arXiv:2606.05468v1 [cs.RO] 3 Jun 2026

FlowPRO: Reward-Free Reinforced Fine-Tuning of Flow-Matching VLAs via Proximalized Preference Optimization

Yihao Wu[1,3] **He Zhang**[1,2*] **Junbo Tan**[3] **Xueqian Wang**[3] **Zhengyou Zhang**[1,2]
[1]Tencent Robotics X, [2]Futian Laboratory, [3]Tsinghua University
Project Website: https://wuyeyexvnainai.github.io/flowpro/

Abstract: Post-training Vision-Language-Action (VLA) models into policies that can be reliably deployed on real robots remains a major bottleneck. SFT and DAgger exploit failure signals only indirectly, and reward-based RL is bottlenecked by the difficulty of real-world reward design and of training reliable critics. We present **FlowPRO**, a reward-free offline reinforced fine-tuning framework for flow-matching VLAs. Algorithmically, we propose **RPRO** (*Robotic Flow-matching Proximalized Preference Optimization*), a preference-optimization objective tailored to the flow-matching action head of VLA models. RPRO pairs a contrastive optimizer with an explicit proximal regularizer that anchors the absolute magnitude of the implicit reward, thereby eliminating the reward-hacking failure mode of plain Flow-DPO. On the data side, a teleoperated intervention-and-rollback paradigm produces naturally paired positive and negative trajectories (τ^w, τ^l) on a real robot from a single operator action; a Smooth Interpolation procedure, combined with batch mixing, then converts these sparse corrections into dense per-state supervision while preserving the base policy's capabilities. On four long-horizon bimanual tasks, FlowPRO attains the highest success rate, outperforming four representative baselines, and ablations confirm the contribution of each loss component.

Keywords: Reinforcement Learning, Preference Alignment, Vision-Language-Action Models, Flow Matching, Robot Manipulation

1 Introduction

Vision-Language-Action (VLA) models have recently emerged as the dominant paradigm for building generalist robots, mapping visual observations and language instructions to low-level robot actions in an end-to-end fashion [1, 2, 3, 4, 5, 6, 7, 8]. However, post-training such models into reliably deployable policies remains challenging in practice. On real robots, both expert demonstrations and task-specific feedback signals are expensive to collect, and post-training gains on the hardest failure modes tend to saturate well before the policy reaches deployment-grade reliability [9].

Existing real-robot post-training pipelines for VLA models fall into three families, each with a characteristic limitation that re-emerges in the flow-matching setting. The first family, SFT and its interactive extensions—vanilla SFT [1, 3] and DAgger-style human correction [10]—scales to real hardware but only weakly exploits the failure signals from autonomous rollouts: vanilla SFT discards them entirely, while DAgger uses them merely to trigger expert correction rather than as a direct optimization signal. The second family, reward- or value-based RL [11, 12, 13, 14, 15, 16, 17], requires training a reliable reward, value, or advantage model, which itself becomes a key obstacle for contact-rich manipulation where dense reward signals are difficult to obtain [18, 19]. The third family, preference-based RL, bypasses reward design via preference data, with GRAPE [20] applying a DPO-style trajectory-level contrast over positive–negative trajectory pairs (τ^w, τ^l) [21]. However, on flow-matching VLAs, the trajectory-level contrast weakens the per-state learning sig-

*Corresponding author

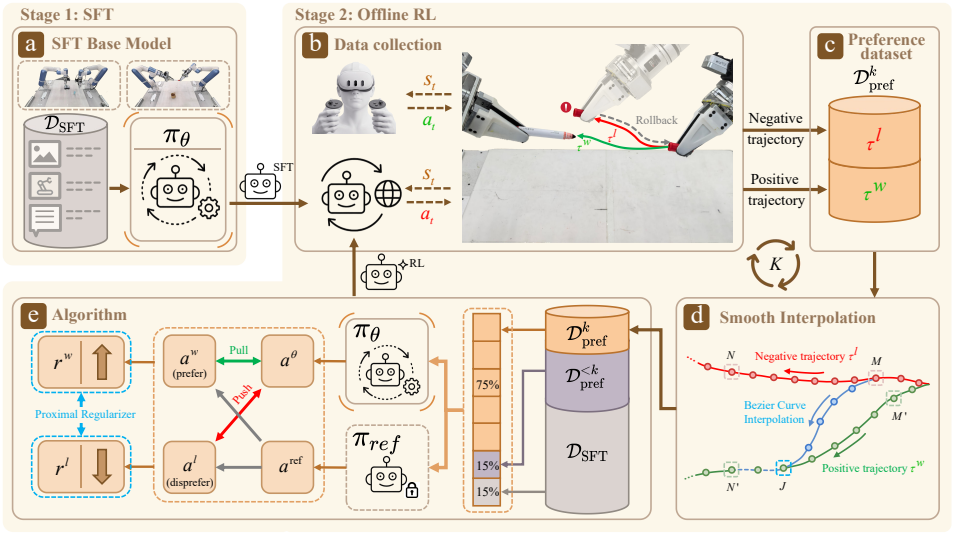

Figure 1: **Overview of the FlowPRO framework. (a) SFT Base Model:** Stage 1 trains π_θ on $\mathcal{D}_{\text{SFT}}$. **(b) Data Collection:** operator-triggered rollback and operator teleoperation yield paired positive and negative trajectories τ^w, τ^l. **(c) Preference Dataset:** pairs are aggregated into $\mathcal{D}^k_{\text{pref}}$. **(d) Smooth Interpolation:** Bézier interpolation synthesizes the missing counterpart at action-chunk granularity (e.g., $M \to J$, $J \to N'$), producing dense per-state tuples. **(e) Algorithm (RPRO):** a^θ, a^{ref} from π_θ and frozen π_{ref} are compared against a^w/a^l to drive $r^w \uparrow, r^l \downarrow$, optimized on a mixed batch of $\mathcal{D}^k_{\text{pref}}$, $\mathcal{D}^{<k}_{\text{pref}}$, and $\mathcal{D}_{\text{SFT}}$; the updated policy is redeployed for K rounds.

nal; moreover, it inherits DPO's reward-hacking failure mode [22], which yields implausible control actions a that drift far from both the positive action a^w and the negative action a^l.

We address these limitations with **FlowPRO**, a preference-based offline RL framework for real-robot post-training of flow-matching VLAs (Fig. 1). FlowPRO proceeds in two stages: Stage 1 performs SFT on a task-specific dataset $\mathcal{D}_{\text{SFT}}$ to obtain a base policy from a flow-matching VLA backbone [1, 5]; Stage 2 then runs an iterative offline-RL loop on top of it for K rounds. In each round, a teleoperated intervention-and-rollback procedure (Fig. 1(b)) lets the operator abort impending failures and teleoperate a correction from a rolled-back recovery state, producing naturally paired trajectories (τ^w, τ^l). A Smooth Interpolation procedure (Fig. 1(d)) then converts these sparse trajectory-level corrections into dense per-state preference tuples by synthesizing the missing counterpart at each state. These tuples are optimized by our **RPRO** algorithm (*Robotic Flow-matching Proximalized Preference Optimization*), which adopts the explicit proximal regularizer of Proximalized Preference Optimization [22] to eliminate the reward-hacking failure mode of plain Flow-DPO, using the previous iteration's policy as the reference policy π_{ref}. We summarize our contributions as follows:

- **Algorithm (RPRO).** We introduce **RPRO**, a preference-optimization algorithm for flow-matching VLAs with two key properties: (i) an explicit proximal regularizer that eliminates the reward-hacking failure mode of plain Flow-DPO; and (ii) a gradient-vanishing property that unifies preference pairs and SFT demonstrations within a single objective.

- **Preference data construction.** We propose an *intervention-and-rollback* paradigm for on-robot preference data collection: a single operator action yields a trajectory pair (τ^w, τ^l), while an operator-chosen rollback horizon Δ diversifies the per-pair initial state—all without recording separate positive and negative rollouts.

- **Data processing and training recipe.** We design two components that make iterative RPRO fine-tuning sample-efficient under scarce real-robot corrections: a Smooth Interpolation procedure that turns sparse corrections into dense per-state preference supervision, and a fixed-ratio batch-mixing schedule that up-weights the round-k dataset $\mathcal{D}^k_{\text{pref}}$ relative to the historical pool and $\mathcal{D}_{\text{SFT}}$.

2 Related Work

Reinforcement Learning for Robot Policy Fine-Tuning. Applying RL to imitation-pretrained robot policies has a long history. DAgger [10] addresses distribution shift by iteratively collecting on-policy expert corrections, and HG-DAgger [23] introduces a human-gated variant that triggers correction only when the policy enters unsafe states; both require continuous human supervision and cannot exploit suboptimal data through a contrastive signal. On the offline-RL side, Implicit Q-Learning [24] and Calibrated Q-Learning [25] enable stable value-based fine-tuning from logged data and bridge offline pre-training with online updates. Recent VLA fine-tuning methods explore environment-reward RL [14], mixed SFT/online-RL pipelines [15], policy-gradient fine-tuning of diffusion policies (DDPO [26], DPPO [16]), and constrained RL for safety [27], but all of them depend on an explicit reward function or on-policy environment interaction, both of which are difficult to scale on real hardware. In contrast, our approach is fully offline and reward-free.

Preference Optimization without Reward Models. *Direct alignment* methods learn policies from preference data without an explicit reward model. DPO [21] and its variants—KTO [28], IPO [29], SimPO [30], and the iterative variant of Xiong et al. [31]—all suffer from *likelihood underdetermination* [22], which PRO [22] resolves in the LLM setting via an explicit proximal regularizer. Contrastive Preference Learning [32] additionally derives a regret-based reformulation that bypasses reward modeling for sequential decision-making. Extensions of DPO to diffusion-based generation (Diffusion-DPO [33], D3PO [34]), to flow-based video generation (Flow-DPO [35]), and to VLAs via trajectory-level contrast (GRAPE/TPO [20]) inherit the same likelihood-underdetermination issue, with TPO additionally diluting the per-state signal. Our work is the first to extend PRO to continuous action generation in flow-matching VLAs.

3 Method

We present FlowPRO, a reward-free offline reinforced fine-tuning framework for flow-matching VLAs (overall procedure in Algorithm 2). We first review the theoretical foundations of preference optimization (§3.1), then introduce RPRO, our extension of PRO to flow matching (§3.2), and finally present our teleoperated data collection paradigm (§3.3) and data processing pipeline (§3.4).

3.1 Preliminaries

RLHF, DPO, and PRO. Standard RLHF [11, 12] fits a reward model r_ϕ under the Bradley-Terry assumption [36] and optimizes a KL-regularized objective over the current policy π_θ and a frozen reference π_{ref}:

$$\max_\theta \; \mathbb{E}_{s\sim\mathcal{D},\, a\sim\pi_\theta(\cdot|s)}\left[r_\phi(s,a)\right] - \beta\,\mathrm{KL}(\pi_\theta\|\pi_{\text{ref}})\,, \tag{1}$$

typically via PPO [13]. DPO [21] eliminates the explicit reward model by exploiting the closed-form mapping $r(s,a) = \beta\log\frac{\pi_\theta(a|s)}{\pi_{\text{ref}}(a|s)} + \beta\log Z(s)$ implied by Eq. (1). Substituting it into the Bradley-Terry model over preferred and dispreferred action pairs $(a^w, a^l) \sim \mathcal{D}$ yields the contrastive loss

$$\mathcal{L}_{\text{DPO}}(\theta) = -\mathbb{E}_{(s,a^w,a^l)\sim\mathcal{D}}\left[\log\sigma\Big(\beta\log\tfrac{\pi_\theta(a^w|s)}{\pi_{\text{ref}}(a^w|s)} - \beta\log\tfrac{\pi_\theta(a^l|s)}{\pi_{\text{ref}}(a^l|s)}\Big)\right]. \tag{2}$$

Since Eq. (2) only constrains the *relative* log-likelihood of a^w vs. a^l, the optimizer can drive both $\pi_\theta(a^w|s)$ and $\pi_\theta(a^l|s)$ down together—a *likelihood underdetermination* pathology [22] that manifests as reward hacking even without an explicit reward model. PRO [22] addresses this by decomposing the DPO loss into a contrastive optimizer plus a proximal regularizer anchored on a virtual *hyper-response* $\mathcal{H}$ that aggregates all unobserved actions; we extend this construction to the flow-matching setting as RPRO (§3.2).

3.2 RPRO: Robotic Flow-matching Proximalized Preference Optimization

The architecture is illustrated in Fig. 1(e).

Preference Optimization in Flow Matching. In flow matching VLAs [1], the action head is a flow-matching model [7, 8]: conditioned on a state $s = (o, l)$ with visual observations o and a language instruction l, a velocity field $v_\theta(a_t, t \mid s)$ indexed by a flow time $t \in [0, 1]$ transports Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$ to an action chunk a along the linear interpolant $a_t = (1-t)\epsilon + ta$, with conditional

velocity $u(a_t \mid a) := a - \epsilon$. Following Flow-DPO [35]—which applies this construction to rectified-flow video generation—and its diffusion-policy precursors [33, 34], we adopt the per-sample form of this regression loss as a tractable proxy for the negative log-likelihood,

$$\ell_\theta(s, a) = \mathbb{E}_{t\sim\mathcal{U}[0,1],\, \epsilon\sim\mathcal{N}(0,I)}\big[\|v_\theta(a_t, t \mid s) - u(a_t \mid a)\|^2\big], \tag{3}$$

which serves as a surrogate for $-\log \pi_\theta(a \mid s)$ up to a constant, yielding the implicit-reward proxy

$$r_\theta(s, a) = \tfrac{\beta}{2}\big(\ell_{\text{ref}}(s, a) - \ell_\theta(s, a)\big), \tag{4}$$

where ℓ_{ref} and ℓ_θ denote the flow matching losses under the reference and current policies.

