ATOMICRL: ATOMIC REINFORCEMENT LEARNING FOR LONG-HORIZON ROBOTIC MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Vision–language–action (VLA) models have shown promise in robotic manipulation, but imitation learning (IL) alone often struggles to master complex long-horizon skills. Reinforcement learning (RL) offers an effective avenue for further improving VLA policies. However, existing VLA-RL methods typically suffer from delayed and biased credit assignment due to trajectory-level sparse rewards, while imbalanced subtask mastery creates bottlenecks that limit overall performance. To address these challenges, we propose AtomicRL, an RL post-training framework that leverages atomic task structures for reliable credit assignment and adaptive exploration. AtomicRL introduces Semantic-Aware Policy Optimization (SAPO), which uses semantic-aware gating to constrain TD residuals from failed atomic subtasks from propagating across task boundaries, mitigating credit contamination while preserving intra-subtask credit assignment and value bootstrapping. In addition, Bottleneck-Adaptive Target Exploration (BATE) dynamically coordinates task sampling and local exploration incentives based on the mastery of individual atomic subtasks, directing learning toward the remaining bottlenecks. Finally, we develop a Real2Sim2Real pipeline that bridges online RL in simulation with real-world deployment, facilitating practical transfer to physical robots. On LIBERO-LONG, AtomicRL achieves absolute performance gains of 3.6% and 4.0% over PPO across two VLA architectures. On CALVIN ABC-D, it improves full-task performance by up to 9.5% over PPO. Combined with the Real2Sim2Real pipeline, AtomicRL further improves real-world task performance by an average of 21.3%. These results demonstrate that AtomicRL effectively improves the learning efficiency and manipulation performance of VLA policies on long-horizon tasks.

1 INTRODUCTION

Recent advances in Vision-Language Models (VLMs) (Beyer et al., 2024; Deitke et al., 2024; Yang et al., 2025) have established Vision–Language–Action (VLA) (Intelligence et al., 2025b; Kim et al., 2025; Liu et al., 2025b) models as a prominent paradigm for robotic manipulation. Although imitation learning (IL) from large-scale robot demonstrations enables VLA models to acquire diverse manipulation skills, its performance is inherently constrained by the coverage of the demonstration data. Reinforcement learning (RL) overcomes this limitation by directly optimizing policies with task feedback obtained through environment interaction, and recent work (Li et al., 2026; Xu et al., 2026) has successfully applied RL to VLA models. However, existing VLA-RL methods still face substantial challenges in long-horizon manipulation tasks.

First, most methods rely on terminal-only success rewards. In an episode that completes the first k subtasks but fails at subtask $k+1$, the absence of a terminal success reward causes generalized advantage estimation (GAE) to propagate failure-related temporal-difference (TD) residuals backward to t_0, suppressing the advantages of correctly executed actions. We refer to this as credit contamination, where useful behavior in failed trajectories is systematically devalued, with the effect worsening as the horizon grows. Prior work (Chen et al., 2025; Yang et al., 2026a; Tan et al., 2025; Zhai et al., 2025) mitigates reward sparsity through stage-level preference optimization or subtask-level intermediate rewards, but these approaches alter reward density without resolving credit contamination. Second, subtask mastery is uneven, with failures increasingly concentrated in a few bottleneck subtasks during later training. Uniform task sampling and fixed exploration then waste interaction budget on saturated subtasks while providing insufficient exploration for the bottlenecks.

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

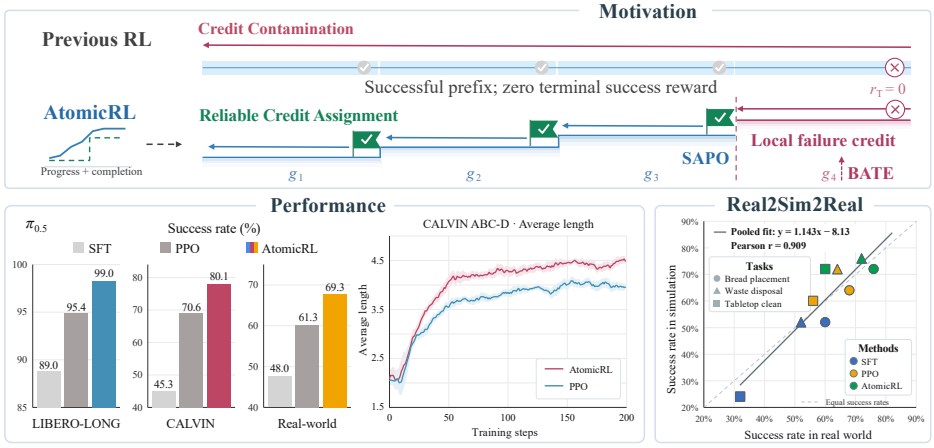


Figure 1: **Overview of AtomicRL.** Unlike previous RL methods that use standard GAE which may suffer from credit contamination, AtomicRL uses SAPO to promote more reliable credit assignment. Together with BATE and the Real2Sim2Real pipeline, AtomicRL achieves strong performance on long-horizon manipulation tasks in both simulation and the real world.

To address these challenges, we propose AtomicRL, as illustrated in Fig. 1, an RL post-training framework built around atomic task structure. Our key insight is that semantic subtasks provide a unified basis for both the propagation of learning signals and the allocation of exploration resources. AtomicRL couples Semantic-Aware Policy Optimization (SAPO) with Bottleneck-Adaptive Target Exploration (BATE), supported by a lightweight Progress–Completion Critic (PCC) that estimates atomic subtask progress and completion boundaries. PCC also provides progress estimates for constructing dense progress-based shaping rewards (PBSR). SAPO modulates TD-residual propagation in advantage estimation with a soft gate conditioned on semantic boundaries. Within each atomic subtask, it preserves standard GAE trace decay; across subtask boundaries, it maintains propagation between successful subtasks while attenuating residuals from failed subtasks. By leaving one-step TD residuals and value bootstrapping unchanged, SAPO reduces failure interference while preserving intra-subtask credit assignment and successful subtask dependencies. BATE tracks the mastery of each subtask via a moving average of its completion rate, jointly adjusts the sampling weight of full tasks and the strength of subtask-level entropy regularization, directing interaction and exploration resources toward bottleneck subtasks. Finally, to support physical deployment, we develop an automated Real2Sim2Real pipeline. Given a single real-world multimodal demonstration and the required auxiliary information, the pipeline reconstructs an interactive simulation environment. It connects simulation-based RL with real-robot deployment, shifting costly physical trial-and-error to parallel simulation with reproducible resets.

We conduct extensive experiments in both simulation and the real world. On LIBERO-LONG (Liu et al., 2023) AtomicRL achieves success rates of 99.0% with $\pi_{0.5}$ (Intelligence et al., 2025b) and 98.8% with OpenVLA-OFT (Kim et al., 2025), improving over PPO by 3.6% and 4.0%. On CALVIN ABC-D (Mees et al., 2022), the average completed sequence length improves from 3.56 to 4.56, while the success rate of completing full five-subtask chain improves by 9.5% over PPO. Combined with the Real2Sim2Real pipeline, AtomicRL further yields an average improvement of 21.3% on real-robot long-horizon tasks. These results demonstrate the effectiveness of AtomicRL for long-horizon VLA reinforcement learning. Overall, our contributions are as follows:

- We introduce AtomicRL, an RL post-training framework built around atomic task structure for long-horizon robotic manipulation.
- We propose SAPO to mitigate cross-subtask credit interference and BATE to adaptively direct task sampling and exploration toward bottleneck subtasks.
- We develop a Real2Sim2Real pipeline that streamlines simulation construction and RL training for efficient real-robot deployment.
- We validate the effectiveness of AtomicRL through extensive experiments conducted in both simulated environments and real-world robots.

2 Related Work

2.1 Reinforcement Learning for VLA Models

Reinforcement learning (RL) optimizes policies through environment interaction and task-level feedback. Recent work has explored both offline and online RL paradigms for VLA policies. Offline RL approaches leverage task feedback to enable continued policy improvement (Huang et al., 2025; Zhang et al., 2025b;a). $\pi^*_{0.6}$ (Intelligence et al., 2025a) integrates expert demonstrations, autonomous interaction data, and human corrections through advantage-conditioned learning. For scalable online training, VLA-RL (Lu et al., 2025) and SimpleVLA-RL (Li et al., 2026) explore frameworks for continually improving VLA policies through environment interaction, while RL4VLA (Liu et al., 2026) systematically investigates how different RL algorithms affect VLA generalization. Furthermore, Flow Q-Learning (Park et al., 2025), Flow-GRPO (Liu et al., 2025a) and π_{RL} (Chen et al., 2026a) investigate online RL fine-tuning for flow-matching policies, addressing challenges in action likelihood computation and exploration.

