Conclusion

Accurate critic values on behavior data leave gradients normal to the behavior manifold undetermined. RetraQ uses the pretrained flow to constrain both the direction and the destination of critic-guided updates: the Jacobian of its denoising map filters the critic gradient, and the map itself brings the updated action back toward the manifold, approximating tangent projection and retraction. In our experiments, RetraQ outperforms the evaluated baselines across tested LIBERO suites. On real plug insertion, it improves a $\pi_{0.5}$ policy at test time and reaches near-perfect success within an hour of online training. These results show that a pretrained generative policy provides not only an expressive action distribution, but also local geometry for reliable policy improvement.

**Limitations and future work.** Repeated offline distillation without new interaction is a limitation: on LIBERO, RetraQ degrades over three refinement rounds more than AWR or FlowDPG (Appendix D.4), so its stability advantage holds against raw ascent but not in general. Separating errors in the refined targets from errors in fitting them could help stabilize offline improvement. The Jacobian–vector product through the frozen flow also adds GPU memory. Our geometric guarantees are local and asymptotic, and the learned filter weakens in high-dimensional action spaces; constraining velocity derivatives during training or adapting the guidance level may help. Refinement also remains tied to the support of the frozen reference flow; combining RetraQ with exploration or carefully updating the reference could broaden the behaviors it can learn.

## AI USE STATEMENT

In this work, we used generative AI tools (large language model assistants) to assist in creating the plotting code for the figures, formatting and checking references. We have not used generative AI tools to generate data sets, to collect or label real-robot data, or to translate text. We take responsibility for the final content of this work.

## REPRODUCIBILITY STATEMENT

Appendix B states the assumptions of every theoretical result and gives complete proofs. Appendix C specifies the operator, step-size calibration, critics, and all three settings, including the full offline-to-online algorithm (Algorithm 2) and the hyperparameters on each platform (Table 6). Appendix D gives, for each benchmark, the task subsets, seeds, evaluation episodes, checkpoint selection, and all recorded arms, as well as the real-robot platform, policy, and trial protocol. Anonymized code for the algorithms will be released.

## REFERENCES

P.-A. Absil and Jérôme Malick. Projection-like retractions on matrix manifolds. *SIAM Journal on Optimization*, 22(1):135–158, 2012. doi: 10.1137/100802529.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: A vision-language-action model with open-world generalization. In *Proceedings of the 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 17–40. PMLR, 2025a.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. $\pi_0$: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, 2025b. doi: 10.15607/RSS.2025.XXI.010.

Nicolas Boumal. *An Introduction to Optimization on Smooth Manifolds*. Cambridge University Press, 2023. doi: 10.1017/9781009166164.

Pierluca D'Oro and Wojciech Jaśkowski. How to learn a useful critic? Model-based action-gradient-estimator policy optimization. In *Advances in Neural Information Processing Systems*, volume 33, pp. 313–324. Curran Associates, Inc., 2020.

Bradley Efron. Tweedie's formula and selection bias. *Journal of the American Statistical Association*, 106(496):1602–1614, 2011. doi: 10.1198/jasa.2011.tm11181.

Herbert Federer. Curvature measures. *Transactions of the American Mathematical Society*, 93(3): 418–491, 1959. doi: 10.1090/S0002-9947-1959-0110078-1. URL https://www.ams.org/tran/1959-093-03/S0002-9947-1959-0110078-1/.

Pete Florence, Corey Lynch, Andy Zeng, Oscar A. Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 158–168. PMLR, 2022.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2052–2062. PMLR, 2019.

Yutong He, Naoki Murata, Chieh-Hsin Lai, Yuhta Takida, Toshimitsu Uesaka, Dongjun Kim, Wei-Hsiang Liao, Yuki Mitsufuji, J. Zico Kolter, Ruslan Salakhutdinov, and Stefano Ermon. Manifold preserving guided diffusion. In *International Conference on Learning Representations*, 2024.

Andrey Kharitenko, Zebang Shen, Riccardo De Santi, Niao He, and Florian Dörfler. Landing with the score: Riemannian optimization through denoising. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=xZNoeX0z9f.

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

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy Q-learning via bootstrapping error reduction. In *Advances in Neural Information Processing Systems*, volume 32, pp. 11761–11771. Curran Associates, Inc., 2019.

Qiyang Li and Sergey Levine. Q-learning with adjoint matching. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=vd4eNAdtO6.

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

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 44776–44791. Curran Associates, Inc., 2023a. doi: 10.52202/075280-1939.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *International Conference on Learning Representations*, 2023b. URL https://openreview.net/forum?id=XVjTT1nw5z.

Jianlan Luo, Charles Xu, Jeffrey Wu, and Sergey Levine. Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning. *Science Robotics*, 10(105):eads5033, 2025. doi: 10.1126/scirobotics.ads5033.

Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 1678–1690. PMLR, 2022.

Hila Manor and Tomer Michaeli. On the posterior distribution in denoising: Application to uncertainty quantification. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=adSGeugiuj.

NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. GR00T N1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

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

Seohong Park, Qiyang Li, and Sergey Levine. Flow Q-learning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 48104–48127. PMLR, 2025b.

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.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, et al. $\pi^*_{0.6}$: A VLA that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025.

Michael Psenka, Alejandro Escontrela, Pieter Abbeel, and Yi Ma. Learning a diffusion model policy from rewards via Q-score matching. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 41163–41182. PMLR, 2024.

Nur Muhammad Mahi Shafiullah, Zichen Jeff Cui, Ariuntuya Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning $k$ modes with one stone. In *Advances in Neural Information Processing Systems*, volume 35, pp. 22955–22968. Curran Associates, Inc., 2022. doi: 10.52202/068431-1668.

Kexin Shi, Junyao Shi, Poorvi Hebbar, Zhuolun Zhao, Tarun Amarnath, Yifan Su, Shikhar Bahl, and Deepak Pathak. FlowDPG: Deterministic policy gradient on flow matching policies for real-world manipulation. *arXiv preprint arXiv:2606.22303*, 2026.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pp. 387–395. PMLR, 2014.

Andrew Wagenmaker, Mitsuhiko Nakamoto, Yunchu Zhang, Seohong Park, Waleed Yagoub, Anusha Nagabandi, Abhishek Gupta, and Sergey Levine. Steering your diffusion policy with latent space reinforcement learning. In *Proceedings of the 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 258–282, 2025.

Zhendong Wang, Jonathan J. Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=AHvFDPi-FA.

Ziqian Wang, Jiayu Sun, Xingjian Mao, Minqian Wang, and Yao Mu. Q-VGM: Q-guided value-gradient matching for flow-matching VLA policies. *arXiv preprint arXiv:2606.08015v1*, 2026. URL https://arxiv.org/abs/2606.08015v1.

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

Shiyuan Zhang, Weitong Zhang, and Quanquan Gu. Energy-weighted flow matching for offline reinforcement learning. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=HA0oLUvuGI.

Zhiyuan Zhou, Andy Peng, Charles Xu, Qiyang Li, Tobias Springenberg, Kevin Frans, and Sergey Levine. Test-time gradient guidance of flow policies in reinforcement learning. *arXiv preprint arXiv:2606.11087*, 2026.

# A Extended Related Work

This appendix extends Section 6. We relate RetraQ to prior work through the template of Eq. (2), $\mathbf{a}^+ = \mathcal{C}\big(\mathbf{a} + \eta F(\mathbf{a})\mathbf{g}(\mathbf{a})\big)$ with $\mathbf{g} = \nabla_{\mathbf{a}} Q$: Appendix A.1 covers critic-guided improvement of diffusion and flow policies, and Appendix A.2 covers the Riemannian-optimization view of the direction transform $F$ and the endpoint correction $\mathcal{C}$.

## A.1 Critic-Guided Improvement of Flow Policies

**Other uses of the critic.** Value-based methods use the critic only through its values, for example by advantage-weighted regression (Peng et al., 2019; Kostrikov et al., 2022) or by weighting the flow-matching loss with $\exp(\beta Q)$ (Zhang et al., 2025); they never differentiate $Q$ with respect to

the action. Pathwise methods backpropagate $\nabla_{\mathbf{a}} Q$ through the policy's generation process, either through the full sampling chain (Wang et al., 2023), a distilled one-step actor (Park et al., 2025b), or an adjoint of the behavior field (Li & Levine, 2026). Our experiments include AWR from the first group and QAM from the second. Latent-steering methods such as DSRL (Wagenmaker et al., 2025) leave the policy weights fixed and instead run RL over the input noise of the generative policy; they change which samples the policy produces rather than refining a given action with the critic gradient, and are complementary to RetraQ.

**Local critic-gradient guidance.** The methods closest to RetraQ apply $\mathbf{g} = \nabla_{\mathbf{a}} Q$ at a single action without differentiating through the sampling process (Table 2). DIPO fits the policy to raw-ascent targets $\mathbf{a} + \eta\mathbf{g}$ (Yang et al., 2023). FlowDPG and Q-VGM evaluate $\mathbf{g}$ at the one-step clean-action estimate $D_t(\mathbf{s}, \mathbf{x}_t)$ of Eq. (1) and regress the velocity toward the ascended action, with a step scaled to the velocity norm (Shi et al., 2026) or with several ascent steps that are clipped to the action range and keep the highest-valued iterate (Wang et al., 2026). QGF adds the gradient at $D_t$ to the velocity during test-time sampling, replacing the transposed Jacobian required by the chain rule with the identity, which it reports to outperform the full Jacobian (Zhou et al., 2026). In all of these, $F$ is the identity up to a scalar or clipping, and $\mathcal{C}$ is at most coordinate clipping or a value-based acceptance rule; $D_t$ only locates where the critic is evaluated. RetraQ instead uses the frozen flow's map as the operators themselves, $F = J_R$ and $\mathcal{C} = R$, which approximate tangent filtering and retraction onto the behavior manifold (Section 4.1). Section 5 compares our implementations of FlowDPG, Q-VGM, and, on the robot, QGF; because the methods also differ in regularization and data collection, these comparisons evaluate complete procedures rather than the operators alone.

Table 2: Local critic-gradient methods in the template of Eq. (2), $\mathbf{a}^+ = \mathcal{C}(\mathbf{a} + \eta F \mathbf{g})$, as described in the cited papers. $D_t(\mathbf{s}, \mathbf{x}_t)$ is the one-step clean-action estimate of Eq. (1) at the flow state $\mathbf{x}_t$; id denotes no endpoint correction. The last three rows are the operator variants of Section 5.1. †QGF adds its guidance term to the velocity at $\mathbf{x}_t$ during sampling.

| Method | $\mathbf{g}$ evaluated at | Direction $F$ | Endpoint $\mathcal{C}$ |
|---|---|---|---|
| DIPO | replay action | $I_D$ | id |
| FlowDPG | $D_t(\mathbf{s}, \mathbf{x}_t)$ | $I_D$, step scaled to velocity norm | id |
| Q-VGM | $D_t(\mathbf{s}, \mathbf{x}_t)$ | $I_D$, gradient clipped | action-range clip, best-$Q$ iterate |
| QGF† | $D_t(\mathbf{s}, \mathbf{x}_t)$ | $I_D$ in place of $J^\top$ | id |
| Raw ascent | clean action $\mathbf{a}$ | $I_D$ | id |
| Raw+$R$ | clean action $\mathbf{a}$ | $I_D$ | $R$ |
| RetraQ | clean action $\mathbf{a}$ | $J_R(\mathbf{a})$ | $R$ |

### A.2 Riemannian Optimization through Denoising

**Retraction and denoising.** Eq. (3) is one step of Riemannian gradient ascent with the metric-projection retraction (Absil & Malick, 2012; Boumal, 2023). On the manifold, the Jacobian of $\Pi$ is $P_T$ (Boumal, 2023, Sec. 5.12), so a single map $R \approx \Pi$ can supply both $\mathcal{C} = R$ and $F = J_R$. The flow's denoising map is such a map. By Tweedie's formula and its second-order form (Efron, 2011; Manor & Michaeli, 2024), it is a posterior mean whose Jacobian is the scaled posterior covariance (Eq. (5)); near a data manifold, these approach $\Pi$ and its Jacobian (Kharitenko et al., 2026), as stated in Eq. (6).

**Manifold-preserving guidance.** Training-free guidance of diffusion models also uses the manifold hypothesis: MPGD (He et al., 2024) applies the guidance gradient to the clean-sample estimate and keeps the guided sample on the data manifold, for example with a pretrained autoencoder. These methods steer the *generation* of a sample toward a known guidance objective. RetraQ instead refines a completed action with a learned critic, motivates filtering by the non-identifiability of the critic's normal gradients, obtains both operators from the frozen policy itself without an auxiliary model, and uses the refined actions as policy-improvement targets.