PRO Loss in Flow Matching. Substituting Eq. (4) into the PRO pairwise objective [22, Appendix D]—DPO's contrastive log-sigmoid plus a proximal regularizer anchored on the hyper-response $\mathcal{H}$—introduces an extra hyper-response reward term $r_\theta(s, \mathcal{H})$, the implicit reward in Eq. (4) evaluated at $\mathcal{H}$. Because the action space is continuous, the two-point set $\{a^w, a^l\}$ has Lebesgue measure zero, so $\pi_\theta(\mathcal{H} \mid s), \pi_{\text{ref}}(\mathcal{H} \mid s) = 1$ and $r_\theta(s, \mathcal{H}) = 0$ in the population limit (proof in Appendix A); we therefore drop it and obtain the PRO loss in the flow-matching setting:

$$\begin{aligned}\mathcal{L}_{\text{PRO}}(\theta) = -\mathbb{E}_{(s,a^w,a^l)\sim\mathcal{D}}\Big[&\underbrace{\log\sigma\big(r_\theta(s, a^w) - r_\theta(s, a^l)\big)}_{\mathcal{L}_{\text{con}}\text{: contrastive optimizer}} \\ &+ \underbrace{\sum_{a\in\{a^w,a^l\}} \tfrac{1}{2}\big[\log\sigma\big(r_\theta(s, a)\big) + \log\sigma\big(-r_\theta(s, a)\big)\big]}_{\mathcal{L}_{\text{reg}}\text{: proximal regularizer}}\Big],\end{aligned} \tag{5}$$

where the regularizer is minimized at $r_\theta(s, a) = 0$ and grows symmetrically with $|r_\theta(s, a)|$, thereby anchoring the absolute magnitude of the implicit reward and preventing the likelihood underdetermination of Eq. (2) from manifesting as reward hacking.

Full RPRO Objective. To preserve base-policy performance and reinforce direct regression toward positive actions a^w, we combine the PRO loss with a supervised fine-tuning term weighted by loss coefficients λ_{PRO} and λ_{SFT}:

$$\mathcal{L}_{\text{RPRO}}(\theta) = \lambda_{\text{PRO}}\mathcal{L}_{\text{PRO}}(\theta) + \lambda_{\text{SFT}}\mathcal{L}_{\text{SFT}}(\theta), \tag{6}$$

where $\mathcal{L}_{\text{SFT}}(\theta) = \mathbb{E}_{(s,a^w)\sim\mathcal{D}}[\ell_\theta(s, a^w)]$ is the flow-matching regression loss on positive actions.

Contrastive Gradient Cancellation. Differentiating Eq. (5) yields (derivation in Appendix B):

$$\begin{aligned}\nabla_\theta\mathcal{L}_{\text{PRO}} = &\underbrace{-\sigma\big(r_\theta(s, a^l) - r_\theta(s, a^w)\big)\big(\nabla_\theta r_\theta(s, a^w) - \nabla_\theta r_\theta(s, a^l)\big)}_{\nabla_\theta\mathcal{L}_{\text{con}}} \\ &+ \underbrace{\sum_{a\in\{a^w,a^l\}} \tfrac{1}{2}(2\sigma(r_\theta(s, a)) - 1)\nabla_\theta r_\theta(s, a)}_{\nabla_\theta\mathcal{L}_{\text{reg}}}.\end{aligned} \tag{7}$$

On identical-pair samples ($a^w = a^l = a$), the contrastive term cancels exactly because $\nabla_\theta r_\theta(s, a^w) - \nabla_\theta r_\theta(s, a^l) = \mathbf{0}$. Only the proximal regularizer survives; with $r = r_\theta(s, a)$ it reduces to $(2\sigma(r) - 1)\nabla_\theta r$, which anchors π_θ near π_{ref}—analogous to the KL penalty in Eq. (1). The effective RPRO gradient therefore simplifies to:

$$\nabla_\theta\mathcal{L}_{\text{RPRO}}\big|_{a^w=a^l} = \lambda_{\text{PRO}}\nabla_\theta\mathcal{L}_{\text{reg}}(\theta) + \lambda_{\text{SFT}}\nabla_\theta\mathcal{L}_{\text{SFT}}(\theta), \tag{8}$$

where $\mathcal{L}_{\text{reg}}$ is the proximal regularizer in Eq. (5). This cancellation makes it safe to route identical-pair samples through the same RPRO loss: the surviving regularizer prevents unconstrained drift from π_{ref}, while λ_{SFT} controls the net step toward the SFT target (Appendix B). This property plays a key role in our data composition (§3.4)

3.3 Human-in-the-Loop Data Collection

We collect preference trajectory pairs (τ^w, τ^l) via a teleoperated intervention-and-rollback pipeline. Although our implementation uses Meta Quest 3 controllers (Appendix G), the pipeline is device-agnostic (e.g., SpaceMouse or leader-follower arms). During rollouts of the current policy, whenever an erroneous or dangerous action is observed, the operator triggers the following sequence:

- **Step 1: Rollback.** The robot rewinds to an earlier state $s_{t-\Delta}$, where the operator-chosen horizon Δ places the robot just before the erroneous behavior begins. The executed segment from $s_{t-\Delta}$ to s_t is recorded as the *negative trajectory* τ^l.
- **Step 2: Scene Restoration.** The system displays the observation image at timestep $t-\Delta$ on the operator's VR headset, enabling the operator to reset the physical scene to $s_{t-\Delta}$.
- **Step 3: Human Correction.** The operator demonstrates the correct behavior from $s_{t-\Delta}$; this corrective demonstration is recorded as the *positive trajectory* τ^w.

This design produces naturally paired trajectories: for the same initial state $s_{t-\Delta}$, we obtain both the policy's erroneous continuation (τ^l) and the human expert's correction (τ^w). These pairs directly serve as preference data for RPRO training. Notably, allowing Δ to vary across interventions naturally diversifies the initial state of each pair, avoiding dependence on a fixed rollback horizon.

3.4 Training Data Composition

Per-state pair construction. The RPRO loss (Eq. 6) requires per-state tuples (s, a^w, a^l). However, because τ^w and τ^l diverge after the shared initial state $s_{t-\Delta}$, each subsequent state belongs to only one trajectory—states on τ^l lack a positive action a^w, and states on τ^w lack a negative action a^l. We address this mismatch via a smooth interpolation strategy that synthesizes the missing counterpart:

- **Case 1: States on the Negative Trajectory.** As illustrated in Fig. 1(d), for a state M on τ^l, we construct a synthetic positive action chunk that moves from M toward the corresponding segment of τ^w. We first locate the closest point M' on τ^w using a weighted distance metric (Appendix C) and take the next H steps on τ^w as the target chunk ending at N'. The synthetic action a^w first bridges from M to a transition point J on this target chunk, then follows τ^w from J to N'. The negative action a^l is the next H steps along τ^l from M, yielding (s_M, a^w, a^l).
- **Case 2: States on the Positive Trajectory.** For a state on τ^w, the positive action a^w is the next H steps along τ^w, and we set $a^l = a^w$. By contrastive gradient cancellation (Eq. 8), only the SFT loss and proximal regularizer remain active, both reinforcing correct behavior.
- **Case 3: SFT Demonstration Data.** For states from $\mathcal{D}_{\text{SFT}}$, we similarly set $a^w = a^l = a_{\text{SFT}}$. As in Case 2, contrastive gradients vanish; the proximal regularizer anchors π_θ toward π_{ref}, rendering these regularized SFT samples.

Case 1 implements this bridge with a cubic Bézier curve for position, Slerp for orientation, and linear interpolation for the gripper. The Bézier curve uses only the arrival tangent from τ^w at J, so the synthetic action merges smoothly into τ^w without inheriting the erroneous direction of τ^l at M (Appendix D). We analyze physical-plausibility safeguards in Appendix E.

The combined dataset takes the following form:

$$\begin{aligned}\mathcal{D} = &\left\{(s, a^w, a^l) \,\middle|\, s \in \tau^l : a^w = \text{Interp}(M \to N'),\ a^l = \tau^l(M \to N)\right\} \\ &\cup \left\{(s, a^w, a^l) \,\middle|\, s \in \tau^w : a^w = a^l = \tau^w(s \to s + H)\right\} \\ &\cup \left\{(s, a^w, a^l) \,\middle|\, s \in \mathcal{D}_{\text{SFT}} : a^w = a^l = a_{\text{SFT}}\right\}. \end{aligned} \tag{9}$$

Batch mixing across iterations. Let $\mathcal{D}^k_{\text{pref}}$ be the preference dataset built via Smooth Interpolation in RPRO iteration k, and $\mathcal{D}^{<k}_{\text{pref}} \triangleq \bigcup_{j<k} \mathcal{D}^j_{\text{pref}}$ the historical preference pool. We keep these three sources in separate buffers and form each mini-batch $\mathcal{B}$ by drawing samples from them at fixed proportions:

$$k{=}1:\ 80\%\text{ from }\mathcal{D}^k_{\text{pref}},\ 20\%\text{ from }\mathcal{D}_{\text{SFT}};\quad k{\geq}2:\ 70\%\text{ from }\mathcal{D}^k_{\text{pref}},\ 15\%\text{ from }\mathcal{D}^{<k}_{\text{pref}},\ 15\%\text{ from }\mathcal{D}_{\text{SFT}}.$$

This fixed-ratio schedule balances three goals. The current preference set $\mathcal{D}^k_{\text{pref}}$ receives the largest share because it contains the newest failure states and thus the most targeted corrective signal. The historical pool $\mathcal{D}^{<k}_{\text{pref}}$ is replayed from $k{\geq}2$ onward to prevent regression on previously corrected failures. The SFT share preserves base-policy capabilities and mitigates catastrophic forgetting.

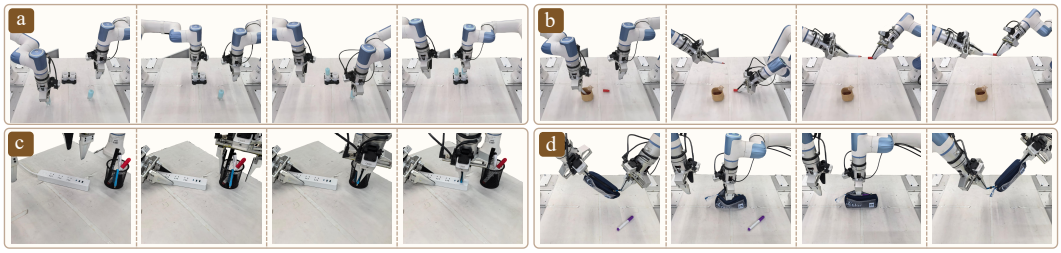

Figure 2: **Experimental setup for the four real-robot tasks: (a)** cosmetic packaging (PACK), **(b)** pen-cap assembly (CAP), **(c)** USB insertion (USB), and **(d)** pencil-case packing (CASE). Due to space constraints, this figure shows only a simplified overview; the complete per-stage pipeline for each task is provided in Fig. 7 of Appendix H.

4 Experimental Evaluation

We evaluate FlowPRO along four axes: **(Q1)** *Overall performance:* Does FlowPRO surpass representative baselines from both the *positive-only* and the *positive-and-negative* regimes? **(Q2)** *Data construction:* Does state-wise *Smooth Interpolation* outperform trajectory-wise preference contrast? **(Q3)** *Loss design:* Does the proximal regularizer in RPRO eliminate the reward-hacking failure mode of plain Flow-DPO? **(Q4)** *Component attribution:* How do the SFT term and the proximal regularizer individually contribute to the full RPRO objective?

4.1 Experimental Setup

Hardware, Tasks, and Protocol. *Platform.* All experiments are conducted on a Dobot XTrainer bimanual platform (Appendix G). *Tasks.* We evaluate on four long-horizon bimanual tasks (Fig. 2, Appendix H): PACK (sub-cm insertion), CAP (in-air coordination), USB (sub-mm precision), and CASE (deformable long-horizon packing). *Training.* We first train a base SFT policy on $\mathcal{D}_{\text{SFT}}$, which serves as the reference policy π_{ref}, and then perform $K = 3$ rounds of RPRO fine-tuning on $\{\mathcal{D}^k_{\text{pref}}\}_{k=1}^{K}$. *Evaluation.* To control for training-time stochasticity, each entry in Table 1 is obtained from 3 independent training seeds, and we report the cross-seed mean $\pm$ standard deviation, where each per-seed success rate is computed over n=100 rollouts with randomized initial placements.

Baselines and Ablations. *Baselines (Q1, Q2).* We compare FlowPRO against four representative comparators covering both the *positive-only* and the *positive-and-negative* regimes: **DAgger** [10] (dataset aggregation), **DAgger-Buffered** (a positive-only control that isolates the preference loss from our batch-composition schedule), **PI0.6*** [17] (advantage conditioning), and **TPO** [20] (trajectory-wise preference optimization). All comparators start from the same SFT checkpoint and follow the same iterative data-collection protocol. *Ablations (Q3, Q4).* We further ablate our loss along the path from a preference-free objective to the full design: **SFT**, **DPO** (Eq. 2 applied to flow matching, without the SFT term or the proximal regularizer), **DPO+SFT**, **PRO** (RPRO without the SFT term, i.e., $\mathcal{L}_{\text{PRO}}$ alone), and **RPRO** (the full objective in Eq. 6). To manage real-robot cost, ablations are run *only on* PACK (which exhibits all representative failure modes) and evaluated after a single round of fine-tuning. We defer the detailed baseline descriptions and the exact loss expressions of every variant to Appendices I and J.