RL for long-horizon manipulation introduces additional challenges due to sparse and delayed task feedback (Chen et al., 2026b; Li et al., 2025c; Dalal et al., 2024). STARE-VLA (Xu et al., 2025) exploits the semantic stage structure of trajectories to perform stage-level preference optimization and reward shaping, while IG-RFT (Su et al., 2026) combines trajectory-level and subtask-level rewards to provide intermediate guidance. However, dense reward shaping provides useful process guidance over long-horizon trajectories, but cannot eliminate the spurious interference caused by later-stage failures. In contrast, AtomicRL gates the propagation of advantage signals at the semantic boundaries of atomic tasks to prevent cross-task interference from unreliable signals, while allocating the exploration budget to bottleneck atomic tasks to improve the efficiency of long-horizon RL.

2.2 Online Reinforcement Learning in Real World

Online reinforcement learning for real-world manipulation is constrained by interaction costs and safety risks. To reduce reliance on real-robot trial and error, some research focuses on learning action-conditioned world models and optimizing policies through imagined interactions (Team et al., 2025; Li et al., 2025a; Zhang et al., 2026c; Jiang et al., 2026a). WoVR (Jiang et al., 2026b) addresses prediction errors and their accumulation to improve the reliability of model-generated rollouts for VLA reinforcement fine-tuning , while PlayWorld (Yin et al., 2026) and RISE (Yang et al., 2026b) learn world models from autonomous robot interactions and transfers policies trained through model-based reinforcement learning to real hardware. Although world models can effectively capture short-term dynamics, long-horizon predictions still exhibit visual distortions and action–state inconsistencies. Other studies adopt an explicit Real2Sim2Real method, combining real-scene reconstruction with physics simulation to provide interactive environments for policy optimization (Li et al., 2025b; Wang et al., 2026). GSWorld (Jiang et al., 2025) integrates Gaussian-on-Mesh assets with physics engines to support visually faithful policy training, while RialTo (Torne et al., 2024) constructs digital twins of real-world environments and uses reinforcement learning in simulation to improve policy robustness. However, constructing environments requires coordinating scene reconstruction, physical asset preparation, and control alignment, making it complex and costly to reproduce. AtomicRL introduces an automated Real2Sim2Real pipeline that uses only a single demonstration to reconstruct a real-world manipulation scene as an interactive simulation environment for long-horizon online RL. The trained policies are then transferred to physical robots for evaluation.

3 Method

3.1 Overview

As illustrated in Figure 2, AtomicRL integrates semantic-aware credit assignment and bottleneck-adaptive exploration for long-horizon robotic manipulation. Specifically, SAPO modulates credit propagation across atomic subtasks (Section 3.2), while bottleneck-adaptive targeted exploration adjusts full-task trajectory sampling and local exploration incentives (Section 3.3). Both components are supported by a Progress–Completion Critic (PCC), which estimates subtask progress and identifies completion boundaries (Section 3.4). For real-world tasks, we further introduce a Real2Sim2Real construction workflow to support simulation training and subsequent transfer evaluation (Section 3.5).

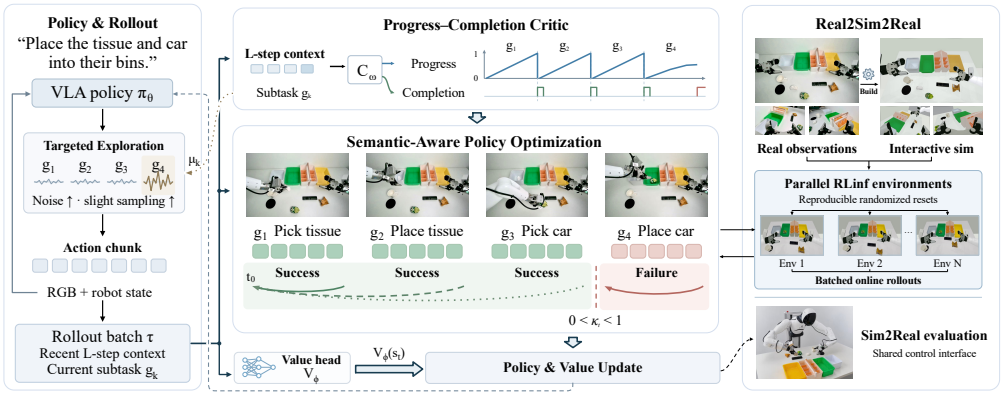


Figure 2: **Overall pipeline of AtomicRL. (a) Progress–Completion Critic:** PCC identifies atomic task boundaries and estimates atomic task progress from interaction histories. **(b) Semantic-Aware Policy Optimization:** SAPO uses these signals to regulate cross-subtask advantage propagation based on subtask execution outcomes. **(c) Bottleneck-Adaptive Targeted Exploration:** BATE adapts task sampling and local exploration according to subtask mastery, focusing training on insufficiently mastered bottleneck subtasks. **(d) Real2Sim2Real workflow:** A complementary Real2Sim2Real pipeline enables parallel simulation-based training and transfer to real-world robots.

3.1.1 Problem Formulation

We consider a language-conditioned partially observable Markov decision process for long-horizon robotic manipulation. At each decision step t, the robot receives an observation o_t, which may include third-person images, wrist-camera images, and proprioceptive states, and outputs an action chunk $a_t = (a_{t,1}, \ldots, a_{t,H_a})$, where H_a denotes the chunk length. A long-horizon instruction is decomposed into an ordered sequence of K atomic subtasks, $G = (g_1, \ldots, g_K)$. We denote the active subtask index by $z_t \in 1, \ldots, K$. For nonterminal transitions, the semantic-boundary indicator is defined as $b_t = \mathbb{I}[z_{t+1} \neq z_t]$, indicating whether execution of a_t leads to a subtask transition. Our objective is to maximize full-task success under a fixed environment-interaction budget by improving credit assignment and directing exploration toward bottleneck subtasks.

3.2 Semantic-Aware Policy Optimization

SAPO addresses temporal credit assignment in long-horizon tasks. It leverages semantic boundaries to selectively attenuate the contribution of TD residuals from failed subtasks to the advantage estimates of preceding actions, thereby regulating cross-subtask credit propagation.

Let r_t denote the chunk-level reward and d_t indicate environment termination after executing a_t. The one-step TD residual is

$$\delta_t = r_t + \gamma(1 - d_t)V_\phi(s_{t+1}) - V_\phi(s_t), \tag{1}$$

where γ is the discount factor and s_t denotes the observation- and language-conditioned input representation available to the value function, rather than a privileged simulator state. For an episode containing T chunk-level transitions indexed by $t = 1, \ldots, T$, the standard GAE estimate used by PPO is

$$\hat{A}_t^{\mathrm{PPO}} = \sum_{\ell=0}^{T-t} (\gamma\lambda)^\ell \delta_{t+\ell}, \tag{2}$$

where $\lambda \in [0, 1]$ controls trace decay. This expression is evaluated within a single episode and does not span environment resets.

SAPO distinguishes semantic segmentation from environment termination and uses atomic subtask outcomes to regulate cross-boundary credit propagation. Specifically, PCC (Section 3.4) identifies semantic boundaries b_t between atomic subtasks. Keeping the one-step TD residual unchanged,

SAPO applies an outcome-aware soft gate to the recursive GAE trace:

$$\hat{A}_t^{\mathrm{SAPO}} = \sum_{\ell=0}^{T-t} (\gamma\lambda)^\ell \left[\prod_{j=0}^{\ell-1} (1 - (1-\kappa_{t+j}) b_{t+j}) \right] \delta_{t+\ell}, \qquad \kappa_t \in [0,1]. \tag{3}$$

Within a subtask, $b_t = 0$, and the trace follows the standard $\gamma\lambda$ decay. At a semantic boundary, $\kappa_t = 1$ when the subsequent subtask succeeds, preserving ordinary propagation; when it fails, we set $0 < \kappa_t < 1$ to attenuate propagation across the boundary. Thus, when subtasks $1{:}k$ succeed but subtask $k+1$ fails, residuals from the failed segment propagate normally within that segment and are additionally attenuated when entering the completed prefix. Residuals within the completed prefix retain their standard trace weights. This design preserves within-task credit propagation and connections between successfully completed subtasks while reducing the direct negative influence of downstream failures on earlier successful behavior. We use the resulting advantages for updates and construct the value targets as $V_\phi(s_t) + \hat{A}_t^{\mathrm{SAPO}}$ before advantage normalization. By making credit propagation sensitive to semantic boundaries and local outcomes, SAPO improves credit assignment and interaction-data efficiency in long-horizon tasks.

3.3 Bottleneck-Adaptive Targeted Exploration

During the early stages of RL, broad exploration provides informative samples for most atomic subtasks. As training approaches saturation, failures may concentrate in a few bottleneck subtasks, making persistent aggressive exploration less effective and potentially disruptive to acquired skills. BATE adapts task sampling and local exploration incentives according to atomic subtask mastery.

