

VLA-Precision: Asymmetric Co-Bootstrapping for Efficient Real-World Online RL of Vision-Language-Action Models

Chenyu Su¹, Zhaolong Shen^{3,4}, Yuan Qian¹, Chen Qian¹, Rui Zhang¹,
Feng Yan¹, Weixing Chen¹, Fei Zhang^{1,5}, Jiamin Wang¹, Shuang Cong¹, and Weiwei Shang^{*,1,2}

¹University of Science and Technology of China

²State Key Laboratory of Precision and Intelligent Chemistry

³Beihang University

⁴Zhongguancun Academy

⁵Hefei SpinX Technology

*Corresponding author: wwshang@ustc.edu.cn

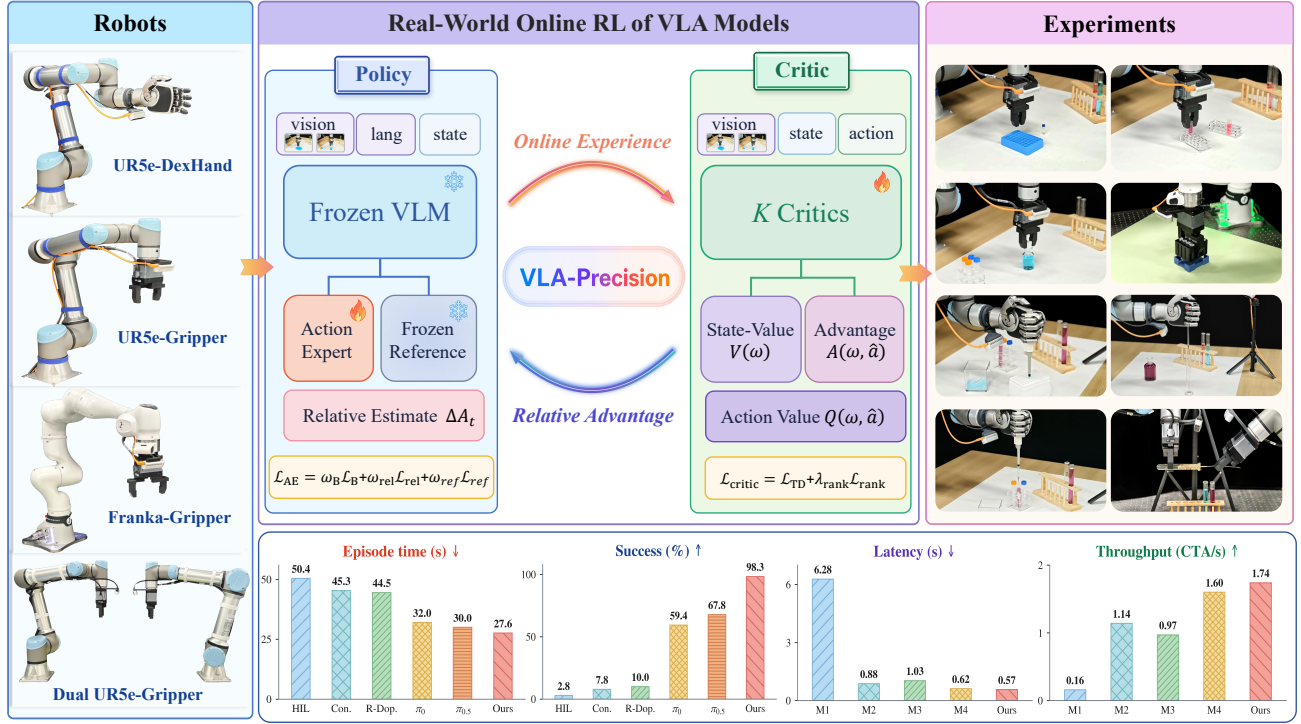


Fig. 1. Framework overview. VLA-Precision combines rapid behavioral learning, progressive value calibration, and reference-regularized relative-advantage policy improvement within a closed-loop experience–policy architecture for stable, efficient real-world online RL of large VLAs. Across nine high-precision chemistry tasks and four robot embodiments, it outperforms baselines in success rate, episode time, critic-to-actor latency, and throughput.

Abstract—Pretrained vision-language-action (VLA) models enable broad manipulation but remain unreliable in tasks demanding precision and repeatability. Applying real-world online reinforcement learning (RL) to VLA post-training enables autonomous trial-and-error improvement beyond demonstrations alone, but exposes two bottlenecks: 1) unreliable value signals can induce policy drift; 2) large-VLA overhead constrains throughput and sample efficiency. To address these challenges, we present VLA-Precision, an efficient real-world online RL framework featuring the Asymmetric Co-Bootstrapping (ACoB) algorithm and the ACoB-Stream architecture. Specifically, ACoB establishes asymmetric co-bootstrapping across timescales: early intervention-guided behavioral learning rapidly improves policy performance while enhancing online experience quality. As autonomous experience accumulates, global return propagation and local preference ranking progressively calibrate value estimates, yielding relative action advantages for reference-regularized policy improvement while suppressing drift. To enable ACoB on large VLAs, we develop ACoB-Stream, a closed-loop experience–policy architecture that establishes invariant-state decoupling and on-demand streaming as design principles, delivering up to 10.9× improvements in throughput and computational efficiency. Ex-

tensive evaluations on nine high-precision chemistry tasks across four categories and four robot embodiments show that VLA-Precision achieves 98.3% mean success rate in 45.8 min/task, with 27.6 s episodes running at 1.2× and 1.8× the speeds of VLA and RL baselines. Resources are available at [VLA-Precision.github.io](https://github.com/WeiweiShang/VLA-Precision).

Index Terms—Vision-language-action models, real-world reinforcement learning, high-precision manipulation, robot learning.

I. INTRODUCTION

RECENT vision-language-action (VLA) models have acquired broadly transferable capabilities through large-scale pretraining on heterogeneous multimodal data [1]–[3]. With only a few task-specific demonstrations, VLAs can perform diverse robotic manipulation tasks and generalize across environmental variations. However, broad adaptability cannot guarantee precise and repeatable task completion: small residual errors at critical stages still cause intermittent failures.

Behavior cloning (BC) learns task behavior directly from dense action labels in demonstrations, but errors compound outside the demonstrated distribution and performance remains bounded by demonstration quality. In contrast, reinforcement learning (RL) uses temporal credit assignment to learn long-term action values from interaction. Therefore, applying RL to VLA post-training enables continual policy optimization through interaction-derived return signals, thereby mitigating compounding errors and surpassing the performance ceiling imposed by demonstration data.

Toward this goal, existing work has explored simulation-based and real-world RL for VLA post-training [4]–[16]. Simulation-based RL enables massively parallel and failure-tolerant interaction, supporting scalable and low-cost policy optimization without the physical constraints of real robots. However, the sim-to-real gap in visual observations and contact dynamics limits reliable transfer to physical robots [7], [8]. In contrast, real-world RL optimizes VLAs through direct physical interaction, enabling policy adaptation to real-world dynamics and sensor noise. Existing real-world RL methods fall into two groups by model scale: RL for large pretrained VLAs [9]–[11] and RL for compact robot policies [12]–[14]. Despite these advances, real-world VLA-RL remains constrained by two critical bottlenecks: 1) Algorithmic stability: unreliable value estimates can induce policy drift; and 2) System efficiency: large-VLA computational overhead limits training throughput and reduces overall online RL efficiency.

To mitigate the algorithmic bottleneck, existing methods [14]–[16] regularize RL updates with behavioral-cloning constraints from demonstrations or interventions. While these constraints preserve prior behaviors and stabilize exploration, they provide limited improvement in value estimation. Consequently, RL updates remain vulnerable to value errors and policy drift. For the systems bottleneck, existing methods [12], [14], [15] rely on replay-intensive optimization, whose computational cost increases substantially with large VLAs, limiting training throughput. Real-world online interaction further amplifies this computational burden. These limitations highlight the need for reliable RL algorithms and efficient training systems for real-world post-training of large VLAs.

To address the first problem, we propose Asymmetric Co-Bootstrapping (ACoB), a real-world online RL algorithm that couples rapid behavioral learning with progressive value calibration across timescales. On the faster timescale, intervention-guided behavioral learning rapidly incorporates successful and human-executed actions, accelerating policy improvement while continually improving online experience quality. Meanwhile, global return propagation and local preference ranking continuously calibrate value estimates as autonomous experience accumulates. The resulting relative action advantages guide reference-regularized policy improvement, suppressing policy drift while preserving learned task-specific competence. These asymmetric interactions establish a cross-timescale co-bootstrapping loop: rapid behavioral learning continuously improves the experience underlying value estimation, while increasingly reliable value estimates guide the policy beyond the limits of direct supervision.

To address the second problem, we propose ACoB-Stream,

a closed-loop experience-policy architecture for real-world online RL of large VLAs. Guided by invariant-state decoupling and on-demand streaming, ACoB-Stream jointly manages the lifecycles of experience-context state and policy state across four stages: (1) **Experience-context formation**: by retaining frozen VLM contexts across updates, it reduces redundant computation during policy updates; (2) **Experience-context persistence**: through context deduplication and sliding-window sampling, it reduces storage redundancy and limits random-access overhead as experience accumulates; (3) **Experience-context access**: through objective-aligned context retrieval, it improves data-access efficiency during optimization; and (4) **Policy-state synchronization**: through trainable-subspace policy dissemination, it reduces synchronization overhead and accelerates policy deployment. Together, these stages form a closed-loop experience-policy stream, reducing end-to-end system overhead in real-world online RL under constrained compute and memory.

The contributions of this work are summarized as follows:

- 1) We propose Asymmetric Co-Bootstrapping (ACoB), a real-world online RL algorithm for VLA post-training that couples rapid behavioral learning with progressive value calibration across timescales, overcoming demonstration-based performance limits while suppressing value-induced policy drift.
- 2) We develop ACoB-Stream, a closed-loop experience-policy architecture organized around state lifecycles, with invariant-state decoupling and on-demand streaming as the core design principles, delivering up to $10.9\times$ improvements in throughput and efficiency.
- 3) Through extensive evaluations on nine high-precision chemistry tasks across four categories and four robot embodiments, VLA-Precision achieves 98.3% average success in 45.8 min/task, with 27.6 s episodes running at $1.2\times$ and $1.8\times$ the speeds of VLA and RL baselines.

The remainder of this paper is organized as follows: In Section II, we review related work on RL-based VLA post-training, including learning algorithms and system architectures. In Section III, we present the proposed ACoB algorithm and ACoB-Stream system architecture in detail. In Section IV, we evaluate the algorithmic effectiveness and system efficiency of VLA-Precision through extensive real-world experiments on high-precision chemistry tasks. In Section V, we discuss the limitations and future directions of this work. Finally, we conclude this paper in Section VI.

II. RELATED WORK

A. Simulation-Based RL for VLA Post-Training

Reinforcement learning (RL) may break demonstration-imposed performance ceilings in vision-language-action (VLA) models through reward-driven exploration and temporal credit assignment. Simulation-based RL enables efficient, high-throughput VLA exploration through large-scale parallel interaction without real-robot training costs and safety risks [17]–[19]. Among the first studies, RIPT-VLA [20] pairs dynamic rollout sampling with leave-one-out advantages, raising

one-shot success from 4% to 97%, while VLA-RL [7] combines trajectory-level RL with process rewards to outperform fine-tuning by 4.5 points. A controlled VLA benchmark [21] shows that RL chiefly strengthens semantic generalization and execution robustness. Subsequent work broadens task and policy coverage. SimpleVLA-RL [22] combines VLA-specific sampling and parallel rendering, lifting LIBERO-Long [23] from 17.1% to 91.7%, while RLinf-VLA [8] coordinates heterogeneous workloads for up to $2.27\times$ speedup. For flow-based policies, π_{RL} [24] constructs tractable likelihoods, improving success by up to 31.0%. Recent work targets physical transfer. RL-Co [16] regularizes simulation-based RL with real demonstrations, raising real-world success by 24% and 20%, while WoVR [25] controls imagined dynamics to improve success across robot platforms. Although these methods reduce real-data demands, the sim-to-real gap still limits transfer reliability. Real-world RL grounds autonomous trial-and-error in real observations and dynamics, converting pretrained VLA competence into precise, reliable execution. This advantage motivates our focus on real-world VLA post-training.

B. Real-World RL for VLA Post-Training

By learning from physical interaction, real-world RL adapts policies to actual sensing and dynamics, improving reliability under deployment conditions. RL for compact robot policies has advanced contact-rich and dynamic manipulation [26]–[30]. HIL-SERL [12] reaches near-perfect success in 1–2.5 hours via intervention-guided online RL, but has limited within-task generalization. ConRFT [14] reaches 96.3% mean success in 45–90 min with offline-to-online RL on Octo representations [31], but Q maximization sacrifices prior performance and permits policy drift. Robo-Dopamine [32] equips ConRFT with a large progress-based reward model, but OOD hallucinations can misguide optimization. RL-100 [13] applies offline-to-online RL to diffusion policies, but converges slowly, requiring 12.1 hours of real-robot rollouts per task on average, with 6.8 hours offline and 5.3 hours online. These tradeoffs motivate RL for large pretrained VLAs, which retain broader action priors. $\pi_{0.6}^*$ [9] exceeds 90% success on most tasks through batchwise value labeling and advantage-conditioned retraining, but requires over 1,000 real-robot rollouts per task on average, severely limiting feedback efficiency. RL Token [11] accelerates critical-phase throughput by up to $3\times$, but requires 400–1,000 online episodes per task, while externalized improvements and policy handoffs limit end-to-end adaptation. Distinct from prior methods, ACoB builds on large pretrained VLAs and couples rapid behavioral learning with progressive value calibration to suppress policy drift, with ACoB-Stream developed to establish an efficient continuous real-world online RL loop.