4.2 Experimental Results

Final and per-iteration success rates on PACK, CAP, USB, and CASE are reported in Table 1 and Fig. 3, and loss-component ablations of RPRO are shown in Fig. 4. RPRO strictly dominates each baseline across all 8 task–base strata, with a Bonferroni-corrected one-sided Cochran–Mantel–Haenszel (CMH) test yielding $p < 10^{-3}$ against each of DAgger, DAgger-Buffered, PI0.6*, and TPO (Appendix L). RPRO likewise strictly dominates each loss-component variant on PACK under in-distribution (ID) and out-of-distribution (OOD) initial conditions, with the same CMH test yielding $p \leq 6 \times 10^{-3}$ against each of PRO, SFT, DPO+SFT, and DPO (Appendix L.4). We discuss the findings below by question.

Comparison with baselines (Q1 & Q2).

Table 1: **Final success rate and completion time after K=3 rounds of iterative post-training on four real-robot bimanual tasks.** SR (↑, %) is reported as mean ± std (in pp) across 3 training seeds, with each per-seed SR computed over n=100 randomized rollouts; CT (↓, s) is reported as the mean over the same rollouts.

Base Policy	Fine-tune	PACK		CAP		USB		CASE	
		SR	CT	SR	CT	SR	CT	SR	CT
PI0	Dagger	88±1.3%	32s	83±3.0%	31s	80±2.8%	35s	83±2.6%	58s
	Dagger-Buffer	90±2.0%	27s	89±2.4%	29s	85±1.6%	33s	89±2.2%	57s
	PI0.6*	94±1.5%	25s	91±1.8%	28s	90±3.2%	27s	90±1.4%	46s
	TPO	92±1.7%	28s	94±2.2%	27s	87±1.7%	30s	88±2.3%	54s
	RPRO	**99±1.0%**	**20s**	**98±1.5%**	**24s**	**92±1.5%**	**26s**	**93±2.0%**	**42s**
PI0.5	Dagger	86±2.5%	30s	86±2.3%	31s	82±3.4%	28s	85±3.3%	57s
	Dagger-Buffer	93±1.3%	28s	92±1.5%	28s	89±2.1%	25s	87±2.4%	55s
	PI0.6*	92±2.8%	23s	94±1.3%	27s	93±1.9%	26s	88±2.7%	43s
	TPO	95±1.6%	24s	93±2.8%	26s	91±2.4%	27s	90±1.5%	45s
	RPRO	**99±1.0%**	**17s**	**99±1.0%**	**23s**	**95±1.6%**	**24s**	**93±1.2%**	**38s**

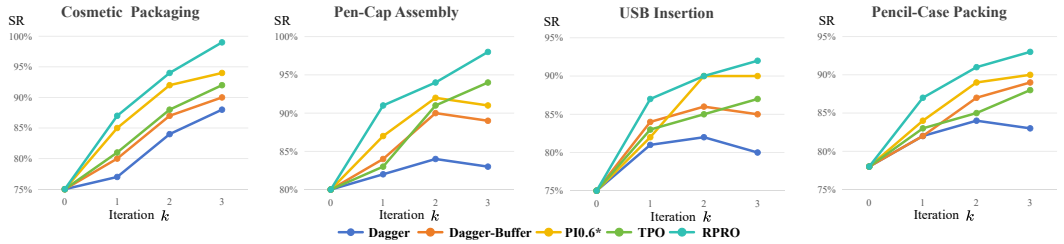

Figure 3: **Per-iteration success rates (SR) on the four real-robot tasks, with PI0.5 as the base policy.** Iteration 0 corresponds to the shared SFT checkpoint. The corresponding curves with PI0 as the base policy are reported in Fig. 8 of Appendix I.

RPRO vs. TPO. Our state-wise method outperforms the trajectory-wise TPO variant by 3–7 pp on every stratum (Table 1), directly addressing Q2 (state-wise vs. trajectory-wise contrast). Trajectory-level contrast only constrains the sum of per-step log-ratios, so the per-state learning signal is diluted by the trajectory length; in contrast, our per-state Smooth Interpolation gives every training state a well-defined pairwise signal at action-chunk granularity. Trajectory-level contrast is also memory-intensive: keeping a full trajectory in memory forces sub-sampling of states on τ^w and τ^l, so the positive trajectory is only partially learned.

RPRO vs. PI0.6.* RPRO outperforms the advantage-conditioned PI0.6* baseline by 2–7 pp on *identical* preference data, isolating *conditioning*-based from *contrastive-loss*-based use of the signal. PI0.6* relies on the model to discover the "improved"/"unimproved" partition from a single conditioning token under a pure regression objective—an indirect pressure that can be diluted by the rest of the VLM context—whereas RPRO injects the preference signal directly into the action-generation loss, pushing π_θ toward a^w and away from a^l per state and per chunk.

RPRO vs. DAgger & DAgger-Buffered. Both DAgger variants rely on positive samples only, while RPRO additionally exploits negative trajectories through contrastive learning, yielding a stronger learning signal and an 8–15 pp gain over vanilla DAgger across strata. Notably, DAgger-Buffered alone outperforms vanilla DAgger by 2–7 pp, confirming that our batch-composition schedule is a more sample-efficient way to consume the round-k corrections than merging them uniformly into $\mathcal{D}_{\text{SFT}}$: corrections form a small but informative set of states where the current policy fails, and therefore deserve heavier per-batch weighting.

Loss-component ablations (Q3 & Q4).

RPRO vs. PRO (ablates $\mathcal{L}_{\text{SFT}}$). Adding the SFT term to PRO yields a clear performance improvement, for two reasons. *First*, the proximal regularizer prevents collapse but does not actively pull π_θ toward a^w beyond maintaining a positive log-ratio gap $r^w > r^l$; the SFT term fills this gap by

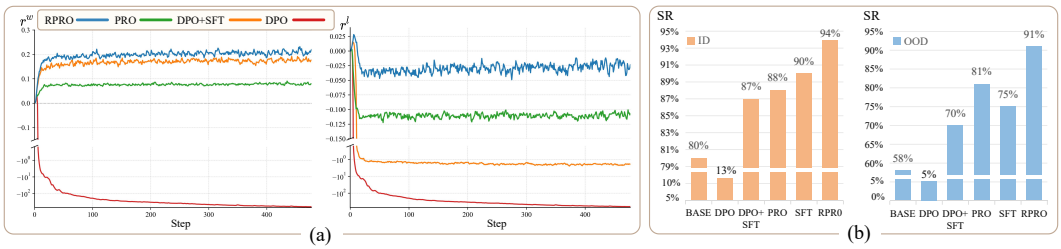

Figure 4: **Loss-component ablations of RPRO. (a)** Implicit-reward dynamics during training: per-step rewards on positive actions r^w (*left*) and negative actions r^l (*right*) for RPRO, PRO, DPO+SFT, and DPO. **(b)** Task success rates under in-distribution (*left*) and out-of-distribution (*right*) initial conditions across loss-component variants. Here, *out-of-distribution (OOD)* refers to object initial positions sampled from spatial regions that were never visited during SFT training.

explicitly fitting π_θ to a^w in the flow-matching regression sense, accelerating convergence. *Second*, the SFT term continually reinforces the base SFT skills, mitigating catastrophic forgetting. SR drops from $94\%/91\%$ to $88\%/81\%$ (ID/OOD).

RPRO vs. SFT (ablates $\mathcal{L}_{\text{PRO}}$). Adding the contrastive term in $\mathcal{L}_{\text{PRO}}$ to SFT clearly improves performance, especially under OOD initial conditions. SFT alone only learns to *imitate* positive actions and never observes what *not* to do, whereas RPRO exposes the policy to both positive and negative actions, providing an explicit *push-away* gradient from a^l that pulls the policy back from nearby failure modes; the OOD-stratum gap is the dominant driver of RPRO's overall gain over SFT. SR drops to $90\%/75\%$.

RPRO vs. DPO+SFT (ablates $\mathcal{L}_{\text{reg}}$). This comparison isolates the proximal regularizer $\mathcal{L}_{\text{reg}}$, with the SFT anchor fixed in both variants. Since the SFT term is *asymmetric*—anchoring π_θ only at a^w—DPO's contrastive optimizer drives r^l unchecked into an unbounded regime, while the SFT pull keeps r^w deceptively high (Fig. 4(a)). Such *unilateral runaway* on the rejected side, even when the margin $r^w - r^l$ remains positive, contaminates the action distribution π_θ rolls out at inference and degrades execution. SR drops to $87\%/70\%$.

RPRO vs. DPO (ablates $\mathcal{L}_{\text{reg}} + \mathcal{L}_{\text{SFT}}$). Removing both the SFT anchor and the proximal regularizer turns the loss into plain DPO, which exhibits the reward-hacking failure mode in its most extreme form. In Fig. 4(a), r^w itself drives below zero and continues to fall to the -10^2 scale—i.e., π_θ ends up *farther* from a^w than the reference policy in absolute terms, even though the relative gap $r^w - r^l$ stays positive throughout training. RPRO's proximal regularizer eliminates this pathology by anchoring the absolute reward levels of both a^w and a^l around π_{ref}. SR collapses to $13\%/5\%$.

5 Conclusion

In this paper, we presented **FlowPRO**, a reward-free offline reinforced fine-tuning framework for flow-matching VLAs, built around the **RPRO** loss and a teleoperated intervention-and-rollback paradigm for preference data construction. Across four long-horizon bimanual real-robot tasks and two π_0-family base policies, FlowPRO consistently attains the highest success rate and the shortest completion time, outperforming all four representative baselines and every loss-component ablation; diagnostic implicit-reward curves further show that the proximal regularizer cleanly removes the reward-hacking failure mode exhibited by Flow-DPO. These results indicate that FlowPRO is an effective and reliable post-training recipe for flow-matching VLA policies. With only a small number of human interventions, it converts scarce on-robot corrections into deployment-grade policy improvements.

6 Limitations and Future Work

Our study has several limitations, each pointing to a concrete extension. First, our evaluation is restricted to a single bimanual platform; broader generalization to mobile and dexterous manipulation is left as future work. Second, deciding *when* to roll back and *how far* (the rollback horizon Δ) currently relies on a human operator. A natural extension is a learned failure-detector that triggers rollback autonomously, leaving the human only the correction.

References

[1] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[2] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. pages 2165–2183, 2023.

[3] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, et al. OpenVLA: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[4] O. M. Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

[5] P. Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

[6] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu. RDT-1B: a diffusion foundation model for bimanual manipulation. 2025:29982–30009, 2025.

[7] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

[8] X. Liu, C. Gong, and Q. Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

[9] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

[10] S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. pages 627–635, 2011.

[11] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.

[12] P. F. Christiano, J. Leike, T. Brown, M. Martic, S. Legg, and D. Amodei. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*, 30, 2017.

[13] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. 2017.

[14] G. Lu, W. Guo, C. Zhang, Y. Zhou, H. Jiang, Z. Gao, Y. Tang, and Z. Wang. VLA-RL: Towards masterful and general robotic manipulation with scalable reinforcement learning. *arXiv preprint arXiv:2505.18719*, 2025.

[15] Y. Guo, J. Zhang, X. Chen, X. Ji, Y.-J. Wang, Y. Hu, and J. Chen. Improving vision-language-action model with online reinforcement learning. pages 15665–15672, 2025.

[16] A. Ren, J. Lidard, L. Ankile, A. Simeonov, P. Agrawal, A. Majumdar, B. Burchfiel, H. Dai, and M. Simchowitz. Diffusion policy policy optimization. 2025:77288–77329, 2025.

[17] P. Intelligence, A. Amin, R. Aniceto, A. Balakrishna, K. Black, K. Conley, G. Connors, J. Darpinian, K. Dhabalia, J. DiCarlo, et al. $\pi^*_{0.6}$: a VLA that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025.

[18] Y. Bai, A. Jones, K. Ndousse, A. Askell, A. Chen, N. DasSarma, D. Drain, S. Fort, D. Ganguli, T. Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

[19] S. Levine, A. Kumar, G. Tucker, and J. Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

[20] Z. Zhang, K. Zheng, Z. Chen, J. Jang, Y. Li, S. Han, C. Wang, M. Ding, D. Fox, and H. Yao. GRAPE: Generalizing robot policy via preference alignment. *arXiv preprint arXiv:2411.19309*, 2024.

[21] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

[22] K. Guo, Y. Li, and Z. Chen. Proximalized preference optimization for diverse feedback types: A decomposed perspective on dpo. *Advances in Neural Information Processing Systems*, 38: 94533–94576, 2026.

[23] M. Kelly, C. Sidrane, K. Driggs-Campbell, and M. J. Kochenderfer. HG-DAgger: Interactive imitation learning with human experts. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8077–8083. IEEE, 2019.

[24] I. Kostrikov, A. Nair, and S. Levine. Offline reinforcement learning with implicit Q-learning. 2021.

[25] M. Nakamoto, S. Zhai, A. Singh, M. Sobol Mark, Y. Ma, C. Finn, A. Kumar, and S. Levine. Cal-QL: Calibrated offline RL pre-training for efficient online fine-tuning. volume 36, pages 62244–62269, 2023.

[26] K. Black, M. Janner, Y. Du, I. Kostrikov, and S. Levine. Training diffusion models with reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pages 4965–4987, 2024.

[27] B. Zhang, Y. Zhang, J. Ji, Y. Lei, J. Dai, Y. Chen, and Y. Yang. SafeVLA: Towards safety alignment of vision-language-action model via constrained learning. *Advances in Neural Information Processing Systems*, 38:153335–153373, 2026.

[28] K. Ethayarajh, W. Xu, N. Muennighoff, D. Jurafsky, and D. Kiela. KTO: Model alignment as prospect theoretic optimization. *arXiv preprint arXiv:2402.01306*, 2024.