We estimate the mastery level of each atomic-task category using an exponential moving average of its completion rate:

$$\mu_k^{(n)} = \beta \mu_k^{(n-1)} + (1-\beta)\hat{s}_k^{(n)}, \qquad \beta \in [0,1), \tag{4}$$

where n indexes statistics updates and $\hat{s}_k^{(n)}$ is the fraction of successful outcomes among subtask attempts whose outcomes are resolved during the update.

The competence gap, $1 - \mu_k$, guides both full-task sampling and local exploration.

$$w_i = 1 + \rho \left[\max_{k \in \mathcal{K}_i} (1 - \mu_k) \right]^\eta, \qquad P_{\mathrm{task}}(i) = \frac{w_i}{\sum_j w_j}, \tag{5}$$

where $\mathcal{K}_i$ contains the subtasks of full task i, $\eta > 0$ controls sensitivity to the competence gap, and $\rho \geq 0$ limits sampling redistribution. The baseline weight of 1 preserves coverage of mastered tasks to monitor forgetting.

Within each trajectory, mastery-dependent entropy regularization adjusts local exploration:

$$\mathcal{L}_{\mathrm{explore}} = -\alpha_H \mathbb{E}_t \left[f_{\exp}\left(1 - \mu_{z_t}^{(n)}\right) \mathcal{H}_{\theta,t} \right], \tag{6}$$

where α_H is the base entropy coefficient, $\mathcal{H}_{\theta,t}$ denotes policy entropy, and $f_{\exp}$ is a bounded exploration weight for the active subtask z_t. The exploration strength is thus increased for poorly mastered subtasks and reduced as subtask mastery improves.

3.4 Progress–Completion Critic

To identify atomic-task boundaries b_t, we introduce a lightweight Progress–Completion Critic (PCC), denoted by C_ω. PCC estimates within-task progress and completion from the recent L observation–action pairs, providing semantic signals for SAPO and BATE and support the construction of progress-based shaping rewards (PBSR).

Conditioned on the current atomic task, PCC evaluates its status after executing a_t and outputs $B-1$ ordinal progress logits $u_{t,j}^{\mathrm{prog}}$ and a completion logit u_t^{comp}, where B is the number of progress levels.

We decode progress as

$$\hat{p}_t = \frac{1}{B-1} \sum_{j=1}^{B-1} \mathbf{1}\left[\sigma\left(u_{t,j}^{\text{prog}}\right) > \tau_{\text{ord}}\right], \qquad \hat{q}_t = \sigma(u_t^{\text{comp}}), \tag{7}$$

where σ denotes the sigmoid function and $\tau_{\text{ord}} = 0.5$ is the ordinal decoding threshold. An atomic task is considered complete when $\hat{q}_t \geq \tau_s$ and $\hat{p}_t \geq \tau_p$ for N consecutive decision steps. Upon confirming a non-terminal boundary, we advance the atomic-task index as $z_{t+1} = z_t + 1$ and reset the tracked within-task progress while retaining the physical observation context. PCC is trained offline on demonstration data using a cumulative ordinal loss together with a class-balanced completion loss:

$$\mathcal{L}_C = \lambda_p \mathcal{L}_{\text{ord}} + \lambda_c \mathcal{L}_{\text{comp}}, \tag{8}$$

where $\mathcal{L}_{\text{ord}}$ applies binary cross-entropy to cumulative progress targets, $\mathcal{L}_{\text{comp}}$ is a class-balanced binary cross-entropy loss for completion prediction. The coefficients λ_p and λ_c control the relative contributions of the two objectives. The progress estimates can optionally be used to construct progress-based shaping rewards (PBSR), yielding the training reward

$$r_t = r_t^{\text{env}} + \alpha_n r_t^{\text{PBSR}}, \tag{9}$$

where r_t^{env} is the sparse task reward, r_t^{PBSR} is the progress-based shaping reward, whose exact formulation and treatment of subtask boundaries are provided in Appendix A.2, and α_n controls its contribution at training iteration n.

3.5 Real2Sim2Real Pipeline for Reinforcement Learning

We develop an automated Real2Sim2Real reinforcement learning pipeline that unifies interactive environment construction, policy optimization in simulation, and deployment and evaluation on physical robots. Given a single multimodal real-world demonstration τ_{real}, metadata m specifying the robot model and supplementary object observations $\mathcal{D}_{\text{aux}}$ where needed, the pipeline constructs a simulation environment aligned with the physical system, a shared observation–action interface specification, and a set of reproducible randomized initial layouts:

$$(\mathcal{E}_{\text{sim}}, \mathcal{C}, \mathcal{S}_0) = \mathcal{B}(\tau_{\text{real}}, m, \mathcal{D}_{\text{aux}}), \tag{10}$$

where $\mathcal{E}_{\text{sim}}$ is the reconstructed interactive simulation environment, $\mathcal{C}$ specifies the shared observation–action interface, $\mathcal{S}_0$ is a set of reproducible randomized initial layouts, and $\mathcal{B}$ denotes the pipeline workflow. The generated environment is then integrated into the RLinf (Yu et al., 2025) distributed training framework, where SAPO and targeted exploration optimize the initial policy to obtain $\theta_{\text{RL}} = \text{Optimize}(\theta_0; \mathcal{E}_{\text{sim}}, \mathcal{C}, \mathcal{S}_0)$. Following pre-deployment validation, the optimized policy is deployed on the physical robot through the same interface specification $\mathcal{C}$ for task evaluation. Further details are provided in the Appendix A.6.

4 Experiment

4.1 Experiments Setup

Simulation Benchmarks. We evaluate AtomicRL on two robotic manipulation benchmarks: LIBERO Liu et al. (2023) and CALVIN Mees et al. (2022). We use the LIBERO-LONG suite(10 long-horizon tasks), and for CALVIN, we follow the ABC-D protocol, training on environments A, B, and C and evaluating on the held-out environment D. All experiments are implemented within the RLinf training framework Yu et al. (2025). On LIBERO-LONG, we compare AtomicRL against PPO with two policy backbones, $\pi_{0.5}$ Intelligence et al. (2025b) and OpenVLA-OFT Kim et al. (2025); on CALVIN, we conduct additional evaluations with $\pi_{0.5}$. Each training run consists of 200 optimization steps. Further implementation details are provided in the Appendix A.4.

Real World Robot. We conduct real-world experiments using an AgiBot G1 robot across three long-horizon tabletop manipulation tasks, each consisting of 4 to 6 atomic subtasks. For each task, we collect 100 expert demonstrations for SFT. The three tasks are:

- **Bread Placement(BP):** Place all pieces of bread on the table onto the white plate;

Table 1: **Evaluation on LIBERO-LONG (%).** Δ_{PPO} denotes the improvement over PPO.

Backbone	Method	SR (%) ↑	Δ_{PPO} (pp)
$\pi_{0.5}$	SFT	89.0	–
	+ PPO	95.4	–
	+ AtomicRL (Ours)	**99.0**	**+3.6**
OpenVLA-OFT	SFT	86.5	–
	+ PPO	94.8	–
	+ AtomicRL (Ours)	**98.8**	**+4.0**

Table 2: **Long-horizon Robotic Manipulation Evaluation on CALVIN Benchmark (%).**

Method	Task	Tasks Completed in a Row					Avg. Len ↑
		1	2	3	4	5	
$\pi_{0.5}$ SFT	ABC→D	92.8	83.5	73.2	61.5	45.3	3.56
+ PPO	ABC→D	99.2	95.7	89.8	81.2	70.6	4.36
+ AtomicRL (Ours)	ABC→D	**99.4**	**97.0**	**92.6**	**87.3**	**80.1**	**4.56**

- **Waste Disposal(WD):** Place the tissue and the car into the trash bin;
- **Tabletop Clean(TC):** Place the remote control into the storage box and the snacks into the box.

Evaluation metrics. For LIBERO, we report the mean success rate across all tasks, with 50 trials per task. For CALVIN, we report the success rates for completing one to five consecutive tasks and the average completed sequence length over 1,000 evaluation trials. These metrics measure cumulative sequence completion rather than independent subtask performance. For real-robot evaluation, each task is tested over 25 trials per method, and we report the mean success rate across the three tasks.

4.2 Results on Simulation

Results on LIBERO. As shown in Tab. 1, AtomicRL achieves success rates of 98.8% with OpenVLA-OFT and 99.0% with $\pi_{0.5}$, exceeding their corresponding PPO baselines by 4.0% and 3.6%. The consistent gains across both backbones support the applicability of AtomicRL to different VLA architectures. Both policies reach near-perfect success rates after 200 optimization iterations, demonstrating further improvements even when PPO already performs well. This efficiency can be attributed to AtomicRL's effective use of training data: SAPO preserves useful learning signals from completed subtasks in failed trajectories, while BATE concentrates exploration on bottleneck atomic subtasks. Together, these mechanisms enable faster convergence and higher final performance.

