PIVOT: Policy Improvement via Trust-Region Optimization from Deployment Experience

Shenghan Lin[1,2*], **Qicheng Xu**[1,2*], **Zhecheng Yuan**[1,2,3*†], **Xiangchen Tian**[2], **Chenhao Lu**[1,2,3] **Zilin Kang**[2,3], **Muhan Wang**[2], **Xiuyu Xie**[2], **Huazhe Xu**[1,2,3]
[1] PokeBot, [2] Tsinghua University IIIS, [3] Shanghai Qi Zhi Institute

Abstract: Post-training with real-world experience is critical for reliable deployment of robot foundation models, but heterogeneous data from demonstrations, human corrections, and autonomous rollouts are difficult to convert into stable policy updates. We present **PIVOT**, an iterative real-world post-training system built on **T**rust-**R**egion **F**low Matching **O**ptimization (**TRFO**). TRFO provides a unified reinforcement-learning objective for heterogeneous deployment experience, using signed advantages to attract high-value transitions and repel low-value ones within a trust region while preserving the flow-matching training form. At each iteration, we first collect on-robot data using the current policy, estimate transition-level values via distributional TD learning, and then apply TRFO to update the policy. The updated policy is subsequently redeployed on the robot to gather the next round of experience, forming an iterative post-training procedure. On a suite of precise, contact-rich bimanual manipulation tasks, PIVOT yields consistent, monotonic gains over successive iterations, substantially improving success rates and throughput across all evaluated tasks. Project page: https://linshey.com/pivot/.

Keywords: Real-World RL, Bimanual Manipulation, Post-Training

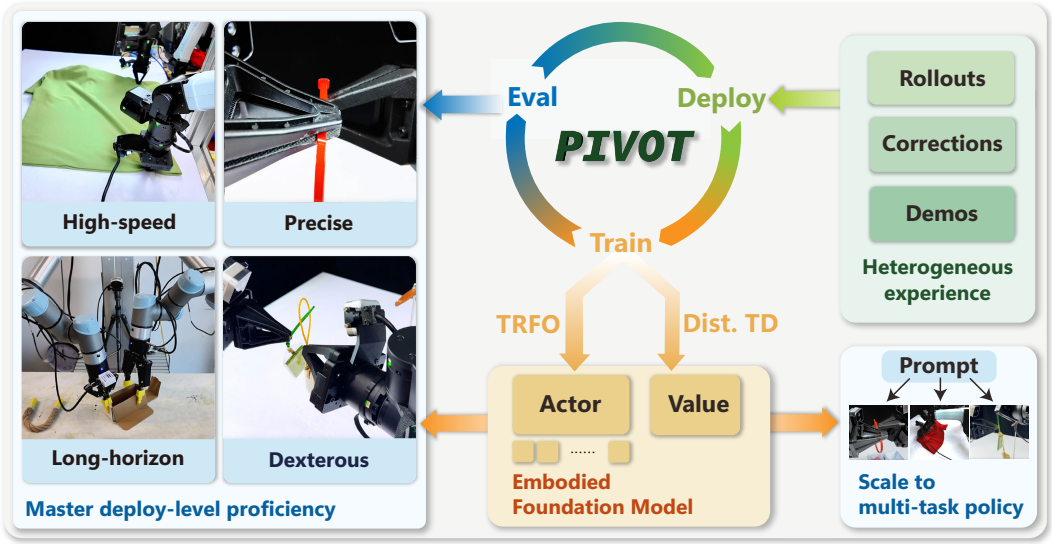


Figure 1: **PIVOT iteratively improves embodied foundation models from real-world deployment experience.** Starting from a pretrained foundation model, we deploy the policy to collect autonomous rollouts and human corrections, train a distributional value expert on the resulting data, and optimize the policy with Trust-Region Flow Matching Optimization (TRFO). Repeating this loop progressively improves manipulation performance across diverse contact-rich bimanual tasks.

[*]Equal Contribution [†]Project Leader

10th Conference on Robot Learning (CoRL 2026), Austin TX, USA.

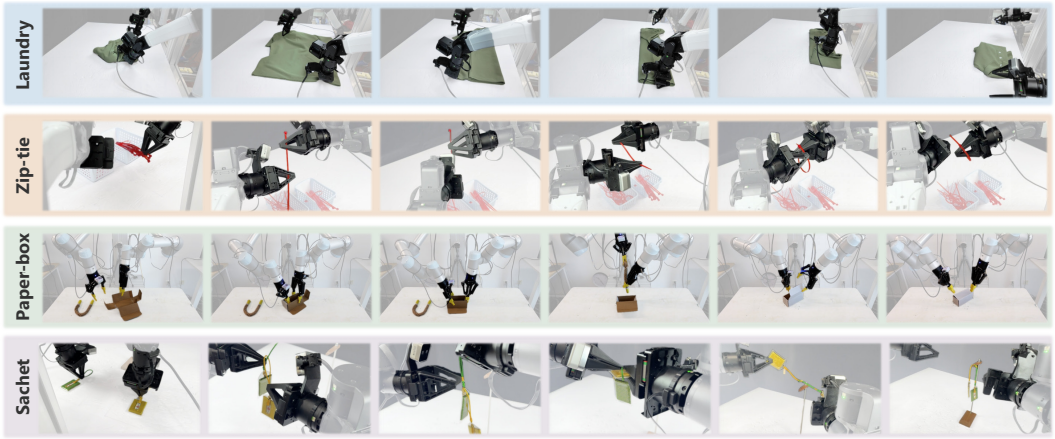


Figure 2: **Bimanual Manipulation Tasks.** Step-by-step keyframes of the four bimanual manipulation tasks: laundry folding, paper box assembly, zip-tie threading, and sachet packaging.

1 Introduction

Embodied foundation models have rapidly advanced, unifying visual observations, language instructions, and low-level motor commands within a single policy [1–8]. Large-scale pretraining provides broad task competence, but bridging the gap to deployment-level proficiency requires post-training from real-world experience [9, 10]. An important trend in embodied post-training is the **heterogeneous experience loop**: a deployed policy collects autonomous rollouts, human operators provide targeted corrections or outcome feedback, and the resulting experience is incorporated into the next policy update [9–12]. Such a loop turns deployment from a terminal evaluation stage into a continuing source of task-relevant supervision. However, the data produced by this loop are inherently heterogeneous: demonstrations specify intended behavior, corrections expose on-policy failure states, and autonomous rollouts contain successes, failures, recoveries, and partial progress.

The algorithmic question at the heart of this loop is therefore how to convert heterogeneous experience into a unified, stable update for the embodied foundation model, while respecting the flow-matching [13, 14] parameterization used by many leading embodied policies. Demonstrations and human corrections provide actions that the policy should move toward [15–19], while autonomous rollouts provide mixed evidence: some segments should be reinforced, some reveal recoverable progress, and others indicate behaviors that should be avoided [20–24]. Pure imitation cannot use this negative evidence, whereas direct policy-gradient or critic-gradient updates can be costly or unstable for generative action models [25, 26]. We therefore propose **Trust-Region Flow Matching Optimization (TRFO)**, a unified objective that modulates the standard flow-matching loss with per-transition advantages estimated by a learned value function: expert data reduce to ordinary flow-matching imitation, while autonomous rollouts induce a trust-region reinforcement learning update bounded by a frozen reference policy. More specifically, TRFO assigns signed advantages to individual transitions and uses them to regulate the direction and strength of the flow-matching update. Positive-advantage transitions pull the policy toward high-value behaviors, whereas negative-advantage transitions induce a repulsive update that discourages low-value behaviors. This yields a unified policy-improvement mechanism that preserves the original flow-matching parameterization while constraining updates within a trust region around the reference policy.

We instantiate TRFO in an iterative real-world post-training loop, **PIVOT**, built on a flow-based VLA model [2]. Starting from a demonstration-tuned policy, each iteration deploys the current policy, collects autonomous rollouts and HG-DAgger-style [16] human corrections, labels episodes with sparse terminal outcomes, and trains a distributional value expert to estimate per-transition advantages. The policy is then updated with TRFO using the previous checkpoint as the frozen reference, and redeployed with a training-free temporal alignment relaxation for chunk blending to collect the next round of on-policy experience. We evaluate PIVOT on a suite of precise, contact-

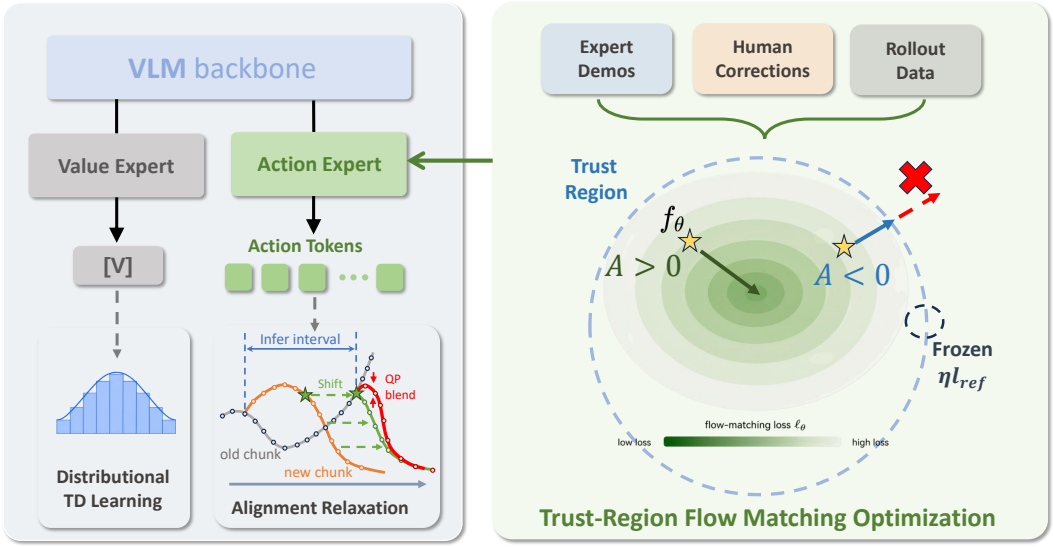


Figure 3: **PIVOT method overview.** PIVOT combines three components: distributional TD learning for estimating transition advantages, TRFO for signed advantage-weighted flow-policy updates, and Temporal Alignment Relaxation for smooth chunk-based deployment.

rich bimanual manipulation tasks, including LAUNDRY (T-shirt folding), ZIP TIE (flexible cable-tie threading), PAPER BOX (multi-stage box assembly), and SACHET (sachet threading). These tasks require long-horizon coordination, fine-grained contact control, and recovery from small execution errors. Across all tasks, the iterative loop yields stable, monotonic gains over successive iterations, substantially improving throughput and success rates. Under the same data budget, TRFO consistently outperforms alternative policy extraction methods. We further show that a single unified multi-task policy trained with TRFO matches the performance of independently trained per-task specialists, suggesting that the objective scales naturally to diverse skill repertoires.