[29] M. G. Azar, M. Rowland, B. Piot, D. Guo, D. Calandriello, M. Valko, and R. Munos. A general theoretical paradigm to understand learning from human preferences. *arXiv preprint arXiv:2310.12036*, 2024.

[30] Y. Meng, M. Xia, and D. Chen. SimPO: Simple preference optimization with a reference-free reward. volume 37, pages 124198–124235, 2024.

[31] W. Xiong, H. Dong, C. Ye, Z. Wang, H. Zhong, H. Ji, N. Jiang, and T. Zhang. Iterative preference learning from human feedback: Bridging theory and practice for RLHF under KL-constraint. 2023.

[32] J. Hejna, R. Rafailov, H. Sikchi, C. Finn, S. Niekum, W. B. Knox, and D. Sadigh. Contrastive preference learning: Learning from human feedback without reinforcement learning. In *International Conference on Learning Representations*, volume 2024, pages 18770–18798, 2024.

[33] B. Wallace, M. Dang, R. Rafailov, L. Zhou, A. Lou, S. Purushwalkam, S. Ermon, C. Xiong, S. Joty, and N. Naik. Diffusion model alignment using direct preference optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8228–8238, 2024.

[34] K. Yang, J. Tao, J. Lyu, C. Ge, J. Chen, W. Shen, X. Zhu, and X. Li. Using human feedback to fine-tune diffusion models without any reward model. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8941–8951, 2024.

[35] J. Liu, G. Liu, J. Liang, Z. Yuan, X. Liu, M. Zheng, X. Wu, Q. Wang, W. Qin, M. Xia, et al. Improving video generation with human feedback. *arXiv preprint arXiv:2501.13918*, 2025.

[36] R. A. Bradley and M. E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952.

[37] J. Robins, S. Greenland, and N. E. BRESLOW. A general estimator for the variance of the Mantel–Haenszel odds ratio. *American journal of epidemiology*, 124(5):719–723, 1986.

[38] W. G. Cochran. Some methods for strengthening the common χ^2 tests. *Biometrics*, 10(4): 417–451, 1954.

[39] N. Mantel and W. Haenszel. Statistical aspects of the analysis of data from retrospective studies of disease. *Journal of the national cancer institute*, 22(4):719–748, 1959.

A Asymptotic Vanishing of the Hyper-Response Reward in Continuous Action Spaces

This appendix provides the formal asymptotic argument supporting the simplification $r_\theta(s, \mathcal{H}) \approx 0$ used in Eq. (5). In the original PRO derivation [22], the hyper-response $\mathcal{H}$ aggregates all responses not in the observed preference pair, and is treated under the discrete vocabulary of an LLM. Here we restate the argument under the continuous-action regime of flow-matching VLAs and show that, in this regime, the hyper-response reward is not merely small but *vanishes exactly* in the population limit.

Setup and notation. Fix a state $s = (o, l)$ and let $\mathcal{A} \subseteq \mathbb{R}^d$ denote the (continuous) action-chunk space. We treat $\pi_\theta(\cdot \mid s)$ and $\pi_{\text{ref}}(\cdot \mid s)$ as absolutely-continuous probability measures on $\mathcal{A}$ with densities $p_\theta(a \mid s)$ and $p_{\text{ref}}(a \mid s)$ w.r.t. the Lebesgue measure μ. For an observed preference pair $(a^w, a^l) \in \mathcal{A}^2$ at state s, define the *hyper-response set*

$$\mathcal{H} \triangleq \mathcal{A} \setminus \{a^w, a^l\}, \tag{10}$$

i.e., the (uncountable) set of all action chunks at state s other than the two observed responses. Following PRO [22, Appendix D], the hyper-response reward is

$$r_\theta(s, \mathcal{H}) \triangleq \beta \log \frac{\pi_\theta(\mathcal{H} \mid s)}{\pi_{\text{ref}}(\mathcal{H} \mid s)}, \tag{11}$$

which is the analogue of $r_\theta(s, a) = \beta \log \frac{\pi_\theta(a|s)}{\pi_{\text{ref}}(a|s)}$ but evaluated on a set rather than a singleton (note that we use β rather than $\frac{\beta}{2}$ here to match the LLM-PRO convention; the flow-matching surrogate Eq. 4 carries the same $\frac{\beta}{2}$ scaling on both numerator and denominator, so this choice is immaterial for the present argument).

Proposition (exact vanishing). Let $\pi_\theta(\cdot \mid s)$ and $\pi_{\text{ref}}(\cdot \mid s)$ be absolutely continuous w.r.t. Lebesgue measure on $\mathcal{A}$. Then for any finite preference pair $(a^w, a^l) \in \mathcal{A}^2$,

$$\pi_\theta(\mathcal{H} \mid s) = \pi_{\text{ref}}(\mathcal{H} \mid s) = 1, \qquad \text{hence} \qquad r_\theta(s, \mathcal{H}) = 0. \tag{12}$$

Proof. The complement of $\mathcal{H}$ in $\mathcal{A}$ is the two-point set $\{a^w, a^l\}$, which has Lebesgue measure zero:

$$\mu(\{a^w, a^l\}) = \mu(\{a^w\}) + \mu(\{a^l\}) = 0 + 0 = 0. \tag{13}$$

Since $\pi_\theta(\cdot \mid s)$ admits a density w.r.t. μ, absolute continuity gives

$$\pi_\theta(\{a^w, a^l\} \mid s) = \int_{\{a^w, a^l\}} p_\theta(a \mid s)\, d\mu(a) = 0, \tag{14}$$

and therefore $\pi_\theta(\mathcal{H} \mid s) = 1 - \pi_\theta(\{a^w, a^l\} \mid s) = 1$. The same argument applies to $\pi_{\text{ref}}(\mathcal{H} \mid s)$. Plugging both into Eq. (11) gives $r_\theta(s, \mathcal{H}) = \beta \log 1 = 0$. □

Why the discrete-action argument is not directly applicable. In the LLM setting of Guo et al. [22], $\mathcal{A}$ is a finite vocabulary $\mathcal{V}^L$ of token sequences and $\pi_\theta(\cdot \mid s)$ is a probability *mass* function. There, $\pi_\theta(\{a^w, a^l\} \mid s) = p_\theta(a^w \mid s) + p_\theta(a^l \mid s)$ is generally *nonzero*, so $\pi_\theta(\mathcal{H} \mid s) = 1 - p_\theta(a^w \mid s) - p_\theta(a^l \mid s)$. The approximation $r_\theta(s, \mathcal{H}) \approx 0$ in PRO [22] relies on the heuristic that $p_\theta(a^w \mid s), p_\theta(a^l \mid s) \ll 1$ when the vocabulary is large, so $\pi_\theta(\mathcal{H} \mid s) \to 1$ as $|\mathcal{V}^L| \to \infty$. The continuous-action regime is qualitatively stronger: *individual* actions carry zero probability mass under any absolutely-continuous density, so the simplification $r_\theta(s, \mathcal{H}) = 0$ is *exact* rather than asymptotic. The flow-matching architecture of π_0 [1], which generates continuous action chunks via a velocity-field ODE, satisfies this absolute-continuity assumption by construction whenever the noise prior $\mathcal{N}(0, I)$ is full-rank—which it is.

Robustness to finite-precision and stochastic-sampler effects. A pedantic objection is that in practice π_θ is realised by a stochastic flow-matching sampler in finite floating-point precision, so the realised distribution is technically supported on a finite grid. In this case the proposition above does not apply verbatim. We address this in two ways. (i) The grid spacing δ induced by floating-point precision is $\sim 10^{-7}$ in fp32; for an action chunk of dimension $d \cdot H$ (e.g., $d = 10$ features per arm—xyz, 6D rotation, and gripper width—times 2 arms, $H = 50$ steps, so dim $= 20 \cdot 50 = 1000$ for bimanual chunks), the probability mass concentrated on the singleton $\{a^w\}$ relative to the support of π_θ scales as δ^{dim}, which is vanishingly small. (ii) More importantly, in the training loss we never need to evaluate $\pi_\theta(\{a^w\} \mid s)$ directly; we only need $\ell_\theta(s, a^w)$, the flow-matching loss surrogate (see §3.2). The surrogate is a continuous functional that is well-defined regardless of whether we view π_θ as truly continuous or as a fine-grid approximation. Consequently the argument carries over with quantitative slack $|r_\theta(s, \mathcal{H})| \leq O(\delta^{\text{dim}})$, which is many orders of magnitude below the typical implicit-reward scales we observe in Fig. 4(a).

Summary. Under the absolute-continuity assumption that holds by construction for flow-matching VLA heads with a full-rank Gaussian prior, the hyper-response reward $r_\theta(s, \mathcal{H})$ is *identically zero*, not merely $\ll 1$. This sharpens the LLM-PRO heuristic and justifies dropping the $r_\theta(s, \mathcal{H})$ term from the contrastive-plus-regularizer objective in Eq. (5). The remaining loss is exact—not approximate—under the stated assumption.

B Derivation of the Gradient Property for Identical Pairs

We derive the gradient of $\mathcal{L}_{\text{PRO}}$ (Eq. 5) and analyze its behavior when $a^w = a^l$.

Step 1: General gradient. The PRO loss is:

$$\mathcal{L}_{\text{PRO}} = -\mathbb{E}_{(s,a^w,a^l)}\Big[\log\sigma\big(r_\theta(s,a^w) - r_\theta(s,a^l)\big) + \sum_{a\in\{a^w,a^l\}}\Big(\tfrac{1}{2}\log\sigma\big(r_\theta(s,a)\big) + \tfrac{1}{2}\log\sigma\big(-r_\theta(s,a)\big)\Big)\Big]. \tag{15}$$

Using $\nabla_x \log\sigma(x) = \sigma(-x) = 1 - \sigma(x)$ and the outer negative sign, we differentiate each term with respect to θ:

Contrastive term. Let $\Delta = r_\theta(s,a^w) - r_\theta(s,a^l)$:

$$-\nabla_\theta \log\sigma(\Delta) = -\sigma(-\Delta)\cdot(\nabla_\theta r_\theta(s,a^w) - \nabla_\theta r_\theta(s,a^l)). \tag{16}$$

Regularizer term. For each $a \in \{a^w, a^l\}$, let $r = r_\theta(s,a)$:

$$\begin{aligned}-\nabla_\theta\Big[\tfrac{1}{2}\log\sigma(r) + \tfrac{1}{2}\log\sigma(-r)\Big] &= -\tfrac{1}{2}(1-2\sigma(r))\cdot\nabla_\theta r \\ &= \tfrac{1}{2}(2\sigma(r)-1)\cdot\nabla_\theta r.\end{aligned} \tag{17}$$

Combining the contrastive and regularizer contributions yields the full gradient:

$$\nabla_\theta\mathcal{L}_{\text{PRO}} = \underbrace{-\sigma\big(r_\theta(s,a^l) - r_\theta(s,a^w)\big)\cdot\big(\nabla_\theta r_\theta(s,a^w) - \nabla_\theta r_\theta(s,a^l)\big)}_{\text{contrastive optimizer}} + \underbrace{\sum_{a\in\{a^w,a^l\}}\tfrac{1}{2}(2\sigma(r_\theta(s,a))-1)\cdot\nabla_\theta r_\theta(s,a)}_{\text{proximal regularizer}}. \tag{18}$$

Step 2: Specialization to $a^w = a^l = a$. When $a^w = a^l = a$, we have $r_\theta(s,a^w) = r_\theta(s,a^l) \triangleq r$, so $\Delta = 0$.

Contrastive term: $\nabla_\theta r_\theta(s,a^w) - \nabla_\theta r_\theta(s,a^l) = \mathbf{0}$, so this term vanishes *automatically*, regardless of the prefactor $-\sigma(0) = -\frac{1}{2}$—no indicator function or sample filtering is required.

Regularizer terms: The two terms (for a^w and a^l) are identical, each equal to $\frac{1}{2}(2\sigma(r)-1)\cdot\nabla_\theta r$. They sum to $(2\sigma(r)-1)\cdot\nabla_\theta r$, which acts as a safeguard preventing $|r|$ from growing unbounded (equivalently, preventing π_θ from drifting too far away from π_{ref})—playing the role of a trust-region / KL-type penalty analogous to the regularizers used in RLHF.

Step 3: The surviving regularizer as a trust-region anchor. The residual regularizer gradient $(2\sigma(r)-1)\cdot\nabla_\theta r$ plays the exact role of a trust-region / KL-type penalty. Recall $r = r_\theta(s,a) = \frac{\beta}{2}(\ell_{\text{ref}}(s,a) - \ell_\theta(s,a))$, so by the chain rule

$$\nabla_\theta r_\theta(s,a) = -\tfrac{\beta}{2}\,\nabla_\theta \ell_\theta(s,a), \tag{19}$$

which is, up to the positive scalar $\frac{\beta}{2}$, the *negative* SFT gradient on (s,a). Substituting into the regularizer gradient gives

$$(2\sigma(r)-1)\cdot\nabla_\theta r_\theta(s,a) \;=\; -\tfrac{\beta}{2}(2\sigma(r)-1)\cdot\nabla_\theta \ell_\theta(s,a). \tag{20}$$

Two useful consequences follow.

(a) Controlled opposition to SFT—the hallmark of a trust-region penalty. The SFT term on (s,a) performs gradient descent on $\ell_\theta(s,a)$, i.e. it updates θ along $-\nabla_\theta \ell_\theta(s,a)$ to make π_θ better reproduce a. The regularizer's descent direction, by Eq. (20), is

$$-\nabla_\theta \mathcal{L}_{\text{reg}} \;=\; -\tfrac{1}{2}(2\sigma(r)-1)\,\nabla_\theta r_\theta(s,a) \;=\; +\tfrac{\beta}{4}(2\sigma(r)-1)\,\nabla_\theta \ell_\theta(s,a). \tag{21}$$

From Eq. (21), the sign of the regularizer's descent step relative to the SFT descent step $-\nabla_\theta \ell_\theta(s,a)$ is determined entirely by the sign of r, and the two regimes have to be analyzed separately.