Results on CALVIN. As shown in Tab. 2, AtomicRL improves the average completed sequence length from 3.56 with SFT to 4.56, compared with 4.36 for PPO. The success rate for completing all five tasks increases from 70.6% with PPO to 80.1%, a gain of 9.5%. The larger gains on longer sequences indicate more reliable execution across the full task chain, demonstrating the effectiveness of AtomicRL for long-horizon reinforcement learning.

4.3 Results on Real-world Robot

We conduct online reinforcement learning in the constructed real-to-sim environments and evaluate both the policy improvement after RL and its sim-to-real transfer performance.

Performance on real-world robot. As shown in Tab. 3, AtomicRL improves over SFT by 21.3% on average across three long-horizon tasks and over PPO by 8.0%. As shown in Fig. 4, SAPO preserves useful learning signals from successfully completed subtasks in unsuccessful trajectories, enabling more appropriate credit assignment for suboptimal trajectories. Consequently, AtomicRL

Table 3: **Real-world robot results(%).**

Method	BP	WD	TC	Avg. SR ↑
$\pi_{0.5}$ SFT	60	52	32	48.0
+ PPO	68	64	52	61.3
+ AtomicRL	**76**	**72**	**60**	**69.3**

improves faster in early training, especially on the longer-horizon Bread Placement task. These results demonstrate the effectiveness of AtomicRL for long-horizon manipulation.

Simulation–reality correspondence. Fig. 3 compares the success rates between simulation and the real robot across nine combinations of three tasks and three methods. The aggregated Pearson correlation coefficient is $r = 0.909$, indicating a strong positive correlation across the evaluated settings. As shown in Fig. 5, the SFT policy fails after completing only part of the task, whereas the AtomicRL-optimized policy successfully completes the illustrated sequence in both simulation and the real world. The qualitative consistency in observations and execution outcomes between the two environments indicates that the reconstructed environment preserves the key visual and interaction structures relevant to the task, supporting effective policy optimization. Combined with the Real2Sim2Real deployment pipeline, AtomicRL shifts the otherwise costly and operationally complex online RL process from physical robots to simulation environments that support fast and parallel interactions. This reduces real-world interaction and reset costs while providing a practical approach for optimizing and deploying long-horizon manipulation policies.

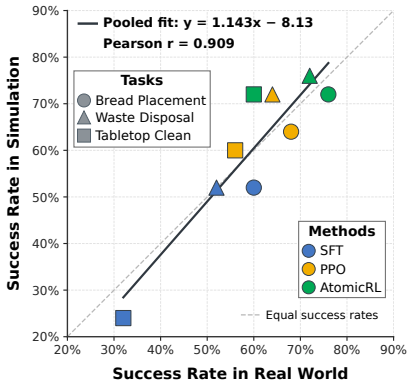
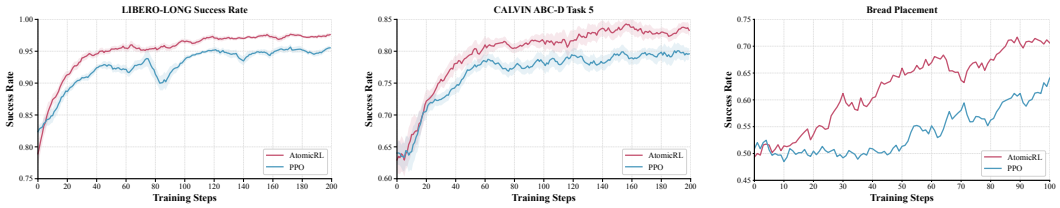


Figure 3: **Sim2Real Correlation.**



(a) LIBERO-LONG. (b) CALVIN ABC-D. (c) Bread placement.

Figure 4: Training curves on LIBERO-LONG and CALVIN ABC-D and Real-World(BP).

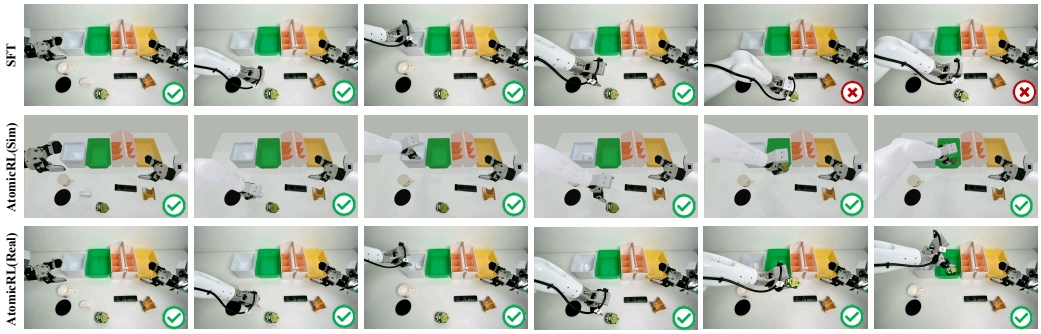


Figure 5: Demonstrations show the execution process of SFT and AtomicRL.

4.4 Ablation Study

We conduct policy-learning ablations on LIBERO-LONG using a weaker SFT checkpoint trained with fewer demonstrations than those used in the main evaluations. This few-shot initialization allows us to examine the contribution of each component under a less proficient initial policy.

Reward shaping and SAPO. To investigate the effect of reward granularity, we use the PCC module to construct process-level dense rewards and subtask-level rewards, and compare them under both PPO and SAPO. As shown in Tab. 4 (a), denser rewards do not always yield better performance: under SAPO, relatively sparse subtask-level rewards outperform process-level dense rewards. When the policy success rate is low, dense rewards provide stronger exploration guidance and accelerate early learning. As the policy improves, however, their benefits diminish, while persistent auxiliary

rewards may dilute the terminal success signal and shift optimization toward local gains. Motivated by this observation, we use linear annealing to gradually reduce the PBSR weight, providing stronger dense-reward guidance early in training and emphasizing the sparse terminal reward later. This balances early exploration with late-stage task alignment.

Table 4: **Ablations on LIBERO-LONG.** (a) Reward shaping with and without SAPO. (b) Component contributions. All results are success rates (%).

(a) Reward shaping and SAPO

Reward	PPO	SAPO
0/1	78.2	87.8
SubR	80.2	89.4
PBSR	81.8	89.0
Ours	–	**89.6**

(b) Full-method ablation

SAPO	PBSR	BATE	SR (%)
×	×	×	78.2
✓	×	×	87.8
✓	✓	×	89.6
✓	✓	✓	**91.2**

Component analysis. Table 4 (b) presents the incremental effects of successively introducing SAPO, annealed PBSR, and BATE. Starting from a PPO success rate of 78.2%, adding SAPO increases the success rate to 87.8% (+9.6 pp); introducing annealed PBSR further improves it to 89.6% (+1.8 pp); and finally incorporating BATE raises it to 91.2% (+1.6 pp), yielding a cumulative improvement of 13.0 percentage points over PPO. SAPO provides the largest performance gain, while annealed reward shaping and bottleneck-adaptive exploration further improve the policy. Combined with the results in Table 4 (a), where SAPO consistently improves performance across different reward settings, these findings suggest that its gains do not simply arise from denser reward feedback. Together, these results support the complementarity of the proposed components and their ability to further unlock the potential of RL for learning long-horizon tasks.

PCC prediction quality and computational cost. We train PCC on 150 trajectories containing both successful and failed executions and evaluate it on a test set of 500 rollout trajectories. The reference annotations for training and evaluation are automatically generated by a VLM and subsequently refined through manual correction. As shown in Tab. 5, PCC achieves 95.92% precision and 89.20% recall, with an empirical inference latency of 42.4 ms. These results show that PCC can estimate subtask progress and completion status with high precision while maintaining low inference latency. As a core component of AtomicRL, PCC converts the semantic task structure in trajectories into progress and boundary information for policy optimization, supporting SAPO in regulating cross-subtask credit propagation and BATE in estimating subtask mastery. Its prediction accuracy and low inference overhead demonstrate that this lightweight design provides a practical perception and evaluation foundation for RL based on atomic task structure.

Table 5: **PCC performance and efficiency.**

Category	Metric	Value
Prediction quality	Accuracy	95.92%
	False detection rate	4.08%
	Recall	89.20%
Efficiency	Trainable Parameters	28.8 M
	Inference latency	42.4 ms

5 Conclusion

In this work, we introduce AtomicRL, a VLA RL post-training framework for long-horizon robotic manipulation centered on atomic task structure. To address temporal credit interference and imbalanced atomic subtask mastery, AtomicRL uses PCC to estimate task progress and completion boundaries, SAPO to regulate the cross-boundary propagation of TD residuals based on subtask execution outcomes, and BATE to shift task sampling and local exploration toward insufficiently mastered bottleneck subtasks. We further develop a Real2Sim2Real pipeline that connects real-task reconstruction, simulation-based RL, and real-robot deployment, enabling policy improvement in simulation environments with fast parallel interactions and reproducible resets. We validate AtomicRL in both simulation and real-world settings, and the results show that it converges faster to a higher performance level, demonstrating its potential for RL-based long-horizon robotic manipulation.