**Denoising Riemannian gradient descent.** The closest predecessor of RetraQ's update is *Landing with the Score* (Kharitenko et al., 2026), whose Denoising Riemannian Gradient Descent (DRGD)

minimizes a known objective $f$ over a data manifold by $\mathbf{x}_{k+1} = \mathbf{v}\big(\mathbf{x}_k - \gamma_k\, \mathbf{v}'(\mathbf{x}_k)\nabla f(\mathbf{x}_k)\big)$, with $\mathbf{v}$ a pretrained diffusion denoiser. RetraQ's update in Eq. (7) shares this filter–step–denoise structure, and we view DRGD as its closest algorithmic predecessor. The two methods, however, address different problems, and RetraQ changes how each ingredient is obtained and justified. *First*, DRGD minimizes a known objective $f$, whose full gradient is meaningful everywhere, and uses the denoiser for feasibility. RetraQ follows a *learned* critic whose values are constrained only on behavior data; Section 3 shows that its normal gradient is then not identified, which gives filtering a statistical rather than purely geometric role, and the local guarantee (Proposition 2) is stated for the true value $Q^\pi$ under on-manifold agreement only. *Second*, RetraQ obtains both operators from a flow-matching policy rather than a score or diffusion denoiser. Rescaling the interpolant as $\mathbf{x}_t/t = \mathbf{a}+\sigma_t\mathbf{z}$ turns an action into a Gaussian-corrupted observation at level $\sigma_{t_g}$, so one velocity evaluation of the context-conditioned flow gives $R$ and one JVP gives $J_R\mathbf{g}$ (Eq. (4)), with no separate score network, noise-schedule conversion, or per-context geometry model. *Third*, DRGD's guarantees assume a uniformly accurate denoiser and Jacobian. For a learned flow, Proposition 4 bounds how errors of the velocity field and of its input derivatives enter $R$ and $J_R$, and Section 5.2 measures how the guidance level and action dimension trade tangent accuracy against normal suppression. *Fourth*, RetraQ embeds the update in policy improvement: refined actions are stop-gradient targets for anchored distillation and offline-to-online learning with a frozen geometry reference, the step is calibrated to a target displacement, and fork critics supply action-gradient supervision on the robot.

# B Theoretical Results

This appendix collects the results behind Sections 3 and 4.1. Section B.1 shows what critic values on the behavior manifold determine; Section B.2 shows that the ideal geometric update improves the true action value; and Section B.3, the main result of this appendix, shows that the denoising map of a flow approximates the two ideal operators. Standard facts from Riemannian optimization are cited rather than re-derived (Boumal, 2023).

**Setting.** Fix a context $\mathbf{s}$ and an action $\mathbf{a}$ on a smooth embedded manifold $\mathcal{M} = \mathcal{M}_\mathbf{s} \subset \mathbb{R}^D$. We omit $\mathbf{s}$ from the critics. All statements are local to $\mathbf{a}$ and use the Euclidean metric in the normalized action coordinates of the flow. Let $P_T(\mathbf{a})$ and $P_N(\mathbf{a})$ be the tangent and normal orthogonal projectors. Choose a sufficiently small neighborhood $U$ of $\mathbf{a}$ where the nearest-point projection $\Pi = \Pi_\mathbf{s}$ is single-valued and smooth. Then $\mathrm{D}\Pi(\mathbf{a}) = P_T(\mathbf{a})$ (Boumal, 2023). Here D denotes the derivative of a map. Figure 2 illustrates the two geometric issues discussed below.

## B.1 What Critic Values on the Manifold Determine

**Proposition 1** (Tangential agreement and normal-gradient freedom)**.** *Let $Q_1$ be continuously differentiable on $U$.*

(i) *If a continuously differentiable critic $Q_2$ agrees with $Q_1$ on $\mathcal{M} \cap U$, then*

$$P_T(\mathbf{a})\nabla Q_1(\mathbf{a}) = P_T(\mathbf{a})\nabla Q_2(\mathbf{a}). \tag{10}$$

(ii) *For every $\mathbf{n} \in N_\mathbf{a}\mathcal{M}$, there exists a continuously differentiable $Q_2$ on $U$ that agrees with $Q_1$ on $\mathcal{M} \cap U$ and satisfies*

$$\nabla Q_2(\mathbf{a}) = \nabla Q_1(\mathbf{a}) + \mathbf{n}. \tag{11}$$

*Moreover, it can be chosen so that, for every $\mathbf{x} \in U$,*

$$|Q_2(\mathbf{x}) - Q_1(\mathbf{x})| \le \|\mathbf{n}\|_2 \,\mathrm{dist}(\mathbf{x}, \mathcal{M}). \tag{12}$$

*Proof.* For part (i), take any $\boldsymbol{\xi} \in T_\mathbf{a}\mathcal{M}$. There is a smooth curve $\mathbf{c}$ on $\mathcal{M}$ with $\mathbf{c}(0) = \mathbf{a}$ and $\mathbf{c}'(0) = \boldsymbol{\xi}$. For a sufficiently short segment, this curve lies in $U$, where $Q_2(\mathbf{c}(t)) - Q_1(\mathbf{c}(t)) = 0$. Differentiating at $t = 0$ gives

$$\boldsymbol{\xi}^\top \big(\nabla Q_2(\mathbf{a}) - \nabla Q_1(\mathbf{a})\big) = 0.$$

The gradient difference is orthogonal to every tangent vector, proving Eq. (10).

For part (ii), define

$$h_{\mathbf{n}}(\mathbf{x}) := \mathbf{n}^\top\big(\mathbf{x} - \Pi(\mathbf{x})\big), \qquad Q_2(\mathbf{x}) := Q_1(\mathbf{x}) + h_{\mathbf{n}}(\mathbf{x}).$$

On $\mathcal{M} \cap U$, $\Pi(\mathbf{x}) = \mathbf{x}$, so $h_{\mathbf{n}} = 0$. At $\mathbf{a}$,

$$\nabla h_{\mathbf{n}}(\mathbf{a}) = \big(I_D - \mathrm{D}\Pi(\mathbf{a})\big)^\top \mathbf{n} = \big(I_D - P_T(\mathbf{a})\big)\mathbf{n} = \mathbf{n}.$$

Finally, the Cauchy–Schwarz inequality and the nearest-point property give

$$|h_{\mathbf{n}}(\mathbf{x})| \le \|\mathbf{n}\|_2 \, \|\mathbf{x} - \Pi(\mathbf{x})\|_2 = \|\mathbf{n}\|_2 \operatorname{dist}(\mathbf{x}, \mathcal{M}).$$

These identities establish both the gradient change and the value bound. □

The proposition assumes exact agreement throughout a local manifold patch. It identifies the tangential derivative of that shared restriction, without asserting that the shared values equal the true action value. Agreement on finitely many samples, or a small value-fitting error, does not by itself guarantee accurate tangential derivatives.

**Data in a thin neighborhood.** Let $\mathcal{S}_\delta \subset U$ be a set of data actions satisfying $\operatorname{dist}(\mathbf{x}, \mathcal{M}) \le \delta$ for every $\mathbf{x} \in \mathcal{S}_\delta$, where $\delta > 0$. Let $\epsilon > 0$ be an allowed discrepancy between the values predicted by two critics on these actions. For any $\mathbf{n} \in N_{\mathbf{a}}\mathcal{M}$ with $\|\mathbf{n}\|_2 \le \epsilon/\delta$, the construction in Proposition 1(ii) gives

$$\sup_{\mathbf{x} \in \mathcal{S}_\delta} |Q_2(\mathbf{x}) - Q_1(\mathbf{x})| \le \delta \|\mathbf{n}\|_2 \le \epsilon, \qquad \nabla Q_2(\mathbf{a}) - \nabla Q_1(\mathbf{a}) = \mathbf{n}. \tag{13}$$

Thus, a normal-gradient change of size $\epsilon/\delta$ is compatible with a value discrepancy of at most $\epsilon$ on all these data. For a fixed value tolerance, this construction permits larger gradient changes as the data become more concentrated around the manifold.

Here $\epsilon$ measures the difference between two critics, rather than either critic's error against a target. If $Q_1$ fits target values $y(\mathbf{x})$ with uniform error at most $\epsilon_0$ on $\mathcal{S}_\delta$, the triangle inequality gives

$$\sup_{\mathbf{x} \in \mathcal{S}_\delta} |Q_2(\mathbf{x}) - y(\mathbf{x})| \le \epsilon_0 + \epsilon.$$

In particular, if $Q_1$ fits these targets exactly, $Q_2$ fits them to accuracy $\epsilon$ while allowing the stated normal-gradient change.

This is a constructive ambiguity result. It does not upper-bound all possible normal-gradient errors by $\epsilon/\delta$; such a bound would require additional assumptions controlling derivatives.

### B.2 Local Improvement by the Geometric Update

Two standard facts connect the update of Eq. (3) to Riemannian optimization under the induced Euclidean metric (Boumal, 2023). First, $\mathbf{g}_T = P_T(\mathbf{a})\nabla Q(\mathbf{a})$ is the Riemannian gradient of the restriction $Q|_{\mathcal{M}}$, so the normalized filtered gradient is the unit tangent direction of steepest initial increase of $Q$ along $\mathcal{M}$. Second, $\mathrm{Retr}_{\mathbf{a}}(\boldsymbol{\xi}) := \Pi(\mathbf{a} + \boldsymbol{\xi})$ is a local retraction: $\mathrm{Retr}_{\mathbf{a}}(\mathbf{0}) = \mathbf{a}$ and $\mathrm{D}\,\mathrm{Retr}_{\mathbf{a}}(\mathbf{0})[\boldsymbol{\xi}] = \boldsymbol{\xi}$ for tangent $\boldsymbol{\xi}$, because $\mathrm{D}\Pi(\mathbf{a}) = P_T(\mathbf{a})$. The following statement specifies when critic ascent along this retraction also improves the true action value.

**Proposition 2** (Local action-value improvement)**.** *Let $\pi$ be a fixed continuation policy, and let $Q^\pi$ denote its true action-value function at the fixed context. Suppose $Q$ and $Q^\pi$ are continuously differentiable near $\mathbf{a} \in \mathcal{M}$ and agree on a local manifold patch containing $\mathbf{a}$. For $\mathbf{g}_T = P_T(\mathbf{a})\nabla Q(\mathbf{a})$ and any local retraction at $\mathbf{a}$,*

$$\frac{\mathrm{d}}{\mathrm{d}\eta} Q^\pi(\mathrm{Retr}_{\mathbf{a}}(\eta \mathbf{g}_T))\bigg|_{\eta=0} = \|\mathbf{g}_T\|_2^2. \tag{14}$$

*If $\mathbf{g}_T \neq \mathbf{0}$, then $Q^\pi(\mathrm{Retr}_{\mathbf{a}}(\eta \mathbf{g}_T)) > Q^\pi(\mathbf{a})$ for all sufficiently small $\eta > 0$.*

*Proof.* Proposition 1(i) gives $P_T(\mathbf{a})\nabla Q^\pi(\mathbf{a}) = \mathbf{g}_T$. By the chain rule and the retraction property,

$$\frac{\mathrm{d}}{\mathrm{d}\eta} Q^\pi(\mathrm{Retr}_{\mathbf{a}}(\eta \mathbf{g}_T))\bigg|_{\eta=0} = \nabla Q^\pi(\mathbf{a})^\top \mathbf{g}_T = \|\mathbf{g}_T\|_2^2.$$

A positive derivative at zero implies the stated strict increase for all sufficiently small positive step sizes. □

Proposition 2 assumes exact critic values on a manifold patch and exact geometric operators, and concerns one context under a fixed continuation policy. A learned filter can leave a residual normal component, so the stepped action need not lie in the domain of a retraction; its endpoint can still be corrected by $\Pi$ as long as it stays in a tube where $\Pi$ is single-valued. Section B.3 shows how the learned denoising map approximates both operators on such a tube.

### B.3 Flow-Based Approximation of the Geometric Operators

This section states and proves the two results that Section 4.1 uses to connect the denoising map $R$ of Eq. (4) to the ideal operators $P_T$ and $\Pi_{\mathbf{s}}$ of Section 3. Proposition 3 concerns the map $R^{\mathrm{opt}}$ defined by the population-optimal velocity; Proposition 4 quantifies the additional error of a learned velocity. As above, the context is fixed and omitted, and $\mu = \mu(\cdot \mid \mathbf{s})$. Norms of matrices are operator norms.

**Assumptions.**

- (A1) $\mathcal{M} \subset \mathbb{R}^D$ is a compact $k$-dimensional embedded $C^4$ submanifold without boundary, with positive reach $\tau_{\mathcal{M}}$ (Federer, 1959).
- (A2) $\mu$ is supported on $\mathcal{M}$ and has a density $p \in C^2(\mathcal{M})$ with respect to the $k$-dimensional volume measure of $\mathcal{M}$, and $p > 0$ on $\mathcal{M}$.

Fix $0 < r < \tau_{\mathcal{M}}$ and the closed tube $U_r := \{\mathbf{a} : \mathrm{dist}(\mathbf{a}, \mathcal{M}) \le r\}$. On $U_r$, the nearest-point projection $\Pi$ is single-valued and $C^3$ (Federer, 1959), and its derivative at $\mathbf{a} = \Pi(\mathbf{a}) + d\,\boldsymbol{\nu}$, with $d = \mathrm{dist}(\mathbf{a}, \mathcal{M})$ and $\boldsymbol{\nu}$ a unit normal vector at $\Pi(\mathbf{a})$, is

$$\mathrm{D}\Pi(\mathbf{a}) = \big(I_T - d\,S_{\boldsymbol{\nu}}\big)^{-1} P_T\big(\Pi(\mathbf{a})\big), \tag{15}$$

where $I_T$ is the identity on the tangent space at $\Pi(\mathbf{a})$ and $S_{\boldsymbol{\nu}}$ is the shape operator in direction $\boldsymbol{\nu}$, the self-adjoint map on that space defined by $\langle S_{\boldsymbol{\nu}}\boldsymbol{\xi}, \boldsymbol{\zeta}\rangle = \langle \mathrm{II}(\boldsymbol{\xi}, \boldsymbol{\zeta}), \boldsymbol{\nu}\rangle$, with $\mathrm{II}$ the normal-valued second fundamental form; $I_T - dS_{\boldsymbol{\nu}}$ is invertible because $d < \tau_{\mathcal{M}}$. On $\mathcal{M}$, $d = 0$ and $\mathrm{D}\Pi = P_T$ (Boumal, 2023).