- *Case* $r > 0$ (i.e. π_θ has reached a *higher* likelihood for a than π_{ref}, $\ell_\theta(s,a) < \ell_{\text{ref}}(s,a)$, so $2\sigma(r)-1 > 0$). The regularizer's descent step is a positive multiple of $+\nabla_\theta \ell_\theta(s,a)$, i.e. *anti-parallel* to the SFT descent step. It acts as a mild brake, pulling $\ell_\theta(s,a)$ back up and preventing $\pi_\theta(a \mid s)$ from running too far above $\pi_{\text{ref}}(a \mid s)$.
- *Case* $r < 0$ (i.e. π_θ has a *lower* likelihood for a than π_{ref}, $\ell_\theta(s,a) > \ell_{\text{ref}}(s,a)$, so $2\sigma(r)-1 < 0$). The regularizer's descent step is now a positive multiple of $-\nabla_\theta \ell_\theta(s,a)$, i.e. *parallel* to the SFT descent step: both terms drive $\ell_\theta(s,a)$ down. Crucially, this is not a failure of regularization. It is simply a way to ensure that the policy does not drift too far away from the reference policy.

What makes this anchoring safe rather than destructive is that it is (i) *state-adaptive and silent at the reference*: it activates only as $|r|$ grows, and exactly vanishes at $r = 0$; and (ii) *sub-dominant to SFT in magnitude*, as made precise in (b) below. So long as the SFT term is weighted strongly enough (which is easily ensured by choosing $\lambda_{\text{SFT}} > \lambda_{\text{PRO}} \cdot \frac{\beta}{4}$ in Eq. 6), the net update on identical-pair samples is dominated by the SFT direction, while the regularizer supplies a bounded correction that keeps the trajectory of $\pi_\theta(a \mid s)$ from straying excessively far from $\pi_{\text{ref}}(a \mid s)$.

(b) Bounded magnitude. Since $|2\sigma(r)-1| \in [0,1)$, the regularizer contribution is strictly bounded by a $\frac{\beta}{2}$-rescaled version of the SFT gradient norm; in particular it is small near the reference (small $|r|$) and grows only as much as needed to counteract deviation. This is exactly the behavior of a trust-region / KL penalty: it is quiescent inside the trust region and activates smoothly as the policy drifts.

Takeaway: a deliberate design advantage, not a residual artifact. Taken together, Steps 1–3 show that for $a^w = a^l$ the FlowPRO objective specializes—without any external gating, indicator function, or sample filtering—to the SFT loss plus a standard trust-region-style regularizer whose magnitude is bounded by $|2\sigma(r)-1| \in [0,1)$ and whose direction is, as required of any genuine regularizer, oriented toward π_{ref} in proportion to the deviation: it opposes SFT when π_θ has overshot π_{ref} ($r > 0$) and assists SFT when π_θ still trails π_{ref} ($r < 0$), exactly switching off at $r = 0$. This supports the three properties asserted in the main text (§3.2, Eq. 8): *unified treatment* of contrastive preference pairs and SFT-like samples under a single loss; a *trust-region safeguard* that prevents every gradient step—contrastive or not—from straying too far from π_{ref}, which is especially valuable

in the offline RL regime where unconstrained SFT updates on scarce corrections would otherwise drift the policy off the data manifold; and *controlled interaction with SFT*, as formalized by Eqs. (20) and (21): the regularizer is one-sided about π_{ref}, state-adaptive, and bounded, so SFT demonstrations and positive-trajectory states can be fed through the *same* FlowPRO objective as genuine preference pairs without destabilizing optimization.

C Trajectory Point Distance Metric

To find the closest point M' on the positive trajectory τ^w for a given state M on the negative trajectory τ^l, we define a weighted distance metric that jointly considers end-effector position, orientation, and gripper state:

$$d(M, M') = \|\boldsymbol{p}_M - \boldsymbol{p}_{M'}\|_2 + 0.5 \cdot d_{\text{geo}}(\boldsymbol{R}_M, \boldsymbol{R}_{M'}) + 0.2 \cdot |g_M - g_{M'}|, \tag{22}$$

where $\boldsymbol{p} \in \mathbb{R}^3$ denotes the end-effector position, $d_{\text{geo}}(\boldsymbol{R}_M, \boldsymbol{R}_{M'}) = \arccos\big(\frac{\text{tr}(\boldsymbol{R}_M^\top \boldsymbol{R}_{M'})-1}{2}\big)$ is the geodesic distance between rotation matrices $\boldsymbol{R}_M, \boldsymbol{R}_{M'} \in SO(3)$, and $g \in [0, 1]$ is the normalized gripper width. The weights 0.5 and 0.2 were determined empirically to balance the contribution of each component.

D Smooth Interpolation Algorithm

Given the current end-effector state at point M on τ^l and the target action chunk from point M' to N' on τ^w, we construct the synthetic positive action chunk via a two-phase interpolation strategy. Let $n_{\text{tr}} = \max(2, \lfloor \rho H \rfloor)$ with $\rho = 0.7$, and define the *phase-transition point* J as the point reached by advancing n_{tr} steps from M' along τ^w (i.e., the n_{tr}-th waypoint of the target chunk). The first phase (n_{tr} steps) smoothly transitions from M to J; the second phase ($H - n_{\text{tr}}$ steps) directly follows the target chunk from J to N'. This avoids abrupt jumps while ensuring the synthetic action merges seamlessly into the positive trajectory. The procedure is detailed in Algorithm 1.

The cubic Bezier curve uses only the *terminal* tangent P_2 (derived from the target trajectory) while setting P_1 to the midpoint of P_0 and P_3. This design choice eliminates the influence of the source trajectory's motion direction at M, which may point toward the erroneous region, and ensures that the interpolated path smoothly converges to the positive trajectory without undesired bending.

E Physical Plausibility of Smooth Interpolation

A natural concern about Smooth Interpolation (Case 1 in Sec. 3.4) is whether the cubic-Bézier bridge from $M \in \tau^l$ to $J \in \tau^w$ can synthesize *physically infeasible* motions—for example, in the USB task, a synthetic chunk that geometrically intersects with the USB socket wall. Fig. 5 illustrates the geometric picture on the USB task; we argue below that this risk is suppressed by three complementary properties of our data-collection and interpolation pipeline, and that any residual risk is irrelevant at deployment time.

(i) High sampling rate bounds the per-chunk geometric displacement. All teleoperation data are recorded at 50 Hz, and Smooth Interpolation operates on a single action chunk of length H (a fraction of a second in our experiments). The synthetic bridge therefore spans only the distance the end-effector travels in $\rho H = 0.7H$ steps—typically a few millimeters to a centimeter in our tasks. By construction, the cubic Bézier curve lies in the convex hull of its four control points $\{P_0, \ldots, P_3\}$, all of which are anchored to $M \in \tau^l$, $J \in \tau^w$, or their midpoint and terminal tangent. The synthetic chunk is therefore confined to a thin lens-shaped neighborhood between τ^l and τ^w—the blue region in Fig. 5(a)—and large excursions into obstacles are geometrically ruled out.

(ii) The sampling range of M excludes the dangerous tail of τ^l. The only situation in which τ^l comes close to an obstacle is at its very end: the rollback signal is triggered *precisely* when the operator detects an imminent collision (e.g., the USB head touching the socket rim), so collisions, if they occur at all on τ^l, are confined to the last few steps before the rollback timestamp. To eliminate

Algorithm 1 Smooth Interpolation for Synthetic Positive Action Chunk

Require: Source state $\boldsymbol{s}_M = (\boldsymbol{p}_0, \boldsymbol{R}_0, g_0)$ at point M; target chunk $\{(\boldsymbol{p}_j, \boldsymbol{R}_j, g_j)\}_{j=1}^{H}$ from τ^w; transition ratio $\rho = 0.7$; smoothness $\mu = 0.4$
Ensure: Interpolated action chunk $\boldsymbol{a}^w$ of length H

$n_{\text{tr}} \leftarrow \max(2, \lfloor \rho \cdot H \rfloor)$ {Number of transition steps}
Let J denote the phase-transition point, i.e., the n_{tr}-th step of the target chunk with state $(\boldsymbol{p}_{n_{\text{tr}}}, \boldsymbol{R}_{n_{\text{tr}}}, g_{n_{\text{tr}}})$.
Initialize $\boldsymbol{a}^w \leftarrow$ copy of target chunk
for each arm **do**
  // Phase 1: Smooth transition from M to J (steps $1, \ldots, n_{\text{tr}}$)
  Position (cubic Bezier):
    $P_0 \leftarrow \boldsymbol{p}_0, \quad P_3 \leftarrow \boldsymbol{p}_{n_{\text{tr}}}$
    $P_1 \leftarrow (P_0 + P_3)/2$ {No source tangent to avoid bending}
    Estimate target tangent $\hat{\boldsymbol{d}}$ at J from $\boldsymbol{p}_{n_{\text{tr}}-1}, \boldsymbol{p}_{n_{\text{tr}}+1}$
    $P_2 \leftarrow P_3 - \mu \cdot \|P_3 - P_0\| \cdot \hat{\boldsymbol{d}}$ {End tangent for smooth arrival}
  for $i = 1$ to n_{tr} **do**
    $t \leftarrow (i-1)/(n_{\text{tr}} - 1)$
    $\boldsymbol{a}_i^w[\text{pos}] \leftarrow (1-t)^3 P_0 + 3(1-t)^2 t\, P_1 + 3(1-t) t^2 P_2 + t^3 P_3$
  end for
  Orientation (Slerp):
  for $i = 1$ to n_{tr} **do**
    $t \leftarrow (i-1)/(n_{\text{tr}} - 1)$
    $\boldsymbol{a}_i^w[\text{rot}] \leftarrow \text{Slerp}(\boldsymbol{R}_0, \boldsymbol{R}_{n_{\text{tr}}}, t)$
  end for
  Gripper (linear):
  for $i = 1$ to n_{tr} **do**
    $t \leftarrow (i-1)/(n_{\text{tr}} - 1)$
    $\boldsymbol{a}_i^w[\text{grip}] \leftarrow (1-t) \cdot g_0 + t \cdot g_{n_{\text{tr}}}$
  end for
  // Phase 2: Follow target from J onward (steps $n_{\text{tr}}+1, \ldots, H$)
  Steps $n_{\text{tr}}+1$ through H retain the original target chunk values.
end for
return $\boldsymbol{a}^w$

this case at the source, we restrict the source point M to be drawn from the prefix $[0,\ L - H]$ of τ^l, where L is the length of τ^l and H is the chunk length (cf. the labels L and H on the τ^l side of Fig. 5(a)). This guarantees (a) that the entire negative chunk a^l used in the loss lies strictly inside τ^l (rather than past its end), and (b) that M is at least one chunk away from the potentially-contact-rich terminal segment. The dangerous tail of τ^l is therefore never used as the starting point of an interpolation bridge.

(iii) RPRO ensures the residual-risk states are never visited at deployment. Even in the worst case where M is allowed to land arbitrarily close to the socket rim, the resulting offending chunk is harmless at deployment time, because the policy never reaches the dangerous state. Concretely, consider state ① in Fig. 5(b), which sits right next to the obstacle: the synthetic chunk fired from ① (blue dashed line) would clip into the socket wall. However, at every preceding state ②–⑦ along τ^l before ①, the per-state positive action a^w (blue solid arcs) already steers the policy onto τ^w, so ① is *never reached* at deployment. Empirically, the per-iteration success curves in Fig. 3 confirm that the policy no longer drifts toward the τ^l tail, and across all K=3 rounds and four tasks in Table 1, we observed no rollout failure attributable to interpolation artifacts.

Taken together, (i) the small per-chunk displacement at 50 Hz keeps every synthetic bridge inside the lens-shaped safe region of Fig. 5(a), (ii) the $[0,\ L - H]$ sampling rule for M removes the dangerous tail of τ^l from interpolation at the source, and (iii) RPRO training renders any residual-risk states (such as state ① in Fig. 5(b)) unreachable at deployment time.

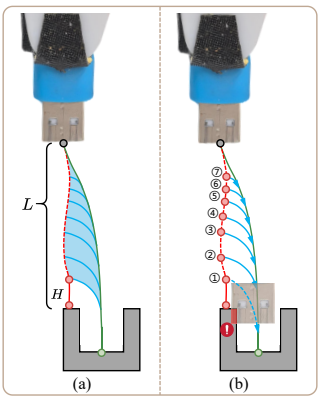

Figure 5: **Why Smooth Interpolation stays physically plausible**, illustrated on the USB task. In both panels: **red dashed** = executed negative rollout τ^l (length L, ending at the socket rim); **green solid** = successful teleoperation τ^w (ending inside the socket); **blue arc** = one synthetic positive chunk of length H, a cubic Bézier bridge from $M \in \tau^l$ to $J \in \tau^w$ followed by direct tracking of τ^w up to N'. **(a) Generic case.** Every synthetic chunk lies in the blue lens-shaped region between τ^l and τ^w, safely away from the socket wall. **(b) Worst case.** Even if state ① (next to the socket rim) gave rise to a wall-clipping chunk (dashed blue arc), the policy never reaches ① at deployment: at every preceding state ②–⑦ along τ^l, the per-state positive action a^w (solid blue arcs) already steers it onto τ^w.

F Iterative Training Procedure

The full iterative training procedure of FlowPRO is given in Algorithm 2.