AI USE STATEMENT

This study used generative AI tools to assist with literature search, manuscript language polishing, and simulation environment construction (see Appendix A.6). All relevant procedures and data were manually reviewed and verified by the authors. All content generated with AI assistance was reviewed and validated by the authors. We take responsibility for the final content of this work, including all text, claims, and artifacts.

REPRODUCIBILITY STATEMENT

We provide details of our method and experimental protocols in the main paper and appendices to enable reproduction of our method and experiments.

REFERENCES

Lucas Beyer, Andreas Steiner, André Susano Pinto, Alexander Kolesnikov, Xiao Wang, Daniel Salz, Maxim Neumann, Ibrahim Alabdulmohsin, Michael Tschannen, Emanuele Bugliarello, Thomas Unterthiner, Daniel Keysers, Skanda Koppula, Fangyu Liu, Adam Grycner, Alexey Gritsenko, Neil Houlsby, Manoj Kumar, Keran Rong, Julian Eisenschlos, Rishabh Kabra, Matthias Bauer, Matko Bošnjak, Xi Chen, Matthias Minderer, Paul Voigtlaender, Ioana Bica, Ivana Balazevic, Joan Puigcerver, Pinelopi Papalampidi, Olivier Henaff, Xi Xiong, Radu Soricut, Jeremiah Harmsen, and Xiaohua Zhai. Paligemma: A versatile 3b vlm for transfer, 2024. URL https://arxiv.org/abs/2407.07726.

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, Xiang Li, Quanlu Zhang, Zhaofei Yu, Guoliang Fan, Tiejun Huang, Yu Wang, and Chao Yu. π_{RL}: Online rl fine-tuning for flow-based vision-language-action models, 2026a. URL https://arxiv.org/abs/2510.25889.

Qianzhong Chen, Justin Yu, Mac Schwager, Pieter Abbeel, Yide Shentu, and Philipp Wu. Sarm: Stage-aware reward modeling for long horizon robot manipulation, 2026b. URL https://arxiv.org/abs/2509.25358.

Zixuan Chen, Jing Huo, Yangtao Chen, and Yang Gao. Robohorizon: An llm-assisted multi-view world model for long-horizon robotic manipulation, 2025. URL https://arxiv.org/abs/2501.06605.

Murtaza Dalal, Tarun Chiruvolu, Devendra Chaplot, and Ruslan Salakhutdinov. Plan-seq-learn: Language model guided rl for solving long horizon robotics tasks, 2024. URL https://arxiv.org/abs/2405.01534.

Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, Jiasen Lu, Taira Anderson, Erin Bransom, Kiana Ehsani, Huong Ngo, YenSung Chen, Ajay Patel, Mark Yatskar, Chris Callison-Burch, Andrew Head, Rose Hendrix, Favyen Bastani, Eli VanderBilt, Nathan Lambert, Yvonne Chou, Arnavi Chheda, Jenna Sparks, Sam Skjonsberg, Michael Schmitz, Aaron Sarnat, Byron Bischoff, Pete Walsh, Chris Newell, Piper Wolters, Tanmay Gupta, Kuo-Hao Zeng, Jon Borchardt, Dirk Groeneveld, Crystal Nam, Sophie Lebrecht, Caitlin Wittlif, Carissa Schoenick, Oscar Michel, Ranjay Krishna, Luca Weihs, Noah A. Smith, Hannaneh Hajishirzi, Ross Girshick, Ali Farhadi, and Aniruddha Kembhavi. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models, 2024. URL https://arxiv.org/abs/2409.17146.

Dongchi Huang, Zhirui Fang, Tianle Zhang, Yihang Li, Lin Zhao, and Chunhe Xia. Co-rft: Efficient fine-tuning of vision-language-action models through chunked offline reinforcement learning, 2025. URL https://arxiv.org/abs/2508.02219.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, Danny Driess, Michael Equi, Adnan Esmail, Yunhao Fang, Chelsea Finn, Catherine Glossop, Thomas Godden, Ivan Goryachev, Lachy Groom, Hunter Hancock, Karol Hausman, Gashon Hussein, Brian Ichter, Szymon Jakubczak, Rowan Jen, Tim Jones, Ben Katz, Liyiming Ke, Chandra Kuchi, Marinda Lamb, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Yao Lu, Vishnu Mano, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Charvi Sharma, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, Will Stoeckle, Alex Swerdlow, James Tanner, Marcel Torne, Quan Vuong, Anna Walling, Haohuan Wang, Blake Williams, Sukwon Yoo, Lili Yu, Ury Zhilinsky, and Zhiyuan Zhou. $\pi^*_{0.6}$: a vla that learns from experience, 2025a. URL https://arxiv.org/abs/2511.14759.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman,

Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025b. URL https://arxiv.org/abs/2504.16054.

Guangqi Jiang, Haoran Chang, Ri-Zhao Qiu, Yutong Liang, Mazeyu Ji, Jiyue Zhu, Zhao Dong, Xueyan Zou, and Xiaolong Wang. Gsworld: Closed-loop photo-realistic simulation suite for robotic manipulation, 2025. URL https://arxiv.org/abs/2510.20813.

Zhennan Jiang, Kai Liu, Yuxin Qin, Shuai Tian, Yupeng Zheng, Mingcai Zhou, Chao Yu, Haoran Li, and Dongbin Zhao. World4rl: Diffusion world models for policy refinement with reinforcement learning for robotic manipulation, 2026a. URL https://arxiv.org/abs/2509.19080.

Zhennan Jiang, Shangqing Zhou, Yutong Jiang, Zefang Huang, Mingjie Wei, Yuhui Chen, Tianxing Zhou, Zhen Guo, Hao Lin, Quanlu Zhang, Yu Wang, Haoran Li, Chao Yu, and Dongbin Zhao. Wovr: World models as reliable simulators for post-training vla policies with rl, 2026b. URL https://arxiv.org/abs/2602.13977.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success, 2025. URL https://arxiv.org/abs/2502.19645.

Chenhao Li, Andreas Krause, and Marco Hutter. Robotic world model: A neural network simulator for robust policy optimization in robotics, 2025a. URL https://arxiv.org/abs/2501.10100.

Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Yang Zhaohui, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, et al. Simplevla-rl: Scaling vla training via reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pp. 125846–125869, 2026.

Wenxuan Li, Hang Zhao, Zhiyuan Yu, Yu Du, Qin Zou, Ruizhen Hu, and Kai Xu. Pin-wm: Learning physics-informed world models for non-prehensile manipulation, 2025b. URL https://arxiv.org/abs/2504.16693.

Yunfei Li, Xiao Ma, Jiafeng Xu, Yu Cui, Zhongren Cui, Zhigang Han, Liqun Huang, Tao Kong, Yuxiao Liu, Hao Niu, Wanli Peng, Jingchao Qiao, Zeyu Ren, Haixin Shi, Zhi Su, Jiawen Tian, Yuyang Xiao, Shenyu Zhang, Liwei Zheng, Hang Li, and Yonghui Wu. Gr-rl: Going dexterous and precise for long-horizon robotic manipulation, 2025c. URL https://arxiv.org/abs/2512.01801.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning, 2023. URL https://arxiv.org/abs/2306.03310.

Jie Liu, Gongye Liu, Jiajun Liang, Yangguang Li, Jiaheng Liu, Xintao Wang, Pengfei Wan, Di Zhang, and Wanli Ouyang. Flow-grpo: Training flow matching models via online rl, 2025a. URL https://arxiv.org/abs/2505.05470.

Jijia Liu, Feng Gao, Bingwen Wei, Xinlei Chen, Qingmin Liao, Yi Wu, Chao Yu, and Yu Wang. What can rl bring to vla generalization? an empirical study, 2026. URL https://arxiv.org/abs/2505.19789.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation, 2025b. URL https://arxiv.org/abs/2410.07864.

Guanxing Lu, Wenkai Guo, Chubin Zhang, Yuheng Zhou, Haonan Jiang, Zifeng Gao, Yansong Tang, and Ziwei Wang. Vla-rl: Towards masterful and general robotic manipulation with scalable reinforcement learning, 2025. URL https://arxiv.org/abs/2505.18719.

Jianlan Luo, Zheyuan Hu, Charles Xu, You Liang Tan, Jacob Berg, Archit Sharma, Stefan Schaal, Chelsea Finn, Abhishek Gupta, and Sergey Levine. Serl: A software suite for sample-efficient robotic reinforcement learning, 2025. URL https://arxiv.org/abs/2401.16013.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. Calvin: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks, 2022. URL https://arxiv.org/abs/2112.03227.