2 Method

PIVOT iteratively post-trains a flow-based VLA from heterogeneous deployment experience. Each iteration first deploys the current policy to collect a dataset that mixes curated demonstrations, autonomous on-policy rollouts, and HG-DAgger-style human corrections, labeling each episode with only a sparse, binary outcome reward. A distributional value network is then trained on this dataset to assign an advantage to every transition (Sec. 2.2), and the policy is updated with *Trust-Region Flow Matching Optimization (TRFO)*, a unified policy-improvement objective for flow-based embodied foundation models (Sec. 2.3). At deployment time, we further apply *temporal alignment relaxation*, a training-free chunk blending trick that substantially improves stitching quality (Sec. 2.4).

2.1 Preliminaries

The backbone VLA generates action chunks [17] via conditional flow matching [13, 14]. Given an observation o and a target action chunk $a \in \mathbb{R}^{H\times d}$ of horizon H, a learned velocity field $f_\theta(o, x_\tau, \tau)$ is trained to denoise the linear interpolant $x_\tau = \tau\epsilon + (1-\tau)a$ toward a, where $\epsilon \sim \mathcal{N}(0, I)$ and $\tau \sim \mathcal{U}(0, 1)$, by minimizing:

$$\ell_\theta(o, a) = \mathbb{E}_{\tau,\epsilon}\big[\|f_\theta(o, x_\tau, \tau) - u_\tau\|_2^2\big], \quad u_\tau = \epsilon - a. \tag{1}$$

At inference time, actions are produced by integrating f_θ from noise to action with a fixed number of Euler steps. To evaluate transitions, we define a state value function $V(o_t)$ with discount factor γ under a task-agnostic sparse reward: $r_t = -1$ for non-terminal steps, $r_n = 0$ on success, and $r_n = -\frac{1}{1-\gamma}$ on failure, which bounds the return in $[V_{\min}{=}-\frac{1}{1-\gamma}, V_{\max}{=}0]$ regardless of task or horizon. Since the policy emits chunks of length H, we use chunk-level bootstrap targets $V_{\text{tgt}}(o_t) = \gamma^H V(o_{t+H}) + \sum_{j=0}^{H-1} \gamma^j r_{t+j}$ for value training and advantage computation.

2.2 Sparse Reward Labeling and Distributional TD Learning

Given the heterogeneous dataset collected at each iteration, we label every episode with a single binary outcome reward—success or failure—requiring no per-task reward shaping (Appendix 6.3). From this sparse signal, we train a *value expert* that predicts a distributional estimate of $V(o_t)$ using the chunk-level bootstrap targets V_{tgt} defined in Sec. 2.1. Because the sparse reward convention fixes $V_{\min}$ and $V_{\max}$ to task-independent constants, a single value expert can serve all tasks without per-task calibration.

The value expert shares the VLA's VLM backbone: a learned aggregation token $e_V \in \mathbb{R}^d$ is appended after the shared vision–language prefix KV-cache and read out by a residual MLP head that produces logits over K uniformly spaced categorical bins on $[V_{\min}, V_{\max}]$ [27], from which the scalar estimate $\hat{V} = \sum_k p_k c_k$ is recovered (Appendix 6.4). The value expert is trained by minimizing the categorical cross-entropy between its predicted bin distribution and the categorical projection [27] of the bootstrap target V_{tgt}, with a Polyak-averaged target network [28] providing the bootstrap values (Appendix 6.5). The resulting per-transition value estimates provide a dense advantage signal for the policy update described next.

2.3 Trust-Region Flow Matching Optimization (TRFO)

Given the value function from Sec. 2.2, we derive a unified objective for updating the flow matching policy from heterogeneous off-policy data. Our *Trust-Region Flow Matching Optimization (TRFO)* retains the standard flow-matching loss form and applies to any flow-based foundation model: for demonstrations and corrections, TRFO reduces to supervised flow-matching imitation; for autonomous rollouts, it becomes a trust-region reinforcement learning update [29, 30] that reinforces positive-advantage actions and repels negative ones while staying close to a frozen reference.

Let $\ell_\theta(o, a)$ denote the per-sample flow matching loss (Eq. 1) of the current policy on transition (o, a), and let $\ell_{\text{ref}}(o, a)$ denote the corresponding loss under a frozen reference policy initialized from the previous iteration's checkpoint. The TRFO objective is

$$\mathcal{L}_{\text{TRFO}}(\theta) \;=\; \mathbb{E}_{(o,a)\sim\mathcal{D}}\Big[\,\text{sg}\big(w(\hat{A},\,\ell_\theta,\,\ell_{\text{ref}})\big)\cdot\ell_\theta(o,a)\,\Big], \tag{2}$$

where $\text{sg}(\cdot)$ denotes stop-gradient and the per-sample weight is

$$w(\hat{A},\,\ell_\theta,\,\ell_{\text{ref}}) \;=\; \begin{cases} \hat{A}\,, & \hat{A} > 0, \\ \hat{A}\,\max\!\left(0,\;1-\dfrac{\ell_\theta}{\eta\,\ell_{\text{ref}}+\epsilon}\right), & \hat{A}\le 0. \end{cases} \tag{3}$$

Here η is a margin coefficient that controls the size of the trust region, ϵ is a small constant for numerical stability, and the advantage is computed as $\hat{A} = \text{clip}\left(\alpha\cdot\left(\gamma^H\widetilde{V}(o_{t+H}) - \widetilde{V}(o_t) + \sum_{i=0}^{H-1} r_i\cdot\gamma^i + b\right),\,-1,\,1\right)$, the temporal difference of *smoothed* value estimates, normalized into $[-1, 1]$. The smoothed value $\widetilde{V}$ is obtained by relabeling every non-expert transition in $\mathcal{D}$ with the trained value expert $\hat{V}$ and applying a low-pass filter along the trajectory; for transitions belonging to expert demonstrations we set $\hat{A} \equiv 1$ by convention. The two branches of w act as imitation and repulsion. Positive-advantage samples pull the policy toward the demonstrated action in proportion to $\hat{A}$; negative-advantage samples repel the policy away, but only until ℓ_θ exceeds $\eta\cdot\ell_{\text{ref}}$, at which point the trust region extinguishes further repulsion.

We provide a heuristic interpretation in Appendix 6.10 showing that, under a local loss-difference proxy for the log-likelihood ratio, the TRFO gradient for autonomous rollouts resembles a first-order advantage-weighted policy gradient surrogate [29, 30], while reducing to ordinary flow-matching imitation for expert data.

2.4 The PIVOT Loop

Combining the components of Sec. 2.2–2.3 yields an iterative improvement loop. We start from a dataset $\mathcal{D}_0$ of curated demonstrations and obtain a base policy π_0 by supervised flow matching fine-

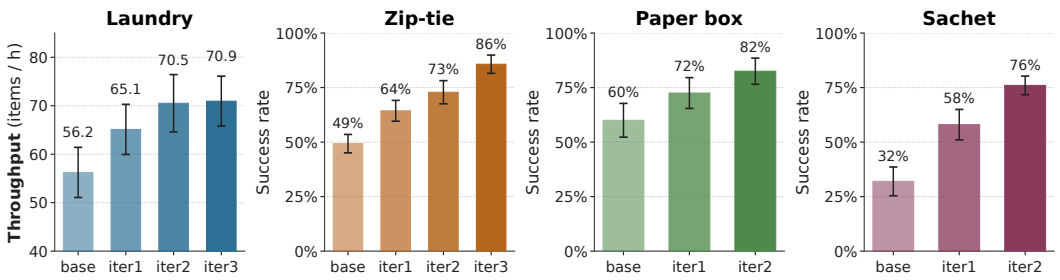


Figure 4: **Iterative improvement across tasks.** Success rate (or throughput) at each post-training iteration for laundry, zip-tie, paper-box, and sachet. PIVOT yields stable, monotonic gains over successive iterations on all evaluated tasks. Error bars denote ±1 SEM.

tuning of the pretrained VLA on $\mathcal{D}_0$. At iteration $i \geq 1$, we (1) deploy π_{i-1} on the target task with the chunk blending procedure described below to collect a batch of autonomous rollouts together with a batch of HG-DAgger-style human corrections, append both to $\mathcal{D}_{i-1}$ to form $\mathcal{D}_i$, and label each new episode with only its terminal success bit; (2) train a value expert on $\mathcal{D}_i$ from the value loss in Sec. 2.2, then relabel every non-expert transition in $\mathcal{D}_i$ with its smoothed advantage $\hat{A}$; (3) update the policy from π_{i-1} on $\mathcal{D}_i$ with the TRFO objective in Sec. 2.3, using π_{i-1} as the frozen reference π_{ref}; and (4) deploy π_i to produce on-policy data for iteration $i+1$. We expect this loop to produce stable, monotone gains in success rate across iterations; we verify this empirically in Sec. 3.1.

Temporal alignment relaxation for chunk blending. Chunk-based policies require a mechanism to stitch consecutive action chunks into smooth motion at deployment time. Existing training-free blending [31, 32] can leave residual jerk, while training-time RTC [33] couples the blending strategy to the model and complicates the post-training pipeline. We augment the QP-based blending of [32] with a simple temporal alignment relaxation—a least-squares offset search that re-registers incoming chunks to the in-flight trajectory before blending—which substantially improves stitching quality while remaining entirely training-free. The resulting procedure produces smooth, jerk-free motion across both high-speed tasks (e.g., garment folding at 67 items/h) and millimeter-precision tasks (e.g., zip-tie threading), and is directly compatible with the iterative PIVOT loop. Details are given in Appendix 6.2.