**The ideal map is a Gaussian posterior mean.** As in Section 4.1, write $\sigma = \sigma_g = (1 - t_g)/t_g$, and let $\mathbf{A} \sim \mu$ and $\mathbf{Z} \sim \mathcal{N}(0, I_D)$ be independent. Then $R^{\mathrm{opt}}(\mathbf{a}) = \mathbb{E}[\mathbf{A} \mid \mathbf{A} + \sigma\mathbf{Z} = \mathbf{a}]$, and differentiating this Gaussian posterior mean in $\mathbf{a}$ gives $J_R^{\mathrm{opt}}(\mathbf{a}) = \sigma^{-2}\,\mathrm{Cov}[\mathbf{A} \mid \mathbf{A} + \sigma\mathbf{Z} = \mathbf{a}]$ (second-order Tweedie identity (Efron, 2011; Manor & Michaeli, 2024)). The posterior of $\mathbf{A}$ given the observation $\mathbf{a}$ is the probability measure on $\mathcal{M}$

$$\rho_{\mathbf{a}}(\mathrm{d}\mathbf{m}) \;\propto\; p(\mathbf{m}) \exp\!\Big(-\frac{\|\mathbf{a} - \mathbf{m}\|_2^2}{2\sigma^2}\Big)\, \mathrm{vol}_{\mathcal{M}}(\mathrm{d}\mathbf{m}), \tag{16}$$

so $R^{\mathrm{opt}}(\mathbf{a})$ and $\sigma^2 J_R^{\mathrm{opt}}(\mathbf{a})$ are the mean and covariance of $\rho_{\mathbf{a}}$.

**Proposition 3** (Local geometry of the ideal denoising map)**.** *Under* (A1)–(A2)*, there exist $\sigma_0 > 0$ and $C < \infty$, depending on $\mathcal{M}$, $p$, and $r$, such that for all $\sigma \in (0, \sigma_0]$ and all $\mathbf{a} \in U_r$,*

$$\big\|R^{\mathrm{opt}}(\mathbf{a}) - \Pi(\mathbf{a})\big\|_2 \le C\sigma^2, \qquad \big\|J_R^{\mathrm{opt}}(\mathbf{a}) - \mathrm{D}\Pi(\mathbf{a})\big\| \le C\sigma^2. \tag{17}$$

*In particular, $J_R^{\mathrm{opt}}(\mathbf{a}) = P_T(\mathbf{a}) + O(\sigma^2)$ for $\mathbf{a} \in \mathcal{M}$, which is Eq.* (6)*.*

*Proof.* Fix $\mathbf{a} \in U_r$, let $\mathbf{m}_0 = \Pi(\mathbf{a})$, and write $\mathbf{a} = \mathbf{m}_0 + d\boldsymbol{\nu}$ as above.

*Step 1: localization.* Let $\phi(\mathbf{m}) := \frac{1}{2}\|\mathbf{a} - \mathbf{m}\|_2^2 - \frac{1}{2}d^2 \ge 0$ on $\mathcal{M}$. Because $\mathbf{m}_0$ is the unique minimizer of $\phi$ and $U_r$ and $\mathcal{M}$ are compact, for every $\varepsilon > 0$ there is $\delta(\varepsilon) > 0$, independent of $\mathbf{a} \in U_r$, such that $\phi(\mathbf{m}) \ge \delta(\varepsilon)$ whenever $\|\mathbf{m} - \mathbf{m}_0\|_2 \ge \varepsilon$. The mass that $\rho_{\mathbf{a}}$ assigns to this region is therefore at most $c\,\sigma^{-k} e^{-\delta(\varepsilon)/\sigma^2}$, since the normalizing constant of Eq. (16) is at least of order $\sigma^k e^{-d^2/(2\sigma^2)}$. The same bound, multiplied by $\mathrm{diam}(\mathcal{M})^2$, controls the contribution of this region to the mean and covariance. These terms are $o(\sigma^m)$ for every $m$ and can be neglected.

*Step 2: chart.* By compactness of $\mathcal{M}$, $\varepsilon$ can be chosen, independently of $\mathbf{m}_0$, so that $\mathcal{M} \cap \{\|\mathbf{m} - \mathbf{m}_0\|_2 < \varepsilon\}$ is a graph over the tangent space at $\mathbf{m}_0$. We parametrize $\mathcal{M}$ near $\mathbf{m}_0$ as $\mathbf{m}(\mathbf{u}) =$

$\mathbf{m}_0 + E\mathbf{u} + \frac{1}{2}\mathrm{II}(\mathbf{u},\mathbf{u}) + O(\|\mathbf{u}\|^3)$, where $E \in \mathbb{R}^{D\times k}$ is an orthonormal basis of the tangent space at $\mathbf{m}_0$, and $\mathbf{u} \in \mathbb{R}^k$. Then

$$\phi(\mathbf{m}(\mathbf{u})) = \tfrac{1}{2}\mathbf{u}^\top H\mathbf{u} + \phi_3(\mathbf{u}) + O(\|\mathbf{u}\|^4), \qquad H = I_k - d\,E^\top S_{\boldsymbol{\nu}} E,$$

with $\phi_3$ a cubic form; there is no linear term because $\mathbf{a} - \mathbf{m}_0$ is normal. $H$ is positive definite, uniformly over $U_r$, because $d \le r < \tau_{\mathcal{M}}$. The density of $\mathbf{u}$ is proportional to $q(\mathbf{u})\exp(-\phi(\mathbf{m}(\mathbf{u}))/\sigma^2)$, where $q(\mathbf{u}) = p(\mathbf{m}(\mathbf{u}))\sqrt{\det g(\mathbf{u})}$ is $C^2$ and positive, with $g$ the metric in the chart.

*Step 3: Laplace expansion.* Substitute $\mathbf{u} = \sigma\mathbf{w}$. The density of $\mathbf{w}$ is proportional to $\exp(-\frac{1}{2}\mathbf{w}^\top H\mathbf{w})\big(1 + \sigma\,\ell(\mathbf{w}) + O(\sigma^2(1+\|\mathbf{w}\|^6))\big)$, where $\ell(\mathbf{w}) = \nabla \log q(\mathbf{0})^\top \mathbf{w} - \phi_3(\mathbf{w})$ is an odd polynomial. Every term of order $\sigma$ in the moments of $\mathbf{w}$ is a Gaussian moment of odd total degree and vanishes. Hence $\mathbb{E}[\mathbf{w}] = O(\sigma)$, $\mathbb{E}[\mathbf{w}\mathbf{w}^\top] = H^{-1} + O(\sigma^2)$, and $\mathbb{E}[\mathbf{w}\otimes\mathbf{w}\otimes\mathbf{w}] = O(\sigma)$, uniformly over $U_r$ because the constants depend continuously on $\mathbf{a}$ through $H$, $\phi_3$, and $q$ on a compact set.

*Step 4: mean and covariance.* Returning to $\mathbf{u} = \sigma\mathbf{w}$, $\mathbb{E}[\mathbf{u}] = O(\sigma^2)$ and $\mathbb{E}[\|\mathbf{u}\|^2] = O(\sigma^2)$, so

$$R^{\mathrm{opt}}(\mathbf{a}) = \mathbf{m}_0 + E\,\mathbb{E}[\mathbf{u}] + \tfrac{1}{2}\mathbb{E}[\mathrm{II}(\mathbf{u},\mathbf{u})] + O(\sigma^3) = \Pi(\mathbf{a}) + O(\sigma^2).$$

For the covariance, $\mathrm{Cov}[\mathbf{m}] = E\,\mathrm{Cov}[\mathbf{u}]E^\top + O(\sigma^4)$: the cross terms between $E\mathbf{u}$ and $\frac{1}{2}\mathrm{II}(\mathbf{u},\mathbf{u})$ involve third moments of $\mathbf{u}$, which are $\sigma^3 \cdot O(\sigma)$, and the remaining terms are of fourth order in $\mathbf{u}$. Since $\mathrm{Cov}[\mathbf{u}] = \sigma^2\big(H^{-1} + O(\sigma^2)\big)$,

$$J_R^{\mathrm{opt}}(\mathbf{a}) = \sigma^{-2}\,\mathrm{Cov}[\mathbf{m}] = EH^{-1}E^\top + O(\sigma^2).$$

Finally, $EH^{-1}E^\top = (I_T - dS_{\boldsymbol{\nu}})^{-1}P_T(\mathbf{m}_0) = \mathrm{D}\Pi(\mathbf{a})$ by Eq. (15). Both bounds are uniform over $U_r$ by Steps 1 and 3. □

**What the proposition does and does not provide.** Proposition 3 is an asymptotic statement as $\sigma \to 0$ for a behavior distribution supported exactly on a manifold with positive reach. Its constant $C$ grows with the curvature of $\mathcal{M}$, the variation of $\log p$, and the proximity of $r$ to the reach; it gives no numerical error at the operating point $\sigma_g \approx 0.11$. The result does not apply when two parts of the support lie within a distance comparable to $\sigma$, for example nearby action modes, because the reach is then small. The manifold is also assumed to have no boundary; near the boundary of a manifold patch, the posterior is truncated on one side, and its mean is shifted inward by $O(\sigma)$ rather than $O(\sigma^2)$. Off the manifold, the Jacobian approximates $\mathrm{D}\Pi$, which rescales tangent directions by $(I_T - dS_{\boldsymbol{\nu}})^{-1}$ and annihilates normal ones; it equals $P_T$ only on $\mathcal{M}$.

**Proposition 4** (Error of the learned denoising map)**.** *Let $v_{\bar\theta}$ be any velocity field that is differentiable in $\mathbf{x}$, let $v^\star$ be the population-optimal velocity of Section 2, and write $\mathbf{e}(\mathbf{x}) := v_{\bar\theta}(\mathbf{s},\mathbf{x},t_g) - v^\star(\mathbf{s},\mathbf{x},t_g)$. For every action $\mathbf{a}$,*

$$R(\mathbf{a}) - R^{\mathrm{opt}}(\mathbf{a}) = (1-t_g)\,\mathbf{e}(t_g\mathbf{a}), \tag{18}$$

$$J_R(\mathbf{a}) - J_R^{\mathrm{opt}}(\mathbf{a}) = t_g(1-t_g)\,\partial_{\mathbf{x}}\mathbf{e}(t_g\mathbf{a}). \tag{19}$$

*Consequently, under the assumptions of Proposition 3, for $\mathbf{a} \in U_r$ and $\sigma_g \le \sigma_0$,*

$$\|R(\mathbf{a}) - \Pi(\mathbf{a})\|_2 \le C\sigma_g^2 + (1-t_g)\|\mathbf{e}(t_g\mathbf{a})\|_2, \qquad \|J_R(\mathbf{a}) - \mathrm{D}\Pi(\mathbf{a})\| \le C\sigma_g^2 + t_g(1-t_g)\|\partial_{\mathbf{x}}\mathbf{e}(t_g\mathbf{a})\|. \tag{20}$$

*Moreover, if $\|\partial_{\mathbf{x}} v_{\bar\theta}(\mathbf{s},\mathbf{x},t_g)\| \le L$ for all $\mathbf{x}$, then $\|J_R(\mathbf{a}) - t_g I_D\| \le t_g(1-t_g)L$ for every $\mathbf{a}$.*

*Proof.* By Eq. (4), $R(\mathbf{a}) = t_g\mathbf{a} + (1-t_g)v_{\bar\theta}(\mathbf{s}, t_g\mathbf{a}, t_g)$, and $R^{\mathrm{opt}}$ is the same expression with $v^\star$. Subtracting gives Eq. (18). Differentiating in $\mathbf{a}$, with the factor $t_g$ from the input scaling (Eq. (21)), gives Eq. (19). The bounds of Eq. (20) follow from the triangle inequality and Proposition 3. For the last statement, $J_R(\mathbf{a}) - t_g I_D = t_g(1-t_g)\,\partial_{\mathbf{x}} v_{\bar\theta}(\mathbf{s}, t_g\mathbf{a}, t_g)$. □

**Consequences for the guidance level.** Eq. (19) shows that the filter inherits the error in the *input derivative* of the velocity, not only the error in its value. Flow-matching regression controls the value error in mean square under the noisy marginal, but not its derivative, so a small regression loss does not by itself make $J_R$ an accurate tangent filter. The normal gain measured in Section 5.2 is a direct

diagnostic of the combined error of Eq. (20) on normal inputs. The last statement of Proposition 4 formalizes the limit $t_g \to 1$ discussed in Section 4.1: for any velocity with input derivatives bounded by $L$, $J_R \to I_D$, the unfiltered direction of raw critic ascent, even though $J_R^{\mathrm{opt}} \to P_T$ on $\mathcal{M}$ by Proposition 3. The two propositions thus bound the two sides of the trade-off in the choice of $t_g$ (Appendix C.1): the $C\sigma_g^2$ term favors large $t_g$, whereas a bounded learned derivative forces $J_R$ toward $t_g I_D$ as $t_g \to 1$.