Algorithm 2 Iterative Training of FlowPRO

Require: SFT dataset $\mathcal{D}_{\text{SFT}}$; reference policy π_{ref}; number of iterations K; loss coefficients $\lambda_{\text{PRO}}, \lambda_{\text{SFT}}$; reward-proxy temperature β; action-chunk horizon H (see App. G)
Ensure: Fine-tuned policy π_θ

Stage 1 (Initial SFT):
Train π_θ on $\mathcal{D}_{\text{SFT}}$ with $\mathcal{L}_{\text{SFT}}$
Stage 2 (Iterative Offline-RL Fine-Tuning):
for $k = 1$ to K **do**
  $\pi_{\text{ref}} \leftarrow \pi_\theta$ {Reference = policy from previous iteration}
  // Data Collection
  Roll out π_θ; human intervenes via teleoperation
  Obtain trajectory pairs (τ_k^w, τ_k^l)
  Construct $\mathcal{D}_{\text{pref}}^k$ via interpolation
  // Training
  for each training step **do**
    // Batch Composition
    if $k = 1$ **then**
      Sample mini-batch $\mathcal{B} = \{(s, a^w, a^l)\}$ with 80% from $\mathcal{D}_{\text{pref}}^k$ and 20% from $\mathcal{D}_{\text{SFT}}$
    else
      Sample mini-batch $\mathcal{B} = \{(s, a^w, a^l)\}$ with 70% from $\mathcal{D}_{\text{pref}}^k$, 15% from $\mathcal{D}_{\text{pref}}^{<k}$, and 15% from $\mathcal{D}_{\text{SFT}}$
    end if
    // Update
    $\mathcal{L} \leftarrow \lambda_{\text{PRO}} \cdot \mathcal{L}_{\text{PRO}}(\theta; \mathcal{B}) + \lambda_{\text{SFT}} \cdot \mathcal{L}_{\text{SFT}}(\theta; \mathcal{B})$
    Update θ via $\nabla_\theta \mathcal{L}$
  end for
end for
return π_θ

G Hardware Platform and Training Setup

G.1 Robot Platform

All real-robot experiments are conducted on a **Dobot XTrainer** bimanual teleoperation platform. The platform is equipped with:

- Two 6-DoF robot arms with parallel-jaw grippers (left and right);
- Three RGB cameras: one top-down global camera, and one wrist camera on each arm;
- A Meta Quest 3 headset used by the human operator for both VR teleoperation during SFT/correction data collection and for the intervention-and-rollback interface described in §3.3.

A photograph of the platform with all of the above components labeled in situ is shown in Fig. 6.

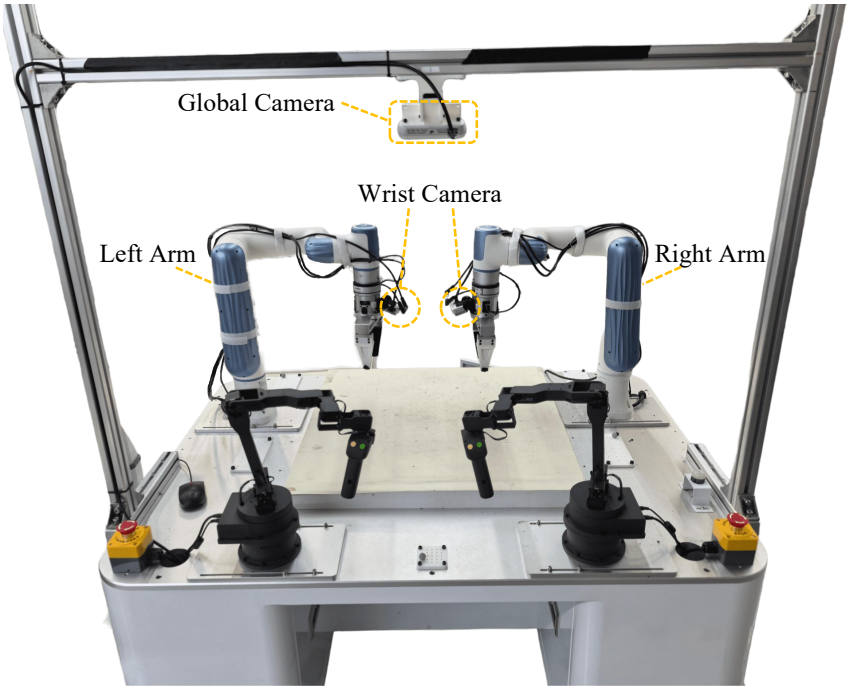

Figure 6: **Hardware platform used for all real-robot experiments.** The Dobot XTrainer bimanual setup includes two 6-DoF arms with parallel-jaw grippers, a top-down global RGB camera, two wrist-mounted RGB cameras (one per arm).

G.2 Training Setup

This subsection details the compute resources, data scale, and all training hyper-parameters used to produce every model evaluated in the main paper. Both the SFT base policy and all RPRO iterations are trained on the same compute node and share the same image and proprioception pipeline.

Compute and precision. All training runs are executed on a single node with $4\times$NVIDIA H20 GPUs (96 GB each) under `bfloat16` mixed precision, with a peak per-GPU memory footprint of approximately 90 GB. Inference of the flow-matching action expert uses 10 denoising steps per chunk.

Action representation. The policy outputs an action chunk of length $H = 50$ steps, which corresponds to 1.0 s of motion at the 50 Hz control rate stated in §3 (item (i) of the physical-plausibility analysis).

Stage 1: SFT base policy. The base policy is trained by flow-matching regression on a per-task demonstration dataset. The numbers of expert episodes per task are summarized in Table 2. Training is run for 60 000 optimizer steps with a per-GPU batch size of 16 (global batch size 64 across the 4 GPUs). The optimizer is AdamW with an initial learning rate of 2.5×10^{-5}, a linear warmup over the first 1,000 steps, followed by a cosine decay over the next 30,000 steps to a floor learning rate of 2.5×10^{-6}, which is then held constant for the remaining steps.

Table 2: **Number of expert demonstration episodes used for Stage 1 (SFT) per task.**

Task	PACK	CAP	USB	CASE
# SFT episodes	773	298	389	300

Stage 2: RPRO iterative fine-tuning. At each RPRO iteration $k \in \{1, 2, 3\}$, the policy is fine-tuned for 25 000 optimizer steps (totalling 75 000 steps across all three rounds) with a *global* batch size of 20 (i.e., 5 samples per GPU). The optimizer is AdamW with an initial learning rate of 1×10^{-5}, a linear warmup over 1,000 steps, followed by a cosine decay over the next 15,000 steps to a floor of 2.5×10^{-6}. The batch composition follows Algorithm 2: at $k = 1$, mini-batches mix the current preference set $\mathcal{D}^1_{\text{pref}}$ and the SFT dataset in an $80/20$ ratio; at $k \geq 2$, they mix $\mathcal{D}^k_{\text{pref}}$, the union of past preference sets $\mathcal{D}^{<k}_{\text{pref}}$, and $\mathcal{D}_{\text{SFT}}$ in a $70/15/15$ ratio.

The number of preference pairs (τ^w, τ^l) collected through the teleoperated intervention pipeline at each round and for each task is reported in Table 3. Each pair is subsequently expanded into per-state (s, a^w, a^l) tuples via the smooth interpolation procedure of §3.4 before being mixed into the training batches.

Table 3: **Number of preference pairs (τ^w_k, τ^l_k) collected at each RPRO round for each task.**

Round	PACK	CAP	USB	CASE
$k = 1$	40	89	30	63
$k = 2$	30	42	30	43
$k = 3$	30	30	30	30

Loss coefficients and reward-proxy temperature. The composite objective of Algorithm 2 (inner-loop update) and Eq. 6 uses $\lambda_{\text{PRO}} = 1/3$ and $\lambda_{\text{SFT}} = 1$, applied to every batch throughout all three RPRO rounds. The reward-proxy temperature in Eq. 4 and Eq. 5 is set to $\beta = 3.5$. These three scalars are the only RPRO-specific hyper-parameters introduced on top of the base flow-matching pipeline and are shared across all four tasks.

Reproducibility. Each cell of the main results table is averaged over 3 random seeds (data shuffling and parameter initialization), and each seed is evaluated for 100 rollouts under both the ID and OOD protocols described in §4.1. All other hyper-parameters not listed above are inherited unchanged from the base flow-matching policy.

H Task Details

This appendix expands the brief task descriptions in §4.1 with the detailed setup, per-stage subgoals, and the dominant failure modes of the SFT policy that FlowPRO is designed to repair. The complete per-stage pipeline for all four tasks is illustrated in Fig. 7.

Cosmetic Packaging (PACK). Two cosmetic containers are placed at fixed marker regions on either side of a bimanual packaging holder, with small per-episode randomization of their in-plane position. The left arm grasps the left container and inserts it into the left slot of the holder, then the

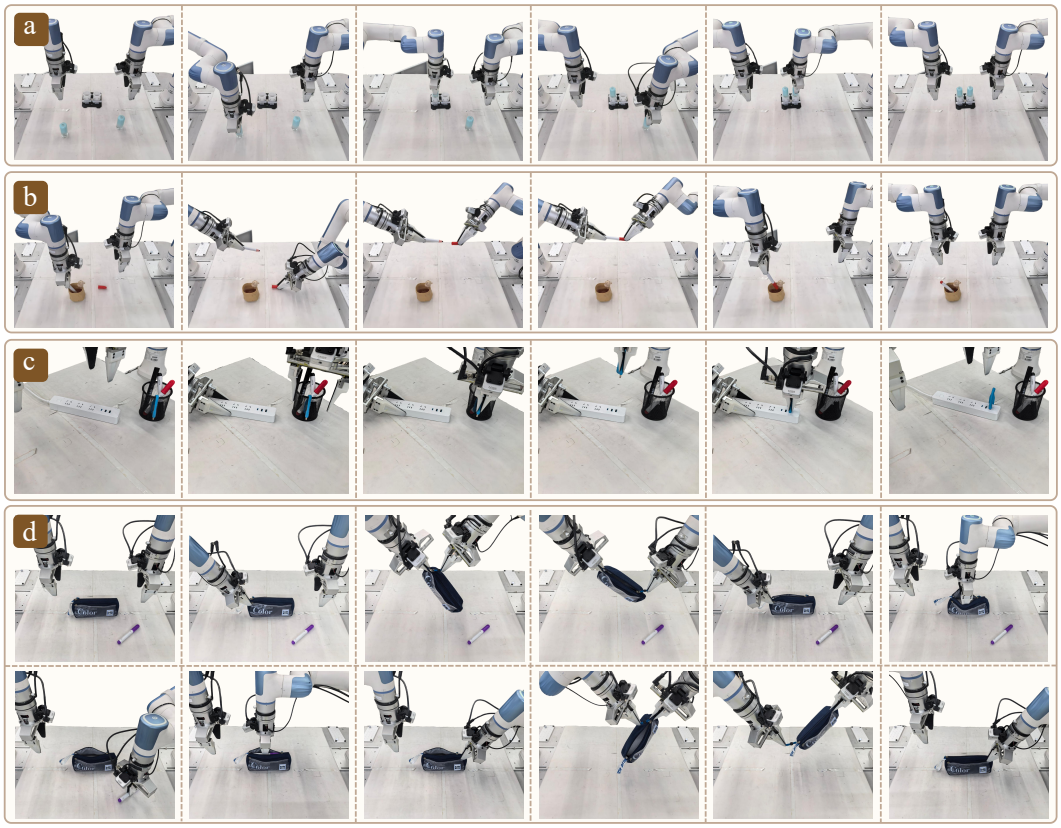

Figure 7: **Complete per-stage pipeline for the four real-robot tasks.** Extended version of Fig. 2 in the main text, showing every sub-stage of PACK, CAP, USB, and CASE.

right arm grasps the right container and inserts it into the right slot. The dominant difficulty lies in the *insertion* phase, which requires sub-centimeter positional accuracy and tight alignment of the container axis with the slot axis. Near the end of the trajectory, two failure modes dominate: (i) the container collides with the rim of the slot and gets pushed aside, and (ii) the container is inserted at a slight tilt and wedges before reaching the bottom. Both failures occur in the final fraction of the trajectory and are the hardest to recover from via SFT alone, since SFT receives no learning signal from negative outcomes.

Pen-Cap Assembly (CAP). The left arm picks an uncapped pen from a pen holder, then the right arm picks a pen cap from the table; the two arms then meet in mid-air and the right arm presses the cap onto the pen tip. The pens and the cap are placed at fixed marker regions with randomization. Three distinct difficulties make this task a challenging stress test for a VLA policy:

- *Left-arm grasp of the upright pen.* Several pens stand upright in the holder with only a short segment exposed above the rim, so the left arm has a very limited graspable region. The SFT policy frequently misses the pen entirely or only catches it near the edge of the fingers, leading to slippage during lift-off.
- *Right-arm grasp of the flat cap.* The pen cap lies flat on the table and is very thin, so the right gripper must align precisely with its low profile. Small vertical or yaw errors cause the fingers to either miss the cap completely or only pinch its edge, which makes the cap tilt or drop shortly after being lifted.
- *In-air bimanual assembly.* Even when both grasps succeed, the two arms must meet at a free-floating rendezvous point and align the cap-to-pen axis with sub-centimeter accuracy. Because both arms are moving and neither object is supported by the table, small pose errors on *either* arm

are amplified at the contact point, and the assembly is highly sensitive to residual errors inherited from the grasping phase.

These three difficulties appear at different temporal segments of the rollout (early, early, late) and involve different arms, which makes the task a good testbed for methods that need to correct localized failure modes without disturbing the already-working behavior elsewhere in the trajectory.