Yiwen Pang, Bo Zhou, Changjin Li, Xuanhao Wang, Shengxiang Xu, Deng-Bao Wang, Peng Cheng, Shimin Di, Jingkuan Song, and Min-Ling Zhang. Atombridge: Agentic vla inference plugin for long-horizon tasks in scientific experiments, 2026. URL https://arxiv.org/abs/2602.09430.

Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning, 2025. URL https://arxiv.org/abs/2502.02538.

Zhian Su, Weijie Kong, Haonan Dong, and Huixu Dong. Ig-rft: An interaction-guided rl framework for vla models in long-horizon robotic manipulation, 2026. URL https://arxiv.org/abs/2602.20715.

Huajie Tan, Sixiang Chen, Yijie Xu, Zixiao Wang, Yuheng Ji, Cheng Chi, Yaoxu Lyu, Zhongxia Zhao, Xiansheng Chen, Peterson Co, Shaoxuan Xie, Guocai Yao, Pengwei Wang, Zhongyuan Wang, and Shanghang Zhang. Robo-dopamine: General process reward modeling for high-precision robotic manipulation, 2025. URL https://arxiv.org/abs/2512.23703.

GigaWorld Team, Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Haoyun Li, Jiagang Zhu, Kerui Li, Mengyuan Xu, Qiuping Deng, Siting Wang, Wenkang Qin, Xinze Chen, Xiaofeng Wang, Yankai Wang, Yu Cao, Yifan Chang, Yuan Xu, Yun Ye, Yang Wang, Yukun Zhou, Zhengyuan Zhang, Zhehao Dong, and Zheng Zhu. Gigaworld-0: World models as data engine to empower embodied ai, 2025. URL https://arxiv.org/abs/2511.19861.

Marcel Torne, Anthony Simeonov, Zechu Li, April Chan, Tao Chen, Abhishek Gupta, and Pulkit Agrawal. Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation, 2024. URL https://arxiv.org/abs/2403.03949.

Maggie Wang, Stephen Tian, Aiden Swann, Ola Shorinwa, Jiajun Wu, and Mac Schwager. Phys2real: Fusing vlm priors with interactive online adaptation for uncertainty-aware sim-to-real manipulation, 2026. URL https://arxiv.org/abs/2510.11689.

Charles Xu, Jost Tobias Springenberg, Michael Equi, Ali Amin, Adnan Esmail, Sergey Levine, and Liyiming Ke. Rl token: Bootstrapping online rl with vision-language-action models, 2026. URL https://arxiv.org/abs/2604.23073.

Feng Xu, Guangyao Zhai, Xin Kong, Tingzhong Fu, Daniel F. N. Gordon, Xueli An, and Benjamin Busam. Stare-vla: Progressive stage-aware reinforcement for fine-tuning vision-language-action models, 2025. URL https://arxiv.org/abs/2512.05107.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report, 2025. URL https://arxiv.org/abs/2505.09388.

Jiarui Yang, Bin Zhu, Jingjing Chen, and Yu-Gang Jiang. Actor-critic for continuous action chunks: A reinforcement learning framework for long-horizon robotic manipulation with sparse reward, 2026a. URL https://arxiv.org/abs/2508.11143.

Jiazhi Yang, Kunyang Lin, Jinwei Li, Wencong Zhang, Tianwei Lin, Longyan Wu, Zhizhong Su, Hao Zhao, Ya-Qin Zhang, Li Chen, Ping Luo, Xiangyu Yue, and Hongyang Li. Rise: Self-improving robot policy with compositional world model, 2026b. URL https://arxiv.org/abs/2602.11075.

Tenny Yin, Zhiting Mei, Zhonghe Zheng, Miyu Yamane, David Wang, Jade Sceats, Samuel M. Bateman, Lihan Zha, Apurva Badithela, Ola Shorinwa, and Anirudha Majumdar. Playworld: Learning robot world models from autonomous play, 2026. URL https://arxiv.org/abs/2603.09030.

Chao Yu, Yuanqing Wang, Zhen Guo, Hao Lin, Si Xu, Hongzhi Zang, Quanlu Zhang, Yongji Wu, Chunyang Zhu, Junhao Hu, Zixiao Huang, Mingjie Wei, Yuqing Xie, Ke Yang, Bo Dai, Zhexuan Xu, Jiakun Du, Xiangyuan Wang, Xu Fu, Letong Shi, Zhihao Liu, Kang Chen, Weilin Liu, Gang Liu, Boxun Li, Jianlei Yang, Zhi Yang, Guohao Dai, and Yu Wang. Rlinf: Flexible and efficient large-scale reinforcement learning via macro-to-micro flow transformation, 2025. URL https://arxiv.org/abs/2509.15965.

Shaopeng Zhai, Qi Zhang, Tianyi Zhang, Fuxian Huang, Haoran Zhang, Ming Zhou, Shengzhe Zhang, Litao Liu, Sixu Lin, and Jiangmiao Pang. A vision-language-action-critic model for robotic real-world reinforcement learning, 2025. URL https://arxiv.org/abs/2509.15937.

Hongyin Zhang, Zifeng Zhuang, Han Zhao, Pengxiang Ding, Hongchao Lu, and Donglin Wang. Reinbot: Amplifying robot visual-language manipulation with reinforcement learning, 2025a. URL https://arxiv.org/abs/2505.07395.

Likui Zhang, Tao Tang, Zhihao Zhan, Xiuwei Chen, Zisheng Chen, Jianhua Han, Jiangtong Zhu, Pei Xu, Hang Xu, Hefeng Wu, Liang Lin, and Xiaodan Liang. Atomicvla: Unlocking the potential of atomic skill learning in robots, 2026a. URL https://arxiv.org/abs/2603.07648.

Yixian Zhang, Huanming Zhang, Feng Gao, Xiao Li, Zhihao Liu, Yi Nie, Chunyang Zhu, Jiaxing Qiu, Yuchen Yan, Jiyuan Liu, Wenhao Tang, Jiaji Rao, Zhengru Fang, Changxu Wei, Yu Wang, Wenbo Ding, and Chao Yu. Harness vla: Steering frozen vlas into reliable manipulation primitives via memory-guided agents, 2026b. URL https://arxiv.org/abs/2607.08448.

Zhilong Zhang, Haoxiang Ren, Yihao Sun, Yifei Sheng, Haonan Wang, Haoxin Lin, Zhichao Wu, Pierre-Luc Bacon, and Yang Yu. Towards practical world model-based reinforcement learning for vision-language-action models, 2026c. URL https://arxiv.org/abs/2603.20607.

Zijian Zhang, Kaiyuan Zheng, Zhaorun Chen, Joel Jang, Yi Li, Siwei Han, Chaoqi Wang, Mingyu Ding, Dieter Fox, and Huaxiu Yao. Grape: Generalizing robot policy via preference alignment, 2025b. URL https://arxiv.org/abs/2411.19309.

A Appendix/supplemental material

A.1 Future Work and Limitations

AtomicRL improves temporal credit assignment across atomic tasks through SAPO and concentrates exploration resources on insufficiently mastered bottleneck subtasks through BATE. Although the experimental results demonstrate its effectiveness in long-horizon robotic manipulation, the current approach still has several limitations.

- AtomicRL improves the utilization of interaction data through more effective credit assignment and exploration-resource allocation, but does not fundamentally address the overall efficiency of online RL training. Even with open-source RL frameworks such as RLinf (Yu et al., 2025) and SERL (Luo et al., 2025), online RL still requires substantial environment interaction and training time, particularly in real-robot settings where interaction and exploration costs remain high. Reducing the interaction required for effective policy learning, improving online RL training efficiency, and lowering the cost of real-robot training therefore remain important directions for future work.
- Our experiments primarily focus on long-horizon tabletop manipulation tasks and initialize RL from SFT policies. The effectiveness of AtomicRL on more open and dynamic tasks requiring stronger embodiment generalization remains to be further evaluated. Recent Agentic RL approaches, such as HarnessVLA (Zhang et al., 2026b) and AtomBridge (Pang et al., 2026) have shown potential for open-world tasks and embodiment generalization, providing an alternative direction for long-horizon robotic manipulation. A promising direction is to transfer AtomicRL's use and optimization of atomic task structure to Agentic RL frameworks, enabling robots to dynamically form and adapt atomic task structures in more open environments and potentially improving the generalization and training efficiency of long-horizon manipulation.

A.2 Progress-Based Shaping Rewards

We optionally supplement the sparse task reward with progress-based shaping rewards (PBSR), using the atomic-task index and discrete progress estimates from PCC.

Progress potential. Let K denote the number of atomic tasks and $c_t = (B-1)\hat{p}_t \in \{0, \dots, B-1\}$ denote the decoded progress level. After completion handling, let z_{t+1} be the active atomic-task index and $\widetilde{c}_t$ the tracked progress level. We set $\widetilde{c}_t = c_t$ unless atomic-task completion is confirmed, in which case it is reset to zero. The task index advances only if another atomic task remains. The updated progress potential is

$$\Phi_{t+1} = \frac{z_{t+1} - 1 + \widetilde{c}_t/B}{K}. \tag{11}$$

The potential combines the atomic-task index with within-task progress, with the latter scaled by $1/B$.