C. Efficient Online RL Systems for VLA Models

Efficient online RL must shorten the closed-loop experience-policy cycle. Simulation-based VLA-RL systems batch multimodal decoding, parallelize rendering, and coordinate heterogeneous workloads, achieving up to $2.27\times$ speedup [7], [8], [22]. However, massively parallel, rapidly

resettable simulators do not transfer directly to real-world online RL. For compact policies, asynchronous actor-learner systems overlap physical interaction with replay optimization to improve throughput [12], [33], [14]. This efficiency does not extend to large pretrained VLAs because optimization at scale imposes greater computational, state-management, and synchronization costs. EXPO-FT [10] and RL Token [11] avoid propagating RL gradients through the VLA by optimizing residual action edits and a lightweight token-conditioned policy, respectively. This confines reward-driven improvements to auxiliary policies, requiring separate editors or switching logic and preventing their accumulation within a unified policy as tasks expand. SOP [34] and USER [35] scale real-world online learning through fleet orchestration and heterogeneous cloud-edge infrastructure, respectively. Scale-out increases throughput with robots and accelerators without reducing per-update large-VLA computation. Departing from the methods above, we develop ACoB-Stream, a closed-loop experience-policy architecture organized around state lifecycles, with invariant-state decoupling and on-demand streaming as the core design principles, enabling efficient continuous real-world online RL of large VLAs.

III. METHOD

In this section, we first formulate real-world RL post-training of large VLAs under an asynchronous actor-learner process as a closed-loop experience-policy optimization problem. Subsequently, we present the Asymmetric Co-Bootstrapping (ACoB) algorithm to improve policy performance beyond demonstration-based limits while suppressing value-induced policy drift. Thereafter, we develop ACoB-Stream, which establishes a closed-loop experience-policy stream for large VLAs through experience-context formation, persistence, access, and policy-state synchronization. Finally, we specify the end-to-end online training and deployment protocol for continuous real-world post-training of large VLAs.

A. VLA-Precision Problem Formulation

Initializing real-world online RL with demonstrations provides task-relevant priors that reduce the burden of early exploration and improve sample efficiency [12], [14]. For a large pretrained VLA, initialization must additionally establish task-specific competence while preserving pretrained manipulation priors and within-task generalization.

To this end, Fig. 1 provides an overview of the VLA-Precision framework, while Fig. 2 details the two-stage post-training pipeline: Stage I performs full-parameter imitation learning on task demonstrations to obtain the task-specific policy prior Θ_{IL} . Stage II then initializes real-world online RL from Θ_{IL} and improves the action expert within this asynchronous actor-learner system. We formulate Stage II as a closed-loop experience-policy optimization problem.

Specifically, we model physical interaction under task instruction ℓ as a standard Markov decision process (MDP),

$$\mathcal{M}_\ell = (\mathcal{S}, \mathcal{A}, P, r, \rho_0, \bar{\gamma}). \quad (1)$$

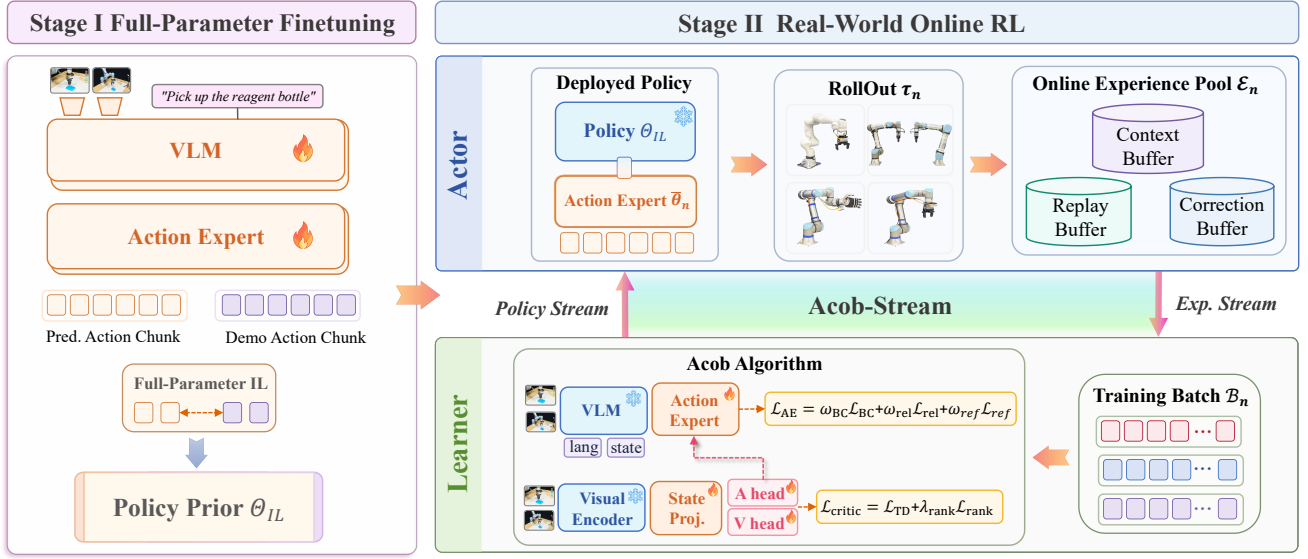


Fig. 2. Two-stage post-training pipeline of VLA-Precision. In Stage I, full-parameter imitation learning on demonstrations establishes the task-specific policy prior Θ_{IL} . Initialized from Θ_{IL} , real-world online RL in Stage II is formulated as a closed-loop experience-policy optimization problem under an asynchronous actor-learner process, where rollouts generate online experience and learner optimization yields action-expert updates for redeployment.

Here, $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces, P the environment transition probability, r the reward function, ρ_0 the initial-state distribution, and $\bar{\gamma}$ the discount factor. At decision step t , the state $s_t = (o_t, q_t, \ell) \in \mathcal{S}$ comprises visual observation o_t , robot state q_t , and task instruction ℓ . A VLA policy action is an H -step action chunk $a_t = (u_{t,0}, \dots, u_{t,H-1}) \in \mathcal{A} \subseteq \mathbb{R}^{H \times d}$, where H is the chunk horizon and d the single-step action dimension. Executing the chunk induces $s_{t+1} \sim P(\cdot | s_t, a_t)$ and yields the cumulative reward $R_t^{(H)} = \sum_{h=0}^{H-1} \gamma^h r_{t,h}$, where γ is the primitive-step discount factor and $\bar{\gamma} = \gamma^H$. The task-conditioned policy $\pi(\cdot | s_t)$ maps each state to an action-chunk distribution and induces trajectory τ under $\mathcal{M}_\ell$.

To instantiate the task-conditioned policy π , we initialize the online policy from the task-finetuned $\pi_{0.5}$ and optimize only the LoRA parameters [36] in the action expert. Under this parameterization, Θ_f denotes the frozen multimodal prefix parameters, ψ the frozen action-expert parameters inherited from Stage I, and $\theta \in \mathbb{R}^p$ the trainable LoRA parameters in the action expert. Action-chunk generation is compactly written as

$$a_t = G_{\psi, \theta}(z_t, \epsilon_t) \sim \pi_{\Theta_f, \psi, \theta}(\cdot | s_t), \quad \epsilon_t \sim \mathcal{N}(0, I). \quad (2)$$

Here, $z_t = F_{\Theta_f}(o_t, \ell, q_t)$ is the task-conditioned multimodal prefix context encoded from the visual observation, task instruction, and robot state. ϵ_t is the initial Gaussian noise, I the identity covariance, and $G_{\psi, \theta}$ the flow-based action expert. The corresponding online RL objective is

$$\theta^* = \arg \max_{\theta \in \mathbb{R}^p} J(\theta), \quad J(\theta) = \mathbb{E}_{\tau \sim p(\tau | \pi_{\Theta_f, \psi, \theta}, P, \rho_0)} \left[\sum_{t=0}^{T-1} \bar{\gamma}^t R_t^{(H)} \right]. \quad (3)$$

Here, θ^* denotes the return-maximizing parameters, and T is the episode horizon measured in action-chunk decision steps. To solve Eq. (3) through continuous real-world interaction,

VLA-Precision constructs the closed-loop solution process illustrated in Fig. 2:

$$\begin{aligned} \text{Actor: } \bar{\theta}_n &\xrightarrow{\text{Rollout}} \tau_n \xrightarrow{\text{experience}} \mathcal{E}_n \ni \mathcal{B}_n, \\ \text{Learner: } (\theta_n, \mathcal{B}_n) &\xrightarrow{\text{ACoB}} \theta_{n+1} \xrightarrow{\text{policy}} \bar{\theta}_{n+1} \end{aligned} \quad (4)$$

Here, n denotes the online update cycle. θ_n and $\bar{\theta}_n$ denote the trainable action-expert state in the learner process and its deployed counterpart in the actor process, respectively, while τ_n , $\mathcal{E}_n$, and $\mathcal{B}_n$ denote the real-world trajectory, online experience pool, and training batch. The online experience pool comprises the replay buffer $\mathcal{R}$, the correction buffer $\mathcal{C}$, and the context buffer $\mathcal{K}$. The actor generates τ_n under $\bar{\theta}_n$, storing all transitions in $\mathcal{R}$, effective corrections in $\mathcal{C}$, and multimodal prefix contexts in $\mathcal{K}$. The learner optimizes θ_n with ACoB over $\mathcal{B}_n \subseteq \mathcal{E}_n$, yielding θ_{n+1} for deployment as $\bar{\theta}_{n+1}$. In this formulation, ACoB optimizes the action expert within the learner process, while ACoB-Stream closes the experience-policy loop between the two processes.

B. Asymmetric Co-Bootstrapping for the Action Expert

Reliable real-world RL post-training of large VLAs must improve the action expert beyond demonstration-based performance limits while preventing unreliable value estimates from inducing policy drift. To this end, ACoB establishes asymmetric co-bootstrapping within the learner process of Eq. (4): behavioral learning rapidly improves the action expert and continuously enhances online experience quality, while the critic continually calibrates value estimates as online experience accumulates, yielding increasingly reliable relative action advantages for further action-expert improvement.

1) *Progressive Value Calibration*: Reliable value learning requires both long-horizon return estimation and fine-grained credit assignment. The former captures cumulative behavioral consequences, while the latter sharpens policy-improvement

signals. ACoB therefore establishes progressive value calibration through global return propagation and local preference ranking for action-expert optimization.

To jointly learn both signals, ACoB instantiates the value model as an ensemble of K task-specific critics. Each critic adopts the value–advantage decomposition $Q_{\phi_k}(\omega, \hat{a}) = V_{\phi_k}(\omega) + A_{\phi_k}(\omega, \hat{a})$, where $\omega_t = (o_t, q_t) = \text{proj}_{o,q}(s_t)$ is the visual–proprioceptive component of the state and Q_{ϕ_k} is the corresponding target critic. For any action chunk $a \in \mathcal{A}$, let $\tilde{a} = \mathcal{T}(a)$ denote its OpenPI-normalized model representation and $\hat{a} = \mathcal{C}(\tilde{a}) = \text{vec}(P_c \tilde{a}_{0:H})$ its flattened critic representation, where P_c excludes inactive action dimensions.

Global Return Propagation. Based on the above critic formulation, ACoB propagates long-horizon returns through temporal-difference optimization. Let $a_t^{\text{exec}} \in \mathcal{A}$ denote the action chunk ultimately executed by the robot, resulting from either autonomous policy inference or human correction. It induces the transition

$$\begin{aligned} \xi_t &= (s_t, a_t^{\text{exec}}, \mathbf{r}_t, s_{t+1}, d_t), \\ \mathbf{r}_t &= (r_{t,0}, \dots, r_{t,H-1}). \end{aligned} \quad (5)$$

where $\xi_t \in \mathcal{E}_n$ and d_t is the terminal flag. Let $\mathcal{B}_n^{\text{RL}} \subseteq \mathcal{B}_n \subseteq \mathcal{E}_n$ denote samples eligible for value learning. Let $\tilde{G}_{\psi,\theta}$ denote the action-expert output in the normalized OpenPI action space. For each sampled transition ξ_t , the bootstrap action at s_{t+1} and its TD target are constructed as

$$\begin{aligned} \hat{a}_{t+1}^{\theta_n} &= \mathcal{C}\left(\tilde{G}_{\psi,\theta_n}(z_{t+1}, \epsilon)\right), \quad \epsilon \sim \mathcal{N}(0, I), \\ y_t &= R_t^{(H)} + \bar{\gamma}(1 - d_t) \min_k Q_{\phi_k}(\omega_{t+1}, \hat{a}_{t+1}^{\theta_n}). \end{aligned} \quad (6)$$

Global return propagation is realized by the ensemble TD objective defined as follows:

$$\mathcal{L}_{\text{TD}}(\phi) = \mathbb{E}_{\mathcal{B}_n^{\text{RL}}} \left[\frac{1}{K} \sum_k (Q_{\phi_k}(\omega_t, \hat{a}_t^{\text{exec}}) - \text{sg}(y_t))^2 \right], \quad (7)$$

where sg denotes stop-gradient and $\phi = \{\phi_k\}_{k=1}^K$. Through recursive bootstrapping, this objective propagates long-horizon returns along executed trajectories.