3 Experiments

We evaluate PIVOT on four precise, contact-rich bimanual manipulation tasks across two hardware platforms, using $\pi_{0.5}$ [2] as the backbone VLA (see Appendix 6.1 for hardware and control details): LAUNDRY (T-shirt folding from randomized crumpled poses; metric: throughput in items/h), ZIP TIE (picking and threading a flexible zip tie; millimeter-level precision), PAPER BOX (long-horizon multi-stage box assembly from a flat cardboard cutout), and SACHET (threading two sachets into a pair; high dexterity). All tasks require contact-rich bimanual coordination and exhibit substantial variability in initial conditions; detailed descriptions are given in Appendix 6.6 and visualized in Figure 2. Our experiments address three questions: 1. Does the iterative post-training loop produce stable, monotonic gains across successive iterations and diverse tasks? 2. How does TRFO compare with alternative policy extraction methods under the same data budget? 3. Can PIVOT scale to a unified multi-task policy that masters diverse skills while preserving per-task performance?

3.1 Iterative Improvement

We first evaluate whether PIVOT driven by TRFO yields consistent gains across successive real-world post-training iterations. Figure 4 reports performance on four contact-rich bimanual manipulation tasks. Across all tasks, PIVOT improves monotonically over successive iterations: LAUNDRY throughput rises by 26.2% (56.2 → 70.9 items/h over three iterations), ZIP TIE success climbs from 49% to 86%, PAPER BOX from 60% to 82%, and SACHET from 32% to 76%.

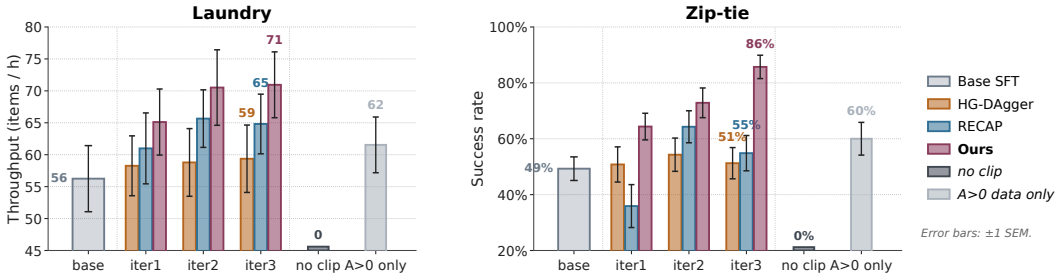


Figure 5: **Policy extraction comparison and ablations.** Laundry throughput and zip-tie success rate across post-training iterations for HG-DAgger, RECAP, TRFO, and ablation variants.

The stable, non-decreasing improvement across tasks with very different manipulation primitives—all of which represent the most challenging contact-rich bimanual coordination problems—confirms that TRFO can reliably convert heterogeneous deployment data into policy gains. Practically, several of these tasks transition from unreliable under the base policy to consistently deployable after only two to three iterations of the loop.

3.2 Policy Extraction Comparison and Ablations

We next compare TRFO with alternative policy extraction methods trained on the same heterogeneous data collected by the PIVOT loop, and ablate two key design choices within TRFO itself. We consider two baselines: **HG-DAgger** [16], which uses demonstrations and human corrections as supervised targets and discards autonomous rollouts; and **RECAP** [10], which performs advantage-conditioned extraction with classifier-free guidance [34] at inference time.

Baselines. Figure 5 reports results under the same data budget. TRFO consistently outperforms both baselines: on LAUNDRY it reaches 71 items/h versus 59 (HG-DAgger) and 65 (RECAP); on ZIP TIE it reaches 86% versus 51% and 55%. Since all methods consume the same data stream, the gap isolates the effect of the extraction operator. HG-DAgger cannot learn from failed rollouts, and RECAP's inference-time conditioning mechanism occasionally degrades across iterations. TRFO consistently yields the strongest results across both tasks and iterations.

Ablations. We ablate two core components of TRFO (Figure 5). **(i)** Removing the trust-region clipping (η-bounded repulsion replaced by unbounded negative-advantage gradients) causes training to diverge within the first few thousand steps—the repulsive signal pushes the policy into unsupported action-space regions, confirming that the frozen-reference trust region is essential. **(ii)** Discarding negative-advantage samples (setting $w = 0$ for $\hat{A} \leq 0$) yields stable training but substantially lower performance: LAUNDRY throughput drops to 62 items/h (vs. 71) and ZIP TIE success falls to 60% (vs. 86%), showing that the repulsive learning signal from negative samples is a meaningful source of improvement that positive-only imitation cannot recover.

3.3 Multi-task Generalization

A key promise of a unified post-training objective is that it should generalize across tasks without per-task modifications, opening the door to scaling a single policy over a growing skill repertoire. We test this by asking whether TRFO can jointly post-train a multi-task policy without sacrificing per-task performance.

Setup. We train a *unified base model* on demonstration data from three tasks—LAUNDRY, ZIP TIE, and SACHET—via supervised flow matching fine-tuning, and then perform a single round of PIVOT post-training on the accumulated experience buffer that pools demonstrations, autonomous rollouts, and human corrections from all three tasks. The model, TRFO update, and training infrastructure are shared across tasks; only the language instruction distinguishes them. We compare this unified PIVOT model against independently trained single-task specialists that each receive the same per-task data budget.

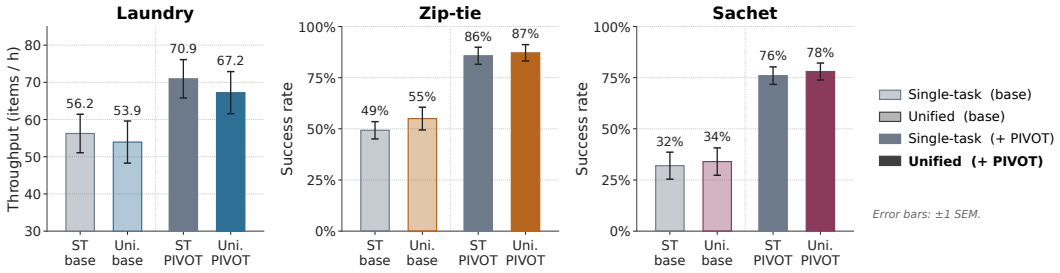


Figure 6: **Unified multi-task post-training.** Comparison between single-task and unified models before and after PIVOT post-training on laundry, zip-tie, and sachet. The laundry plot reports throughput, while zip-tie and sachet report success rate.

Results. Figure 6 reports the comparison. The unified PIVOT model performs on par with the per-task specialists across all three tasks (LAUNDRY 67.2 vs. 70.9 items/h; ZIP TIE 87% vs. 86%; SACHET 78% vs. 76%), and even slightly surpasses the specialists on two of the three tasks. A single policy can thus master both high-speed manipulation (folding garments at 67 items/h) and millimeter-precision tasks (threading flexible zip ties at 87% success) without per-task compromise. This is enabled by the generality of the TRFO objective: because it modulates the standard flow matching loss with per-sample advantages rather than requiring task-specific reward shaping or separate critic architectures, the same objective and infrastructure apply unchanged regardless of the number or diversity of skills, suggesting that PIVOT provides a scalable foundation for post-training generalist manipulation policies.

3.4 Empirical Analysis of Value Estimates and TRFO

To better understand how PIVOT drives iterative improvement from sparse binary rewards, we visualize the learned values and the corresponding TRFO updates in Figures 7 and 8.

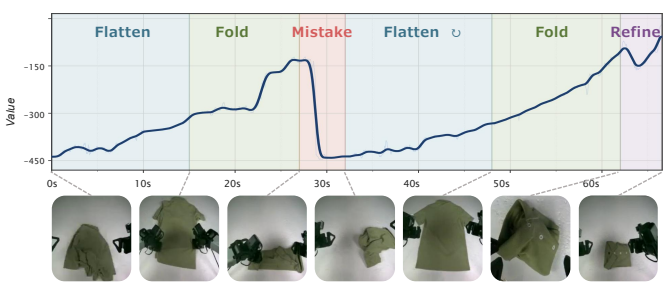


Figure 7: **Visualization of value estimates over a human-teleoperated trajectory.** The value expert tracks task progress over a rollout, increasing during successful progress, dropping after mistakes, and recovering after corrective behavior.

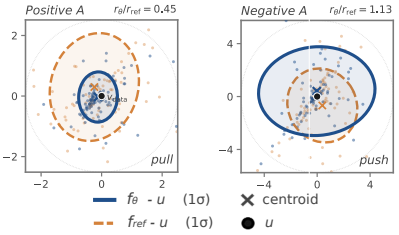


Figure 8: **Empirical Study of TRFO updates.** TRFO pulls the policy toward positive-advantage actions and pushes it away from negative-advantage actions within the trust region.

As shown in Figure 7, the value estimates produced by our distributional value expert closely track episode progress. The value increases steadily during successful execution, drops sharply when the robot makes a mistake, and recovers when corrective behavior is applied. This yields a well-shaped, dense advantage signal for policy optimization.

Figure 8 visualizes the effect of TRFO on individual transitions. We randomly sample transitions from the training set, and for each sample compute the velocity residuals $f_\theta(o, x_\tau, \tau) - u_\tau$ (blue; post-update policy) and $f_{\text{ref}}(o, x_\tau, \tau) - u_\tau$ (orange; pre-update reference), where the origin corresponds to the data velocity u_τ, i.e. perfect prediction. The resulting residual vectors are projected to 2D via PCA across all samples, and a 1σ covariance ellipse is overlaid for each distribution. The ratio $r_\theta / r_{\text{ref}}$ ($r = \sqrt{\ell}$, the root-mean-square flow matching loss from Eq. 1) measures how the update shifts the policy relative to the reference. For positive-advantage transitions, TRFO contracts the distribution toward the origin ($r_\theta / r_{\text{ref}} < 1$), pulling the policy closer to the demonstrated action; for negative-advantage transitions, it expands the distribution away ($r_\theta / r_{\text{ref}} > 1$), repelling the policy from low-value behavior within the trust region.

4 Related Work

Robot learning from demonstrations, corrections, and rollouts. Behavior cloning from offline demonstrations remains the dominant approach for training visuomotor policies [17, 18]. Interactive imitation methods [15, 16, 35, 36] such as HG-DAgger [16] incorporate human corrections into the training loop, while reinforcement learning learns directly from autonomous or on-policy rollouts on physical hardware [20–24, 37, 38]. Our work unifies all three data sources—demonstrations, corrections, and rollouts—under a single policy-update objective that treats each source according to its estimated advantage.