# C Implementation Details of RetraQ

This appendix gives the details that Section 4 omits: how the operator is computed and calibrated (C.1), the critics (C.2), how the three settings use refined actions, including the full offline-to-online algorithm (C.3), and the hyperparameters (Table 6).

**Conventions.** Flow time follows Section 2, with $t = 1$ at the data. The $\pi_{0.5}$ code base uses the reversed time $t' = 1 - t$, so its configuration value $t'_g = 0.1$ is the same operating point as $t_g = 0.9$ here ($\sigma_g \approx 0.11$). *Real robot* refers to phone-charging plug insertion on an AgiBot G02 dual-arm robot with three cameras (head and both wrists; Section 5.4). Displacements are Euclidean norms of whole action chunks in the flow's normalized action coordinates.

## C.1 The Operator in Practice

**Cost.** We never form the $D \times D$ matrix $J_R$. One Jacobian–vector product through $R$ at $\mathbf{a}$ with tangent $\mathbf{g}$, computed by forward-mode automatic differentiation, returns both $R(\mathbf{s}, \mathbf{a})$ and $\widetilde{\mathbf{g}} = J_R\mathbf{g}$ at a small constant multiple of the cost of one forward pass. An update therefore costs one critic gradient, one JVP, and one further forward pass of $R$ at $\mathbf{a} + \eta\widetilde{\mathbf{g}}$. We use $M = 1$ refinement step on all platforms.

**Guidance level.** For $0 < t < 1$ write $R_t(\mathbf{s}, \mathbf{a}) := D_t^{\bar{\theta}}(\mathbf{s}, t\mathbf{a}) = t\mathbf{a} + (1-t)v_{\bar{\theta}}(\mathbf{s}, t\mathbf{a}, t)$, so that $R = R_{t_g}$ in Eq. (4). Its action-input Jacobian is

$$J_t(\mathbf{s}, \mathbf{a}) := \partial_{\mathbf{a}} R_t(\mathbf{s}, \mathbf{a}) = tI_D + t(1-t)\,\partial_{\mathbf{x}} v_{\bar{\theta}}(\mathbf{s}, t\mathbf{a}, t). \tag{21}$$

The factor $t$ in the second term comes from the input scaling; the JVP includes it. The level $t_g$ trades two errors (Propositions 3 and 4): a large $t_g$ makes the $O(\sigma_g^2)$ terms small but drives a learned $J_R$ toward $t_g I_D$, and a small $t_g$ lets the posterior average across curvature and nearby action modes, which blurs both $R$ and $J_R$. We use $t_g = 0.9$ on all platforms. This point was fixed from operator diagnostics on OGBench and inherited by the other platforms; it was not tuned on task success, and we ran no $t_g$ sweep on the robot.

**Step-size calibration.** The scale of $\widetilde{\mathbf{g}}$ varies with the critic and the local geometry, so we fix the displacement rather than $\eta$. On a calibration batch $\{(\mathbf{s}_j, \mathbf{a}_j^{(0)})\}_{j=1}^B$, with first-step filtered gradients $\widetilde{\mathbf{g}}_j = J_R(\mathbf{s}_j, \mathbf{a}_j^{(0)})\nabla_{\mathbf{a}}\hat{Q}(\mathbf{s}_j, \mathbf{a}_j^{(0)})$, we solve

$$\operatorname*{stat}_{j=1,\dots,B} \left\| R(\mathbf{s}_j, \mathbf{a}_j^{(0)} + \eta\widetilde{\mathbf{g}}_j) - R(\mathbf{s}_j, \mathbf{a}_j^{(0)}) \right\|_2 = d_{\mathrm{target}} \tag{22}$$

for $\eta$ by a log-scale grid search followed by bisection. The statistic is the batch mean in offline-to-online learning and the median in offline distillation and robot test-time guidance. $\eta$ is calibrated once and then held fixed: on the first actor batch in offline-to-online learning, on a held-out split of 2048 states in offline distillation, and on a bank of fork-state samples for robot test-time guidance ($\eta = 0.01$). A calibration that does not converge is flagged, never clipped, and excluded from displacement-matched comparisons.

Subtracting $R(\mathbf{s}_j, \mathbf{a}_j^{(0)})$ removes the map's own offset $R(\mathbf{s}, \mathbf{a}) - \mathbf{a}$, which is present even when $\widetilde{\mathbf{g}} = \mathbf{0}$, so $d_{\mathrm{target}}$ measures only critic-induced movement. The offset is not small: on 135 robot fork states, $\|R(\mathbf{a}) - \mathbf{a}\|_2$ has median 1.62, about 65 times the deployed $d_{\mathrm{target}} = 0.025$. Multiplying $\hat{Q}$ by $\beta > 0$ multiplies every $\widetilde{\mathbf{g}}_j$ by $\beta$ and the calibrated $\eta$ by $1/\beta$, so the calibrated step $\eta\widetilde{\mathbf{g}}$ does not depend on the critic's scale. The controlled toy problem instead calibrates the total displacement $\|R(\mathbf{s}, \mathbf{a} + \eta\widetilde{\mathbf{g}}) - \mathbf{a}\|_2$.

Table 3: **Critics used by RetraQ.** TTG: test-time guidance; MC: Monte Carlo; TD: temporal difference. FQL-style twin-$Q$ TD evaluates the bootstrap target at an action of the current actor.

| Platform and setting | Critic | Training data and targets |
|---|---|---|
| Real robot, TTG | fork critic: Critic 1 (linear), Critic 2 (nonlinear) | 270 forked rollouts; binary success |
| Real robot, offline distill. | fork critic (Critic 1) | same |
| Real robot, offline-to-online | IQL, twin $Q$ and $V$, Eq. (27) | online replay; terminal success |
| LIBERO, offline | chunk critic, MC | forked BC rollouts; discounted success |
| LIBERO, online | FQL-style twin-$Q$ TD, warm-started from MC | mixed replay; discounted success |
| RoboMimic Can, offline | chunk critic, MC | forked BC rollouts; discounted success |
| OGBench, offline and online | FQL-style twin-$Q$ TD | offline data, then mixed replay; benchmark reward |
| Toy | analytic $\hat{Q}$ (no learned critic) | — |

**Padded actions.** For padded action representations (on the robot, 16 executable coordinates padded to 32, and 30 executed frames padded to the 50-frame horizon), the fixed orthogonal projector $P_{\mathrm{act}}$ retains executable coordinates and zeroes padded ones. The effective critic is $\widehat{Q}(\mathbf{s}, \mathbf{a}) := \hat{Q}(\mathbf{s}, P_{\mathrm{act}}\mathbf{a})$, and step ❶ of Eq. (7) uses its gradient. All other operations and norms use model action coordinates.

**Guards.** *Numerical* checks stop the run if a non-finite value appears in the critic value, its gradient, the JVP output, the proposal, the refined action, or $R(\mathbf{a})$, or if $\eta$ leaves $[10^{-6}, 10^{2}]$; values are never patched, since a patched target would train the actor toward an arbitrary action without a trace. A *displacement* check discards an update if the critic-induced displacement exceeds a cap ($0.5$ by default, $0.25$ on the robot) rather than rescaling it, because rescaling would still train on a displacement at which neither the retraction nor the critic's local linearization is trusted. The cap matters because matching a batch statistic does not bound each action: on 240 robot fork frames, $\|\nabla_{\mathbf{a}} Q\|$ has median $0.0155$ and maximum $2.03$, and $2.5\%$ of the calibrated steps exceed the robot cap.

### C.2 Critics

RetraQ uses only the action gradient of a critic. Table 3 lists the critic used in each setting.

#### C.2.1 Fork Critic for Test-Time Guidance

**Collection.** A fork group is collected as in Figure 6a. Seeded replay reproduces the same prefix and reaches the fork state $\mathbf{s}$. In each branch, the first action chunk after $\mathbf{s}$ is replaced, the frozen policy $\pi_{\bar{\theta}}$ finishes the episode, and the branch is labeled by its task outcome $y$. The reference chunk $\mathbf{a}_{\mathrm{c}}^{\mathrm{ref}}$ is the chunk that the frozen policy predicts at the fork state. Because branches share the state and differ only in the replaced chunk, outcome differences within a group can be attributed to the action.

**Robot fork states and branches.** The robot fork states are taken at two task-critical moments: the right arm has grasped the plug and is about to move it down toward the socket, and the plug is very close to the socket. Each collection run resets the robot to a fixed joint configuration at one of these moments. Runs are assigned to fork states by clustering their frame-0 joint proprioception; five collection runs yield four fork states. Scene variables that proprioception does not capture, such as the pose of the phone and cable, are not matched, so branches are not exact counterfactuals. A robot fork group contains three kinds of branches (Table 4). *Injected-noise* branches add a zero-mean Gaussian offset, constant over the chunk, to the coordinates $\mathcal{J} = \{7, 8, 9\}$, joints 1–3 of the right arm that performs the insertion, with standard deviation $\sigma_{\mathrm{b}} = 0.3$ times the per-coordinate action standard deviation. *Policy-resample* branches execute a new sample of the frozen policy, and *human-teleoperated* branches an operator-provided first chunk. The first chunk has $H_{\mathrm{exec}} = 30$

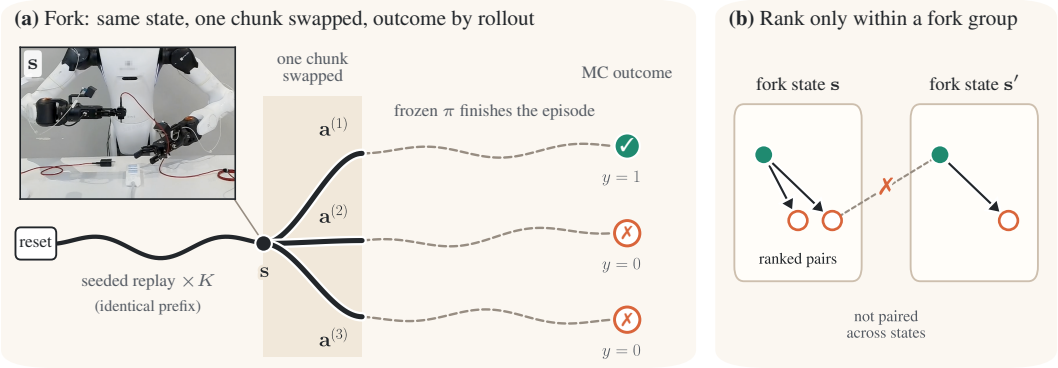



Figure 6: **Forked rollouts provide local action comparisons.** **(a)** Seeded replay from reset reproduces the same prefix $K$ times and reaches the fork state $\mathbf{s}$. In each branch, one action chunk is swapped for a perturbed chunk, the frozen policy $\pi_{\bar{\theta}}$ finishes the episode, and the branch is labeled by its Monte Carlo task outcome $y$. **(b)** Ranked pairs (arrows) are formed only within a fork group and place a successful branch (filled) above a failed one (hollow). Branches from different fork states $\mathbf{s}$ and $\mathbf{s}'$ are never paired, because their outcome difference would mix the effect of the state with that of the action.

Table 4: **Robot fork corpus by branch type.** Deviation is the Euclidean norm of the first-chunk deviation from the reference chunk on the perturbed joints $\mathcal{J}$, in radians. The tier weight enters the critic loss, Eq. (26).

| Branch type | $n$ | Success | Deviation, median | Deviation, max | Tier weight |
|---|---|---|---|---|---|
| Injected noise | 135 | 0.348 | 0.107 | 0.320 | 1.0 |
| Human teleoperation | 32 | 0.469 | 0.035 | 0.084 | 0.5 |
| Policy resample | 103 | 0.291 | 0.024 | 0.094 | 0.25 |

frames (1 s at 30 Hz). Of 300 collected rollouts, 270 are usable ($n = 103, 105, 30, 32$ over the four fork states; overall success $34.1\%$): we exclude rollouts with an operator intervention at any frame, whose later actions are not generated by the policy, and two whose first chunk is incomplete (24 and 27 frames).

**Simulation forks.** In simulation, forks are taken along replayed BC trajectories: every 12 environment steps on LIBERO, and at steps 48, 96, and 144 on RoboMimic Can. LIBERO branches perturb the chunk with independent Gaussian noise, and RoboMimic branches with temporally correlated AR(1) noise (coefficient 0.7, rescaled to unit standard deviation). Simulation labels are discounted Monte Carlo success $\gamma^T$, with $T$ the number of steps to success; robot labels are binary terminal success judged by the operator.