Local Preference Ranking. Although corrections can recover a rollout to success, TD over the executed trajectory cannot correct the value of the overwritten proposal. The proposal may thus remain overvalued and misguide action-expert optimization. ACoB addresses this limitation through state-matched local preference ranking, enforcing the correction above the original proposal. Before intervention, the deployed VLA proposes $a_t^{\text{prop}} \sim \pi_{\Theta_f, \psi, \bar{\theta}_n}(\cdot | s_t)$. An effective intervention is marked by $i_t = 1$ and supplies a corrective chunk a_t^{hum} , giving

$$a_t^{\text{cmd}} = \begin{cases} a_t^{\text{prop}}, & i_t = 0, \\ a_t^{\text{hum}}, & i_t = 1, \end{cases}, \quad a_t^{\text{exec}} = \text{Exec}(s_t, a_t^{\text{cmd}}). \quad (8)$$

where $i_t = 0$ denotes autonomous execution and Exec denotes environment-side action resolution. Under intervention, the original proposal is retained for ranking. We set $c_t = 1$ only when $i_t = 1$ and $\|\hat{a}_t^{\text{exec}} - \hat{a}_t^{\text{prop}}\|_2 > \varepsilon_a$, and $c_t = 0$ otherwise, where $\varepsilon_a > 0$ is a small tolerance. Because a_t^{prop} has no

observed successor, it is excluded from TD learning and used only in the state-matched ranking objective

$$\begin{aligned} \mathcal{L}_{\text{rank}}(\phi) &= \mathbb{E}_{\xi_t \sim \mathcal{B}_n^{\text{RL}} | c_t=1} \left[\frac{1}{K} \sum_{k=1}^K \left[m_c - \Delta A_{t,k}^{\text{pair}} \right]_+^2 \right], \quad (9) \\ \Delta A_{t,k}^{\text{pair}} &= A_{\phi_k}(\omega_t, \hat{a}_t^{\text{exec}}) - A_{\phi_k}(\omega_t, \hat{a}_t^{\text{prop}}), \end{aligned}$$

where $m_c \geq 0$ is the ranking margin and $[x]_+ = \max(x, 0)$. Together, global return propagation and local preference ranking define the ACoB critic objective

$$\mathcal{L}_{\text{critic}} = \mathcal{L}_{\text{TD}} + \lambda_{\text{rank}} \mathcal{L}_{\text{rank}}. \quad (10)$$

where $\lambda_{\text{rank}} \geq 0$ balances the two signals. TD grounds the critic in long-horizon returns, while local ranking directly calibrates the relative value of the correction and original proposal, resolving the local credit ambiguity that long-horizon TD propagation cannot address. Under the decomposition $Q = V + A$, V represents the state-only component shared across actions, whereas the ranking loss acts only on A . This isolates action-dependent differences from the state-only component and aligns the signal with action-expert optimization.

2) Relative-Advantage Policy Improvement: Real-world online RL post-training of a large VLA must translate critic value estimates into reliable action-expert optimization signals. Direct maximization of absolute Q values can amplify early value-estimation errors and drive the policy toward spuriously high-value actions. ACoB therefore proposes relative-advantage policy improvement, replacing absolute-value pursuit with paired advantage comparison. By focusing on improvement between actions, this mechanism reduces sensitivity to value scale and optimistic estimation errors.

However, value-signal reliability must be established gradually as online experience accumulates, so this indirect RL signal alone yields slow, unstable early action-expert updates. ACoB therefore couples relative-advantage improvement with flow-matching behavior cloning [37]. Unlike prior methods that primarily use behavior cloning to constrain the policy distribution [14], [16], ACoB uses critical-stage human corrections to drive behavior cloning early in training, rapidly absorbing high-quality corrections to improve the action expert and accelerate critic calibration through better experience. Long-horizon return propagation and local preference ranking then continually calibrate value estimates, providing increasingly reliable guidance for autonomous improvement as intervention recedes. This interaction establishes asymmetric co-bootstrapping between rapid behavior cloning and progressive value calibration. Meanwhile, the task-finetuned action expert is retained as a frozen reference. Reference regularization keeps online refinement focused on execution precision rather than reshaping behavior, preserving existing competence and suppressing value-induced policy drift.

Relative-Advantage Improvement. Directly differencing independently aggregated Q estimates can mix state value, critic-specific scales, and stochastic decoding variation, obscuring action improvement. ACoB instead computes paired advantage differences within each critic before pessimistic ensemble aggregation. Given a multimodal context z_t , the current action expert and frozen reference decode action

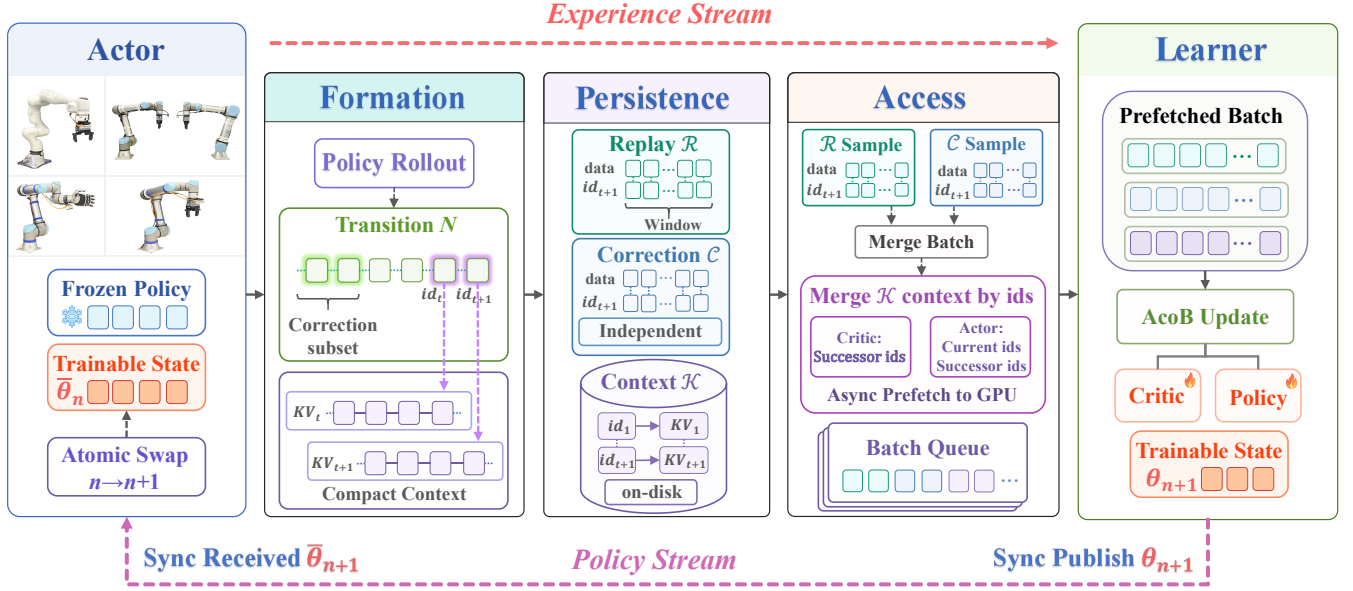


Fig. 3. ACoB-Stream architecture. Through invariant-state decoupling and on-demand streaming, ACoB-Stream jointly manages the lifecycles of experience-context state and policy state across experience-context formation, persistence, objective-aligned access, and policy-state synchronization, thereby establishing a closed-loop experience-policy stream between the actor and learner.

chunks using the same random sample ϵ : $\tilde{a}_t^\theta = \tilde{G}_{\psi,\theta}(z_t, \epsilon)$ and $\tilde{a}_t^{\text{ref}} = \tilde{G}_{\psi,\theta_0}(z_t, \epsilon)$, where θ_0 denotes the frozen reference parameters. Their critic representations are $\hat{a}_t^\theta = \mathcal{C}(\tilde{a}_t^\theta)$ and $\hat{a}_t^{\text{ref}} = \mathcal{C}(\tilde{a}_t^{\text{ref}})$. For compactness, let $A_{t,k}^x = A_{\phi_k}(\omega_t, \hat{a}_t^x)$ for $x \in \{\theta, \text{ref}, \text{prop}\}$. On this basis, ACoB defines the pessimistic relative advantage as the minimum current-to-baseline advantage gap across the K critics:

$$b_{t,k} = A_{t,k}^{\text{ref}} + c_t [A_{t,k}^{\text{prop}} - A_{t,k}^{\text{ref}}]_+, \quad (11)$$

$$\Delta A_t = \min_k [A_{t,k}^\theta - \text{sg}(b_{t,k})].$$

where $[x]_+ = \max\{x, 0\}$. Here, $c_t = 1$ for a valid human correction, whereas $c_t = 0$ otherwise. The term $b_{t,k}$ is the comparison baseline of critic k . It equals $A_{t,k}^{\text{ref}}$ for $c_t = 0$ and $\max\{A_{t,k}^{\text{ref}}, A_{t,k}^{\text{prop}}\}$ for $c_t = 1$. Taking the minimum over the K critics makes $\Delta A_t > 0$ only when every critic favors the current action over its baseline. To prevent the action expert from pursuing critic scores after sufficient relative improvement, ACoB optimizes the smooth soft-margin objective

$$\mathcal{L}_{\text{rel}}(\theta) = \mathbb{E}_{\xi_t \sim \mathcal{B}_n^{\text{RL}}, \epsilon} \left[\kappa \text{softplus} \left(\frac{m_\pi - \Delta A_t}{\kappa} \right) \right], \quad (12)$$

where $\text{softplus}(x) = \log(1 + e^x)$, $m_\pi \geq 0$ is the desired advantage margin, and $\kappa > 0$ controls its softness. Critic parameters and comparator branches are held fixed, so gradients reach θ only through $\hat{a}_t^\theta$.

Flow-Matching Behavior Cloning. ACoB optimizes flow-matching behavior cloning over task demonstrations, successful online executions, and effective human corrections. Its flow-matching objective is

$$\ell_t^{\text{FM}}(\theta) = \frac{1}{H d_m} \left\| [v_{\psi,\theta}(x_\eta, z_t, \eta) - u_\eta]_{0:H} \right\|_F^2, \quad (13)$$

$$\mathcal{L}_{\text{BC}}(\theta) = \mathbb{E}_{\xi_t \sim \mathcal{B}_n, \epsilon, \eta} [\ell_t^{\text{FM}}(\theta) \mid \chi_t^{\text{beh}} = 1].$$

where $v_{\psi,\theta}$ is the velocity field underlying $\tilde{G}_{\psi,\theta}$, $\epsilon \sim \mathcal{N}(0, I)$, $\eta \sim p_\eta$, $x_\eta = \eta\epsilon + (1 - \eta)\tilde{a}_t^{\text{exec}}$, $u_\eta = \epsilon - \tilde{a}_t^{\text{exec}}$, and d_m is the model action dimension. Here, $y_t^{\text{suc}} \in \{0, 1\}$ denotes the episode-success label of ξ_t . The behavioral mask $\chi_t^{\text{beh}} = \max\{y_t^{\text{suc}}, c_t\}$ selects all chunks from successful trajectories and only effective corrections from failed ones.

Reference Regularization. To retain the reference behavior during online refinement, ACoB regularizes body-action coordinates toward the corresponding frozen-reference action:

$$\mathcal{L}_{\text{ref}}(\theta) = \mathbb{E}_{\xi_t \sim \mathcal{B}_n, \epsilon} \left[\frac{1}{H d_b} \left\| P_b(\tilde{a}_{t,0:H}^\theta - \tilde{a}_{t,0:H}^{\text{ref}}) \right\|_F^2 \right], \quad (14)$$

where P_b selects the body-action coordinates and d_b is their single-step dimension. The complete action-expert objective is

$$\mathcal{L}_{\text{AE}} = w_{\text{BC}} \mathcal{L}_{\text{BC}} + w_{\text{rel}} \mathcal{L}_{\text{rel}} + w_{\text{ref}} \mathcal{L}_{\text{ref}}, \quad (15)$$

where the nonnegative weights sum to one. Here, $\mathcal{L}_{\text{rel}}$, $\mathcal{L}_{\text{BC}}$, and $\mathcal{L}_{\text{ref}}$ denote relative-advantage RL, behavior cloning, and reference regularization, respectively.

Together, Eqs. (10) and (15) establish ACoB's asymmetric co-bootstrapping across timescales: intervention-driven behavior cloning rapidly improves both the action expert and the experience underlying value calibration, while progressively calibrated value estimates provide relative-advantage guidance for stable autonomous improvement, with the frozen reference preserving existing competence.