VLA post-training and deployment experience loops. Recent work has increasingly framed VLA post-training as a closed-loop learn-from-experience process, where curated demonstrations are combined with human corrections and autonomous rollouts collected by deployed policies, and the policy improves through repeated redeployment [9–12]. RECAP/$\pi^*_{0.6}$ [10] integrates heterogeneous experience through advantage-conditioned policy extraction; χ_0 [11] addresses long-horizon deployment failures through distributional alignment and progress estimation; and SOP [9] and LWD [12] scale this loop to fleet-level online post-training. In parallel, recent VLA fine-tuning methods explore on-policy RL, value-based optimization, residual policy learning, preference/reward optimization, and trajectory-level policy updates [39–53]. Our work focuses on the policy-improvement operator inside the experience loop, providing a unified trust-region flow-matching objective for demonstrations, corrections, and autonomous rollouts.

Policy optimization for flow and diffusion policies. Classical offline RL methods perform policy extraction through advantage-weighted regression, improving a policy by upweighting high-value actions while remaining close to the data distribution [54–57]. For diffusion and flow policies, recent work optimizes denoising trajectories, treats the sampling process as an auxiliary decision process, applies critic-gradient or adjoint-matching updates, or steers generation with learned value or energy guidance [25, 26, 43, 58–66]. In contrast, TRFO preserves the standard flow-matching interface and performs policy improvement through a signed, advantage-weighted trust-region objective.

5 Conclusion

We presented **PIVOT**, an iterative post-training system that converts heterogeneous real-world experience—demonstrations, human corrections, and autonomous rollouts—into stable policy improvement for flow-based embodied foundation models. At its core is *Trust-Region Flow Matching Optimization (TRFO)*, a unified objective that modulates the flow-matching loss with signed advantages; This allows PIVOT to safely leverage the full spectrum of heterogeneous deployment data—including sub-optimal rollouts, failure trajectories, and human corrections—without degrading the generative policy. Across a suite of precise, contact-rich bimanual manipulation tasks, PIVOT delivers consistent, monotonic gains over successive iterations, substantially improving success rates and throughput across all evaluated tasks. Notably, several tasks that were initially unreliable under the base policy become practically deployable after only a few iterations of the loop, and the improvements hold uniformly across long-horizon, contact-rich tasks that demand high precision and dexterous bimanual coordination. These results demonstrate that a closed-loop heterogeneous experience loop, equipped with a trust-region RL objective that preserves the simplicity of flow matching, can bridge the gap between broad pretraining and reliable real-world deployment.

Limitations. While PIVOT shows consistent gains across the evaluated real-world tasks, our experiments remain limited in scale. In particular, the unified post-training setting currently covers only a small set of tasks, and future work should study substantially larger and more diverse task repertoires. In addition, our environments are relatively structured: all evaluations are conducted in fixed desktop workspaces with controlled robot setups. Extending the approach to more open-ended and dynamically changing environments remains an important direction for future work.

Acknowledgments

This work was supported by the Tsinghua University–Keystone Electrical (Zhejiang) Co., Ltd. Joint Research Center for Embodied Multimodal Artificial Intelligence (JCEMAI) and the Beijing Natural Science Foundation Undergraduate Research Program (Grant No. QY26437).

References

[1] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2024. URL https://arxiv.org/abs/2410.24164.

[2] Physical Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, M. Y. Galliker, D. Ghosh, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, D. LeBlanc, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, A. Z. Ren, L. X. Shi, L. Smith, J. T. Springenberg, K. Stachowicz, J. Tanner, Q. Vuong, H. Walke, A. Walling, H. Wang, L. Yu, and U. Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

[3] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. S. Ryoo, G. Salazar, P. R. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. H. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich. RT-1: Robotics Transformer for Real-World Control at Scale. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi:10.15607/RSS.2023.XIX.025.

[4] 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. In J. Tan, M. Toussaint, and K. Darvish, editors, *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 2165–2183. PMLR, 06–09 Nov 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

[5] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn. OpenVLA: An Open-Source Vision-Language-Action Model, 2024. URL https://arxiv.org/abs/2406.09246.

[6] D. Ghosh, H. R. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, J. Luo, Y. L. Tan, L. Y. Chen, Q. Vuong, T. Xiao, P. R. Sanketi, D. Sadigh, C. Finn, and S. Levine. Octo: An Open-Source Generalist Robot Policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi:10.15607/RSS.2024.XX.090.

[7] Gemini Robotics Team, S. Abeyruwan, J. Ainslie, J.-B. Alayrac, M. G. Arenas, T. Armstrong, A. Balakrishna, R. Baruch, M. Bauza, M. Blokzijl, S. Bohez, K. Bousmalis, A. Brohan, T. Buschmann, A. Byravan, S. Cabi, K. Caluwaerts, F. Casarini, O. Chang, J. E. Chen, X. Chen, H.-T. L. Chiang, K. Choromanski, D. D'Ambrosio, S. Dasari, T. Davchev, C. Devin, N. D. Palo, T. Ding, A. Dostmohamed, D. Driess, Y. Du, D. Dwibedi, M. Elabd, C. Fantacci, C. Fong, E. Frey, C. Fu, M. Giustina, K. Gopalakrishnan, L. Graesser, L. Hasenclever, N. Heess, B. Hernaez, A. Herzog, R. A. Hofer, J. Humplik, A. Iscen, M. G. Jacob, D. Jain, R. Julian, D. Kalashnikov, M. E. Karagozler, S. Karp, C. Kew, J. Kirkland, S. Kirmani,

Y. Kuang, T. Lampe, A. Laurens, I. Leal, A. X. Lee, T.-W. E. Lee, J. Liang, Y. Lin, S. Maddineni, A. Majumdar, A. H. Michaely, R. Moreno, M. Neunert, F. Nori, C. Parada, E. Parisotto, P. Pastor, A. Pooley, K. Rao, K. Reymann, D. Sadigh, S. Saliceti, P. Sanketi, P. Sermanet, D. Shah, M. Sharma, K. Shea, C. Shu, V. Sindhwani, S. Singh, R. Soricut, J. T. Springenberg, R. Sterneck, R. Surdulescu, J. Tan, J. Tompson, V. Vanhoucke, J. Varley, G. Vesom, G. Vezzani, O. Vinyals, A. Wahid, S. Welker, P. Wohlhart, F. Xia, T. Xiao, A. Xie, J. Xie, P. Xu, S. Xu, Y. Xu, Z. Xu, Y. Yang, R. Yao, S. Yaroshenko, W. Yu, W. Yuan, J. Zhang, T. Zhang, A. Zhou, and Y. Zhou. Gemini robotics: Bringing ai into the physical world, 2025. URL https://arxiv.org/abs/2503.20020.

[8] C. Cheang, S. Chen, Z. Cui, Y. Hu, L. Huang, T. Kong, H. Li, Y. Li, Y. Liu, X. Ma, H. Niu, W. Ou, W. Peng, Z. Ren, H. Shi, J. Tian, H. Wu, X. Xiao, Y. Xiao, J. Xu, and Y. Yang. GR-3 Technical Report, 2025. URL https://arxiv.org/abs/2507.15493.

[9] M. Pan, S. Feng, Q. Zhang, X. Li, J. Song, C. Qu, Y. Wang, C. Li, Z. Xiong, Z. Chen, Y. Liu, and J. Luo. SOP: A scalable online post-training system for vision-language-action models, 2026. URL https://arxiv.org/abs/2601.03044.

[10] Physical Intelligence, A. Amin, R. Aniceto, A. Balakrishna, K. Black, K. Conley, G. Connors, J. Darpinian, K. Dhabalia, J. DiCarlo, D. Driess, M. Equi, A. Esmail, Y. Fang, C. Finn, C. Glossop, T. Godden, I. Goryachev, L. Groom, H. Hancock, K. Hausman, G. Hussein, B. Ichter, S. Jakubczak, R. Jen, T. Jones, B. Katz, L. Ke, C. Kuchi, M. Lamb, D. LeBlanc, S. Levine, A. Li-Bell, Y. Lu, V. Mano, M. Mothukuri, S. Nair, K. Pertsch, A. Z. Ren, C. Sharma, L. X. Shi, L. Smith, J. T. Springenberg, K. Stachowicz, W. Stoeckle, A. Swerdlow, J. Tanner, M. Torne, Q. Vuong, A. Walling, H. Wang, B. Williams, S. Yoo, L. Yu, U. Zhilinsky, and Z. Zhou. $\pi^*_{0.6}$: a VLA that learns from experience, 2025. URL https://arxiv.org/abs/2511.14759.

[11] C. Yu, C. Sima, G. Jiang, H. Zhang, H. Mai, H. Li, H. Wang, J. Chen, K. Wu, L. Chen, L. Zhao, M. Shi, P. Luo, Q. Bu, S. Peng, T. Li, and Y. Yuan. χ_0: Resource-aware robust manipulation via taming distributional inconsistencies, 2026. URL https://arxiv.org/abs/2602.09021.

[12] Y. Wang, X. Li, P. Xie, P. Yang, B. Nie, Y. Cai, Q. Zhang, C. Qu, J. Wu, J. Song, X. Ren, J. Huang, M. Pan, S. Feng, Z. Chen, and J. Luo. Learning while deploying: Fleet-scale reinforcement learning for generalist robot policies, 2026. URL https://arxiv.org/abs/2605.00416.

[13] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling, 2023. URL https://arxiv.org/abs/2210.02747.

[14] Q. Liu. Rectified flow: A marginal preserving approach to optimal transport, 2022. URL https://arxiv.org/abs/2209.14577.

[15] S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In G. Gordon, D. Dunson, and M. Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 627–635, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL https://proceedings.mlr.press/v15/ross11a.html.

[16] 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, 2019. doi:10.1109/ICRA.2019.8793698.

[17] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware, 2023. URL https://arxiv.org/abs/2304.13705.

[18] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. C. Burchfiel, and S. Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi:10.15607/RSS.2023.XIX.026.

[19] Z. Hu, R. Wu, N. Enock, J. Li, R. Kadakia, Z. Erickson, and A. Kumar. RaC: Robot learning for long-horizon tasks by scaling recovery and correction, 2025. URL https://arxiv.org/abs/2509.07953.