**Action input and heads.** Let $\mathbf{\Delta} = \mathbf{a} - \mathbf{a}_c^{\text{ref}}$ be the deviation in Eq. (8), $\mathbf{\Delta}_h$ its $h$-th control step, and $P_{\mathcal{J}}$ the selector of the perturbed coordinates. The action term reads the deviation through its mean over the executed chunk,

$$\mathbf{u} = \bar{P}\mathbf{\Delta} := \frac{1}{H_{\text{exec}}} \sum_{h=1}^{H_{\text{exec}}} P_{\mathcal{J}} \mathbf{\Delta}_h \in \mathbb{R}^3, \qquad A_\omega(\mathbf{\Delta}) = f_\omega(\bar{P}\mathbf{\Delta}), \quad f_\omega(\mathbf{0}) = 0, \tag{23}$$

which matches how the injected perturbation is applied. Because $\bar{P}$ is fixed and linear, $\nabla_{\mathbf{a}} Q_\phi = \kappa\, \bar{P}^\top \nabla f_\omega(\mathbf{u})$ is zero on unperturbed and padded coordinates and beyond the executed chunk. The baseline is linear in the 16-dimensional joint proprioception, $B_\psi(\mathbf{s}) = \boldsymbol{\psi}^\top [\mathbf{p}(\mathbf{s}); 1]$, and carries almost no outcome information: alone, its held-out AUROC on the four fork states is $0.511$, $0.527$, $0.578$, and $0.410$, whereas the action term reaches $0.727$, $0.685$, $0.814$, and $0.508$. The robot uses two action heads. *Critic 1* is linear, $f_\omega(\mathbf{u}) = \mathbf{w}^\top \mathbf{u}$:

$$Q_\phi(\mathbf{s}, \mathbf{a}) = \text{sigmoid}\big(B_\psi(\mathbf{s}) + \mathbf{w}^\top \bar{P}(\mathbf{a} - \mathbf{a}^{\text{ref}}_{c(\mathbf{s})})\big), \qquad \phi = (\psi, \mathbf{w}). \tag{24}$$

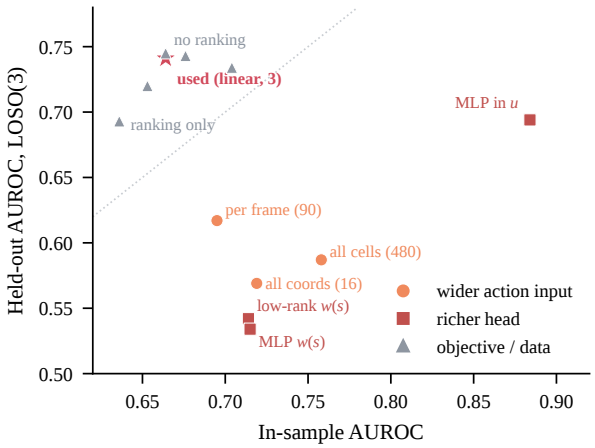



Figure 7: **Fit versus transfer of fork-critic variants.** In-sample AUROC versus leave-one-state-out AUROC over fork states 0–2. Wider action inputs and richer heads move right (better fit) and down (worse transfer); objective and data choices leave the used linear head's transfer nearly unchanged. "MLP in $\mathbf{u}$" is Critic 2. Dotted line: $y = x$.

*Critic 2* is nonlinear in the deviation, $f_\omega(\mathbf{u}) = \mathrm{MLP}_\omega(\mathbf{u}) - \mathrm{MLP}_\omega(\mathbf{0})$, and is trained on the same data. Fork states whose action standard deviation on $\mathcal{J}$ is below $0.25\sigma_{\mathrm{b}} = 0.075$ do not train the action head; this excludes state 3 (0.052).

**Loss.** The fork critic is trained with

$$\mathcal{L}(\phi) = \mathcal{L}_{\mathrm{out}}(\phi) + \lambda_{\mathrm{rank}}\mathcal{L}_{\mathrm{rank}}(\omega), \qquad \lambda_{\mathrm{rank}} \geq 0. \tag{25}$$

The outcome loss fits the predictions to the labels. The ranking loss compares branches of the same fork group and asks the one with the better outcome to receive the higher action score; since both branches share $\mathbf{s}$, the baseline cancels and $\mathcal{L}_{\mathrm{rank}}$ depends on $\omega$ alone. Pairs across fork groups are excluded (Figure 6b), because their outcome difference would also reflect the state. With logits $\nu_i = B_\psi(\mathbf{s}_i) + f_\omega(\mathbf{u}_i)$, tier weights $\alpha_i$ (Table 4), and ranked pairs $\mathcal{P} = \{(i,j) : c_i = c_j,\ y_i > y_j + \epsilon\}$ with $\epsilon = 0.02$, which drops pairs with nearly equal labels,

$$\mathcal{L}(\phi) = \lambda_{\mathrm{out}} \frac{\sum_i \alpha_i \,\mathrm{BCE}(\nu_i, y_i)}{\sum_i \alpha_i} + \lambda_{\mathrm{rank}} \frac{\sum_{(i,j)\in\mathcal{P}} \alpha_{ij}\ \log\bigl(1 + e^{-(\nu_i - \nu_j)}\bigr)}{\sum_{(i,j)\in\mathcal{P}} \alpha_{ij}} + \lambda_{\mathbf{w}}\, \mathbb{E}_i \bigl\| \nabla_{\mathbf{u}} f_\omega(\mathbf{u}_i) \bigr\|_2^2, \tag{26}$$

where $\alpha_{ij} = \min(\alpha_i, \alpha_j)$, $\lambda_{\mathrm{out}} = 0.1$, $\lambda_{\mathrm{rank}} = 1$, and $\lambda_{\mathbf{w}} = 10^{-2}$. For the linear head, the last term is $\|\mathbf{w}\|_2^2$. We train with learning rate $3 \times 10^{-3}$ for 2000 steps.

**What the ablation shows.** With four fork states and 270 rollouts, only a low-dimensional action direction can be resolved (Figure 7). Every richer action input or head raises in-sample AUROC but lowers the AUROC on held-out fork states: the linear head on the three-dimensional chunk mean reaches 0.741 in leave-one-state-out evaluation over states 0–2, whereas per-frame, all-coordinate, and state-dependent variants fall to 0.534–0.617, and the nonlinear head of Critic 2 reaches 0.694 despite the highest in-sample AUROC (0.884). This is why Critic 1 is the default. Critic 2 failed a gate fixed in advance (held-out AUROC above 0.667 when state 3 is also held out; it reaches 0.649), but it is evaluated as a second test-time critic (Section 5.4) because, unlike Critic 1, its gradient varies with the action. Objective and data choices barely change transfer; in particular, removing the ranking term leaves the held-out AUROC at 0.745, so the fork critic's transfer does not depend on it. Across three training seeds, the cosine similarity of the linear head's $\nabla_{\mathbf{a}} Q$ exceeds 0.9999.

**What a linear head implies for RetraQ.** With Eq. (24), the action gradient is $\kappa\, \bar{P}^\top \mathbf{w}$. Its direction is the same at every action of a fork group, and $Q_\phi$ increases monotonically along it with no interior maximum. The critic therefore says which way to move but not how far: the step length comes from the displacement calibration of Eq. (22), not from a critic maximum. The data support this direction only within the perturbation range, while a linear head extrapolates beyond it.

Table 5: **The three RetraQ settings** share the operator REFINE (Algorithm 1) and differ in its inputs and in how its output is used. Critics per platform are listed in Table 3.

| Setting | Initial action $\mathbf{a}^{(0)}$ | Critic | Use of $\mathbf{a}^\star$ |
|---|---|---|---|
| Test-time guidance | $\pi_{\bar\theta}(\cdot \mid \mathbf{s})$ at fork states | fork critic, Eq. (8) | executed |
| Offline distillation | current actor at dataset states | fixed during refinement | distilled |
| Offline-to-online | executed chunks in replay $\mathcal{D}$ | replay critic, snapshot $\hat{Q}$ | distilled |

The deployed step stays well inside that range: a median step changes each coordinate by at most $d_{\text{target}} = 0.025$, whereas the injected perturbations have standard deviation $\sigma_{\text{b}} = 0.3$ on each perturbed coordinate.

### C.2.2 REPLAY CRITICS FOR OFFLINE-TO-ONLINE LEARNING

Online rollouts are not forked, so no two samples share a state and the ranking term is unavailable. A replay transition stores a context $\mathbf{s}$, the executed part of a chunk ($k \le H$ frames, padded to the horizon $H$ with a frame mask $\mathbf{m}$; Appendix C.3), the per-frame rewards $r_0, \dots, r_{k-1}$, the next context $\mathbf{s}'$, and a terminal flag $d$. On the robot, we train twin critics $Q_{\phi_1}, Q_{\phi_2}$ and a state value $V_\psi$ with implicit Q-learning (Kostrikov et al., 2022), with target critics $Q_{\bar\phi_i}$:

$$\begin{aligned} \mathcal{L}_V(\psi) &= \mathbb{E}\big[L_2^\tau\big(\bar q - V_\psi(\mathbf{s})\big)\big], & \bar q &= \mathrm{sg}\big[\min_i Q_{\bar\phi_i}(\mathbf{s},\mathbf{a},\mathbf{m})\big], \\ \mathcal{L}_Q(\phi_1,\phi_2) &= \mathbb{E}\Big[\textstyle\sum_{i=1}^{2}\big(Q_{\phi_i}(\mathbf{s},\mathbf{a},\mathbf{m}) - y\big)^2\Big], & y &= \mathrm{sg}\Big[\textstyle\sum_{h=0}^{k-1}\gamma^h r_h + \gamma^k(1-d)\,V_\psi(\mathbf{s}')\Big], \end{aligned} \tag{27}$$

where $L_2^\tau(x) = |\tau - \mathbb{1}(x<0)|\,x^2$ is the expectile loss. We use $\tau = 0.7$, $\gamma = 0.99$ per executed frame, and Polyak averaging $\bar\phi_i \leftarrow (1-\rho)\bar\phi_i + \rho\phi_i$ with $\rho = 0.005$ after every update. The reward is sparse: $r = 1$ at the terminal frame of a success judged by the operator and 0 otherwise. The networks read frozen vision–language features of the BC model (masked mean pooling, concatenated with the normalized robot state; 2080 dimensions). $Q$ additionally takes the $30 \times 16$ executed actions and the 30-dimensional frame mask (2590 inputs in total); $Q$ and $V$ are MLPs with two hidden layers of width 256, LayerNorm, and tanh, with no output activation, and share no trainable parameters. Each is trained with Adam, learning rate $3 \times 10^{-4}$, and gradient-norm clipping at 1.0, in FP32. RetraQ reads the frozen snapshot $\hat{Q} = \min_i Q_{\phi_i}$. IQL evaluates the critics only at executed actions, so their values are supported by data while their normal gradients are unconstrained, which is the case step ❷ handles.

In simulation, LIBERO and OGBench use FQL-style twin-$Q$ TD critics, whose bootstrap target evaluates the next action of the current actor. The LIBERO online critic is warm-started from the offline Monte Carlo chunk critic (8000 training steps); the OGBench critic is first trained for $10^6$ offline steps, frozen during offline RetraQ, and then trained online. Both mix offline and online data equally in critic batches.

## C.3 THE THREE SETTINGS

Table 5 summarizes how the settings of Section 4.3 use the shared operator REFINE.

**Test-time guidance on the robot.** With all parameters fixed, RetraQ refines a sample of the BC policy before execution, only at detected fork states. The detector computes the Euclidean distance from the 16-dimensional joint proprioception to the four stored fork-state centroids and selects the nearest one if the distance is below 0.15 (within-state spread at most 0.007 rad; centroids at least 0.38 apart). Guidance is restricted to states 0–2, since state 3 lies below the action-standard-deviation floor and its held-out AUROC is 0.446. Otherwise, and if the critic logit saturates ($|\nu| > 15$) or a guard fails, the BC chunk is executed unchanged. Critic 1 and Critic 2 use the same detector, operator, and step size.

**Offline distillation.** Offline distillation runs outer iterations on a fixed dataset without collection; the critic is not updated while targets are refined. In each iteration, the current actor samples an action at each dataset state (detached), REFINE improves it, and the actor is distilled toward the

Table 6: **RetraQ hyperparameters.** Values shared by all methods on a platform (policy backbone, evaluation protocol) are in Appendix D; critics are listed in Table 3.

| | Toy | LIBERO | RoboMimic | OGBench | Real robot |
|---|---|---|---|---|---|
| Guidance level $t_g$ ($\sigma_g$) | | | 0.9 (0.11) | | 0.9 (0.11)$^{\ddagger}$ |
| Target displacement $d_{\text{target}}$ | 0.05$^{\dagger}$ | 0.25 | 0.10 | 0.05 | 0.025 |
| Refinement steps $M$ | | | 1 | | |
| Displacement cap | | | 0.5 | | 0.25 |
| Executed frames per chunk | – | 5 | 8 | 1 | 30 |
| Flow integration steps | 100 | 32 | 64 | 128 | 10 |
| Offline outer iterations | 30 | 5 | 5 | 5 | – |
| Critic steps before improvement | – | 8,000 | 12,000 / iter. | $10^6$ | see$^{\S}$ |
| Critic steps per online block | – | 1,500 | – | 5,000 | 2 / 20 trans. |
| Actor batch size | – | 256 | 256 | 512 | 128 |
| Loss weights $\lambda_{\text{imp}}$ / $\lambda_{\text{BC}}$ | – | 1 / 1 | 1 / 1 | 1 / 0 (offline) | 1 / 1 |

$^{\dagger}$The toy calibrates the total displacement $\|R(\mathbf{s}, \mathbf{a} + \eta\widetilde{\mathbf{g}}) - \mathbf{a}\|_2$; all other platforms calibrate the critic-induced displacement of Eq. (22). $^{\ddagger}$Configured as $t'_g = 0.1$ in the reversed time convention of the $\pi_{0.5}$ code base. $^{\S}$Test-time guidance: fork critic, 2000 steps on 270 forked rollouts. Offline-to-online: IQL critic warmed up on demonstrations, then 2 updates per 20 new transitions. On the robot, the actor is initialized from a $\pi_{0.5}$ BC checkpoint trained for 80,000 steps on the task demonstrations and trained with AdamW ($\beta = (0.9, 0.95)$, learning rate $3.75 \times 10^{-5}$, 1000-step linear warm-up, cosine decay to $7.5 \times 10^{-6}$) with the vision encoder frozen.

refined actions. Each iteration fits a new set of targets, so the actor's Adam optimizer is re-initialized at its start. We run 30 iterations on the toy and 5 on LIBERO, RoboMimic, and OGBench, and report intermediate iterations where stated (LIBERO $k \leq 3$, RoboMimic $k = 2$). On OGBench, offline RetraQ uses no anchor term ($\lambda_{\text{BC}} = 0$). On the robot, the offline-distillation arm instead fine-tunes $\pi_{0.5}$ continuously on refined targets from the Critic 1 fork critic, with the deployed guidance level $t_g = 0.9$ ($t'_g = 0.1$ in the $\pi_{0.5}$ convention) and a total-displacement target of 2.10 instead of the test-time $d_{\text{target}} = 0.025$.