C. ACoB-Stream Architecture

Real-world online RL relies on asynchronous actor-learner systems for efficient experience collection and policy updates. For compact policies, four scale-dependent costs remain manageable: inexpensive model forward passes, CPU-resident lightweight training samples, low-latency I/O at limited data scale, and rapid full-model transfer and deployment. By

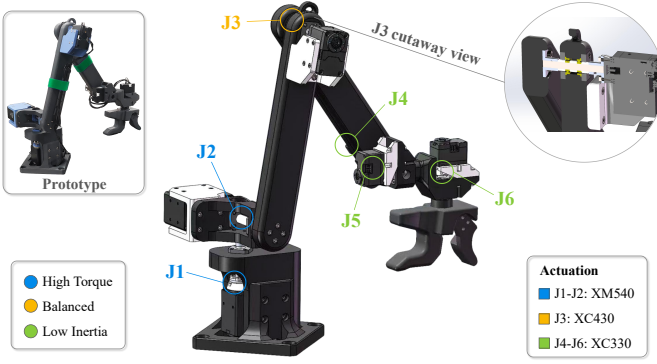


Fig. 4. Active UR5e-isomorphic master: CAD model and prototype.

contrast, large VLAs amplify these costs through expensive frozen-prefix forward, prefix KV accumulation beyond CPU RAM, high-latency large-context access, and lag-inducing full-model synchronization. Together, these costs prevent new experience from promptly affecting the deployed policy.

To address this mismatch, ACoB-Stream introduces a closed-loop experience-policy architecture organized around state lifecycles, jointly managing experience-context state and policy state across four stages: experience-context formation, persistence, and access, followed by policy-state synchronization. The architecture follows two design principles: invariant-state decoupling avoids repeated computation, storage, and transfer, while on-demand streaming limits context access and policy synchronization to the state required by the current objective. Fig. 3 illustrates this four-stage architecture.

1) *Experience-Context Formation*: Repeated model forward passes through the frozen prefix for every sampled item constitute a primary throughput bottleneck in large-VLA learner updates. In ACoB-Stream, the experience-context state comprises the data maintained in the replay buffer $\mathcal{R}$, correction buffer $\mathcal{C}$, and context buffer $\mathcal{K}$. Because the frozen prefix remains unchanged during online optimization, ACoB-Stream retains prefix KV caches generated during actor inference. Before persistence, observation-schema-aware compaction removes inactive prefix positions. At episode commitment, all transitions enter $\mathcal{R}$, their effective-correction subset enters $\mathcal{C}$, and compacted prefix contexts enter $\mathcal{K}$. Transition records in $\mathcal{R}$ and $\mathcal{C}$ retain current and successor context identifiers that reference $\mathcal{K}$. These identifiers let the learner retrieve the required contexts and train the action expert without recomputing the frozen prefix. This reduces frozen-prefix computation from once per sampled item to once per new observation.

2) *Experience-Context Persistence*: Within the experience-context state, the prefix contexts in $\mathcal{K}$ pose the primary persistence challenge: they must remain available across updates, yet their accumulated footprint outgrows CPU RAM even after compaction. ACoB-Stream therefore designs persistence around context deduplication and sliding-window sampling. Each cache is stored once in the disk-backed context buffer $\mathcal{K}$. The replay and correction buffers retain only current and successor context identifiers for all online transitions and the effective-correction subset, respectively. For replay sampling,

the sliding window bounds the active set according to the Linux page-cache capacity available in CPU RAM, improving page-cache reuse and reducing repeated disk-to-CPU data transfers, thereby accelerating data access without requiring explicit KV-cache residency. Correction sampling remains independent. Unlike a RAM-only sliding buffer, this design retains full history for checkpoint recovery while reducing duplicate storage and cold-disk access. Sampled transitions and identifiers then pass to the access stage.

3) *Experience-Context Access*: Although the disk-backed context buffer $\mathcal{K}$ makes accumulated prefix contexts tractable, random I/O and unnecessary retrieval can still delay learner updates. ACoB-Stream therefore aligns experience-context access with each optimization objective: value optimization retrieves only successor contexts from $\mathcal{K}$ for its bootstrap targets, whereas action-expert optimization additionally retrieves current contexts. Samples from $\mathcal{R}$ and $\mathcal{C}$ are merged before their required contexts are retrieved in batches from $\mathcal{K}$ by identifier. Background CPU workers sample transitions, retrieve contexts, and assemble host batches, while device workers transfer the completed batches to GPU and enqueue them after placement. Preparing the next GPU-ready batch during the current network update reduces learner idle time and improves GPU utilization. The learner uses these batches for ACoB optimization and passes the updated action-expert state to synchronization.

4) *Policy-State Synchronization*: Completing learner optimization affects robot behavior only after the actor process deploys the updated policy. Full-VLA serialization, transfer, and loading delay deployment and increase policy-version lag. ACoB-Stream therefore designs synchronization around the trainable policy subspace. Invariant VLA state remains resident in both processes. The learner periodically publishes the complete trainable action-expert state instead of the full VLA or parameter deltas. Upon receipt, the actor process merges this state into its local VLA, completes device placement, and atomically replaces the active state under lock. Rollouts continue on the last valid version during preparation. Transmitting and loading only changed policy state reduces synchronization and deployment overhead, shortens policy-refresh latency, and brings behavior to the robot sooner.

In summary, ACoB-Stream manages state lifecycles through cross-update reuse of frozen-prefix contexts, deduplicated disk-backed persistence and objective-aligned retrieval, and trainable policy-state dissemination instead of full-model synchronization. This reduces computation, memory, I/O, and policy-refresh overhead, allowing new experience to affect the deployed policy sooner.

D. Real-World Training and Deployment

Real-world training and deployment of VLA-Precision comprise three components: teleoperation and data collection, two-stage algorithm training, and robot deployment, which respectively collect real-world experience, optimize the policy, and execute learned behaviors.

1) *Teleoperation and Data Collection*: Task requirements for motion range, adjustment precision, and human-machine

Algorithm 1: Two-Stage Training of VLA-Precision

Require: pretrained VLA $\pi_{0.5}$; demonstrations $\mathcal{D}$; trainable action-expert LoRA θ and frozen Stage-I reference θ_0 ; critic and target ensembles $\{Q_{\phi_k}, Q_{\bar{\phi}_k}\}_{k=1}^K$; replay buffer $\mathcal{R}$, empty correction buffer $\mathcal{C}$, and context buffer $\mathcal{K}$; batch size B , update ratio N_e , critic warm-up steps N_w , and policy publication interval N_{pub} .

- 1: *Stage I:* Fully fine-tune $\pi_{0.5}$ on $\mathcal{D}$ to obtain (Θ_f, ψ) .
- 2: *Stage II:* Freeze $(\Theta_f, \psi, \theta_0)$; launch actor and learner; publish θ .
- 3: **Actor:**
- 4: **for** each episode e **do**
- 5: Set episode trajectory $\tau_e \leftarrow \emptyset$.
- 6: **for** each interaction step t **do**
- 7: $(z_t, a_t^{\text{prop}}) \leftarrow \text{Infer}(\pi_{\Theta_f, \psi, \bar{\theta}_n}, s_t)$.
- 8: Obtain $\xi_t = (s_t, a_t^{\text{exec}}, \mathbf{r}_t, s_{t+1}, d_t)$ from Eqs. (5) and (8).
- 9: Append $(\xi_t, a_t^{\text{prop}}, c_t)$ to τ_e .
- 10: **end for**
- 11: Attach y_e^{suc} to all transitions in τ_e .
- 12: Append τ_e , $\tau_e[c_t = 1]$, and $\{z_t\}$ to $\mathcal{R}$, $\mathcal{C}$, and $\mathcal{K}$, respectively.
- 13: **end for**
- 14: **Learner:**
- 15: **wait until** $|\mathcal{R}| \geq B$ and $|\mathcal{C}| \geq B$.
- 16: **for** $w = 1, \dots, N_w$ **do**
- 17: Sample $B/2$ from each of $\mathcal{R}$ and $\mathcal{C}$, then retrieve contexts from $\mathcal{K}$.
- 18: Update critics N_e times by Eq. (10).
- 19: **end for**
- 20: **for** $n = 1, 2, \dots$ **do**
- 21: Sample $B/2$ from each of $\mathcal{R}$ and $\mathcal{C}$, then retrieve contexts from $\mathcal{K}$.
- 22: Update critics N_e times by Eq. (10) and θ once by Eq. (15).
- 23: **if** $n \bmod N_{\text{pub}} = 0$ **then**
- 24: Publish the updated θ to the actor.
- 25: **end if**
- 26: **end for**

compliance motivate two distinct teleoperation schemes: isomorphic master-based teleoperation for long sequences and compliant interaction, and keyboard-based teleoperation for tasks with limited orientation variation and fine adjustment.

Active Isomorphic Master Interface. For the UR5e platform, we design and build the six-DoF active, kinematically isomorphic master interface shown in Fig. 4. Its joint topology and axes match the UR5e, while parameterized links support an arbitrary task-specific scale factor and preserve direct joint-space mapping without online inverse kinematics. The structural parts can be 3D-printed directly from the design files and assembled with off-the-shelf components, enabling straightforward replication. To preserve the device and action distribution across demonstration collection and online correction, the master follows the slave and maintains configuration alignment during policy execution, then maps operator input to the slave for immediate takeover upon intervention.

This active master poses two hardware challenges. First, insufficient actuation prevents the master from following policy-driven slave motion, whereas high manual resistance impedes intervention, disrupting the transition from policy execution to operator takeover. Second, transmitting link loads through the servos increases drivetrain backlash and gear wear, degrading joint-mapping accuracy and long-term reliability. In GELLO [38], uniform low-torque servos cannot reliably drive the master to follow policy-controlled slave motion, while servo-borne link loads accumulate backlash and gear wear under repeated operation. In contrast, we address the former with load-aware graded actuation and the latter with a load-decoupled joint structure.

In the graded actuation design, we match servo torque and

mass to the load at each joint. Specifically, high-torque XM540 servos at J1–J2 carry the dominant gravity loads, an XC430-T240BB balances torque and mass at J3, and lightweight XC330-T288 servos at J4–J6 reduce wrist inertia. Accordingly, the proximal joints provide the torque required for policy following and gravity compensation while distal inertia and manual resistance remain limited.

However, torque alone cannot ensure structural stability over prolonged data collection. We therefore use an independent shaft, dual-bearing support, and coupling to decouple load bearing from actuation. In this structure, the support carries radial and bending loads, while the servo provides only actuation. Consequently, the resulting load path limits drivetrain backlash and gear wear while preserving joint stiffness and transmission consistency during long-term operation.

Incremental Cartesian Keyboard Interface. For stages with limited orientation variation but high precision demands, continuous teleoperation interfaces, including VR controllers and isomorphic masters, are ill-suited to sustained alignment at millimeter and submillimeter scales because hand tremor, device noise, and mapping error directly perturb control commands. To address this limitation, our keyboard interface maps keystrokes to predefined six-DoF translational or rotational increments in a fixed task frame and provides coarse and fine step sizes. Coarse steps accelerate target approach, whereas fine steps provide repeatable corrections at millimeter or submillimeter scales without requiring the operator to hold a precise pose. This discrete control improves precision and repeatability during final alignment. However, repeated keystrokes become cumbersome for long sequences or large reorientations, for which the isomorphic interface is preferred.

State–Action Representation. We adopt a step-wise delta task-space action representation, whose bounded single-step end-effector increments facilitate stable optimization and fine-motion learning. Under this representation, the robot state and action are recorded as

$$\begin{aligned} q_t &= [x_{0 \rightarrow t}^{\text{rel}}, \dot{x}_t, f_t, \mu_t, g_t] \in \mathbb{R}^{19}, \\ u_t &= [\delta x_t^{\text{cmd}}, g_t^{\text{cmd}}] \in \mathbb{R}^7, \end{aligned} \quad (16)$$

where $x_{0 \rightarrow t}^{\text{rel}} \in \mathbb{R}^6$ represents the TCP pose relative to the episode start, $\dot{x}_t \in \mathbb{R}^6$ is the TCP twist, $f_t, \mu_t \in \mathbb{R}^3$ are measured force and torque, and g_t is the gripper state. The incremental end-effector command $\delta x_t^{\text{cmd}} \in \mathbb{R}^6$ is expressed in the configured frame relative to the preceding-step TCP pose. Joint-correspondence commands from the isomorphic master are converted to slave TCP increments before storage, thereby matching the keyboard interface. Across both interfaces, a unified pipeline records synchronized state–action pairs for training at a nominal 15 Hz.

2) *Two-Stage Real-World Training:* VLA-Precision adopts two-stage post-training that combines demonstration-driven initialization with real-world online improvement. Stage I fully fine-tunes pretrained $\pi_{0.5}$ on task demonstrations through flow-matching imitation learning. Stage II freezes the resulting base parameters (Θ_f, ψ) , initializes zero-effect trainable LoRA parameters θ in the action expert, updates only θ , and retains the corresponding frozen LoRA parameters θ_0 as the Stage-I reference. The complete procedure is shown in Algorithm 1.