[20] J. Luo, Z. Hu, C. Xu, Y. L. Tan, J. Berg, A. Sharma, S. Schaal, C. Finn, A. Gupta, and S. Levine. SERL: A software suite for sample-efficient robotic reinforcement learning. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 16961–16969, 2024. doi: 10.1109/ICRA57147.2024.10610040.

[21] J. Luo, C. Xu, J. Wu, and S. Levine. Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning. *Science Robotics*, 10(105):eads5033, 2025. doi:10.1126/scirobotics.ads5033. URL https://www.science.org/doi/abs/10.1126/scirobotics.ads5033.

[22] K. Lei, H. Li, D. Yu, Z. Wei, L. Guo, Z. Jiang, Z. Wang, S. Liang, and H. Xu. RL-100: Performant robotic manipulation with real-world reinforcement learning, 2025. URL https://arxiv.org/abs/2510.14830.

[23] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine. QT-Opt: Scalable Deep Reinforcement Learning for Vision-Based Robotic Manipulation, 2018. URL https://arxiv.org/abs/1806.10293.

[24] D. Kalashnikov, J. Varley, Y. Chebotar, B. Swanson, R. Jonschkowski, C. Finn, S. Levine, and K. Hausman. MT-Opt: Continuous Multi-Task Robotic Reinforcement Learning at Scale, 2021. URL https://arxiv.org/abs/2104.08212.

[25] A. Z. Ren, J. Lidard, L. L. Ankile, A. Simeonov, P. Agrawal, A. Majumdar, B. Burchfiel, H. Dai, and M. Simchowitz. Diffusion policy policy optimization, 2024. URL https://arxiv.org/abs/2409.00588.

[26] D. McAllister, S. Ge, B. Yi, C. M. Kim, E. Weber, H. Choi, H. Feng, and A. Kanazawa. Flow matching policy gradients, 2025. URL https://arxiv.org/abs/2507.21053.

[27] M. G. Bellemare, W. Dabney, and R. Munos. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, pages 449–458. Pmlr, 2017.

[28] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller. Playing atari with deep reinforcement learning, 2013. URL https://arxiv.org/abs/1312.5602.

[29] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz. Trust region policy optimization. In *International Conference on Machine Learning*, pages 1889–1897. PMLR, 2015.

[30] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

[31] K. Black, M. Galliker, and S. Levine. Real-time execution of action chunking flow policies. *Advances in Neural Information Processing Systems*, 2025. URL https://arxiv.org/abs/2506.07339.

[32] G. Gao, J. Wang, J. Zuo, J. Jiang, J. Zhang, X. Zeng, Y. Zhu, L. Ma, K. Chen, M. Sheng, R. Zhang, and Z. An. Towards human-level intelligence via human-like whole-body manipulation, 2025. URL https://arxiv.org/abs/2507.17141.

[33] K. Black, A. Z. Ren, M. Equi, and S. Levine. Training-time action conditioning for efficient real-time chunking, 2025. URL https://arxiv.org/abs/2512.05964.

[34] J. Ho and T. Salimans. Classifier-free diffusion guidance, 2022. URL https://arxiv.org/abs/2207.12598.

[35] M. Laskey, S. Staszak, W. Y.-S. Hsieh, J. Mahler, F. T. Pokorny, A. D. Dragan, and K. Goldberg. SHIV: Reducing supervisor burden in DAgger using support vectors for efficient learning from demonstrations in high dimensional state spaces. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 462–469, 2016. doi:10.1109/ICRA.2016.7487167.

[36] M. Laskey, J. Lee, R. Fox, A. Dragan, and K. Goldberg. DART: Noise Injection for Robust Imitation Learning. In S. Levine, V. Vanhoucke, and K. Goldberg, editors, *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 143–156. PMLR, 13–15 Nov 2017. URL https://proceedings.mlr.press/v78/laskey17a.html.

[37] T. Lampe, A. Abdolmaleki, S. Bechtle, S. H. Huang, J. Tobias Springenberg, M. Bloesch, O. Groth, R. Hafner, T. Hertweck, M. Neunert, M. Wulfmeier, J. Zhang, F. Nori, N. Heess, and M. Riedmiller. Mastering stacking of diverse shapes with large-scale iterative reinforcement learning on real robots. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7772–7779, 2024. doi:10.1109/ICRA57147.2024.10610297.

[38] H. Zhu, J. Yu, A. Gupta, D. Shah, K. Hartikainen, A. Singh, V. Kumar, and S. Levine. The ingredients of real world robotic reinforcement learning. In *International Conference on Learning Representations*, 2020. URL https://arxiv.org/abs/2004.12570.

[39] S. Tan, K. Dou, Y. Zhao, and P. Krähenbühl. Interactive post-training for vision-language-action models, 2025. URL https://arxiv.org/abs/2505.17016.

[40] 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, 2025. URL https://arxiv.org/abs/2505.18719.

[41] K. Chen, Z. Liu, T. Zhang, Z. Guo, S. Xu, H. Lin, H. Zang, X. Li, Q. Zhang, Z. Yu, G. Fan, T. Huang, Y. Wang, and C. Yu. π_{RL}: Online RL fine-tuning for flow-based vision-language-action models, 2025. URL https://arxiv.org/abs/2510.25889.

[42] D. Huang, Z. Fang, T. Zhang, Y. Li, L. Zhao, and C. Xia. CO-RFT: Efficient fine-tuning of vision-language-action models through chunked offline reinforcement learning, 2025. URL https://arxiv.org/abs/2508.02219.

[43] Y. Chen, S. Tian, S. Liu, Y. Zhou, H. Li, and D. Zhao. ConRFT: A Reinforced Fine-tuning Method for VLA Models via Consistency Policy. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi:10.15607/RSS.2025.XXI.019.

[44] C. Xu, Q. Li, J. Luo, and S. Levine. RLDG: Robotic Generalist Policy Distillation via Reinforcement Learning. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi:10.15607/RSS.2025.XXI.028.

[45] L. Ankile, Z. Jiang, R. Duan, G. Shi, P. Abbeel, and A. Nagabandi. Residual Off-Policy RL for Finetuning Behavior Cloning Policies, 2025. URL https://arxiv.org/abs/2509.19301.

[46] W. Xiao, H. Lin, A. Peng, H. Xue, T. He, Y. Xie, F. Hu, J. Wu, Z. Luo, L. J. Fan, G. Shi, and Y. Zhu. Self-Improving Vision-Language-Action Models with Data Generation via Residual RL, 2025. URL https://arxiv.org/abs/2511.00091.

[47] S. K. S. Ghasemipour, A. Wahid, J. Tompson, P. Sanketi, and I. Mordatch. Self-improving embodied foundation models, 2025. URL https://arxiv.org/abs/2509.15155.

[48] S. Zhai, Q. Zhang, T. Zhang, F. Huang, H. Zhang, M. Zhou, S. Zhang, L. Liu, S. Lin, and J. Pang. A vision-language-action-critic model for robotic real-world reinforcement learning, 2025. URL https://arxiv.org/abs/2509.15937.

[49] 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. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 15665–15672. IEEE, 2025.

[50] Y. Zhang, C. Wang, O. Lu, Y. Zhao, Y. Ge, Z. Sun, X. Li, C. Zhang, C. Bai, and X. Li. Align-then-steer: Adapting the vision-language action models through unified latent guidance, 2025. URL https://arxiv.org/abs/2509.02055.

[51] H. Li, Y. Zuo, J. Yu, Y. Zhang, Z. Yang, K. Zhang, X. Zhu, Y. Zhang, T. Chen, G. Cui, D. Wang, D. Luo, Y. Fan, Y. Sun, J. Zeng, J. Pang, S. Zhang, Y. Wang, Y. Mu, B. Zhou, and N. Ding. Simplevla-rl: Scaling vla training via reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026.

[52] Z. Chen, R. Niu, H. Kong, Q. Wang, Q. Xing, and Z. Fan. TGRPO: Fine-tuning Vision-Language-Action Model via Trajectory-wise Group Relative Policy Optimization, 2025. URL https://arxiv.org/abs/2506.08440.

[53] Y. Li, X. Ma, J. Xu, Y. Cui, Z. Cui, Z. Han, L. Huang, T. Kong, Y. Liu, H. Niu, W. Peng, J. Qiao, Z. Ren, H. Shi, Z. Su, J. Tian, Y. Xiao, S. Zhang, L. Zheng, H. Li, and Y. Wu. GR-RL: Going dexterous and precise for long-horizon robotic manipulation, 2025. URL https://arxiv.org/abs/2512.01801.

[54] X. B. Peng, A. Kumar, G. Zhang, and S. Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning, 2019. URL https://arxiv.org/abs/1910.00177.

[55] A. Nair, A. Gupta, M. Dalal, and S. Levine. AWAC: Accelerating Online Reinforcement Learning with Offline Datasets, 2021. URL https://arxiv.org/abs/2006.09359.

[56] I. Kostrikov, A. Nair, and S. Levine. Offline Reinforcement Learning with Implicit Q-Learning, 2021. URL https://arxiv.org/abs/2110.06169.

[57] Z. Wang, A. Novikov, K. Zolna, J. S. Merel, J. T. Springenberg, S. E. Reed, B. Shahriari, N. Siegel, C. Gulcehre, N. Heess, et al. Critic regularized regression. *Advances in Neural Information Processing Systems*, 33:7768–7778, 2020.

[58] Q. Li and S. Levine. Q-learning with adjoint matching, 2026. URL https://arxiv.org/abs/2601.14234.

[59] C. Domingo-Enrich, M. Drozdzal, B. Karrer, and R. T. Q. Chen. Adjoint matching: Fine-tuning flow and diffusion generative models with memoryless stochastic optimal control, 2025. URL https://arxiv.org/abs/2409.08861.

[60] J. Liu, G. Liu, J. Liang, Y. Li, J. Liu, X. Wang, P. Wan, D. Zhang, and W. Ouyang. Flow-GRPO: Training Flow Matching Models via Online RL, 2025. URL https://arxiv.org/abs/2505.05470.

[61] T. Zhang, C. Yu, S. Su, and Y. Wang. ReinFlow: Fine-tuning flow matching policy with online reinforcement learning, 2025. URL https://arxiv.org/abs/2505.22094.