USB Insertion (USB). The left arm grasps a power strip placed at a marker region on the table and lifts it to a stable holding pose, while in parallel the right arm picks a small USB lamp standing in a pen holder; the right arm then inserts the lamp's USB plug into one of the strip's sockets. Object positions are fixed at marker regions with per-episode randomization of in-plane position and yaw. The dominant difficulty lies in the final *plug-in-socket insertion*: USB connectors require sub-millimeter accuracy in both position and orientation, and the socket aperture leaves essentially no clearance, so even a small residual error in either arm prevents the plug from entering. Two failure modes dominate near the end of the trajectory: (i) the plug edge collides with the socket bezel and is pushed off-axis, and (ii) the plug enters at a slight tilt and stalls before fully seating. Because the left arm is also actively holding the strip rather than resting on a fixture, both arms contribute pose error simultaneously, making this task a stringent test of *coordinated high-precision insertion*.

Pencil-Case Packing (CASE). This task is the most long-horizon in our suite, comprising six chained sub-stages executed by two arms in close coordination:

1. the left arm picks up the closed soft-fabric pencil case from a marker region on the table;
2. the right arm grasps the zipper slider, and the two arms cooperate to *unzip* the case;
3. the left arm places the now-open case at the table center;
4. the right arm reaches inside the case and holds it open from within;
5. the right arm picks a pen lying on the table and drops it into the case;
6. the right arm grasps the pencil case again and the two arms cooperate to *zip the case closed* and return it to the table center.

Object positions are fixed at marker regions with per-episode randomization. Three properties together make this task uniquely challenging: *(a) length and chaining*—a single early-stage error (e.g., a slipped grasp at stage 1 or an off-axis pull at stage 2) propagates and contaminates all subsequent stages, so the SFT policy's success rate compounds multiplicatively across stages; *(b) high-precision zipping*—both unzip (stage 2) and zip (stage 6) require the right gripper to align with and pull the small slider along a curved fabric track, which is geometrically as demanding as the insertion tasks above but must be executed twice in a single rollout; *(c) deformability*—the fabric case continuously changes shape under contact, so the visual configuration the policy observes at each stage drifts away from the SFT distribution depending on prior interactions, and conventional SFT supervision provides little signal for recovering from such state-dependent deformation. Failures observed under SFT span all stages: zipper grip slip, asymmetric two-arm pulling that jams the zipper, the case folding while the pen is being inserted.

I Baseline Methods

This appendix expands the brief baseline summary in §4.1 with the full methodological description of each comparator. The corresponding loss formulations in the flow-matching setting are deferred to Appendix J.

DAgger [10]. *Positive-only* interactive-imitation baseline via *dataset aggregation*. At each RPRO round, expert corrections collected on failed states through our teleoperated intervention pipeline (§3.3) are merged directly into the SFT dataset, and the policy is re-trained from scratch by behavior cloning on the aggregated set. This setting captures the dominant "correct-then-retrain" paradigm used in prior interactive imitation work.

DAgger-Buffered. *Positive-only* baseline via *controlled batch mixing*—an internal control we introduce to isolate the contribution of the preference loss from that of our batch-composition schedule. New expert corrections are kept in a separate buffer rather than being merged into the SFT dataset, and each training batch combines the SFT dataset and the buffered data using the *same* batch-composition schedule as FlowPRO. Comparing DAgger-Buffered against DAgger isolates the contribution of batch composition, while comparing it against FlowPRO isolates the contribution of the preference-based loss itself.

PI0.6* [17]. *Positive-and-negative* baseline via *advantage conditioning*. The flow-matching expert is reconditioned on a binary improvement indicator $I_t \in \{0, 1\}$ derived from our preference labels: states from positive trajectories τ^w are labelled $I_t = 1$ and states from negative trajectories τ^l are labelled $I_t = 0$. At inference, classifier-free guidance is applied with respect to I_t to steer generation toward improved actions. This comparator isolates the difference between a *conditioning*-based and a *contrastive-loss*-based use of identical preference data.

TPO [20]. *Positive-and-negative* baseline via *trajectory-wise preference optimization*. Our state-wise reward proxy (Eq. 4) is summed over each trajectory to form a trajectory-level reward $r_\theta(\tau) = \sum_t r_\theta(s_t, a_t)$, which is then plugged into the FlowPRO objective over (τ^w, τ^l) pairs in place of state-level rewards. This comparator isolates per-state vs. trajectory-level preference construction while keeping the loss family fixed.

Per-iteration baseline curves on PI0. For completeness, Fig. 8 reports the per-iteration success rates of all comparators when the base policy is PI0 instead of PI0.5, matching the PI0.5 curves shown in Fig. 3 of the main text.

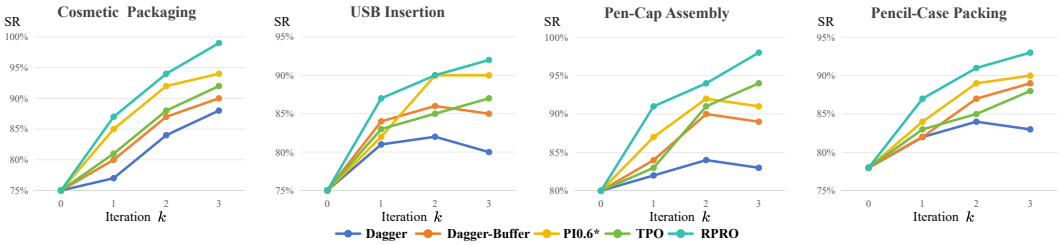

Figure 8: **Per-iteration success rates (SR) on the four real-robot tasks, with PI0 as the base policy.** Iteration 0 corresponds to the shared SFT checkpoint. Companion to Fig. 3 in the main text, which reports the analogous curves with PI0.5 as the base policy.

J Loss Formulations for Baselines and Ablations

For completeness, we list here the exact loss expressions adopted for each ablation variant in the flow-matching setting. Throughout this appendix, $\ell_\theta(s, a) = \mathbb{E}_{t,\epsilon}[\|v_\theta(a_t, t|s) - u(a_t|a)\|^2]$ denotes the per-sample flow-matching loss and $r_\theta(s, a) = \frac{\beta}{2}(\ell_{\text{ref}}(s, a) - \ell_\theta(s, a))$ is the implicit reward proxy (Eq. 4).

DAgger. A *positive-only* interactive imitation baseline that adapts the classical DAgger [10] pipeline to our preference-collection protocol. At round k, the policy $\pi_{\theta_{k-1}}$ is rolled out on the target tasks, an expert relabels the failed states via the teleoperated intervention pipeline (§3.3), and the resulting correction set $\mathcal{D}_k$ (containing only the *preferred* actions a^w) is *aggregated* directly into the running SFT dataset:

$$\mathcal{D}_{\text{SFT}}^{(k)} = \mathcal{D}_{\text{SFT}}^{(k-1)} \cup \{(s, a^w) : (s, a^w, a^l) \in \mathcal{D}_k\}, \quad \mathcal{D}_{\text{SFT}}^{(0)} = \mathcal{D}_{\text{SFT}}. \tag{23}$$

The policy at round k is then re-trained by behavior cloning on the aggregated set:

$$\mathcal{L}_{\text{DAgger}}^{(k)}(\theta) = \mathbb{E}_{(s,a)\sim\mathcal{D}_{\text{SFT}}^{(k)}}\big[\ell_\theta(s, a)\big]. \tag{24}$$

Negative trajectories τ^l are discarded; DAgger therefore consumes the same expert annotation budget as FlowPRO but only exploits the positive half of each preference pair, and uniform aggregation gives no control over the SFT-to-correction sample ratio in each batch.

DAgger-Buffered. A *positive-only* ablation of DAgger that isolates the effect of *controlled batch mixing* from the underlying loss family. Instead of merging expert corrections into the SFT pool, we keep three separate buffers in exact correspondence with FlowPRO: the round-k correction set $\mathcal{D}_{\text{corr}}^{k} = \{(s, a^w) : (s, a^w, a^l) \in \mathcal{D}_k\}$, the historical pool $\mathcal{D}_{\text{corr}}^{<k} \triangleq \bigcup_{j<k} \mathcal{D}_{\text{corr}}^{j}$, and $\mathcal{D}_{\text{SFT}}$. Let $\mathcal{P}_{\text{mix}}^{(k)}$ denote the mixture sampling distribution over these three buffers under the *same fixed proportions as FlowPRO* (Sec. 3.4; $80\%/20\%$ for $k{=}1$ and $70\%/15\%/15\%$ for $k{\geq}2$). The policy is trained by behavior cloning under this mixture:

$$\mathcal{L}_{\text{DAgger-Buf}}^{(k)}(\theta) = \mathbb{E}_{(s,a^w)\sim\mathcal{P}_{\text{mix}}^{(k)}}\big[\ell_\theta(s, a^w)\big]. \tag{25}$$

Compared to vanilla DAgger, only the data buffering and mixing schedule changes; the loss family (positive-only behavior cloning), the expert annotations used, and the optimization protocol are identical. This baseline therefore quantifies how much of the benefit of our pipeline comes from *batch composition* alone, independent of the contrastive preference objective.

TPO (trajectory-wise). Following the TPO objective of GRAPE [20], this baseline applies a trajectory-wise preference loss to the *whole-trajectory log-likelihood ratio*. Under the MDP assumption $\pi(\zeta) = \prod_{t=1}^{T} \pi(a_t \mid o_t)$, so $\log \frac{\pi_\theta(\zeta)}{\pi_{\text{ref}}(\zeta)} = \sum_{t=1}^{T} \log \frac{\pi_\theta(a_t|o_t)}{\pi_{\text{ref}}(a_t|o_t)}$. In the flow-matching setting, we approximate the per-step log-ratio using the flow-matching loss surrogate and define the trajectory-level reward proxy

$$r_\theta(\tau) = \sum_{(s_t,a_t)\in\tau} r_\theta(s_t, a_t) = \frac{\beta}{2} \sum_{(s_t,a_t)\in\tau} \big(\ell_{\text{ref}}(s_t, a_t) - \ell_\theta(s_t, a_t)\big). \tag{26}$$

This trajectory-level proxy is plugged into the RPRO objective (Eq. 6) rather than the DPO objective used in the original GRAPE formulation, so that TPO and FlowPRO share the same preference loss family and differ only in the granularity of the reward proxy (trajectory-level vs. step-level), combined with the SFT term on the preferred trajectory:

$$\mathcal{L}_{\text{TPO}}(\theta) = \mathcal{L}_{\text{PRO}}\big(r_\theta(\tau^w), r_\theta(\tau^l)\big) + \lambda_{\text{SFT}} \cdot \mathcal{L}_{\text{SFT}}(\theta). \tag{27}$$

The per-step reward proxy $r_\theta(s_t, a_t)$ is shared with RPRO (Eq. 4); the only difference is whether the optimizer operates on a single-state reward gap or on the trajectory-summed gap.

PI0.6*. Following RECAP [17], PI0.6* adds *advantage conditioning* to the flow-matching action expert rather than modifying its loss. The full RECAP pipeline trains a distributional value function $p_\phi(V \mid o_t, \ell)$ and computes an N-step advantage $A^\pi(o_t, a_t)$ that is binarized into an improvement indicator $I_t = \mathbb{1}[A^\pi > \epsilon_\ell]$. In our setting, the teleoperated intervention-and-rollback pipeline (§3.3) already provides trajectory-level positive/negative labels that directly serve as I_t:

$$I_t = \begin{cases} \texttt{True}, & \text{if the state is on the positive trajectory } \tau^w \text{ (or is an SFT expert sample)}, \\ \texttt{False}, & \text{if the state is on the negative trajectory } \tau^l. \end{cases} \tag{28}$$

We therefore bypass RECAP's value-function training and advantage-estimation machinery, and reuse only the policy-side optimization. The flow-matching action expert is reconditioned on I_t as an additional input token, and trained with a standard conditional flow-matching loss plus a 30% indicator dropout (to enable classifier-free-guidance sampling at inference):

$$\mathcal{L}_{\text{RECAP}}(\theta) = \mathbb{E}_{(o,a,I_t)}\big[\|v_\theta(a_\eta, \eta \mid o, \ell, I_t) - (a - \omega)\|^2\big], \quad \text{with } I_t \text{ masked w.p. } 0.3, \tag{29}$$

where $a_\eta = \eta a + (1-\eta)\omega$ with $\omega \sim \mathcal{N}(0, I)$ is the flow-matching interpolant and $\eta \in [0, 1]$ is the flow time index. At inference time we fix $I_t = \texttt{True}$ and optionally apply classifier-free guidance between the conditional and unconditional velocity fields to further amplify the "improvement" direction. This formulation isolates the contribution of *advantage conditioning as a policy-improvement mechanism*, holding the preference dataset and label assignment identical to FlowPRO.

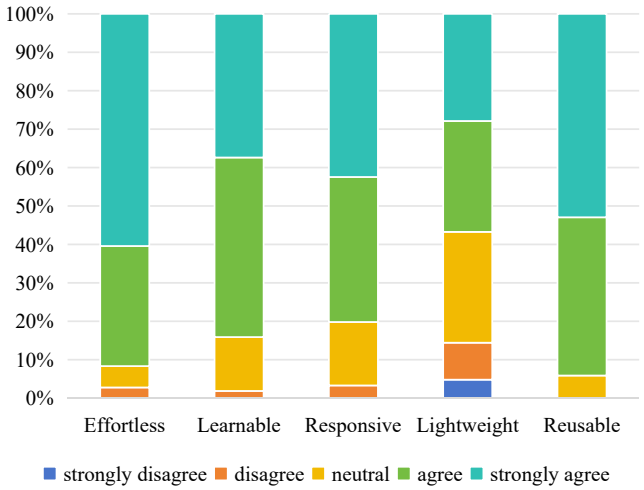

Figure 9: **User study on the teleoperated data-collection system.** Stacked-bar distribution of 5-point Likert responses across the five evaluation dimensions. Larger *agree/strongly agree* (top, green/teal) regions indicate more favorable ratings.