Shaping reward. For each valid non-terminal PCC update, we compute

$$r_t^{\mathrm{PBSR}} = \gamma\Phi_{t+1} - \Phi_t, \tag{12}$$

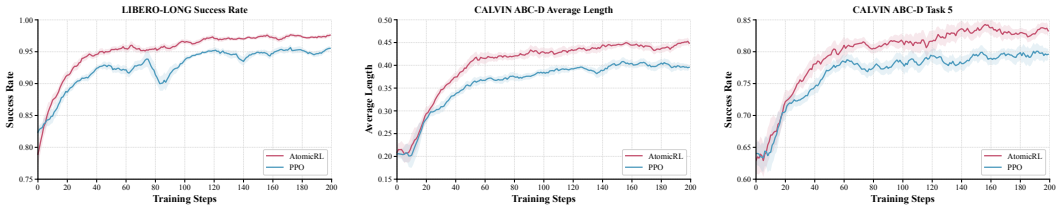
where γ is the discount factor and Φ_t is the previously stored potential. The stored potential is initialized to zero when the task tracker is reset and replaced by Φ_{t+1} after each valid update. The shaping contribution is zero on terminal transitions, including environment timeouts, and whenever no valid update is available. The training reward is

$$r_t = r_t^{\mathrm{env}} + \alpha_n r_t^{\mathrm{PBSR}}, \tag{13}$$

where r_t^{env} is the sparse task reward and α_n controls the shaping contribution at training iteration n. The shaping reward can be positive or negative and is not clipped.

Shaping schedule. We linearly anneal the shaping coefficient:

$$\alpha_n = \alpha_0 \left[1 - (1-\rho)\,\mathrm{clip}\left(\frac{n - n_s}{n_e - n_s},\, 0,\, 1\right)\right], \tag{14}$$



(a) LIBERO-LONG success rate. (b) CALVIN average length. (c) CALVIN task-5 success rate.

Figure 6: Training curves on LIBERO-LONG and CALVIN ABC-D.

where α_0 is the initial coefficient, $n_s < n_e$ specify the annealing interval, $\rho \in [0, 1]$ is the final retained fraction, and $\text{clip}(x, 0, 1)$ clamps x to $[0, 1]$. Setting $\rho = 0$ removes PBSR after annealing.

PBSR modifies the reward used to compute TD residuals, whereas SAPO regulates their cross-boundary propagation during advantage estimation.

A.3 Additional Ablations

Sensitivity to the SAPO Gating Coefficient. We examine the effect of the SAPO gating coefficient κ on LIBERO-LONG under the same configuration as the component ablations. Here, κ controls trace propagation across boundaries preceding failed subtasks, while gates preceding successful subtasks remain fixed at one. Setting $\kappa = 1$ recovers standard GAE propagation, whereas $\kappa = 0$ blocks cross-boundary trace propagation from failed subtasks without removing the one-step value bootstrap. As shown in Table 6, the highest observed success rate is 87.8% at $\kappa = 0.3$, improving over the unattenuated baseline (78.2%) by 9.6 percentage points. Performance decreases to 85.0% and 81.2% as κ increases to 0.5 and 0.8, respectively. Hard trace blocking ($\kappa = 0$) also performs well, achieving 87.2%, only 0.6 percentage points below $\kappa = 0.3$. These results support the use of soft attenuation of cross-boundary trace propagation from failed subtasks in this setting.

Table 6: **Effect of the SAPO gating coefficient.**

κ	**Success Rate**
0.0	87.2%
0.3	**87.8%**
0.5	85.0%
0.8	81.2%
1.0	78.2%

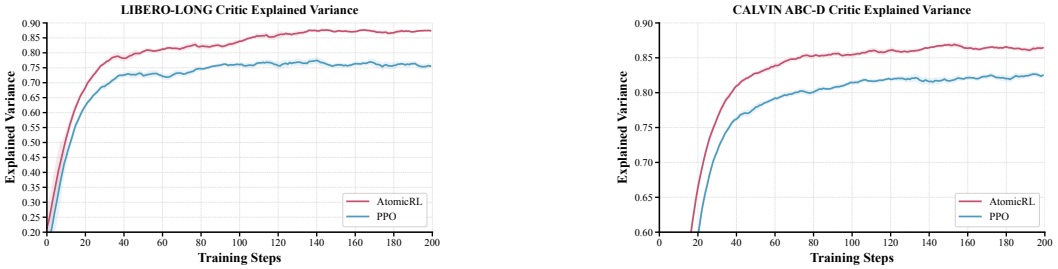
A.4 Training Details

Training Setup. For all LIBERO and real-world experiments, we train PCC offline on annotated demonstrations. PCC uses a frozen SigLIP visual encoder with approximately 400M parameters, initialized from the $\pi_{0.5}$ visual backbone, together with lightweight trainable temporal and prediction modules. For data annotation, we adapt the annotation pipeline of AtomicVLA (Zhang et al., 2026a) to generate fine-grained annotations for demonstration trajectories. We annotate 150 trajectories for LIBERO and 50 trajectories per real-world task, which are subsequently used for offline PCC training. The trained PCC is kept fixed during RL post-training.

All RL post-training experiments are implemented in the RLinf framework. Tab. 7 and Tab. 8 summarize the training hyperparameters for the simulation benchmarks and real-world tasks, respectively. All training is performed on eight NVIDIA H200 GPUs, while inference is conducted on a single NVIDIA GeForce RTX 5090 GPU.

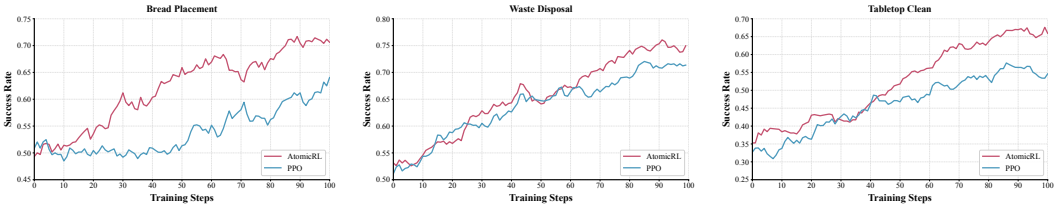
A.5 Additional Results

Detail Results on benchmarks. Fig. 6 and Fig. 7 show the training dynamics and value-function fitting of AtomicRL. On both LIBERO-LONG and CALVIN ABC-D, AtomicRL improves more rapidly during early training and consistently outperforms PPO throughout subsequent training. Meanwhile, the critic explained variance in the later stages of training is higher than PPO by approximately 12% on LIBERO and 5% on CALVIN, indicating that the value function explains a larger fraction of the



(a) LIBERO-LONG critic explained variance. (b) CALVIN critic explained variance.

Figure 7: Critic explained variance.



(a) BP success rate. (b) WD success rate. (c) TC success rate.

Figure 8: Training curves on Real-world robot.

variance in the training targets. These trends are consistent with the design of SAPO and BATE: SAPO attenuates the cross-boundary propagation of TD residuals from failed atomic subtasks to completed prefixes, improving temporal credit assignment and reducing interference from downstream failures with previously successful behaviors; BATE concentrates sampling and exploration resources on insufficiently mastered bottleneck subtasks. The concurrent improvement in task performance and value-function fitting further supports the effectiveness of atomic-task-structured optimization for long-horizon policy learning.

Detail Results on real-world robot. As shown in Fig. 8, when online RL is performed in the Real2Sim-reconstructed environments, AtomicRL exhibits better training performance than PPO across all three real-world tasks. The advantage is particularly pronounced during early training on Bread Placement (BP), which consists of six atomic subtasks: PPO remains around 50% success over the first 40 training steps, whereas AtomicRL reaches 60% at around step 30 and continues to improve thereafter. This gap reflects the difficulty of learning long task chains under sparse feedback and is consistent with the design of AtomicRL, which attenuates cross-boundary credit interference from downstream failures and thereby makes better use of successful prefixes in failed trajectories. AtomicRL also achieves higher late-stage training success rates on Waste Disposal (WD) and Tabletop Clean (TC), indicating that its training advantage is not limited to a single task. Combined with the simulation–real-world performance correlation analysis in Fig. 3, these training curves provide an indirect reference for the trend of real-world policy improvement.

A.6 Details of automated Real2Sim2Real pipeline

As shown in Fig. 11, we develop an automated Real2Sim2Real workflow, with GPT-6 Astra assisting tool orchestration and code implementation. The workflow connects real-task reconstruction, simulation-based online RL in RLinf (Yu et al., 2025), and real-robot deployment, with the key steps and validation requirements summarized in the figure.