[62] S. Park, Q. Li, and S. Levine. Flow q-learning, 2025. URL https://arxiv.org/abs/2502.02538.

[63] S. Zhang, W. Zhang, and Q. Gu. Energy-weighted flow matching for offline reinforcement learning, 2025. URL https://arxiv.org/abs/2503.04975.

[64] K. Frans, S. Park, P. Abbeel, and S. Levine. Diffusion guidance is a controllable policy improvement operator, 2025. URL https://arxiv.org/abs/2505.23458.

[65] A. Wagenmaker, Y. Zhang, M. Nakamoto, S. Park, W. Yagoub, A. Nagabandi, A. Gupta, and S. Levine. Steering your diffusion policy with latent space reinforcement learning. In J. Lim, S. Song, and H.-W. Park, editors, *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pages 258–282. PMLR, 27–30 Sep 2025. URL https://proceedings.mlr.press/v305/wagenmaker25a.html.

[66] Q. Li, Z. Zhou, and S. Levine. Reinforcement learning with action chunking, 2025. URL https://arxiv.org/abs/2507.07969.

[67] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel. GELLO: A General, Low-Cost, and Intuitive Teleoperation Framework for Robot Manipulators, 2024. URL https://arxiv.org/abs/2309.13037.

[68] Gemma Team, T. Mesnard, C. Hardin, R. Dadashi, S. Bhupatiraju, S. Pathak, L. Sifre, M. Rivière, M. S. Kale, J. Love, P. Tafti, L. Hussenot, P. G. Sessa, A. Chowdhery, A. Roberts, A. Barua, A. Botev, A. Castro-Ros, A. Slone, A. Héliou, A. Tacchetti, A. Bulanova, A. Paterson, B. Tsai, B. Shahriari, C. L. Lan, C. A. Choquette-Choo, C. Crepy, D. Cer, D. Ippolito, D. Reid, E. Buchatskaya, E. Ni, E. Noland, G. Yan, G. Tucker, G.-C. Muraru, G. Rozhdestvenskiy, H. Michalewski, I. Tenney, I. Grishchenko, J. Austin, J. Keeling, J. Labanowski, J.-B. Lespiau, J. Stanway, J. Brennan, J. Chen, J. Ferret, J. Chiu, J. Mao-Jones, K. Lee, K. Yu, K. Millican, L. L. Sjoesund, L. Lee, L. Dixon, M. Reid, M. Mikuła, M. Wirth, M. Sharman, N. Chinaev, N. Thain, O. Bachem, O. Chang, O. Wahltinez, P. Bailey, P. Michel, P. Yotov, R. Chaabouni, R. Comanescu, R. Jana, R. Anil, R. McIlroy, R. Liu, R. Mullins, S. L. Smith, S. Borgeaud, S. Girgin, S. Douglas, S. Pandya, S. Shakeri, S. De, T. Klimenko, T. Hennigan, V. Feinberg, W. Stokowiec, Y. hui Chen, Z. Ahmed, Z. Gong, T. Warkentin, L. Peran, M. Giang, C. Farabet, O. Vinyals, J. Dean, K. Kavukcuoglu, D. Hassabis, Z. Ghahramani, D. Eck, J. Barral, F. Pereira, E. Collins, A. Joulin, N. Fiedel, E. Senter, A. Andreev, and K. Kenealy. Gemma: Open models based on gemini research and technology, 2024. URL https://arxiv.org/abs/2403.08295.

[69] J. Farebrother, J. Orbay, Q. Vuong, A. A. Taïga, Y. Chebotar, T. Xiao, A. Irpan, S. Levine, P. S. Castro, A. Faust, A. Kumar, and R. Agarwal. Stop regressing: Training value functions via classification for scalable deep rl, 2024. URL https://arxiv.org/abs/2403.03950.

[70] I. Loshchilov and F. Hutter. Decoupled weight decay regularization, 2019. URL https://arxiv.org/abs/1711.05101.

[71] B. Yi, H. Choi, H. G. Singh, X. Huang, T. E. Truong, C. Sferrazza, Y. Ma, R. Duan, P. Abbeel, G. Shi, K. Liu, and A. Kanazawa. Flow policy gradients for robot control, 2026. URL https://arxiv.org/abs/2602.02481.

6 Appendix

6.1 Robot Setup

We conduct experiments on two bimanual manipulation platforms.

YAM i2rt. The first platform is a bimanual YAM i2rt system equipped with two 6-DoF arms. The observation space consists of three Intel RealSense RGB cameras (one head-mounted and two wrist-mounted). The policy runs at 30 Hz. Demonstration data is collected via GELLO [67] teleoperation with two leader arms kinematically matched to the two follower arms. Tasks evaluated on this platform include LAUNDRY, ZIP TIE, and SACHET.

UR. The second platform uses a pair of Universal Robots 6-DoF arms. The observation space likewise consists of three Intel RealSense RGB cameras (one head-mounted and two wrist-mounted), and the policy runs at 30 Hz. Demonstrations are collected via GELLO [67] teleoperation. The task evaluated on this platform is PAPER BOX.

Control interface. Both platforms share a common asynchronous chunk-based control interface. The robot is controlled by a joint-space PD controller. Let $s_t \in \mathbb{R}^d$ denote the measured joint configuration at timestep t and $a_t \in \mathbb{R}^d$ the PD setpoint to be commanded at timestep t. The policy outputs an action chunk $[a_t, \dots, a_{t+H-1}]$ in *delta* form $a'_{t+j} = a_{t+j} - s_t$, parameterized relative to the configuration s_t at the chunk's start. At each inference trigger, the local controller sends (o_t, ℓ) to a remote inference server; the server returns the delta chunk $[a'_{t+j}]_{j=0}^{H-1}$, from which the controller recovers the absolute setpoints $a_{t+j} = s_t + a'_{t+j}$ and feeds them to the PD loop.

6.2 Temporal Alignment Relaxation for Chunk Blending

Chunk-based action generation introduces a practical deployment challenge: the policy runs on a remote inference server, and variable network and computation latency causes the freshly returned chunk to be delayed by several control steps relative to the moment at which it was conditioned. Naively switching from the old to the new chunk produces a discontinuity in commanded position and audible jerk on the robot. As discussed in Sec. 2.4, existing training-free blending [31, 32] can leave residual jerk, while training-time approaches such as RTC [33] couple the blending strategy to the model and complicate iterative post-training. We augment the QP-based blending of [32] with a temporal alignment relaxation step that substantially improves stitching quality while remaining entirely training-free.

Temporal alignment relaxation. The key idea is to re-register the incoming chunk to the in-flight trajectory before blending. Let $[a_t, a_{t+1}, \dots]$ denote the chunk currently in execution at wall-clock time t, and let $[a'_{t-k}, \dots, a'_{t-k+H-1}]$ denote the newly received chunk indexed in its own timeframe (i.e., conditioned on o_{t-k}). We search over a small offset window $[-K, K]$ for the alignment that best matches the leading M steps of the new chunk to the corresponding steps of the in-flight chunk:

$$c^\star \;=\; \arg\min_{c \in [-K,\, K]} \; \sum_{j=0}^{M-1} \left\| a_{t+j} - a'_{t+c+j} \right\|_2^2, \tag{4}$$

with $K = 5$ and $M = 3$. The new chunk is then truncated to $[a'_{t+c^\star}, \dots, a'_{t-k+H-1}]$. Because the alignment is a pure temporal shift, it does not alter action values—only the time index at which they are applied. This ensures that the subsequent QP blending operates on well-registered trajectories in action space, producing smooth handovers without sacrificing the precision of the original predictions.

QP-based trajectory blending. On the overlap between the aligned new chunk and the in-flight chunk, we solve a lightweight quadratic program (QP) following [32] to produce a smooth handover. Let $x = [x_0, x_1, \dots, x_L]$ denote the remaining segment of the old chunk and $y = [y_0, y_1, \dots, y_L]$

denote the aligned segment of the new chunk over a blending horizon L, both prepended with the last executed action a_{prev} to enforce continuity. We solve for a blended trajectory $z = [z_0, \dots, z_L]$ that minimizes:

$$\min_{z} \underbrace{w_{\text{smooth}} \sum_{j=1}^{L-1} \|z_{j+1} - 2z_j + z_{j-1}\|^2}_{\text{acceleration penalty}} + \sum_{j=0}^{L} \Big[\underbrace{\lambda^j \|z_j - x_j\|^2}_{\text{old chunk}} + \underbrace{(1-\lambda^j)\|z_j - y_j\|^2}_{\text{new chunk}} \Big], \tag{5}$$

subject to the continuity constraint $z_0 = a_{\text{prev}}$. The decay factor $\lambda \in (0, 1)$ controls the transition speed: at $j = 0$ the objective strongly favors the old chunk, while for large j it favors the new chunk. The acceleration penalty w_{smooth} suppresses jerk in the commanded trajectory. We solve this QP via SLSQP. For the first chunk of each episode, where no old chunk exists, we apply a simplified single-chunk QP that only minimizes the acceleration penalty while satisfying the continuity constraint from the current robot configuration.

6.3 Sparse Reward Details

We adopt a task-agnostic sparse reward that requires only a single binary success label per episode. Concretely, given an episode $\{(o_t, a_t, r_t)\}_{t=1}^{n}$, we set $r_t = -1$ for every non-terminal step $t < n$ and assign a terminal reward of $r_n = 0$ on success or $r_n = -\frac{1}{1-\gamma}$ on failure. Under this convention, the discounted return $\sum_{t=1}^{n} \gamma^{t-1} r_t$ is bounded in $\left[V_{\min} = -\frac{1}{1-\gamma}, V_{\max} = 0\right]$ regardless of task or horizon, so the value function can be read directly as a normalized signal of remaining progress to task completion. Because the reward is fully determined by a single per-episode label, the same definition applies to long- and short-horizon tasks without per-task tuning—an important property for training a single multi-task value function across heterogeneous skills. The chunk-level bootstrap targets V_{tgt} and the sparse reward convention are defined in Sec. 2.1.