**Offline-to-online learning.** Refinement starts from replayed actions. A replayed transition contains the executed part of a chunk, $\mathbf{a}^{\text{exec}}$, with $k \leq H$ frames. We convert it to the normalized action coordinates of the BC flow (on the robot, absolute to delta actions, then normalization), zero-pad it to the horizon, $\mathbf{a}^{(0)} = \text{Pad}_H(\mathbf{a}^{\text{exec}}, \mathbf{0})$, and mark the executed frames with $m_h = \mathbb{1}[h \leq k]$. The critic takes the mask as input, and the improvement loss is restricted to the executed frames,

$$\mathcal{L}_{\text{FM}}(\theta; \mathbf{s}, \mathbf{a}, \mathbf{m}) := \mathbb{E}_{t,\mathbf{z}} \big\| \mathbf{m} \odot \big( v_\theta(\mathbf{s}, \mathbf{x}_t, t) - (\mathbf{a} - \mathbf{z}) \big) \big\|_2^2, \tag{28}$$

where $\odot$ multiplies each frame by its mask entry; the loss of Section 4.3 is the case $\mathbf{m} = \mathbf{1}$. The actor loss is $\mathbb{E}_{\mathcal{B}_\pi} \big[ \lambda_{\text{imp}} \mathcal{L}_{\text{FM}}(\theta; \mathbf{s}, \mathbf{a}^\star, \mathbf{m}) + \lambda_{\text{BC}} \mathcal{L}_{\text{FM}}(\theta; \mathbf{s}, \mathbf{a}^{\text{BC}}, \mathbf{1}) \big]$, where $\mathbf{a}^{\text{BC}}$ is a freshly generated full-horizon chunk of the frozen BC policy on every platform. The improvement term supervises only the executed frames; the anchor term supplies targets for every frame. Gradients reach only $\theta$, never target generation or the sampling chain. The BC flow $\bar{\theta}$ stays frozen throughout. Algorithm 2 gives the robot version. Updates are triggered by new data: on the robot, every 20 new chunk transitions allow one actor update attempt and two critic updates, each drawing a fresh global batch of 128 chunk transitions (16 per GPU on 8 GPUs) entirely from online replay. An attempt whose guards fail is skipped, so the number of actor updates can be smaller than the number of attempts. No human interventions occurred in the reported online runs. In simulation, updates proceed in blocks (LIBERO: 1500 critic steps per 2500 environment steps; OGBench: 5000 critic steps per 5000 environment steps), with $\eta$ calibrated once as on the robot.

### C.4 Hyperparameters

Table 6 lists the RetraQ hyperparameters on each platform.

**Algorithm 2** Offline-to-online RetraQ on the robot, full version of Algorithm 1, with frame masks, guards, and the IQL critic update of Eq. (27). REFINE is defined in Algorithm 1; here $\hat{Q}$ also takes the mask $\mathbf{m}$.

**Require:** BC parameters $\bar{\theta}$; demonstrations $\mathcal{D}_{\text{demo}}$; replay buffer $\mathcal{D}$; horizon $H$; refinement steps $M$; guidance level $t_g$; target displacement $d_{\text{target}}$; IQL parameters $\gamma, \tau, \rho$; loss weights $\lambda_{\text{imp}}, \lambda_{\text{BC}}$; warm-up updates $W$; critic updates per cycle $K_Q$

$\theta \leftarrow \bar{\theta}$ ($\bar{\theta}$ frozen); initialize $Q_{\phi_1}, Q_{\phi_2}, V_\psi$; $\bar{\phi}_i \leftarrow \phi_i$; $\eta \leftarrow$ unset
**for** $w = 1, \dots, W$ **do**
    IQLUPDATE(batch from $\mathcal{D}_{\text{demo}}$) ▷ actor fixed
**end for**
**while** training budget remains **do**
    Collect rollouts of $\pi_\theta$; label success or failure; append transitions to $\mathcal{D}$
    **for** each update cycle allowed by the replay schedule **do**
        **for** $j = 1, \dots, K_Q$ **do**
            IQLUPDATE(batch from $\mathcal{D}$)
        **end for**
        Freeze $\hat{Q}(\mathbf{s}, \mathbf{a}, \mathbf{m}) = \min_i Q_{\phi_i}(\mathbf{s}, \mathbf{a}, \mathbf{m})$
        Sample actor batch $\mathcal{B}_\pi$ of $(\mathbf{s}, \mathbf{a}^{\text{exec}}, k)$ from $\mathcal{D}$
        $\mathbf{a}^{(0)} \leftarrow \text{Pad}_H(\mathbf{a}^{\text{exec}}, \mathbf{0})$; $m_h \leftarrow \mathbb{1}[h \le k]$ ▷ normalized coordinates
        **if** $\eta$ is unset **then** solve Eq. (22) on $\mathcal{B}_\pi$ ▷ then held fixed
        $\mathbf{a}^\star \leftarrow \text{REFINE}(\mathbf{s}, \mathbf{a}^{(0)}; \hat{Q}, \eta)$; $\mathbf{a}^{\text{BC}} \sim \pi_{\bar{\theta}}(\cdot \mid \mathbf{s})$ ▷ full horizon
        $\mathcal{L}_\pi \leftarrow \mathbb{E}_{\mathcal{B}_\pi}\big[\lambda_{\text{imp}}\mathcal{L}_{\text{FM}}(\theta; \mathbf{s}, \mathbf{a}^\star, \mathbf{m}) + \lambda_{\text{BC}}\mathcal{L}_{\text{FM}}(\theta; \mathbf{s}, \mathbf{a}^{\text{BC}}, \mathbf{1})\big]$
        Update $\theta$ with $\nabla_\theta \mathcal{L}_\pi$ if the numerical and displacement guards pass
    **end for**
    Periodically synchronize actor weights to the rollout workers
**end while**

**procedure** IQLUPDATE($\mathcal{B}$)
    Update $V_\psi$ on $\mathcal{L}_V$, then $Q_{\phi_1}, Q_{\phi_2}$ on $\mathcal{L}_Q$ (Eq. (27))
    $\bar{\phi}_i \leftarrow (1-\rho)\bar{\phi}_i + \rho\phi_i$ for $i = 1, 2$
**end procedure**

Table 7: Targets for component ablations. The step size $\eta$ is calibrated separately for each guided arm.

| Name | Target $\mathbf{a}^\star$ |
|---|---|
| Self-distillation | $\mathbf{a}$ |
| Map-only | $R(\mathbf{a})$ |
| Raw ascent | $\mathbf{a} + \eta\mathbf{g}$ |
| Filter-only | $\mathbf{a} + \eta J(\mathbf{a})\mathbf{g}$ |
| Raw+$R$ | $R(\mathbf{a} + \eta\mathbf{g})$ |
| RetraQ | $R(\mathbf{a} + \eta J(\mathbf{a})\mathbf{g})$ |

# D ADDITIONAL EXPERIMENTAL DETAILS AND RESULTS

## D.1 OPERATOR DEFINITIONS AND CALIBRATION

We use the abbreviations $R$, $J$, and $\mathbf{g}$ introduced in Section 5.1. Table 7 specifies the target used by each component ablation. The unchanged-target control measures changes caused by repeated self-distillation. The map-only control measures the result of applying the frozen action map without a critic step. Neither control is displacement-matched to the critic-guided arms.

The controlled toy campaign calibrates the median total movement $\|\mathbf{a}^\star - \mathbf{a}\|_2$ to its target displacement. The deployed calibration described in Section 4 instead measures the movement induced by the critic relative to the mapped baseline, $\|R(\mathbf{a} + \eta J(\mathbf{a})\mathbf{g}) - R(\mathbf{a})\|_2$. These quantities differ whenever $R(\mathbf{a}) \neq \mathbf{a}$, so the controlled results evaluate the operator under the toy calibration. AWR is not displacement-calibrated in the evaluated implementation. QAM (Li & Levine, 2026) fits the actor by adjoint matching, with the frozen BC flow as the base field; its temperature is matched to the same target displacement by bisection on the achieved actor drift and reverts to its default value when the target lies outside the achievable range.

Table 8: Forward-Jacobian diagnostics for one $D = 16$ flow. Each cell is a quantile over query points.

| Query set | $G_N$, med. | $G_N$, 95th | $E_T$, med. | $E_T$, 95th |
|---|---|---|---|---|
| On manifold | 0.0341 | 0.0446 | 0.0597 | 0.3979 |
| Actor samples | 0.0334 | 0.0424 | 0.0470 | 0.3483 |
| Perturbed points | 0.0342 | 0.0448 | 0.0604 | 0.3977 |

Table 9: Separate dimension sweep. $G_N$ is the median normal gain at the evaluation time $t_g$. Training budget is fixed across dimensions. Dashes mark two curved cells whose flows did not pass the fit check.

| $D$ | $k$ | Geometry | $t_g$ | $G_N$ | Model check |
|---|---|---|---|---|---|
| 2 | 1 | flat | 0.9844 | 0.0567 | accepted |
| 2 | 1 | curved | 0.9531 | 0.0343 | accepted |
| 8 | 2 | flat | 0.9844 | 0.0789 | accepted |
| 8 | 2 | curved | 0.9844 | 0.0958 | accepted |
| 32 | 2 | flat | 0.9844 | 0.2120 | accepted |
| 32 | 2 | curved | — | — | not accepted |
| 128 | 2 | flat | 0.9844 | 0.8642 | accepted |
| 128 | 2 | curved | — | — | not accepted |
| 512 | 2 | flat | 0.9844 | 0.9659 | accepted |
| 512 | 2 | curved | 0.9844 | 0.9661 | accepted |

Online component runs can acquire different replay data and critic parameters as training proceeds, so their differences measure the resulting learning procedures; the controlled problems of Section 5.2 isolate the immediate effect of each operator at fixed actions and a fixed critic.

### D.2 Forward-Jacobian Diagnostics

Fix a context and a point $\mathbf{a}$ on its known $k$-dimensional manifold, with $1 \le k < D$. Let $P_T(\mathbf{a})$ and $P_N(\mathbf{a})$ be the orthogonal tangent and normal projectors defined in Section 2. We measure

$$G_N(\mathbf{a}) := \frac{\|J(\mathbf{a})P_N(\mathbf{a})\|_F}{\sqrt{D-k}}, \tag{29}$$

$$E_T(\mathbf{a}) := \frac{\|(J(\mathbf{a}) - I_D)P_T(\mathbf{a})\|_F}{\sqrt{k}}, \tag{30}$$

where $\|\cdot\|_F$ is the Frobenius norm, the square root of the sum of squared matrix entries. Equivalently, these are root-mean-square responses over an orthonormal normal or tangent basis. They also equal the root-mean-square responses to uniformly sampled unit directions in the respective subspaces. The implementation evaluates the basis responses with forward JVPs without forming the Jacobian. For $J = P_T$, both diagnostics vanish. For $J = I_D$, normal gain is one and tangent error is zero. Small tangent error alone therefore does not establish normal filtering. These diagnostics describe average directional responses, rather than worst-direction operator-norm bounds.

Table 8 reports the $D = 16$ diagnostics. For sampled or perturbed points away from the manifold, the reference tangent and normal spaces are evaluated at their known manifold projections. Each row summarizes 512 query points for one frozen flow at $t_g = 0.9$.

For this flow, a separate same-input comparison between $J\mathbf{v}$ and $J^\top\mathbf{v}$ reports a relative difference of approximately $0.9167$, after the campaign's aggregation over points and four probe directions. Because the learned Jacobian is far from symmetric, we report forward-Jacobian diagnostics, which match the JVP used by RetraQ.

The dimension sweep in Table 9 uses a different geometric family from the $D = 16$ refinement campaign. Training budget was held fixed across dimensions, so fitting error also varies across configurations and the sweep does not isolate a causal effect of dimension. The $D = 16$ problem is not part of this sweep.