A.7 Additional Visualizations

We provide additional visualizations of policy execution in simulation and the real world. As shown in Fig. 9, SFT completes the initial actions but fails at a later stage, whereas AtomicRL successfully completes the illustrated sequence in both simulation and real-world deployment. Figure 10

presents representative real-robot rollouts for Bread Placement, Waste Disposal, and Tabletop Clean, illustrating how the policy executes successive atomic tasks to achieve the overall goal.

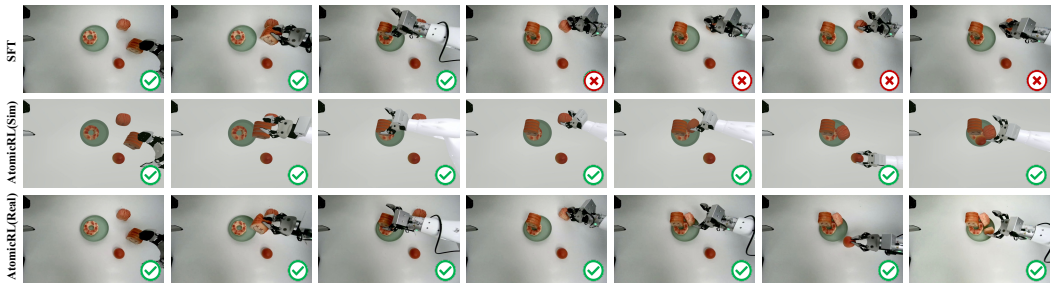


Figure 9: Demonstrations show the execution process of SFT and AtomicRL.

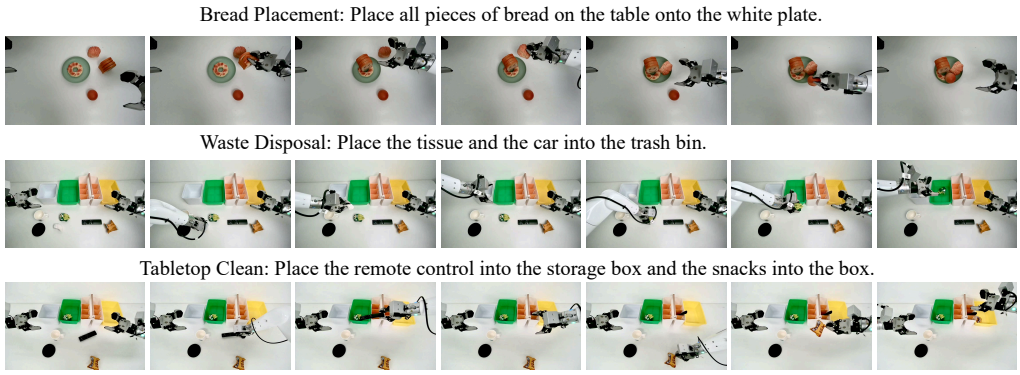


Figure 10: Demonstrations of real-world experiments.

Table 7: Verified training settings for the main LIBERO and CALVIN runs.

Parameter	LIBERO-$\pi_{0.5}$	LIBERO-OpenVLA-OFT	CALVIN-$\pi_{0.5}$
Optimization and rollout			
Action representation	End-effector delta	End-effector delta	End-effector delta
Training steps	200	200	200
Maximum episode length	480	512	480
Parallel environments / rollout epochs	64 / 8	64 / 8	64 / 8
Trajectories per iteration	512	512	512
Global / micro batch size	2048 / 64	512 / 8	2048 / 64
Policy / value learning rate	10^{-5} / 10^{-4}	8×10^{-6} / 10^{-4}	5×10^{-6} / 10^{-4}
Adam (β_1, β_2) / ϵ	$(0.9, 0.95)$ / 10^{-8}	$(0.9, 0.95)$ / 10^{-8}	$(0.9, 0.95)$ / 10^{-8}
Weight decay / gradient norm	0.01 / 1.0	0.01 / 1.0	0.01 / 1.0
Policy update epochs	4	1	4
γ / λ / PPO clip	0.99 / 0.95 / 0.20	0.99 / 0.95 / 0.20	0.99 / 0.95 / 0.20
Reward / log-probability granularity	Chunk / chunk	Chunk / token	Chunk / chunk
SAPO setting			
Boundary source	PCC	PCC	Env
PCC success threshold	0.50	0.50	–
Confirmation	1 chunk	1 chunk	–
Cross-boundary GAE carry	0.30	0.30	0.30
PBSR setting (Optional)			
Effective reward mode	PBSR	PBSR	Env
PBSR strength α	0.3	0.3	–
PBSR annealing stop step	50	50	–
BATE setting			
Base entropy coefficient	5×10^{-3}	10^{-4}	5×10^{-3}
Increase: start step / maximum gain	0 / 1.5	0 / 1.5	50 / 1.5
Increase: EMA threshold	0.05	0.05	0.20
Reduce: start step / minimum gain	150/0.70	150/0.70	150/0.50
Reduce: EMA threshold	0.02	0.02	0.05

Table 8: Simulation-based RL settings and real-robot evaluation protocol for real-world tasks.

Parameter	Setting
Robot platform and evaluation protocol	
Robot platform	AGIBOT G1
Number of tasks	3
Subtasks per task	4–6
Trials per task and method	25
Trial order	Randomised
Policy and environment	
Policy backbone	$\pi_{0.5}$
SFT checkpoint step	6,000
Action representation	End-effector delta
Action / execution horizon	30 / 30
Denoising steps	5
Control frequency	10 Hz
Initial layouts	16
Maximum episode length	480 / 600 / 480 (Tabletop / Bread / Waste)
Optimization and rollout	
Training steps	100
Parallel environments	32
Trajectories per iteration	32
Global / micro batch size	256 / 1
Initial policy / value learning rate	5×10^{-6} / 10^{-4}
Adam (β_1, β_2) / ϵ	$(0.9, 0.95)$ / 10^{-8}
Weight decay / gradient norm	0.01 / 1.0
Policy update epochs	1
γ / λ / PPO clip	0.99 / 0.95 / 0.20
Value clip / Huber δ	0.20 / 10.0
Advantage normalization	Enabled
Reward / log-probability granularity	Chunk / chunk
Terminal-action masking	Enabled
SubR setting (optional)	
SubR strength	0.3
SAPO settings	
Boundary source	PCC
PCC success threshold	0.50
Confirmation	1 chunk
Cross-boundary GAE carry	0.30

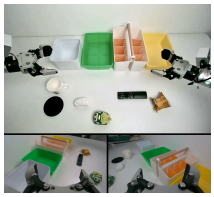
Automated Real2Sim2Real Pipeline

Role:

You are a robotics simulation and reinforcement learning engineer. Convert the supplied real-task data into an interactive simulation with aligned camera views and action-space semantics for online RL.

Input:

One LeRobot demonstration containing images, states, actions, and task text; object multiview observations and material information; an SFT checkpoint with its configuration and normalization.



Real demonstration: head and wrist views

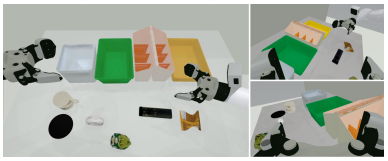


Object multiview observations

Workflow:

1. **Initialize the environment.** Build the robot in MuJoCo from its URDF, preserving joint structure, order, and limits. Use COLMAP/PyCOLMAP and Open3D/Trimesh to reconstruct task objects, with explicit shape priors where needed.
2. **Restore scene appearance.** Map foreground photos to surface UV atlases; adjust color, specularity, and shininess. Bind deformable textured surfaces to physical nodes. For complex static backgrounds, 3DGS is an optional rendering extension; collision geometry remains separate.
3. **Assign physical properties.** Rigid objects use CoACD/primitive proxies with mass, inertia, and friction. Soft objects use MuJoCo flex tetrahedra with mass/density, Young's modulus, Poisson's ratio, damping, and friction. Recoverable soft materials additionally use viscoelastic forces, relaxation times, and internal deformation states. Mark measured versus assumed parameters.
4. **Recover and diversify layouts.** Combine RGB-D/scale evidence, robot kinematics, and SciPy multiview contour/landmark fitting to align poses, support height, and cameras. Check overlays; sample seeded layouts within feasible regions and retain physically stable initial states.
5. **Validate the environment.** Replay recorded actions and run the SFT checkpoint to check action semantics, contact, compression/recovery, and numerical stability. Restore material memory when resetting viscoelastic objects.
6. **Expose the RL interface.** Provide observation, action, reward, termination, and reset APIs for parallel RLinf workers. Select PPO, GRPO, or SAPO according to the implemented algorithm adapter; preserve the deployment control interface.

Output: scene preview and required checks



Simulation: head and wrist views

✓ Camera-view / action-space alignment
✓ Explicit physical material parameters
✓ Rigid / deformable object interaction
✓ Parallel RL training interface

Figure 11: Automated Real2Sim2Real workflow with essential tool calls and a real-to-simulation scene example.