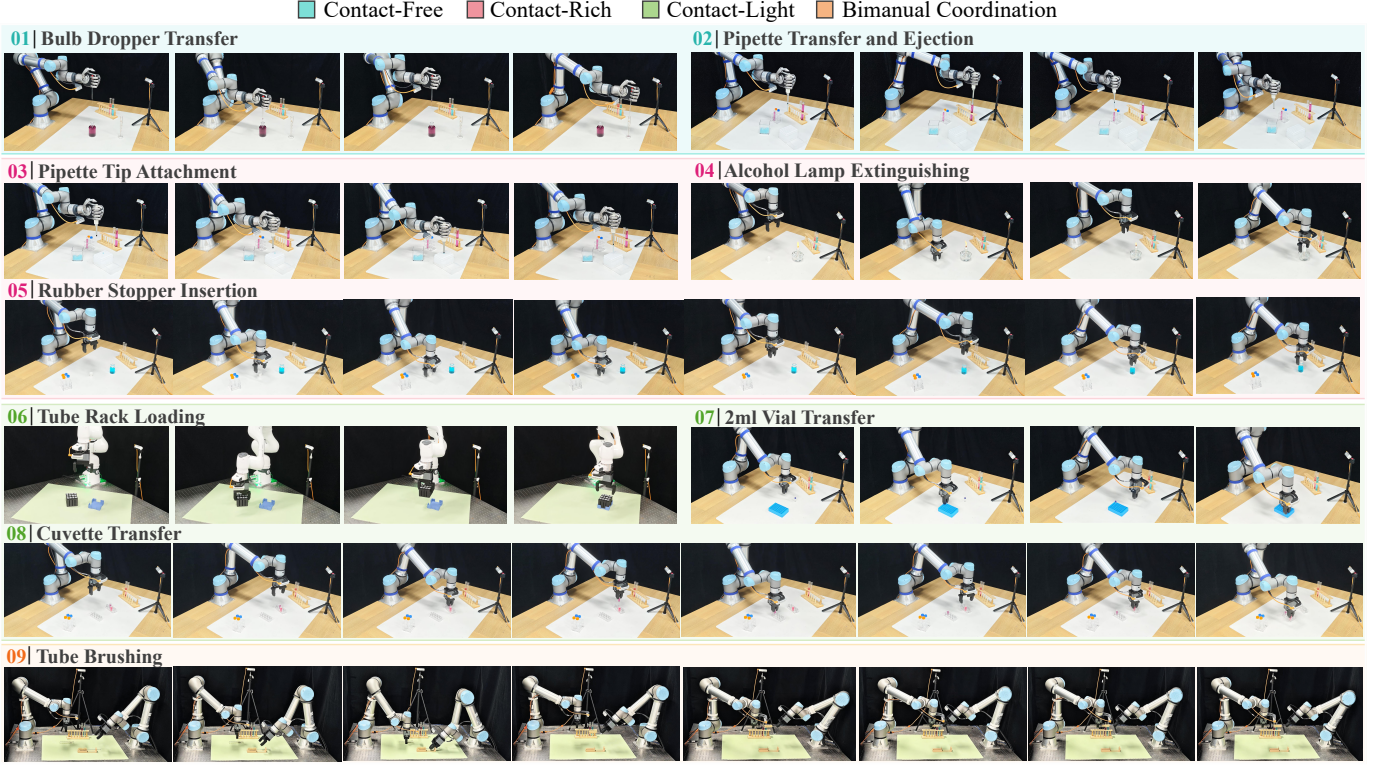


Fig. 5. Real-world chemistry manipulation task suite.

3) *Robot Deployment*: To ensure safe, high-precision execution under physical contact, we implement robot-specific Cartesian impedance controllers: force-mode wrench control for UR5e and torque-level control for Franka.

UR5e Force-Mode Impedance Control. At control frequency f_c , we compute the following Cartesian PD wrench $w_B \in \mathbb{R}^6$ and apply it through the UR `forceMode` interface:

$$w_B = \begin{bmatrix} f_B \\ \mu_B \end{bmatrix} = \begin{bmatrix} K_p e_p - K_v \text{sat}_{\delta_v}(v_t) \\ f_c^{-1}(K_R e_R - K_\omega \omega_t) \end{bmatrix}. \quad (17)$$

Here, (p_t^d, R_t^d) and (p_t, R_t) denote the desired and measured end-effector poses, and (v_t, ω_t) the measured twist. With $\text{sat}_b(x) = \text{clip}(x, -b, b)$, the upper row converts position error $e_p = \text{sat}_{\delta_p}(p_t^d - p_t)$ and linear velocity into force f_B , whereas the lower row converts rotation error $e_R = \text{Log}(R_t^d R_t^\top)^\vee$ and angular velocity into moment μ_B . The four gains specify the corresponding stiffness and damping.

Franka Torque-Level Impedance Control. We realize the commanded pose with the following Cartesian impedance law, which produces the joint-torque command $\tau_t \in \mathbb{R}^7$:

$$\tau_t = (J_t^C)^\top (K_p e_t^C - K_d \nu_t^C) + N_t [K_N(q_0^J - q_t^J) - D_N \dot{q}_t^J] + \tau_C. \quad (18)$$

Here, $e_t^C = \text{sat}_c([(p_t^{d,C} - p_t^C)^\top, (e_{R,t}^C)^\top]^\top)$ and $\nu_t^C = [(v_t^C)^\top, (\omega_t^C)^\top]^\top$ denote the clipped pose error and twist in frame C . The first term maps the Cartesian PD wrench, with stiffness K_p and damping K_d , through J_t^C to the task torque. The second projects posture regulation toward q_0^J through $N_t = I - (J_t^C)^\top ((J_t^C)^\top)^\dagger$, stabilizing redundant joints without altering the end-effector task. The final term

τ_C provides Polymetis Coriolis compensation. Together, these terms form the final torque command τ_t for all seven joints.

IV. EXPERIMENTS

In this section, we first detail the experimental setup. Subsequently, we evaluate the overall performance of VLA-Precision in real-world environments. Furthermore, we evaluate the throughput and efficiency of the ACoB-Stream system architecture. Finally, we validate the core mechanisms of the ACoB algorithm through ablation studies.

A. Experimental Setup

Hardware Configuration. As shown in Fig. 6, our real-world experiments are conducted on four robotic platforms: (1) a UR5e arm with a PGI-140-80 two-finger parallel gripper, (2) a UR5e arm with a LinkerHand L20 dexterous hand, (3) two UR5e arms, each equipped with a PGI-140-80 gripper, and (4) a Franka Research 3 arm with a PGI-140-80 gripper. Each single-arm platform is equipped with one wrist camera and one external camera, whereas the dual-arm platform is equipped with two wrist cameras and one external camera for arm-local and shared scene views.

Baselines. We compare VLA-Precision with three real-world RL methods and two large VLA baselines. HIL-SERL [12] combines high-update-to-data off-policy RL with human corrections and reports near-perfect success on dexterous tasks after 1–2.5 hours of real-world training. ConRFT [14] fine-tunes the Octo VLA through offline-to-online RL with a consistency-policy action head, reporting 96.3%

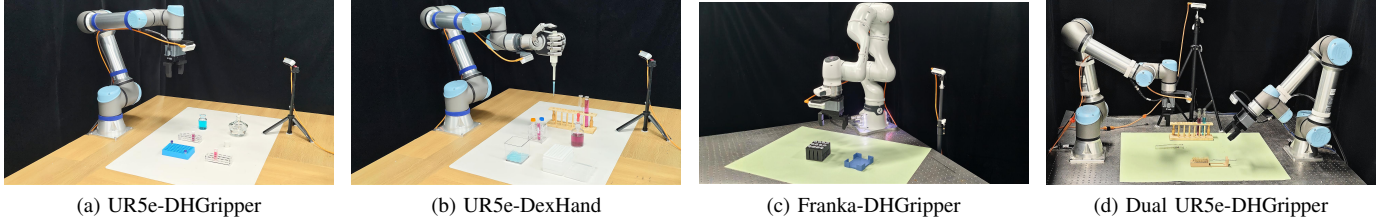


Fig. 6. Hardware platforms. Single-arm: one wrist camera and one external camera; dual-arm: two wrist cameras and one external camera.

TABLE I
REAL-WORLD PERFORMANCE ACROSS CHEMISTRY MANIPULATION TASKS.

Task	Ours Training (min)	Success Rate (%)						Average Episode Time (s)					
		HIL- SERL	ConRFT	Robo- Dopamine	π_0	$\pi_{0.5}$	Ours	HIL- SERL	ConRFT	Robo- Dopamine	π_0	$\pi_{0.5}$	Ours
Tube Rack Loading	27	0	5(+5%)	10(+10%)	75(+75%)	85(+85%)	100(+100%)	N/S	31.4($\times 0.8$)	29.8($\times 0.9$)	25.6($\times 1.0$)	24.9($\times 1.0$)	23.5($\times 1.1$)
2 mL Vial Transfer	34	0	5(+5%)	5(+5%)	80(+80%)	85(+85%)	100(+100%)	N/S	49.2($\times 0.8$)	46.8($\times 0.8$)	37.1($\times 1.0$)	33.6($\times 1.1$)	33.3($\times 1.1$)
Cuvette Transfer	65	0	0(+0%)	0(+0%)	35(+35%)	40(+40%)	90(+90%)	N/S	N/S	N/S	53.3($\times 1.0$)	52.6($\times 1.0$)	46.2($\times 1.2$)
Rubber Stopper Insertion	52	5	10(+5%)	15(+10%)	50(+45%)	70(+65%)	100(+95%)	44.6($\times 0.6$)	38.9($\times 0.7$)	37.7($\times 0.7$)	28.1($\times 1.0$)	23.8($\times 1.2$)	19.2($\times 1.5$)
Alcohol Lamp Extinguishing	23	0	5(+5%)	10(+10%)	85(+85%)	90(+90%)	100(+100%)	N/S	29.4($\times 0.8$)	27.8($\times 0.8$)	22.4($\times 1.0$)	21.1($\times 1.1$)	21.0($\times 1.1$)
Pipette Tip Attachment	44	15	35(+20%)	40(+25%)	35(+20%)	50(+35%)	100(+85%)	21.6($\times 0.8$)	17.8($\times 0.9$)	18.4($\times 0.9$)	16.6($\times 1.0$)	14.0($\times 1.2$)	11.5($\times 1.4$)
Bulb Dropper Transfer	28	5	10(+5%)	10(+5%)	90(+85%)	90(+85%)	100(+95%)	32.7($\times 0.8$)	30.8($\times 0.9$)	29.6($\times 0.9$)	26.4($\times 1.0$)	24.7($\times 1.1$)	24.4($\times 1.1$)
Pipette Transfer and Ejection	51	0	0(+0%)	0(+0%)	80(+80%)	90(+90%)	100(+100%)	N/S	N/S	N/S	36.7($\times 1.0$)	35.4($\times 1.0$)	33.6($\times 1.1$)
Tube Brushing	87	0	0(+0%)	0(+0%)	5(+5%)	10(+10%)	95(+95%)	N/S	N/S	N/S	41.7($\times 1.0$)	39.7($\times 1.0$)	35.6($\times 1.2$)
Average	45.8	2.8	7.8(+5%)	10(+7.2%)	59.4(+56.7%)	67.8(+65%)	98.3(+95.6%)	50.4($\times 0.6$)	45.3($\times 0.7$)	44.5($\times 0.7$)	32.0($\times 1.0$)	30.0($\times 1.1$)	27.6($\times 1.2$)

TABLE II
TRAINING SETTINGS FOR VLA-PRECISION AND VLA BASELINES.

Task	$\pi_0 / \pi_{0.5}$		Ours			
	SFT Demos.	SFT Steps	Stage-I Demos.	Stage-I Steps	Max Horizon	Variation Range
Tube Rack Loading	120	25000	80	10000	150	40 $\times$ 40 cm
2 mL Vial Transfer	150	25000	80	10000	250	40 $\times$ 40 cm
Cuvette Transfer	200	30000	120	15000	260	25 $\times$ 25 cm
Rubber Stopper Insertion	150	25000	80	5000	140	40 $\times$ 40 cm
Alcohol Lamp Extinguishing	120	25000	80	10000	120	40 $\times$ 40 cm
Pipette Tip Attachment	120	25000	60	5000	90	15 tip holes
Bulb Dropper Transfer	150	25000	80	15000	150	40 $\times$ 20 cm
Pipette Transfer and Ejection	150	25000	120	15000	220	2 $\times$ 4 holes
Tube Brushing	200	30000	120	15000	250	L: 10 $\times$ 10 cm; R: 15 $\times$ 10 cm

TABLE III
SHARED HYPERPARAMETERS FOR REAL-WORLD ONLINE TRAINING.

Hyperparameter	Value
λ_{rank}	50
$(w_{\text{BC}}, w_{\text{rel}}, w_{\text{ref}})$	(0.25, 0.50, 0.25)
Initial reset range	3 cm in x , y , and z
Training GPUs	4 $\times$ NVIDIA A800 GPUs

average success across eight real-world tasks within 45–90 min. Robo-Dopamine [32] performs RL on the Octo VLA using dense rewards generated by a process reward model adapted with one demonstration, reporting 95.2% average success after 150 real-world rollouts. π_0 [1] and $\pi_{0.5}$ [2] are pretrained flow-based large VLAs that we fine-tune on each task without Stage-II online RL.