SFT. Pure flow-matching regression on preferred actions a^w only:

$$\mathcal{L}_{\text{SFT}}(\theta) = \mathbb{E}_{(s,a^w)\sim\mathcal{D}}\big[\ell_\theta(s, a^w)\big]. \tag{30}$$

DPO (in flow matching). Standard pairwise DPO loss with the implicit reward proxy:

$$\mathcal{L}_{\text{DPO}}(\theta) = -\mathbb{E}_{(s,a^w,a^l)\sim\mathcal{D}}\Big[\log\sigma\big(r_\theta(s, a^w) - r_\theta(s, a^l)\big)\Big]. \tag{31}$$

Only the contrastive term is present; there is no proximal regularizer and no SFT term.

DPO + SFT. DPO loss augmented with an SFT regression on the *chosen* action only:

$$\mathcal{L}_{\text{DPO+SFT}}(\theta) = \lambda_{\text{PRO}} \cdot \mathcal{L}_{\text{DPO}}(\theta) + \lambda_{\text{SFT}} \cdot \mathbb{E}_{(s,a^w)\sim\mathcal{D}}\big[\ell_\theta(s, a^w)\big]. \tag{32}$$

Note that $\mathcal{L}_{\text{DPO}}$ here is exactly the contrastive optimizer $\mathcal{L}_{\text{con}}$ inside $\mathcal{L}_{\text{PRO}}$ (Eq. 5); DPO+SFT therefore corresponds to RPRO with the proximal regularizer $\mathcal{L}_{\text{reg}}$ removed, and uses the same coefficients $\lambda_{\text{PRO}} = 1/3$, $\lambda_{\text{SFT}} = 1$ as RPRO.

PRO (no SFT). The contrastive-plus-regularizer component $\mathcal{L}_{\text{PRO}}$ (Eq. 5) used *alone*, i.e., the full RPRO loss (Eq. 6) with $\lambda_{\text{SFT}} = 0$:

$$\begin{aligned}\mathcal{L}_{\text{PRO}}(\theta) = -\mathbb{E}_{(s,a^w,a^l)}\Big[&\log\sigma(r_\theta(s, a^w) - r_\theta(s, a^l)) \\ &+ \sum_{a\in\{a^w,a^l\}} \big(\tfrac{1}{2}\log\sigma(r_\theta(s, a)) + \tfrac{1}{2}\log\sigma(-r_\theta(s, a))\big)\Big].\end{aligned} \tag{33}$$

RPRO (full). The full RPRO loss combining the PRO contrastive-plus-regularizer objective with the SFT regression (Eq. 6):

$$\mathcal{L}_{\text{RPRO}}(\theta) = \lambda_{\text{PRO}} \cdot \mathcal{L}_{\text{PRO}}(\theta) + \lambda_{\text{SFT}} \cdot \mathcal{L}_{\text{SFT}}(\theta). \tag{34}$$

K User Study on the Data-Collection System

To assess whether our teleoperated human-in-the-loop data-collection system (Sec. 3.3) is also *usable* from the operator's perspective—and not merely effective in producing useful preference pairs for training—we conducted a small subjective user study with the data-collection operators who participated in our experiments.

Protocol. After completing several full data-collection sessions across the four tasks (PACK, CAP, USB, CASE), each operator was asked to rate the system along five dimensions on a standard 5-point Likert scale (*strongly disagree*, *disagree*, *neutral*, *agree*, *strongly agree*). All five items are phrased so that "agree" consistently corresponds to a more favorable evaluation, which avoids reverse-coding when aggregating responses. The five items, together with the one-word labels used in Fig. 9, are:

- **Effortless** – "Collecting demonstrations with this system is physically effortless."
- **Learnable** – "I was able to learn this system quickly and start collecting usable data with little training."
- **Responsive** – "I can trigger a rollback promptly the moment I sense an upcoming failure."
- **Lightweight** – "Resetting between trials (or after a failure) is quick and requires little extra effort."
- **Reusable** – "I would willingly use this system again for collecting data on future tasks."

Results. The response distributions are summarized in Fig. 9. Across all five dimensions the combined *agree* and *strongly agree* shares dominate the responses—reaching roughly 80% or above for *Effortless*, *Learnable*, *Responsive*, and *Reusable*, and remaining the majority (around 56%) even for *Lightweight*, where only about 14% of responses are *disagree* or *strongly disagree* and the rest are neutral. Beyond producing the preference pairs that drive RPRO training, the data-collection system is therefore perceived by its users as low-effort, quick to learn, responsive to rollback, and worth reusing.

L Statistical Significance Analysis

This appendix collects the formal statistical analyses underlying the significance claims in the main text (Table 1 and §4.2). We report (i) Wilson 95% confidence intervals for every per-cell success rate, (ii) Cochran–Mantel–Haenszel (CMH) tests stratified by task and base policy with Mantel–Haenszel pooled odds ratios, (iii) a stratified sign test on the directional consistency of RPRO's advantage, and (iv) the analogous CMH analysis for the loss-component ablations on PACK.

L.1 Test setup, assumptions, and multiplicity control

Data and independence. For each combination of base policy (PI0, PI0.5), task (PACK, CAP, USB, CASE), and method, we evaluate n=100 independent rollouts with randomized initial placements. Each cell is trained with 3 random seeds; the per-cell counts are obtained by pooling the rollouts across seeds. Independence holds both within cells (fresh randomized placement per rollout) and across cells (separate fine-tuned checkpoints).

Test family. We use binomial models throughout: each cell count $X \sim \mathrm{Binomial}(n = 100, p)$. Per-cell success rates are summarized by Wilson 95% confidence intervals. Our primary cross-stratum analysis is the Cochran–Mantel–Haenszel (CMH) test with Mantel–Haenszel pooled odds ratios, and directional consistency is further assessed by a one-sided sign test over strata.

Multiplicity correction. Across the 4 pairwise CMH families (one per baseline), we interpret the CMH p-values under a Bonferroni correction at level $\alpha/4 = 0.0125$; all four CMH p-values reported below remain $< \alpha/4$ even under this stricter level.

Reporting conventions. All p-values below are one-sided unless otherwise stated. CMH χ^2 statistics are computed without continuity correction (the standard Mantel–Haenszel score statistic). Wilson 95% intervals use $z_{0.975} = 1.96$. Mantel–Haenszel pooled odds-ratio confidence intervals use a Robins–Breslow–Greenland-type Wald approximation [37] on $\ln \widehat{OR}_{MH}$.

L.2 Cochran–Mantel–Haenszel test, stratified by task and base policy

To pool evidence across the 8 task–base strata, we adopt the Cochran–Mantel–Haenszel (CMH) test, which is the standard tool for stratified $2\times2\times K$ binary outcomes and gains statistical power by combining directionally consistent evidence across strata [38, 39]. We treat each task–base combination as one of $K{=}8$ strata of 2×2 tables (RPRO vs. baseline; success vs. failure; $n_R{=}n_B{=}100$ in every stratum). Let a_k denote the number of RPRO successes in stratum k, $b_k = 100 - a_k$ failures, c_k baseline successes, $d_k = 100 - c_k$ baseline failures, and $N_k = 200$. Under the null hypothesis of equal success probabilities within each stratum, the conditional mean and variance of a_k given the marginals are

$$E_k = \frac{(a_k+b_k)(a_k+c_k)}{N_k} = \frac{a_k+c_k}{2}, \qquad V_k = \frac{(a_k+b_k)(c_k+d_k)(a_k+c_k)(b_k+d_k)}{N_k^2(N_k-1)} = \frac{(a_k+c_k)(b_k+d_k)}{796}, \quad (35)$$

and the CMH score statistic is

$$\chi^2_{\text{CMH}} = \frac{\left(\sum_k (a_k - E_k)\right)^2}{\sum_k V_k} \overset{H_0}{\sim} \chi^2_1, \quad (36)$$

which we convert to a one-sided p-value by halving (the observed direction matches $H_1 : p_R > p_B$ in every stratum). The Mantel–Haenszel pooled odds ratio is $\widehat{OR}_{MH} = \sum_k a_k d_k/N_k \big/ \sum_k b_k c_k/N_k$, with 95% CIs from the Robins–Breslow–Greenland-type Wald approximation on $\ln \widehat{OR}_{MH}$ [37].

Table 4 summarizes the four pairwise CMH analyses. Across all four baselines, RPRO is significantly superior at $p < 10^{-3}$ even under a Bonferroni correction across the four families ($\alpha/4 = 0.0125$), with pooled odds ratios uniformly bounded away from 1 (lower 95% CI > 1.4 in every comparison).

Table 4: **Cochran–Mantel–Haenszel tests for RPRO vs. each baseline, stratified by task ({PACK, CAP, USB, CASE}) and base policy ({PI0, PI0.5}).** $K{=}8$ strata, $n{=}100$ per cell.

Comparison	$\sum_k (a_k - E_k)$	$\widehat{OR}_{MH}$ (95% CI)	χ^2_{CMH}	$p_{\text{1-sided}}$
RPRO vs. DAgger	47.5	4.58 [3.06, 6.87]	63.13	$< 10^{-15}$
RPRO vs. DAgger-Buffered	27.0	2.92 [1.92, 4.45]	26.80	1.1×10^{-7}
RPRO vs. PI0.6*	18.0	2.24 [1.45, 3.47]	13.88	9.7×10^{-5}
RPRO vs. TPO	19.0	2.33 [1.51, 3.59]	15.23	4.8×10^{-5}

L.3 Stratified sign test on directional consistency

We further check that RPRO's advantage is *directionally consistent* across strata (i.e., not driven by one or two large gaps). For each of the 8 strata, we compare RPRO to the strongest baseline within that stratum:

- Strict wins ($\text{SR}_{\text{RPRO}} > \max_b \text{SR}_b$): 8 strata—PI0/PACK (99 vs. PI0.6* 94), PI0/CAP (98 vs. TPO 94), PI0/USB (92 vs. PI0.6* 90), PI0/CASE (93 vs. PI0.6* 90), PI0.5/PACK (99 vs. TPO 95), PI0.5/CAP (99 vs. PI0.6* 94), PI0.5/USB (95 vs. PI0.6* 93), PI0.5/CASE (93 vs. TPO 90).
- Ties: 0 strata.
- Strict losses: 0.

A one-sided sign test of $H_1 : \Pr(\text{RPRO is strictly best}) > 0.5$ yields $p = 0.5^8 = 3.9\times10^{-3}$, below 0.05, confirming that the CMH-based aggregate significance reported in §L.2 is not an artifact of a single dominant stratum but reflects a directionally consistent advantage.

L.4 Loss-component ablations on PACK (Q3 & Q4)

For the loss-component ablation reported in Fig. 4(b), we evaluate $n{=}100$ rollouts under both in-distribution (ID) and out-of-distribution (OOD) initial-condition strata on PACK. Table 5 reports

the CMH analyses pooled across the ID/OOD strata (K=2). All four one-sided CMH p-values are below 6.0×10^{-3} (raw), and remain significant at $\alpha = 0.05$ after Bonferroni correction over the 4 ablation families.

Table 5: **CMH tests for the loss-component ablation on PACK, stratified by ID vs. OOD initial conditions** (K=2 strata, n=100 per cell). $\widehat{OR}_{MH}$ confidence intervals use the Robins–Breslow–Greenland-type Wald approximation; the confidence interval for RPRO vs. DPO is omitted because the table contains very small baseline success counts (5/100), for which the $\ln \widehat{OR}_{MH}$ Wald interval is unreliable—the score-based p-value remains valid and is reported.

Comparison	$\sum_k (a_k - E_k)$	$\widehat{OR}_{MH}$ (95% CI)	χ^2_{CMH}	$p_{\text{1-sided}}$
RPRO vs. SFT	10.0	3.81 [1.61, 8.99]	9.27	1.2×10^{-3}
RPRO vs. DPO+SFT	14.0	4.82 [2.23, 10.40]	16.05	3.1×10^{-5}
RPRO vs. PRO	8.0	2.27 [1.20, 4.31]	6.30	6.0×10^{-3}
RPRO vs. DPO	83.5	$\sim 1.4\times10^{2}$ (CI omitted)[2]	278.4	$< 10^{-50}$

Reading these tables. The pooled CMH analysis confirms that every component of RPRO contributes a statistically detectable improvement: *(a) SFT term*—adding the SFT term to PRO (i.e., RPRO vs. PRO) improves PACK success at $p \approx 6\times10^{-3}$ (CMH); *(b) Contrastive PRO term*—adding the PRO contrastive term to SFT (i.e., RPRO vs. SFT) improves PACK success at $p \approx 1.2 \times 10^{-3}$ (CMH), with the gap being driven primarily by the OOD stratum ($z = 3.01$, $p \approx 10^{-3}$ at the cell level); *(c) Proximal regularizer*—replacing PRO with plain DPO (i.e., RPRO vs. DPO+SFT) degrades PACK success at $p \approx 3.1 \times 10^{-5}$ (CMH), and removing both the proximal regularizer and the SFT term (i.e., RPRO vs. DPO) collapses success rates to $13\%/5\%$ at $p < 10^{-50}$. Together these isolate the contribution of each loss component and substantiate the qualitative claims in §4.2.

[2]The Mantel–Haenszel pooled odds-ratio point estimate is $16823/123 \approx 136.8$; with $b_k c_k$ counts as small as $9 \cdot 5 = 45$, the RBG Wald interval on $\ln \widehat{OR}_{MH}$ is unreliable, so we report only the score-based χ^2_{CMH} and its one-sided p-value, which do not require a normal approximation on $\ln \widehat{OR}$.