6.4 Value Expert: Detailed Architecture

The value expert is instantiated as a Gemma decoder [68] that shares the vision–language prefix KV-cache produced by the VLA's VLM backbone, so that the perceptual encoding of o_t is computed only once per inference and reused for both action generation and value estimation. We introduce a learned aggregation token $e_V \in \mathbb{R}^d$, which acts as a readout slot for the state value. The value expert's input token sequence is simply $\left[\, e_V \,\right]$. The aggregation token is read out by a residual MLP head that produces logits $z \in \mathbb{R}^K$ over $K = 512$ uniformly spaced bins on the support $[V_{\min}, V_{\max}] = \left[-\frac{1}{1-\gamma}, 0\right]$. The scalar estimate is recovered as $\hat{V} = \sum_{k=1}^{K} p_k \, c_k$ with $p = \text{softmax}(z)$ and $\{c_k\}$ the bin centers.

The aggregation token cross-attends to the VLM prefix KV-cache, so $V(o_t)$ is produced in a single forward pass through the value expert. At inference time, the shared prefix KV-cache lets us amortize the expensive VLM forward across both action generation and value estimation: the VLM is run only once per observation and reused by the value expert.

6.5 Distributional Value Loss: Implementation Details

The value expert is trained with the categorical cross-entropy loss

$$\mathcal{L}_{\text{value}} \;=\; -\sum_{k} \hat{p}_k^{V} \, \log p_k^{V}, \tag{6}$$

where the categorical targets $\hat{p}^V$ are obtained by projecting the scalar bootstrap target V_{tgt} from Sec. 2.2 onto the K support bins via the standard two-bin projection of [27]. Concretely, given a scalar target V_{tgt} and bin centers $c_1 < c_2 < \cdots < c_K$, let $k^\star$ be the largest index with $c_{k^\star} \le V_{\text{tgt}}$. We set

$$\hat{p}_{k^\star}^{V} \;=\; \frac{c_{k^\star+1} - V_{\text{tgt}}}{c_{k^\star+1} - c_{k^\star}}, \qquad \hat{p}_{k^\star+1}^{V} \;=\; \frac{V_{\text{tgt}} - c_{k^\star}}{c_{k^\star+1} - c_{k^\star}}, \tag{7}$$

and zero on all other bins. The bootstrap target V_{tgt} naturally lies within $[V_{\min}, V_{\max}]$, so no clipping is needed. This linear interpolation ensures both that the cross-entropy gradient is well-defined when the target falls strictly between two bin centers, and that the expectation of the projected categorical distribution recovers the original scalar exactly.

Bootstrapping uses a separate target value expert with parameters ϕ^-, maintained as a Polyak average of the online parameters ϕ via $\phi^- \leftarrow \tau\,\phi + (1-\tau)\,\phi^-$ after every gradient step. The bootstrap target V_{tgt} is evaluated under ϕ^- with stop-gradient before being projected to the categorical targets above, in line with standard practice in distributional TD learning [27, 69].

6.6 Task Descriptions

- LAUNDRY. The robot folds T-shirts from a randomized initial configuration. The task set includes six T-shirts that vary in size, shape, and color, and each episode begins with the garment placed in an arbitrary crumpled pose. A fold is counted as successful if the robot completes it within 180 s. We report *throughput* (items/h) as the primary metric, which captures both speed and reliability. Each evaluation round consists of approximately 60 episodes.
- ZIP TIE. A fine-grained precision task that requires the robot to pick a single zip tie from a bin, align the tip of the tie with its locking head, and thread it through—all demanding millimeter-level accuracy and robust handling of a flexible, deformable object. Each evaluation round consists of approximately 70 episodes.
- PAPER BOX. Starting from a flat cardboard cutout whose initial pose and position are randomized, the robot must execute a long-horizon sequence of folds to form an open-top box, coil a rope and place it inside, and finally close the box. The task tests complex, multi-stage manipulation with generalization over the initial configuration of the cardboard. Each evaluation round consists of approximately 40 episodes.
- SACHET. An intricate dexterous task in which the robot threads two sachets into a pair—symbolizing the Chinese idiom "good things come in pairs." The task demands both high dexterity and fine precision, and the initial positions of the sachets are heavily randomized to test generalization. Each evaluation round consists of approximately 50 episodes.

6.7 Dataset & Iteration Details

Base model. The base policy π_0 for each task is obtained via HG-DAgger [16]-style supervised fine-tuning of the pretrained VLA. The training set $\mathcal{D}_0$ consists of two data sources: (i) curated human demonstrations and (ii) human corrections collected by deploying intermediate checkpoints and allowing the operator to intervene. While this demo-plus-correction recipe provides a reasonable starting point, the performance ceiling reachable by supervised imitation alone is limited.

Per-iteration data budget. Table 1 summarizes the approximate number of episodes collected at each stage of the iterative post-training loop. For each task, $\mathcal{D}_0$ denotes the initial dataset used to train the base policy π_0, and $\Delta\mathcal{D}_i$ denotes the new episodes (autonomous rollouts and human corrections) appended at iteration i. The cumulative dataset $\mathcal{D}_i = \mathcal{D}_{i-1} \cup \Delta\mathcal{D}_i$ is used for both value-function training and the TRFO policy update at each iteration.

6.8 Training Hyperparameters

Table 2 lists the hyperparameters used for the base SFT policy, the TRFO policy update, and the distributional value expert. All three stages share the same optimizer and learning rate schedule; in each iteration, we first train the value expert to convergence, then update the action expert with TRFO.

Update-to-data ratio for the value expert. A critical design choice is controlling the update-to-data (UTD) ratio for the distributional TD learning of the value expert. Aggressive optimization

Table 1: Approximate dataset sizes (number of episodes) at each stage of PIVOT. $\mathcal{D}_0$ lists the initial demonstration and correction episodes used to train the base policy. For each subsequent iteration $\Delta\mathcal{D}_i$, we report the number of correction episodes (Corr.) and autonomous rollout episodes (Roll.) separately. The rightmost columns report the total cumulative episodes and the corresponding approximate robot hours.

	$\mathcal{D}_0$ (base)		$\Delta\mathcal{D}_1$		$\Delta\mathcal{D}_2$		$\Delta\mathcal{D}_3$			
Task	Demo.	Corr.	Roll.	Corr.	Roll.	Corr.	Roll.	Corr.	**Total eps.**	**Total hrs.**
Laundry	254	115	105	71	97	76	61	63	842	8.6
Zip Tie	200	88	139	61	100	18	70	51	727	7.2
Paper Box	200	0	80	0	73	0	—	—	353	5.2
Sachet	453	106	75	6	50	0	—	—	690	5.2

of the value expert quickly leads to overfitting—the value predictions become overconfident and noisy, which in turn corrupts the advantage estimates used by TRFO. We therefore constrain the value expert to approximately 1 UTD at the data collection frequency of 30 Hz: given a dataset of N transitions, the value expert is trained for roughly N/B gradient steps where B is the batch size, ensuring each transition is seen approximately once on average. We found this regime essential for maintaining stable, monotonic improvement across iterations; higher UTD ratios (e.g., 3–5$\times$) led to value overestimation and degraded policy performance.

Table 2: Hyperparameters for PIVOT. All values are shared across tasks unless otherwise noted.

Category	**Hyperparameter**	**Value**
Shared (all stages)		
	Optimizer	AdamW [70] (β_1=0.9, β_2=0.95, ϵ=10^{-8})
	Peak learning rate	2.5×10^{-5}
	LR schedule	Cosine decay to 2.5×10^{-6} (1,000 warmup)
	Batch size	64
Base SFT		
	Training steps	∼40,000
	Flow-matching time distribution	$\tau \sim \mathcal{U}(0,\, 1)$
TRFO (action expert)		
	Discount factor γ	0.998
	Action chunk horizon H	40
	Sampling steps	10
	Trust-region margin η	1.1
	Advantage scale α	0.2
	Advantage bias b	$[-5,\, 5]$
	Training steps per iteration	∼40,000 (≈1 UTD at 30 Hz)
Value expert		
	Number of categorical bins K	512
	Value MLP hidden dim / residual blocks	1024 / 2
	Polyak averaging τ_{target}	0.005
	Training steps per iteration	≈1 UTD at 30 Hz
Advantage smoothing		
	Smoothing filter	Gaussian ($\sigma = 15$)

6.9 Full Algorithm

Algorithm 1 provides the complete pseudocode for PIVOT, covering the outer iterative loop (Sec. 2.4), the distributional value training and advantage labeling (Sec. 2.2), and the TRFO policy update (Sec. 2.3).

6.10 A Heuristic Loss-Space Interpretation of TRFO

This appendix gives a heuristic interpretation of Trust-Region Flow Matching Optimization (TRFO) as a first-order, loss-space analogue of an advantage-weighted likelihood-ratio update. The discussion is intended to clarify the design of the objective rather than to establish a formal policy-improvement theorem. In particular, we do *not* assume that the flow-matching loss is an exact negative log-likelihood, and we do *not* claim a global KL trust-region guarantee. The statements below should be read as local approximations around a frozen reference policy and as an interpretation of the resulting gradient directions.

Setup. Let $z = (o, a)$ denote a transition consisting of an observation and an action chunk. For a flow-matching policy f_θ, define the per-sample flow-matching loss

$$\ell_\theta(z) = \mathbb{E}_{\tau, x_\tau}\left[\|f_\theta(o, x_\tau, \tau) - u_\tau\|_2^2\right], \tag{8}$$

where x_τ and u_τ are constructed from the usual flow-matching interpolation between noise and the data action. Let $\ell_{\text{ref}}(z)$ be the same loss evaluated under a frozen reference policy. In our iterative setting, the reference is the policy checkpoint before the current post-training update.

Classical clipped policy-gradient methods [29, 30] are commonly written in terms of a likelihood ratio

$$r_\theta(z) = \frac{\pi_\theta(a \mid o)}{\pi_{\text{ref}}(a \mid o)}. \tag{9}$$

For flow-matching action models, this density ratio is generally not available in closed form without additional machinery. TRFO therefore avoids direct likelihood-ratio evaluation and keeps the standard supervised flow-matching training interface.