Tasks and Metrics. We design nine high-precision ma-

nipulation tasks in chemistry laboratories (Fig. 5) to evaluate performance across four task categories: contact-rich, contact-light, contact-free, and bimanual coordination. To rigorously evaluate generalization, we deliberately vary object poses and initial arm poses in every task, with the evaluated ranges for object poses and arm resets reported in the Variation Range column of Table II and Table III, respectively.

1) High-Precision Contact-Rich Tasks. These tasks demand millimeter and submillimeter precision, since even slight pose errors can cause jamming or damage during insertion.

Cuvette Transfer. The robot sequentially transfers two cuvettes from one rack to two designated diagonal slots in another rack. The fully transparent glass cuvettes and racks complicate visual perception, while their fragility demands precise pose and force control.

2 mL Vial Transfer. The robot sequentially transfers two 2 mL vials from one rack to diagonal slots in another. The

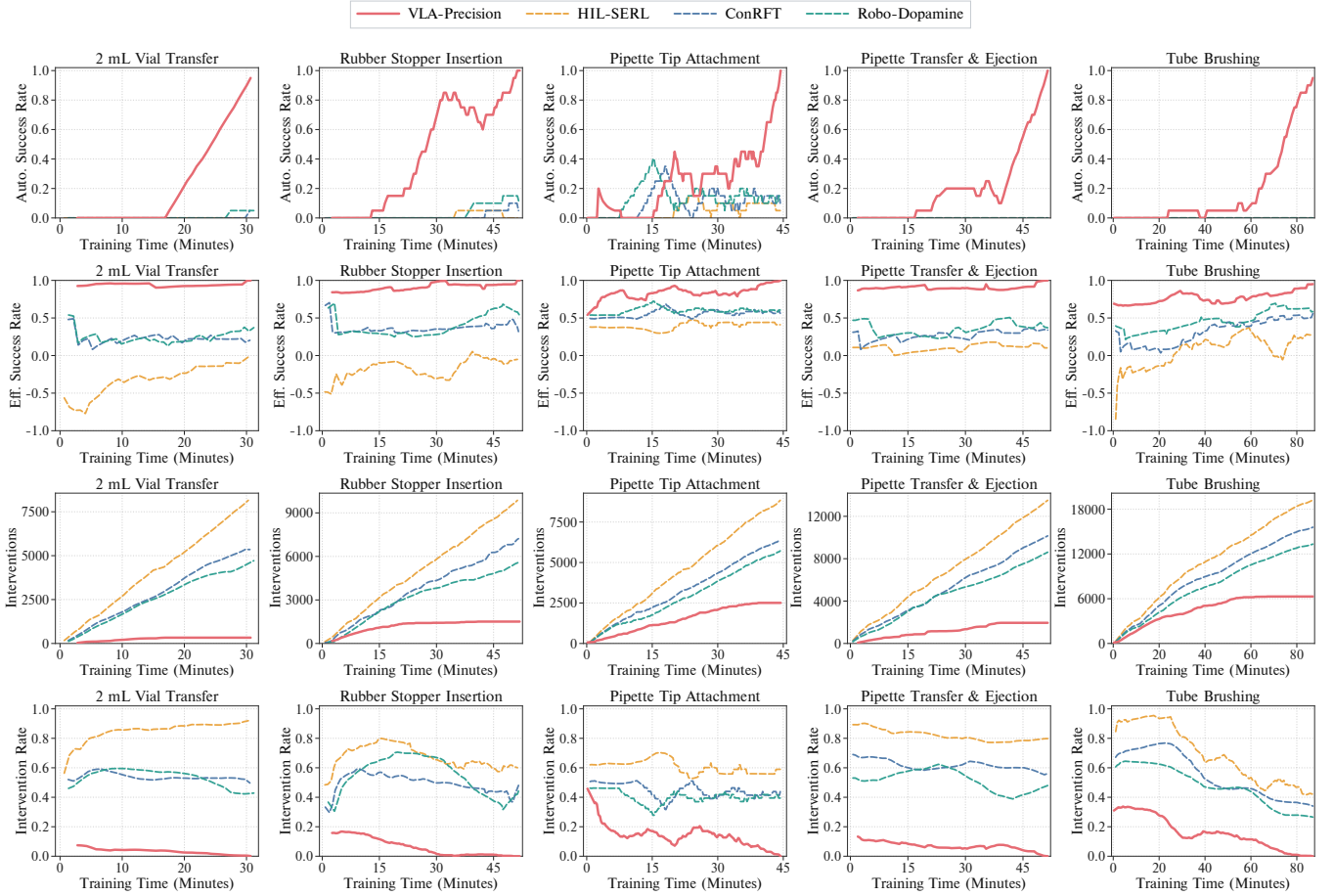


Fig. 7. Real-world online RL training dynamics on five representative tasks. Autonomous success rate is the 20-episode rolling mean of autonomous task completion. Effective success rate is the 20-episode rolling mean of task completion, including assisted completion, minus the intervention rate. Interventions denote cumulative human-corrected low-level action steps, and intervention rate is the 20-episode rolling mean of their per-episode fraction.

small vial diameter imposes tight alignment tolerances.

Tube Rack Loading. The robot lifts a centrifuge-tube rack and inserts it into a loading slot. Insertion requires complete pose alignment with the slot and precise force control.

2) *High-Precision Contact-Light Tasks.* Limited and intermittent contact demands stable action generation by coordinating visual and force feedback to prevent object instability.

Rubber Stopper Insertion. The robot inserts a rubber stopper into a reagent-bottle opening. A high grasp can crush the stopper, whereas a low grasp leaves insufficient insertion clearance, and bottle motion demands stable control.

Alcohol Lamp Extinguishing. The robot covers an alcohol-lamp wick with an extinguisher cap. Visual interference from the flame demands precise action generation through fused visual and force feedback.

Pipette Tip Attachment. The robot aligns a pipette with a tip and presses it into place. The narrow tip aperture requires precise alignment and appropriate axial force.

3) *High-Precision Contact-Free Tasks.* Without contact feedback, these tasks rely on visual observations and robot state for action generation while limiting accumulated error.

Bulb Dropper Transfer. The robot uses a flexible bulb dropper to aspirate and dispense liquid. Visual guidance must

coordinate finger actuation with precise end-effector alignment, leaving little tolerance for timing or pose errors.

Pipette Transfer and Ejection. The robot transfers liquid and ejects the used pipette tip. The long sequence requires visually guided coordination of hand state and end-effector motion while limiting accumulated error.

4) *Bimanual High-Precision Coordination Task.* This task demands high-precision bimanual coordination, since errors from either arm directly disrupt execution.

Tube Brushing. One arm stabilizes a test tube while the other inserts and manipulates a brush. The transparent tube complicates visual localization, while brushing requires high-precision coordination between the two arms.

B. Real-World Evaluation of VLA-Precision

Overall Performance. Table I reports the real-world experimental results across nine chemistry manipulation tasks. Compared with the baselines, VLA-Precision achieves the highest average success rate of 98.3% after an average of 45.8 min of online training per task and completes successful trials in 27.6 s on average. It succeeds in 177 of 180 held-out trials, reaches 100% on seven tasks, and maintains at least 90% success on every task. Relative to $\pi_{0.5}$ and π_0 , VLA-Precision

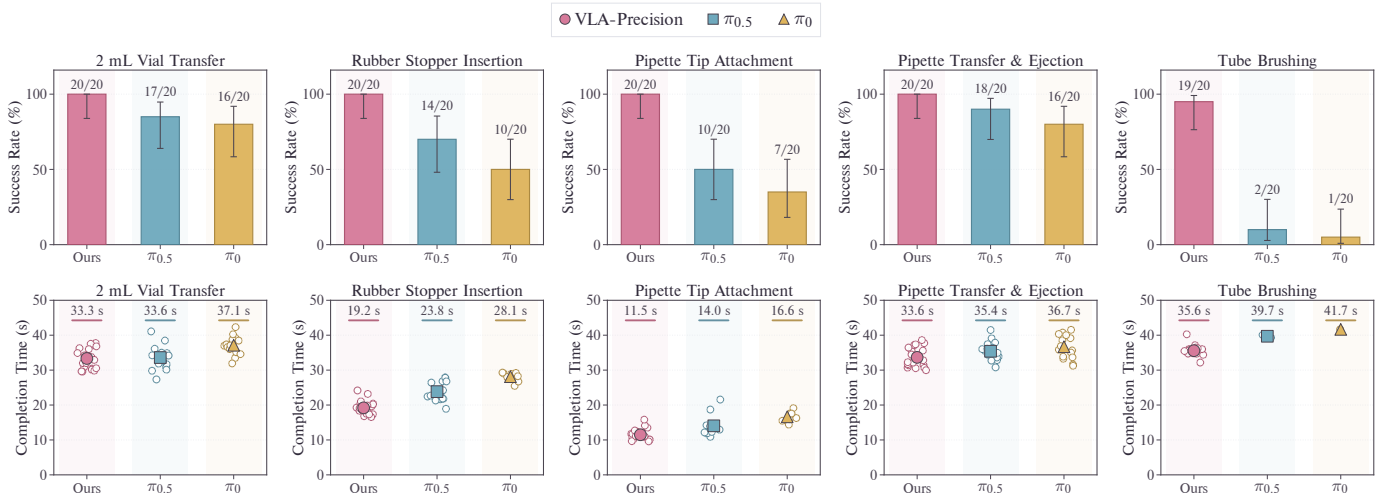


Fig. 8. Real-world test performance on five representative tasks. Each method–task pair comprises 20 trials. Top: success rates with two-sided Wilson 95% confidence intervals. Bottom: successful-trial completion times (open circles) and their means (filled markers).

improves average success by 30.5% and 38.9% and improves average execution speed by 8.7% and 15.9%, respectively. Its worst-task success rate is 90%, compared with 10% for $\pi_{0.5}$ and 5% for π_0 . Relative to the strongest real-world RL baseline, Robo-Dopamine, VLA-Precision improves average success by 88.3% and average execution speed by 61.2%.

The results reveal the limitations of the baselines. HIL-SERL learns a compact task-specific policy but starts online RL with limited task-specific competence. Although effective in shorter-horizon settings with limited task variation, it reaches only 2.8% average success on our longer-horizon, pose-randomized suite. ConRFT and Robo-Dopamine demonstrate online RL on the Octo VLA, with Robo-Dopamine providing dense progress rewards. Although both enter online RL after task-specific offline training on Octo, direct maximization of absolute Q estimates remains vulnerable to optimistic value errors and policy drift. Their average success is limited to 7.8% and 10.0%. Full-parameter fine-tuning of π_0 and $\pi_{0.5}$ retains stronger pretrained VLA priors and improves average success to 59.4% and 67.8%, but remains bounded by demonstration-only adaptation. The 98.3% average success of VLA-Precision shows that ACoB breaks the demonstration-limited ceiling through online RL while preserving the generalization inherited from the task-finetuned large VLA.

Online RL Training Performance. Fig. 7 shows four online RL metrics on five representative tasks. At the final task-specific comparison horizons, VLA-Precision leads all four metrics. Across the five tasks, autonomous success rate, effective success rate, interventions, and intervention rate average 98.0%, 99.0%, 2,517.4, and 0.024%, respectively. Relative to Robo-Dopamine, the strongest baseline, VLA-Precision raises autonomous and effective success rates by 92.0 and 49.5 percentage points while reducing interventions by 66.9% and intervention rate by 40.46 percentage points.

For the high-precision contact-rich 2 mL vial transfer task, successive pick-and-place operations form a multistep sequence. HIL-SERL begins with limited task-specific competence and never achieves autonomous success. Its effective

success rate remains below zero because assisted success cannot offset the 91.95% final intervention rate. ConRFT and Robo-Dopamine enter with task-specific behavior learned offline and lower intervention rates to 49.70% and 42.84%, yet both end at only 5% autonomous success. Their BC regularization anchors the actor, but direct absolute-Q maximization remains vulnerable to optimistic value errors and policy drift, preventing reliable autonomous multistep behavior.

For the high-precision contact-light rubber-stopper insertion and pipette-tip attachment tasks, HIL-SERL ends stopper insertion at 0% autonomous success and the 60.07% intervention rate. ConRFT and Robo-Dopamine end at only 5% and 10%, with rates of 48.08% and 45.63%, respectively. Offline-learned task behavior reduces intervention for the latter two, but unstable updates perturb the compliant stopper and easily displaced bottle, preventing stable behavior. Tip attachment is the only short-sequence task without gripper control. The three baselines peak at 15%, 35%, and 40% autonomous success but fall to 5%, 10%, and 15% across 15 randomized holes. The higher VLA peaks show faster local acquisition, but susceptibility to optimistic value errors and policy drift prevents robust pose generalization.

For the high-precision contact-free pipette transfer and tip ejection task, the alignment demand of tip attachment expands into a long sequence. All three baselines remain at 0% autonomous success. Starting with weaker task-specific competence, HIL-SERL accumulates 13,512 interventions with a 79.82% intervention rate, while ConRFT and Robo-Dopamine require 10,154 and 8,614 interventions at rates of 56.10% and 47.96%, respectively. Stable insertion into and withdrawal from narrow tube openings require precise actions conditioned on vision and robot state. Task-specific offline initialization reduces intervention, and Robo-Dopamine adds dense progress rewards. However, direct absolute-Q maximization leaves both methods vulnerable to accumulated value and motion errors, so alignment cannot remain stable over the full sequence.