Table 10: Controlled $D = 16$ problem after 30 refinement–distillation rounds. The objective is averaged over 4096 samples within each actor seed. The distance statistic is the within-seed 95th percentile of Euclidean distance to the known manifold. Both columns report medians over eight actor seeds sharing one reference flow and critic. The map-only probe did not report this distance percentile.

| Method | True objective ↑ | Distance, 95th pct. ↓ |
|---|---|---|
| BC reference | 0.3947 | 0.0149 |
| Self-distillation | 0.2656 | 0.0812 |
| Map-only | 0.2859 | — |
| Raw ascent | -16.7940 | 1.0160 |
| Filter-only | -0.5822 | 0.4300 |
| Raw+$R$ | 0.7744 | 0.0726 |
| **RetraQ** | 0.8272 | 0.0284 |

**Guidance-level sweep.** Figure 4a,b repeats the diagnostics on flat manifolds with $D \in \{2, 8, 32, 128, 512\}$ at five guidance levels $t_g \in \{0.922, 0.938, 0.953, 0.969, 0.984\}$, with one flow per dimension trained for 20,000 steps (512 on-manifold query points; 128 for $D = 512$). The deployed level $t_g = 0.9$ lies just below this grid. Across the grid, the tangent error falls from about 0.07–0.09 to about 0.01–0.02, whereas the normal gain rises, from 0.27 to 0.86 at $D = 128$ and from 0.84 to 0.97 at $D = 512$; for $D \leq 8$ it stays below 0.1. Because the training budget is fixed, the flow fits the high-dimensional cells worst (normalized held-out loss 0.708 at $D = 512$ against 0.072–0.418 elsewhere), so the growth with $D$ is confounded with fitting quality. Curved cells at $D = 32$ and 128, whose flows did not pass the fit check, are omitted.

**Critic exploitation.** On the $D = 16$ problem, the critic's predicted objective and the true objective of the refined actions are 1.498 and $-16.794$ for raw ascent, 0.679 and $-0.582$ for filter-only, 0.831 and 0.774 for Raw+$R$, and 0.838 and 0.827 for RetraQ (Figure 4c).

### D.3 Controlled Refinement Results

Table 10 uses the batch mean of the analytic task objective before taking the median over actor seeds. Using the batch median instead changes the filter-only result from $-0.5822$ to 0.1088. Both summarize the same campaign but answer different questions about its sample distribution. The batch mean is retained as the primary objective statistic so that the poor lower tail remains represented. The reported distance percentile describes proximity to the known manifold. It is not a lower bound or a measure of statistical uncertainty.

We report continuous distances rather than threshold-based out-of-distribution fractions, whose radius would depend on a reference control that itself drifts.

### D.4 LIBERO Evaluation and Offline Stability

Figure 9 shows that the two retracted arms produce actions near fixed points of the frozen map: more than 99% of their actions have a smaller residual than the median demonstration chunk, against 29–91% for the other arms. Actors trained without the retraction, including self-distillation, remain within the range of the demonstrations.

Online LIBERO evaluation uses 30 test initializations per task and actor seed, indexed 0–29. The reported validation protocol uses five initializations starting at index 45. The comparison covers all ten Goal and all ten Object tasks. The online Spatial evaluation covers six tasks, with task indices $\{0, 1, 4, 5, 6, 7\}$. These are the Spatial tasks with the lowest BC success; on the remaining tasks, BC already leaves little room for improvement through offline or online fine-tuning. The offline Spatial campaign covers eight tasks, so the two Spatial averages are not directly comparable.

For each training seed, task success rates are averaged with equal task weights. Table 11 reports the mean and sample standard deviation of these seed-level averages.

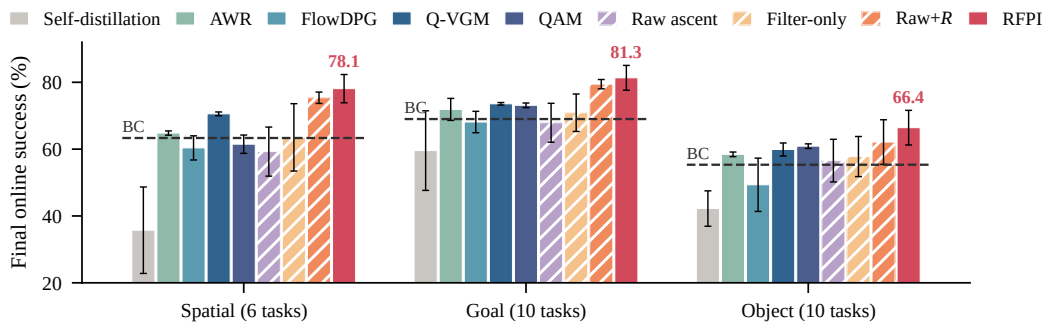



Figure 8: **Final online LIBERO success** (%), mean ± standard deviation over three actor seeds. Cool colors: baselines; hatched warm colors: component ablations of RetraQ; dashed lines: BC. All numbers are in Table 11.

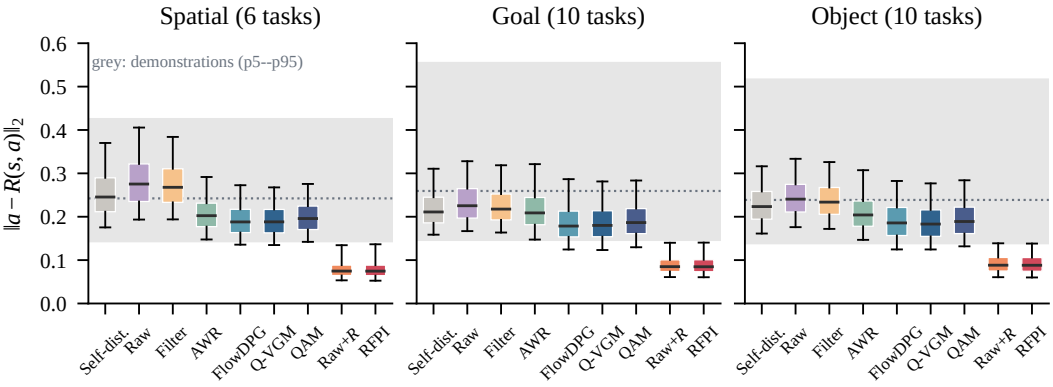



Figure 9: **Retraction residual of the final online actors on LIBERO**, $\|\mathbf{a} - R(\mathbf{s}, \mathbf{a})\|_2$ at $t_g = 0.9$, pooled over tasks and seeds (boxes: quartiles; whiskers: 5th–95th percentiles). Grey band and dotted line: the same residual on held-out demonstration chunks. The residual is computed with the frozen map that RetraQ and Raw+$R$ use to build their targets, so it measures consistency with the behavior model rather than a distance to the data.

**Limits of repeated offline improvement.** Repeated offline improvement behaves differently from online improvement. After three refinement–distillation rounds, RetraQ success falls from the BC reference rates of 69.58%, 69.00%, and 55.33% to 39.86%, 50.00%, and 21.67% on the evaluated Spatial, Goal, and Object task sets, respectively (Table 12 and Figure 10). AWR and FlowDPG do not exhibit the same across-suite deterioration under their evaluated configurations. The self-distillation control also changes over rounds, although its decline is smaller than RetraQ's. These results limit claims of reliable repeated offline improvement. They motivate distinguishing errors in refined targets from errors introduced when fitting those targets, which the final policy scores alone cannot separate.

The offline curves in Table 12 evaluate fixed rounds and show the complete available suite aggregates. Round is denoted by $r$ to distinguish it from the intrinsic dimension $k$ used in the geometric analysis. The shared BC reference is measured once per task. Raw+$R$ and map-only were not evaluated in this offline LIBERO campaign. On Spatial, AWR and QAM were evaluated after a simulator reinstallation that changed the outcomes of re-evaluated earlier cells, so their Spatial offline entries are not directly comparable with those of the other arms.

The offline BC reference means are 69.58% for the eight Spatial tasks, 69.00% for Goal, and 55.33% for Object. Restricting the Spatial BC evaluations to the six online task indices gives 63.33%. Figure 8 uses 63.33%, 69.00%, and 55.33% as the BC references for the online Spatial, Goal, and Object suites.

Table 11: All online LIBERO arms. Entries are success (%) as mean ± across-seed standard deviation. Goal and Object cover all ten tasks. Spatial covers six tasks.

| Method | Goal (10 tasks) | Object (10 tasks) | Spatial (6 tasks) |
|---|---|---|---|
| Self-distillation | $59.56 \pm 11.91$ | $42.22 \pm 5.30$ | $35.74 \pm 12.94$ |
| AWR | $71.89 \pm 3.29$ | $58.44 \pm 0.69$ | $64.81 \pm 0.64$ |
| FlowDPG | $68.11 \pm 3.20$ | $49.33 \pm 7.97$ | $60.37 \pm 3.61$ |
| Q-VGM | $73.56 \pm 0.38$ | $59.89 \pm 1.95$ | $70.56 \pm 0.56$ |
| QAM | $73.11 \pm 0.69$ | $60.89 \pm 0.69$ | $61.48 \pm 2.74$ |
| Raw ascent | $67.89 \pm 5.82$ | $56.56 \pm 6.38$ | $59.26 \pm 7.34$ |
| Filter-only | $70.89 \pm 5.60$ | $57.78 \pm 6.00$ | $63.52 \pm 10.08$ |
| Raw+$R$ | $79.44 \pm 1.39$ | $62.11 \pm 6.68$ | $75.37 \pm 1.70$ |
| **RetraQ** | $\mathbf{81.34 \pm 3.71}$ | $\mathbf{66.42 \pm 5.18}$ | $\mathbf{78.10 \pm 4.23}$ |

Table 12: Offline LIBERO success (%) at each fixed refinement round $r$. Each value averages three actor seeds and 30 evaluation episodes per task and seed, with equal task weights. Shared BC reference rates are given in the text.

| | Spatial (8 tasks) | | | Goal (10 tasks) | | | Object (10 tasks) | | |
|---|---|---|---|---|---|---|---|---|---|
| Method | $r=1$ | $r=2$ | $r=3$ | $r=1$ | $r=2$ | $r=3$ | $r=1$ | $r=2$ | $r=3$ |
| Self-distillation | 64.86 | 61.11 | 62.22 | 66.67 | 66.11 | 66.11 | 50.22 | 51.44 | 51.00 |
| AWR | 65.56 | 70.28 | 69.86 | 72.00 | 73.00 | 76.11 | 55.44 | 57.67 | 58.78 |
| FlowDPG | 65.00 | 64.31 | 64.58 | 73.67 | 75.44 | 74.33 | 56.67 | 55.89 | 56.44 |
| Q-VGM | 19.86 | 23.89 | 21.94 | 34.22 | 35.33 | 32.44 | 19.89 | 18.56 | 17.56 |
| QAM | 40.83 | 38.06 | 37.92 | 36.67 | 34.44 | 34.22 | 24.11 | 22.89 | 20.33 |
| Raw ascent | 38.89 | 22.08 | 16.67 | 56.11 | 40.33 | 23.33 | 39.22 | 17.44 | 8.89 |
| **RetraQ** | 71.39 | 54.86 | 39.86 | 67.78 | 55.33 | 50.00 | 55.44 | 29.89 | 21.67 |

### D.5 RoboMimic Held-Out Comparison

The preferred Can comparison fixes the number of refinement rounds at two using evaluation seeds 90000–90039, then evaluates on held-out seeds 90100–90199. Each trained arm has three actor-training seeds and 100 held-out episodes per seed. The BC reference has one checkpoint evaluated on 100 episodes. The broader 90000–90199 evaluation band includes selection episodes and is not used as the primary held-out estimate.

For RetraQ minus Raw+$R$, the paired 95% interval is $[-2.00, 8.67]$ percentage points around a difference of 3.33 points. It was constructed by resampling episode pairs within each training seed and is conditional on those trained policies. The reported McNemar value is $p = 0.268$. The difference is therefore not statistically significant at this sample size.

**Critic quality.** Table 14 repeats the Can refinement with three critics trained on 588, 3,234, and 4,410 samples. Each entry is the best success over refinement rounds $r \geq 1$ on the selection band (seeds 90000–90039, 40 episodes per actor seed), and the round is selected on the same episodes, so the entries are optimistic peaks rather than held-out estimates. The BC rate on this band is 57.5%. With the weakest critic, RetraQ does not exceed BC and ranks last; with the two stronger critics, it ranks first. At 4,410 samples, only Q-VGM has three actor seeds. For the RetraQ arm, the filtered and raw gradients have mean cosine 0.670, 0.697, and 0.712 for the three critics, with norm ratios $\|J_R\mathbf{g}\|/\|\mathbf{g}\|$ of 0.400, 0.342, and 0.349 (first round, averaged over seeds).

### D.6 OGBench Online Comparison

The cube-single campaign starts from an offline-refined actor. The reported initial success is 30.7% for that actor and 8.0% for the BC reference. Table 15 reports final policies with two training seeds and 200 evaluation episodes per seed.

The table's standard deviations summarize two training seeds.