A local ratio proxy. The key heuristic is to treat the *difference* in flow-matching losses as a local proxy for the log-likelihood ratio. A precedent for this device exists in Flow Matching Policy Gradients (FPO) [26], which exponentiates conditional-flow-matching loss differences to construct a surrogate likelihood ratio for PPO-style policy optimization. Following this intuition, we approximate:

$$\log r_\theta(z) \approx \ell_{\text{ref}}(z) - \ell_\theta(z). \tag{10}$$

Equation (10) induces the ratio proxy

$$\widetilde{r}_\theta(z) = \exp(\ell_{\text{ref}}(z) - \ell_\theta(z)). \tag{11}$$

Consider the advantage-weighted surrogate based on this proxy,

$$J_{\text{proxy}}(\theta) = \mathbb{E}_{z\sim\mathcal{D}}\left[\widehat{A}(z)\,\widetilde{r}_\theta(z)\right]. \tag{12}$$

When the current policy remains close to the reference on the sampled transitions, the quantity $\Delta_\ell(z) = \ell_{\text{ref}}(z) - \ell_\theta(z)$ is small. A first-order expansion gives

$$\widetilde{r}_\theta(z) = 1 + \ell_{\text{ref}}(z) - \ell_\theta(z) + O\big(\Delta_\ell(z)^2\big). \tag{13}$$

Substituting this expansion into Eq. (12) yields

$$J_{\text{proxy}}(\theta) \approx \mathbb{E}_{z\sim\mathcal{D}}\left[\widehat{A}(z)\,(1 + \ell_{\text{ref}}(z) - \ell_\theta(z))\right] \tag{14}$$

$$= \text{const.} - \mathbb{E}_{z\sim\mathcal{D}}\left[\widehat{A}(z)\,\ell_\theta(z)\right], \tag{15}$$

where the constant is independent of θ. Thus, under the local calibration in Eq. (10), maximizing the first-order ratio surrogate is equivalent to minimizing the signed flow-matching loss

$$\mathcal{L}_{\text{signed}}(\theta) = \mathbb{E}_{z\sim\mathcal{D}}\left[\widehat{A}(z)\,\ell_\theta(z)\right]. \tag{16}$$

This is the sense in which TRFO can be viewed as a first-order, flow-matching-loss analogue of an advantage-weighted likelihood-ratio update.

Imitation and repulsion from the signed loss. The signed loss in Eq. (16) has the desired local gradient directions. For a positive-advantage sample, $\widehat{A}(z) > 0$, minimizing $\widehat{A}(z)\ell_\theta(z)$ decreases $\ell_\theta(z)$, which corresponds under Eq. (10) to increasing the proxy ratio $\widetilde{r}_\theta(z)$. Thus positive-advantage samples are imitated. Expert demonstrations are a special case in our implementation, where they are assigned positive advantage and therefore reduce to ordinary flow matching up to a scalar weight.

For a negative-advantage sample, $\widehat{A}(z) < 0$, minimizing $\widehat{A}(z)\ell_\theta(z)$ increases $\ell_\theta(z)$, which corresponds to decreasing $\widetilde{r}_\theta(z)$. Thus negative-advantage samples induce an unlikelihood-style or repulsive update. This interpretation is local and directional: it describes the sign of the update on the sampled transition, not a global statement about the learned action distribution.

Trust-region geometry. The signed loss in Eq. (16) already has the correct gradient directions, but applying it without constraint would allow arbitrarily large repulsive updates on negative-advantage samples, pushing the policy into loss regions where the local calibration in Eq. (10) ceases to be meaningful. A similar concern motivates the asymmetric trust regions in FPO [26] and FPO++ [71], which clip or penalize differently for the two advantage signs. TRFO realizes a similar asymmetry directly in loss space.

Define the reference-scaled margin $c_{\mathrm{ref}}(z) = \eta\,\ell_{\mathrm{ref}}(z) + \epsilon$ with $c_{\mathrm{ref}}(z) > 0$. The TRFO per-sample weight is

$$w\Big(\widehat{A}, \ell_\theta, \ell_{\mathrm{ref}}\Big) = \begin{cases} \widehat{A}, & \widehat{A} > 0, \\ \widehat{A}\left(1 - \dfrac{\ell_\theta(z)}{c_{\mathrm{ref}}(z)}\right)_+, & \widehat{A} \le 0, \end{cases} \tag{17}$$

and the implemented objective is

$$\mathcal{L}_{\mathrm{TRFO}}(\theta) \;=\; \mathbb{E}_{z\sim\mathcal{D}}\left[\mathrm{sg}(w)\cdot\ell_\theta(z)\right], \tag{18}$$

where $\mathrm{sg}(\cdot)$ denotes stop-gradient, so the effective per-step contribution is $w\,\nabla_\theta\ell_\theta(z)$.

The geometry is asymmetric by design. For positive-advantage samples the weight is simply $\widehat{A} > 0$, and the gradient decreases ℓ_θ—ordinary imitation scaled by advantage. For negative-advantage samples the weight is negative only while $\ell_\theta(z) < c_{\mathrm{ref}}(z)$: repulsion is active inside a sample-dependent trust region anchored at the frozen reference loss, and is extinguished once the current loss exceeds the margin $c_{\mathrm{ref}}(z)$. In terms of the proxy ratio $\widetilde{r}_\theta(z) = \exp(\ell_{\mathrm{ref}}(z) - \ell_\theta(z))$, this corresponds to a one-sided lower bound $\widetilde{r}_\theta(z) \ge r_{\min}(z) \equiv \exp(\ell_{\mathrm{ref}}(z) - c_{\mathrm{ref}}(z))$, beyond which no further repulsion is applied. The trust region is therefore *unbounded on the imitation side and bounded on the repulsion side*, preventing the policy from being driven into data-unsupported regions by negative evidence while placing no ceiling on how closely it can approach high-advantage actions.

Scope of the interpretation. The preceding analysis is heuristic. It shows that, under a local loss-difference proxy for the log-likelihood ratio (Eq. (10)), the TRFO weight in Eq. (17) yields the same first-order gradient direction as an advantage-weighted likelihood-ratio objective, and that the reference-scaled hinge clips repulsive updates at a sample-dependent boundary in loss space, realizing a trust region that is asymmetric across the two advantage signs. The analysis does not imply a formal monotonic-improvement guarantee or an exact density-ratio estimate; TRFO should be understood as a practical, reference-anchored policy-improvement rule that preserves the standard flow-matching training interface while acting, to first order, as an advantage-weighted likelihood-ratio RL objective.

Algorithm 1 PIVOT: Iterative Real-World Post-Training

Require: Pretrained VLA, initial dataset $\mathcal{D}_0$ (demos + corrections), number of iterations N
Ensure: Post-trained policy π_N

Phase 0: Base Policy (SFT)

Train π_0 on $\mathcal{D}_0$ via supervised flow matching: $\mathcal{L}_{\mathrm{SFT}} = \mathbb{E}_{(o,a)\sim\mathcal{D}_0}\big[\ell_\theta(o,a)\big]$

Phase 1: Iterative Post-Training

for iteration $i = 1, \ldots, N$ **do**
  // Step 1: Data collection
  Deploy π_{i-1}
  Collect autonomous rollouts and HG-DAgger human corrections
  Label each episode with binary terminal outcome: success ($r_n = 0$) or failure ($r_n = -\frac{1}{1-\gamma}$)
  $\mathcal{D}_i \leftarrow \mathcal{D}_{i-1} \cup \Delta\mathcal{D}_i$ ▷ Append new data to cumulative buffer

  // Step 2: Value expert training (distributional TD, ≈ 1 UTD)
  Initialize value expert $\hat{V}_\phi$ and target network $\hat{V}_{\phi^-}$ from π_{i-1}'s checkpoint
  for each gradient step on $\mathcal{D}_i$ **do**
    Sample batch $\{(o_t, r_{t:t+H}, o_{t+H})\}$
    $V_{\mathrm{tgt}} \leftarrow \sum_{j=0}^{H-1} \gamma^j r_{t+j} + \gamma^H \hat{V}_{\phi^-}(o_{t+H})$ ▷ H-step bootstrap
    Update ϕ by minimizing $\mathcal{L}_{\mathrm{value}} = -\sum_k \hat{p}_k^V \log p_k^V$ ▷ Categorical cross-entropy
    $\phi^- \leftarrow \tau_{\mathrm{target}}\,\phi + (1 - \tau_{\mathrm{target}})\,\phi^-$ ▷ Polyak averaging
  end for

  // Step 3: Advantage labeling
  for each episode $\{o_1, \ldots, o_n\}$ in $\mathcal{D}_i$ **do**
    if episode is expert demonstration **then**
      $\hat{A}_t \leftarrow 1$ for all t ▷ Expert data always positive
    else
      Compute $\hat{V}(o_t)$ for all t using the trained value expert
      $\widetilde{V} \leftarrow \textsc{GaussianSmooth}(\hat{V},\ \sigma{=}15)$ ▷ Low-pass filter
      $\hat{A}_t \leftarrow \mathrm{clip}\Big(\alpha\Big(\gamma^H \widetilde{V}(o_{t+H}) - \widetilde{V}(o_t) + \sum_{j=0}^{H-1} \gamma^j r_{t+j} + b\Big),\ -1,\ 1\Big)$
    end if
  end for

  // Step 4: TRFO policy update
  Initialize $\pi_\theta \leftarrow \pi_{i-1}$; freeze reference $\pi_{\mathrm{ref}} \leftarrow \pi_{i-1}$
  for each gradient step on $\mathcal{D}_i$ **do**
    Sample batch $\{(o, a, \hat{A})\}$; sample $\tau \sim \mathcal{U}(0,1)$, $\epsilon \sim \mathcal{N}(0, I)$
    $x_\tau \leftarrow \tau\epsilon + (1-\tau)a$ ▷ Noisy interpolant
    $\ell_\theta \leftarrow \|f_\theta(o, x_\tau, \tau) - (\epsilon - a)\|^2$ ▷ Policy flow-matching loss
    $\ell_{\mathrm{ref}} \leftarrow \|f_{\mathrm{ref}}(o, x_\tau, \tau) - (\epsilon - a)\|^2$ ▷ Reference flow-matching loss (no grad)
    if $\hat{A} > 0$: $w \leftarrow \hat{A}$ ▷ Imitate high-advantage transitions
    else: $w \leftarrow \hat{A} \cdot \max\big(0,\ 1 - \ell_\theta / (\eta\,\ell_{\mathrm{ref}} + \epsilon_{\mathrm{num}})\big)$ ▷ Repulse within trust region
    Update θ by $\nabla_\theta\,[\mathrm{sg}(w) \cdot \ell_\theta]$
  end for
  $\pi_i \leftarrow \pi_\theta$
end for
return π_N