For the bimanual high-precision test-tube brushing task, the coupled dual-arm data distribution and long sequence leave

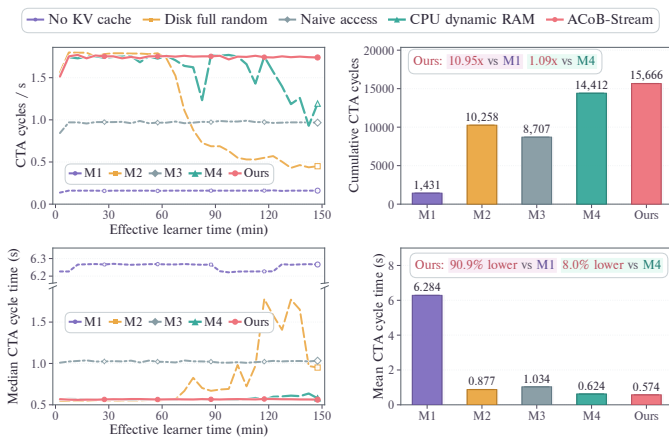


Fig. 9. Training throughput and latency comparison. Top left: CTA throughput versus effective learner time. Top right: cumulative CTA cycles after 150 min. Bottom left: median CTA cycle time versus effective learner time. Bottom right: mean CTA cycle time. A CTA cycle comprises one critic-only update and one joint critic-actor update.

all baselines at 0% autonomous success. Starting with weaker task-specific competence, HIL-SERL accumulates 19,196 interventions. ConRFT and Robo-Dopamine begin with task behavior learned offline and reduce the counts to 15,587 and 13,322, yet neither succeeds autonomously. Their BC regularization anchors the actor, but absolute-Q policy drift compounds across both arms and destabilizes the sequence.

ACoB instead rapidly absorbs online corrections through intervention-guided behavior cloning, continually calibrates value estimates through global returns and state-matched preferences, and stabilizes relative-advantage updates with a frozen reference. Together, these mechanisms raise the autonomous success rate and reduce the intervention rate across all five tasks, validating asymmetric co-bootstrapping.

Offline VLA Test Performance. We further evaluate VLA-Precision, $\pi_{0.5}$, and π_0 offline on the same five representative tasks. As summarized in Table II, π_0 and $\pi_{0.5}$ receive 120–200 demonstrations and 25,000–30,000 SFT steps, whereas VLA-Precision uses 60–120 demonstrations and 5,000–15,000 Stage-I steps before online RL.

Fig. 8 reports 20 trials per method and task. Overall, VLA-Precision succeeds in 99 of 100 trials, compared with 61 for $\pi_{0.5}$ and 50 for π_0 , improving average success by 38 and 49 percentage points, respectively. Notably, it reaches 100% on all four single-arm tasks and 95% on bimanual tube brushing. Across the single-arm tasks, $\pi_{0.5}$ ranges from 50–90% and π_0 from 35–80%. On brushing, they fall to 10% and 5%. VLA-Precision also completes successful trials in 26.65 s on average, compared with 29.32 s for $\pi_{0.5}$ and 32.05 s for π_0 .

More importantly, the task-level gaps expose the limits of demonstration-only fine-tuning under pose randomization, precision demands, and long-horizon coordination. Collectively, these results show that online RL strengthens the task-specific VLA policy beyond demonstration-only adaptation, yielding consistently robust execution across the five tasks.

C. Throughput and Efficiency Evaluation of ACoB-Stream

To quantify the system-level advantages of ACoB-Stream in large-VLA online RL, we compare it with four actor-

learner system baselines in online RL update throughput and computational efficiency. These baselines extend the asynchronous actor-learner architecture. HIL-SERL [12] is the most representative real-world implementation of this architecture. It decouples robot interaction from policy optimization so that data collection and policy updates proceed without synchronous blocking. We adapt this architecture to large VLAs and construct three system-level ablations plus a competitive CPU-resident alternative.

The first three baselines selectively remove ACoB-Stream mechanisms. **M1: No KV Cache** removes cross-update reuse of frozen-prefix contexts and recomputes the required contexts during every update. **M2: Disk Full Random** retains the disk-backed context store but replaces sliding-window sampling with uniform sampling over the full replay history. **M3: Naive Context Access** removes objective-aligned context retrieval by always loading both current and successor contexts, and disables on-demand context prefetch. **M4: Dynamic CPU RAM** is a competitive alternative that maintains a dynamically sized replay and KV active window in learner CPU RAM, applies FIFO eviction, and retains disk copies for persistence. **M5: Disk ACoB-Stream** is the full system, combining cross-update reuse of frozen-prefix contexts, context deduplication with disk-backed sliding-window sampling, and objective-aligned context retrieval with on-demand context prefetch.

Fig. 9 reports the system evaluation. Disk ACoB-Stream achieves the highest update throughput and the lowest mean critic-to-actor (CTA) cycle latency, completing 15,666 CTA cycles in 150 min at 1.7407 CTA/s and 0.574 s per cycle. Compared with No KV Cache, it delivers 10.95 $\times$ the throughput and reduces mean latency by 90.9%, demonstrating the computational value of cross-update frozen-prefix reuse. Against Disk Full Random, it provides 1.53 $\times$ the throughput and 34.5% lower mean latency. Moreover, full-history random sampling begins to degrade at 43 min as the replay working set exceeds the Linux page-cache capacity, causing more frequent disk-to-CPU transfers, with its throughput eventually declining from 1.56 to 0.45 CTA/s, while Disk ACoB-Stream reaches 1.74 CTA/s in the final 5-min bin, showing that sliding-window sampling preserves disk locality as replay grows. Objective-aligned retrieval and on-demand prefetch jointly provide 1.80 $\times$ the throughput and 44.4% lower mean latency than Naive Context Access, confirming the benefit of loading and transferring only required contexts. Although Dynamic CPU RAM remains competitive, it requires a more complex memory hierarchy to manage dynamically sized CPU-resident replay and KV windows, coordinate FIFO eviction, and maintain disk persistence. Disk ACoB-Stream delivers 1.09 $\times$ the throughput and 8.0% lower latency while eliminating CPU-disk active-window coordination overhead.

D. Ablation Experiments for the ACoB Algorithm

To evaluate the contributions of the core mechanisms in ACoB, we compare the full algorithm with three ablated variants on four representative tasks. **w/o critic preference** removes the intervention-guided critic preference loss and computes the same paired action comparison through equivalent relative Q-value differences rather than relative advantage

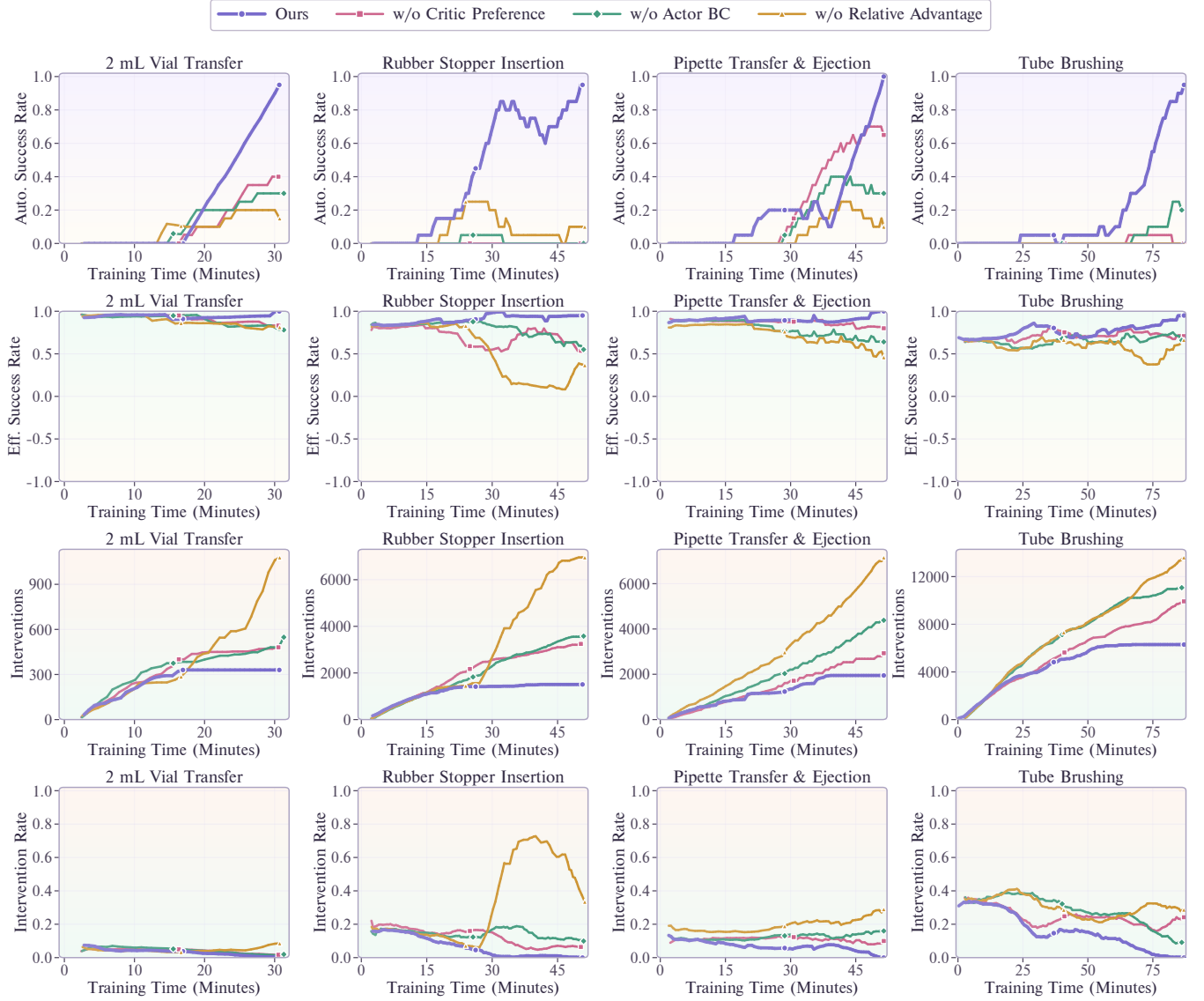


Fig. 10. ACoB ablation training dynamics across four representative tasks. Columns denote tasks; rows show autonomous success rate, effective success rate, cumulative interventions, and intervention rate over training time. Rate metrics are 20-episode rolling means.

differences. **w/o actor BC** removes intervention-guided behavior cloning and renormalizes the remaining actor-loss weights. **w/o relative advantage** replaces relative-advantage policy improvement with direct sampled-action Q maximization.

Overall Results. Fig. 10 reports results at the per-task horizons shared by all four variants. Full ACoB reaches 96.25% final autonomous success with a 0.03% intervention rate on average. Removing critic preference, actor BC, or relative advantage reduces autonomous success to 26.25%, 20.00%, and 8.75%, while increasing intervention rates to 10.57%, 9.24%, and 24.93%. Interventions across the four tasks also rise from 10,080 to 16,580, 19,581, and 28,806. These degradations demonstrate that the evaluated mechanisms are required for reliable autonomous improvement.

w/o critic preference. Without state-matched preference ranking, TD observes only executed actions and cannot correct the value of overwritten policy proposals. Returns obtained after human correction can therefore leave failed proposals

overvalued, weakening subsequent action comparisons. The 70.00-point success drop demonstrates that critic preference resolves this local credit ambiguity and progressively calibrates action-dependent value estimates.

w/o actor BC. Removing actor BC prevents successful and human-corrected actions from being directly absorbed by the action expert. Behavioral improvement must then rely on slower, indirect critic guidance, increasing intervention and degrading the online experience underlying value calibration. The 76.25-point success drop confirms that actor BC rapidly converts corrections into stable behavior and improves the experience supporting subsequent value learning.

w/o relative advantage. Replacing relative-advantage improvement with direct Q maximization produces the largest degradation: an 87.50-point success drop and 24.93% intervention rate. Direct maximization exposes the action expert to absolute value scale and optimistic critic errors, favoring high predicted Q values over reliable local improvement. Relative

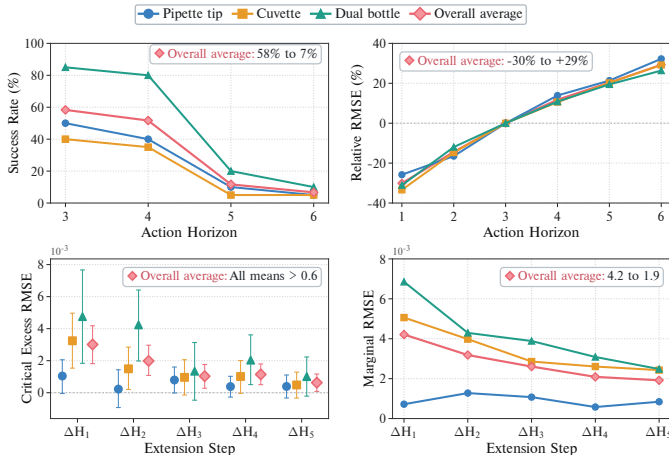


Fig. 11. Action-horizon analysis of $\pi_{0.5}$ on three representative precision-insertion tasks. Top left: autonomous success rate versus executed action horizon H_e . Top right: relative action-prediction RMSE, measuring predicted-demonstration action discrepancy as the percentage change from $H_e = 3$. Bottom left: excess marginal RMSE in interaction-critical phases relative to normal motion. Bottom right: marginal RMSE increase for each extension ΔH_i , where ΔH_i denotes increasing H_e from i to $i + 1$. Colored markers denote task means, red diamonds their equal-task-weighted mean, and bottom-left whiskers 95% episode-bootstrap confidence intervals.

advantage is therefore essential for translating value estimates into stable policy improvement while suppressing policy drift.