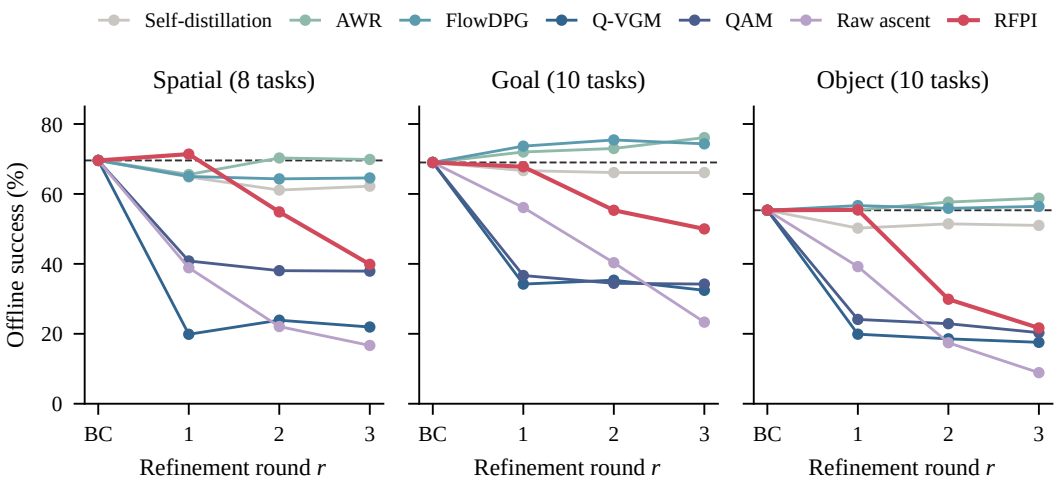



Figure 10: **Repeated offline refinement on LIBERO** (Table 12). Round "BC" is the shared BC reference (dashed line).

Table 13: RoboMimic Can held-out success (%) after two refinement rounds. The final column is the range across the three actor seeds, not a confidence interval. The BC reference has one checkpoint.

| Method | Mean | Across-seed range |
|---|---|---|
| BC reference | 53.00 | one checkpoint |
| FlowDPG | 64.33 | 62.00–67.00 |
| Q-VGM | 69.00 | 68.00–71.00 |
| Raw+$R$ | 67.00 | 65.00–69.00 |
| **RetraQ** | 70.33 | 64.00–77.00 |

**Filter diagnostics.** During online training of the RetraQ arm, we record the cosine between the filtered and raw critic gradients and their norm ratio at 10k, 25k, 50k, and 100k steps for both seeds. The median cosine is 0.988–0.991 at every checkpoint, and the ratio of median norms $\|J_R\mathbf{g}\|/\|\mathbf{g}\|$ is 0.42–0.48. On this task, the filter therefore changes mainly the length of the step, which the displacement calibration then sets, consistent with the near-equal final success of RetraQ and Raw+$R$ in Table 15.

On the harder cube-double task, both evaluated methods remain near zero (RetraQ 2.50%, Q-VGM 1.25%; two seeds, 400 episodes each), so we do not draw conclusions from it.

### D.7 Is RetraQ Behavior Regularization?

A natural concern is that, at $\sigma_g \approx 0.11$, the map $R$ might act mainly as a pull toward the behavior policy and $J_R$ as an isotropic shrinkage, so that RetraQ would reduce to critic ascent plus behavior regularization. The evidence collected above bears on this question as follows.

**An explicit behavior anchor does not reproduce the retraction.** On LIBERO, the component ablations share the distillation objective of Eq. (9), including its BC anchor, and differ only in their targets (Table 7). With this anchor, raw ascent and filter-only remain within about two points of BC on average, whereas replacing the target $\mathbf{a}+\eta\mathbf{g}$ by $R(\mathbf{a}+\eta\mathbf{g})$ adds 5.6–16.1 points (Table 11). The retraction therefore contributes beyond the behavior regularization that the anchor already provides.

**The map alone does not improve the policy.** Map-only targets $R(\mathbf{a})$ apply the same map without a critic step. They do not improve over the behavior policy: the true objective is 0.286 against 0.395 for BC on the $D = 16$ problem (Table 10), and OGBench success is 0.25% against 3.50% for continued BC (Table 15). Self-distillation also falls below BC on LIBERO. The gains thus require the critic direction; $R$ contributes by correcting the critic step rather than by regularizing on its own.

Table 14: RoboMimic Can success (%) versus critic training-set size. Best round per method, selected on the same selection-band episodes; BC is 57.5%. ‡One actor seed; all other cells have three.

| Critic samples | RetraQ | Q-VGM | FlowDPG |
|---|---|---|---|
| 588 | 57.5 | 59.2 | 62.5 |
| 3,234 | **73.3** | 56.7 | 71.7 |
| 4,410 | **87.5**‡ | 79.2 | 80.0‡ |

Table 15: Final OGBench cube-single success (%), mean ± standard deviation over two seeds.

| Method | Success |
|---|---|
| Continued BC | $3.50 \pm 1.41$ |
| Self-distillation | $19.75 \pm 13.79$ |
| Map-only | $0.25 \pm 0.35$ |
| Raw ascent | $97.00 \pm 3.54$ |
| Raw+$R$ | $99.00 \pm 0.71$ |
| **RetraQ** | $98.75 \pm 0.35$ |
| Q-VGM | $95.00 \pm 5.66$ |
| FlowDPG | $4.50 \pm 0.71$ |

**RetraQ moves along the manifold, not back to BC.** On the $D = 16$ problem, where the manifold is known, a pull toward the behavior policy would move refined actions toward BC and their objective toward BC's $0.395$. RetraQ instead reaches a true objective of $0.827$, while the 95th-percentile distance to the manifold changes only from $0.015$ for BC to $0.028$, against $0.073$ for Raw+$R$ (Table 10). The improvement therefore comes from motion along the manifold. On this problem, the learned $J_R$ has median normal gain $0.034$ and tangent error $0.060$ on the manifold (Table 8): it is close to $P_T$, not to a scaled identity.

### D.8 Real-Robot Setup and Records

**Platform and policy.** The task is phone-charging plug insertion on an AgiBot G02 dual-arm robot controlled at 30 Hz, with three cameras (head and both wrists). The policy is a $\pi_{0.5}$ flow behavior-cloned on task demonstrations for 80,000 steps (batch size 256). It predicts 50-frame chunks, of which the first 30 frames (1 s) are executed, and its flow is integrated with 10 steps.

**Test-time guidance.** RetraQ uses $t_g = 0.9$ ($\sigma_g \approx 0.11$), $\eta = 0.01$ calibrated once offline and then frozen, $d_{\text{target}} = 0.025$, and a displacement cap of $0.25$ (Appendix C.1). Guidance is enabled only at detected fork states 0–2 (Appendix C.3); elsewhere the BC chunk is executed unchanged. The baselines use the same frozen policy, the same critic, and the same gate: QGF updates the velocity as $v \leftarrow v - \lambda \mathbf{g}$ without a $1/t$ factor, FlowDPG takes a normalized velocity step with a $1/t$ factor, and Q-VGM follows its first version (Wang et al., 2026): it takes gradient-clipped ascent steps along $\mathbf{g}$ from the one-step clean-action estimate, clips each iterate to the action range, and keeps the highest-valued iterate, with gating and a $1/t$ factor. Their step sizes are set by bisection so that their median guided displacement equals the median displacement of the deployed RetraQ arm.

**Critics and trials.** Every real-robot evaluation consists of 60 trials. In offline-to-online learning, this holds for every checkpoint of each of the three training runs per method, so each reported online entry averages $3 \times 60$ trials. Critic 1 and Critic 2 are the linear and the nonlinear fork critic of Appendix C.2.1, trained on the same data.

**Why Q-VGM fails with the linear critic.** With Critic 1, $A_\omega$ in Eq. (8) is linear in the action deviation, so $\nabla_{\mathbf{a}} Q$ points in the same direction at every action, and the critic value increases monotonically along it. Q-VGM takes several ascent steps along this direction, clips each iterate to the action range, and keeps the highest-valued iterate. Under a monotone critic, this rule always selects the last, farthest iterate, and neither the coordinate clip nor the value-based selection removes the normal component of the fixed direction. The executed chunks can therefore leave the region cov-

Table 16: Real-robot phone-charging success rates (%) without online interaction, as plotted in Figure 5b. **Top:** test-time guidance of the frozen BC policy; the BC reference is shared across the two critic columns, and all guidance methods are displacement-matched to RetraQ. Each entry is over 60 trials. **Bottom:** offline distillation with the deployed guidance level $t_g = 0.9$ and a larger displacement budget shared by the three arms; the RetraQ arm uses the Critic 1 fork critic.

| Method | Critic 1 | Critic 2 |
|---|---|---|
| *Test-time guidance (policy parameters fixed)* | | |
| BC | 55.00 | |
| QGF (Zhou et al., 2026) | 58.33 | 56.67 |
| Q-VGM | 0.00 | 58.33 |
| FlowDPG | 63.33 | 66.67 |
| **RetraQ** | **76.67** | **73.33** |
| *Offline distillation (shared displacement budget)* | | |
| Q-VGM | 0.00 | — |
| FlowDPG | 28.33 | — |
| **RetraQ** | **60.00** | — |

Table 17: Real-robot offline-to-online success (%) versus online training time, as plotted in Figure 5c. All methods start from the same BC policy and the same warmed-up IQL critic (0 h); each then trains its own critic on its own rollouts under the same total interaction budget. Each checkpoint of each training run is evaluated with 60 trials, and each entry is the mean over three training runs per method.

| Method | 0 h | 0.5 h | 1.0 h |
|---|---|---|---|
| QGF | 55.00 | 88.33 | 91.67 |
| FlowDPG | 55.00 | 88.33 | 86.67 |
| Q-VGM | 55.00 | 95.00 | 93.33 |
| **RetraQ** | 55.00 | **98.33** | **100.00** |

ered by the behavior data at the fork states, which is consistent with the observed failure of every trial. Because the step size is matched to RetraQ's median displacement, this failure stems from the direction of the update rather than its size. With the nonlinear Critic 2, the gradient direction varies with the action and the value need not increase monotonically along the ascent path, and Q-VGM stays near BC ($58.33\%$). RetraQ receives the same gradient but removes its normal component with $J_R$ and maps the endpoint back with $R$.

**Offline distillation.** The offline-distillation arms are a separate experiment. They distill refined targets into the policy without online interaction, under one shared displacement budget: the total RetraQ displacement ($2.10$ at $\eta = 0.05$) measured at this setting. The RetraQ arm uses the Critic 1 fork critic and the same guidance level as deployment, $t_g = 0.9$ ($t'_g = 0.1$ in the $\pi_{0.5}$ convention). The displacement budget, however, is much larger than that of test-time guidance ($d_{\text{target}} = 0.025$), far outside the perturbation range that supervises the fork critic, so it probes how each method's targets behave when the critic is queried away from its training data. The Q-VGM-distilled policy reaches $0\%$: it repeatedly hits the table. The FlowDPG-distilled policy reaches $28.33\%$: it approaches the socket but frequently misaligns the plug at insertion. The RetraQ-distilled policy reaches $60.00\%$, on par with the BC rate of $55\%$ (Fisher's exact test, $p = 0.71$). At this budget, unconstrained targets thus degrade the distilled policy, whereas RetraQ targets preserve it; a comparison that varies only $d_{\text{target}}$ would be needed to determine whether smaller displacements also let offline RetraQ improve substantially over BC.

**Online RetraQ.** Averaged over three training runs, online RetraQ reaches $98.33\%$ after 30 minutes and $100\%$ after one hour of online training, the highest success of all methods at each checkpoint (Table 17). It uses the IQL critic and the update schedule of Appendix C.3: every 20 new chunk transitions allow one actor update attempt and two critic updates, with batches drawn entirely from online replay. No human interventions occurred during training; episodes are labeled by the operator as successes or failures.

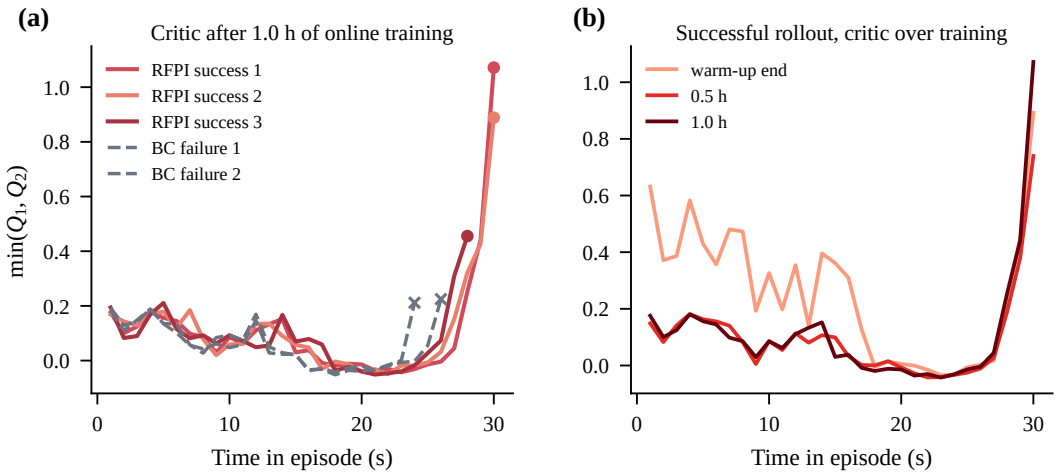



Figure 11: **Online IQL critic along fixed evaluation rollouts** ($\min(Q_1, Q_2)$ at the end of each 1 s action chunk). **(a)** Critic after 1.0 h of online training along three successful rollouts of the online RetraQ policy and two failed rollouts of the BC policy; markers show the last chunk. The value rises sharply only in the final seconds of successful insertions. The two groups come from different policies, so this panel is illustrative. **(b)** One successful rollout scored by the critic at successive checkpoints. After warm-up, the critic assigns high values early in the episode; with online training, it concentrates value at the insertion.