In summary, the ablations validate asymmetric co-bootstrapping in ACoB: actor BC rapidly improves behavior and online experience quality, while critic preference progressively calibrates value estimates, yielding reliable relative advantages for stable policy improvement.

V. DISCUSSION

Our results indicate that reliable real-world RL of large VLAs depends on resolving an early temporal asymmetry: task fine-tuning yields a capable policy before online experience can adequately calibrate the critic. Consequently, conservative value-guided updates learn slowly, whereas aggressive updates risk policy drift. ACoB closes this gap by rapidly improving behavior from successful and corrected actions while progressively calibrating value estimates, creating a positive feedback loop that advances the policy beyond direct supervision without eroding task-specific competence.

Practical real-world RL of large VLAs also depends on how efficiently new experience is converted into policy updates. As data accumulate, redundant computation, transfer, and random access increase update latency. ACoB-Stream shows that jointly managing reusable computation, experience, and policy state can sustain this experience-policy loop under finite compute and memory during continuous online training.

More broadly, our findings show that real-world RL can improve an already capable VLA while preserving its task-specific competence. In high-precision tasks, small value-induced policy shifts can destabilize objects, damage equipment, or cause unsafe interaction. Maintaining behavioral stability during improvement is therefore central to extending online RL to higher-risk precision tasks.

Building on these findings, we discuss two broader research directions that warrant further investigation:

1) *Long-Horizon Bimanual Real-World RL for VLAs*. The evaluation includes one bimanual precision task but not longer sequences with multiple precision-critical stages, where local errors accumulate and degrade later operations. Longer sequences make the reference scheme for delta actions increasingly important. Let x_t^{ref} denote the TCP pose at the start of the current action horizon. Step-wise and chunk-wise forms decode as

$$\begin{aligned} \hat{x}_{t+k}^{\text{step}} &= x_t^{\text{ref}} + \sum_{j=1}^k z_{t,j}^{\text{step}}, \\ \hat{x}_{t+k}^{\text{chunk}} &= x_t^{\text{ref}} + z_{t,k}^{\text{chunk}}, \quad k = 1, \dots, H_e, \end{aligned} \quad (19)$$

where step-wise deltas reference the preceding TCP pose, keeping the target distribution compact and bounded but recursively accumulating prediction errors. Chunk-wise deltas reference the horizon-start pose and avoid recursive accumulation, although their target distribution broadens as H_e increases. We therefore use step-wise deltas with $H_e = 3$ to preserve frequent closed-loop correction while limiting recursive accumulation in the current precision setting.

To quantify this horizon-dependent error accumulation, we evaluate the $\pi_{0.5}$ fine-tuned on task demonstrations using step-wise deltas at different executed action horizons on three representative high-precision insertion tasks, as shown in Fig. 11. Specifically, increasing H_e from 3 to 6 reduces average autonomous success from 58.3% to 6.7% and increases relative action-prediction RMSE by 29.3%. Although the marginal RMSE introduced by each horizon extension decreases, it remains positive and is consistently higher during interaction-critical phases than during normal motion. These results expose the limitation of step-wise deltas as the execution horizon grows and motivate chunk-wise deltas for long-horizon bimanual manipulation. Feng et al. [39] provide complementary theoretical and empirical support by deriving $O(H_e)$ and $O(1)$ worst-case error growth for step-wise and chunk-wise deltas, respectively, and reporting superior chunk-wise performance in long-horizon imitation learning.

2) *Multi-Task Real-World RL for Large VLAs*. An important extension beyond current task-specific training is to apply ACoB to joint multi-task optimization of a shared large VLA, enabling interaction experience to transfer across related skills while reducing per-task data collection. Task-conditioned value learning with balanced interaction data and retained demonstrations provides a feasible path to limiting interference and catastrophic forgetting, as supported by recent multi-task real-world VLA post-training [34].

VI. CONCLUSION

In this work, we present VLA-Precision, an efficient framework for real-world online RL of large VLAs. Within this framework, ACoB couples rapid behavioral learning with progressive value calibration across timescales, improving high-precision policies beyond demonstration-based limits while suppressing drift. To make this optimization efficient for large VLAs, ACoB-Stream establishes invariant-state decoupling

and on-demand streaming as design principles, delivering up to $10.9\times$ improvements in throughput and computational efficiency. Extensive evaluations across nine high-precision chemistry tasks and four robot embodiments demonstrate 98.3% average success in 45.8 min/task, with 27.6 s episodes running at $1.2\times$ and $1.8\times$ the speeds of VLA and RL baselines.

REFERENCES

- [1] K. Black, N. Brown, D. Driess, A. Esmail, M. R. 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, L. Smith, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky, “ π_0 : A vision-language-action flow model for general robot control,” in *Proceedings of RSS*, 2025.
- [2] K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. R. 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,” in *Proceedings of CoRL*, vol. 305, pp. 17–40, 2025.
- [3] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. P. Foster, P. R. 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,” in *Proceedings of CoRL*, vol. 270, pp. 2679–2713, 2025.
- [4] 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 *Proceedings of ICRA*, pp. 15665–15672, 2025.
- [5] W. Xiao, H. Lin, A. Peng, H. Xue, T. He, Y. Xie, F. Hu, J. Wu, Z. Luo, L. Fan, G. Shi, and Y. Zhu, “Self-improving vision-language-action models with data generation via residual RL,” in *Proceedings of ICLR*, 2026.
- [6] F. Zhu, Z. Yan, Z. Hong, Q. Shou, X. Ma, and S. Guo, “WMPO: World model-based policy optimization for vision-language-action models,” in *Proceedings of ICLR*, 2026.
- [7] 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.
- [8] H. Zang, M. Wei, S. Xu, Y. Wu, Z. Guo, Y. Wang, H. Lin, P. Wang, H. Yuan, Y. Zhang, L. Shi, Y. Xie, Z. Xu, Z. Liu, K. Chen, W. Tang, Q. Zhang, W. Zhang, C. Yu, and Y. Wang, “RLux-VLA: A unified and efficient framework for reinforcement learning of vision-language-action models,” in *Proceedings of RSS*, 2026.
- [9] A. Amin, R. Aniceto, A. Balakrishna, K. Black, K. Conley, G. B. Connors, J. Darpinian, K. Dhabalia, J. Di Carlo, D. Driess, M. R. 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,” in *Proceedings of RSS*, 2026.
- [10] P. Dong, K.-H. Hung, T. Gao, D. Sadigh, and C. Finn, “EXPO-FT: Sample-efficient reinforcement learning finetuning for vision-language-action models,” *arXiv preprint arXiv:2605.25477*, 2026.
- [11] C. Xu, J. T. Springenberg, M. Equi, A. Amin, A. Esmail, S. Levine, and L. Ke, “RL Token: Bootstrapping online RL with vision-language-action models,” *arXiv preprint arXiv:2604.23073*, 2026.
- [12] J. Luo, C. Xu, J. Wu, and S. Levine, “Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning,” *Sci. Robot.*, vol. 10, no. 105, p. eads5033, 2025.
- [13] K. Lei, H. Li, D. Yu, Z. Wei, L. Guo, Z. Jiang, Z. Wang, S. Liang, and H. Xu, “Performer robotic manipulation with real-world reinforcement learning,” *Sci. Robot.*, vol. 11, no. 116, p. ead6267, 2026.
- [14] 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 RSS*, 2025.
- [15] Y. Zhao, H. Jin, L. Jiang, X. Zhang, K. Wu, P. Ren, Z. Xu, Z. Che, L. Sun, D. Wu, C. H. Liu, and J. Tang, “Real-world reinforcement learning from suboptimal interventions,” *arXiv preprint arXiv:2512.24288*, 2025.
- [16] L. Shi, S. Chen, F. Gao, Y. Chen, K. Chen, T. Zhang, H. Zang, J. Zhou, W. Zhang, C. Yu, and Y. Wang, “Beyond imitation: Reinforcement learning-based sim-real co-training for VLA models,” *arXiv preprint arXiv:2602.12628*, 2026.
- [17] Y. Ma, Z. Song, Y. Zhuang, J. Hao, and I. King, “A survey on vision-language-action models for embodied AI,” *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 37, no. 7, pp. 3031–3051, 2026.
- [18] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, “Isaac gym: High performance GPU based physics simulation for robot learning,” in *Proceedings of NeurIPS*, vol. 1, 2021.
- [19] J. Gu, F. Xiang, X. Li, Z. Ling, X. Liu, T. Mu, Y. Tang, S. Tao, X. Wei, Y. Yao, X. Yuan, P. Xie, Z. Huang, R. Chen, and H. Su, “ManiSkill2: A unified benchmark for generalizable manipulation skills,” in *Proceedings of ICLR*, 2023.
- [20] S. Tan, K. Dou, Y. Zhao, and P. Krähenbühl, “Interactive post-training for vision-language-action models,” *arXiv preprint arXiv:2505.17016*, 2025.
- [21] J. Liu, F. Gao, B. Wei, X. Chen, Q. Liao, Y. Wu, C. Yu, and Y. Wang, “What can RL bring to VLA generalization? an empirical study,” in *Proceedings of NeurIPS*, vol. 38, 2025.
- [22] 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 *Proceedings of ICLR*, 2026.
- [23] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone, “LIBERO: Benchmarking knowledge transfer for lifelong robot learning,” in *Proceedings of NeurIPS*, vol. 36, pp. 44776–44791, 2023.
- [24] 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,” *arXiv preprint arXiv:2510.25889*, 2025.
- [25] Z. Jiang, S. Zhou, Y. Jiang, Z. Huang, M. Wei, Y. Chen, T. Zhou, Z. Guo, H. Lin, Q. Zhang, Y. Wang, H. Li, C. Yu, and D. Zhao, “WoVR: World models as reliable simulators for post-training VLA policies with RL,” *arXiv preprint arXiv:2602.13977*, 2026.
- [26] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine, “Scalable deep reinforcement learning for vision-based robotic manipulation,” in *Proceedings of CoRL*, vol. 87, pp. 651–673, 2018.
- [27] Y. Zheng, F. F. Veiga, J. Peters, and V. J. Santos, “Autonomous learning of page flipping movements via tactile feedback,” *IEEE Trans. Robot.*, vol. 38, no. 5, pp. 2734–2749, 2022.
- [28] C. Zhao, C. Jiang, L. Luo, S. Yuan, Q. Chen, and H. Yu, “Learning thin deformable object manipulation with a multisensory integrated soft hand,” *IEEE Trans. Robot.*, vol. 41, pp. 4606–4621, 2025.
- [29] S. Aslam, K. Kumar, P. Zhou, H. Yu, M. Y. Wang, and Y. She, “DartBot: Overhand throwing of deformable objects with tactile sensing and reinforcement learning,” *IEEE Trans. Autom. Sci. Eng.*, vol. 22, pp. 13644–13661, 2025.
- [30] P. Liu, H. Bou-Ammar, J. Peters, and D. Tateo, “Safe reinforcement learning on the constraint manifold: Theory and applications,” *IEEE Trans. Robot.*, vol. 41, pp. 3442–3461, 2025.
- [31] D. Ghosh, H. 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 RSS*, 2024.
- [32] H. Tan, S. Chen, Y. Xu, Z. Wang, C. Chi, Y. Ji, Y. Lyu, Z. Zhao, X. Chen, P. Co, S. Xie, G. Yao, P. Wang, Z. Wang, and S. Zhang, “General process reward modeling for robotic reinforcement learning,” in *Proceedings of CVPR*, pp. 22412–22422, 2026.
- [33] 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 *Proceedings of ICRA*, pp. 16961–16969, 2024.
- [34] 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,” *arXiv preprint arXiv:2601.03044*, 2026.
- [35] H. Zang, S. Yu, H. Lin, T. Zhou, Z. Huang, Z. Guo, X. Xu, Y. Sheng, J. Zhou, S. Zhang, F. Gao, W. Tang, Y. Yue, Q. Zhang, X. Chen, C. Yu,

- and Y. Wang, “USER: A unified and extensible system for online real-world policy learning in embodied AI,” in *Proceedings of RSS*, 2026.
- [36] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, “LoRA: Low-rank adaptation of large language models,” in *Proceedings of ICLR*, 2022.
 - [37] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, “Flow matching for generative modeling,” in *Proceedings of ICLR*, 2023.
 - [38] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel, “GELLO: A general, low-cost, and intuitive teleoperation framework for robot manipulators,” in *Proceedings of IROS*, pp. 12156–12163, 2024.
 - [39] Y. Feng, J. Zheng, Z. Wang, D. Liu, J. Li, J. Pang, T. Wang, and X. Zhan, “Demystifying action space design for robotic manipulation policies,” in *Proceedings of ICML*, 2026.