THINKING IN TRAJECTORIES: STRUCTURED EXPLORATION FOR VISION-LANGUAGE-ACTION REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Reinforcement-learning post-training has recently improved vision-language-action (VLA) models beyond supervised imitation, yet most existing methods explore by sampling low-level actions or action chunks. In a high-dimensional continuous action space, such perturbations mostly yield local control variations: changing the high-level intent of a long-horizon behavior requires coordinated deviations over many steps, which random action noise discovers only inefficiently. We propose **TCoT-RL**, which formulates trajectory reasoning as a *directly optimizable exploration policy* for VLA post-training: a trajectory policy autoregressively samples a variable-length sequence of end-effector waypoints representing a high-level motion plan, while a flow-based action policy generates continuous controls conditioned on the sampled trajectory. This hierarchical factorization enables structured exploration over alternative behavioral intents rather than only perturbing low-level actions. To directly optimize both levels of the hierarchy, TCoT-RL performs joint trajectory–action reinforcement learning with level-specific credit assignment. We introduce dense trajectory feedback based on task progress and stage-aware endpoint grounding, and optimize the trajectory and action policies with separate group-relative advantages and clipped GRPO objectives. This allows trajectory reasoning to evolve from a supervised intermediate representation into an active exploration policy through online interaction. Experiments show that TCoT-RL consistently improves over supervised trajectory-conditioned policies and matched action-only RL baselines. On LIBERO-Plus, TCoT-RL achieves a state-of-the-art success rate of 89.6%, outperforming the strongest competing RL baseline by 1.8 percentage points. Paired rollout analysis further shows that trajectory-level exploration expands end-effector state coverage and visitation entropy and produces greater behavioral diversity. These results demonstrate that explicitly optimizing the distribution over high-level motion hypotheses provides an effective structured exploration mechanism for VLA reinforcement learning.

1 INTRODUCTION

Vision-language-action (VLA) models have emerged as a promising paradigm for general-purpose robotic manipulation. By adapting large vision-language models to robot demonstrations, recent systems can follow open-vocabulary instructions, recognize diverse objects, and execute manipulation skills across heterogeneous tasks and environments (Brohan et al., 2023; Zitkovich et al., 2023; Black et al., 2025b; Pertsch et al., 2025; Bjorck et al., 2025; Bu et al., 2025; Li et al., 2024). Yet most VLA policies are still trained mainly through supervised imitation, so their behavior is constrained by the coverage and quality of demonstrations, leaving a central question: *how can a pretrained VLA policy continue discovering and improving behaviors through interaction after supervised training?*

Online reinforcement learning provides a natural solution by allowing a policy to explore new behaviors and optimize directly for task outcomes. Test-time selection, such as sampling many plans and keeping the best, is not a substitute: it needs a success verifier at deployment and repeated physical attempts, whereas RL amortizes the search into the policy and reinforces effective behaviors that the demonstrations under-weight. Recent RL post-training methods have improved manipulation performance, robustness, and generalization (Li et al., 2026a; Liu et al., 2025; Guo et al., 2025;

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.

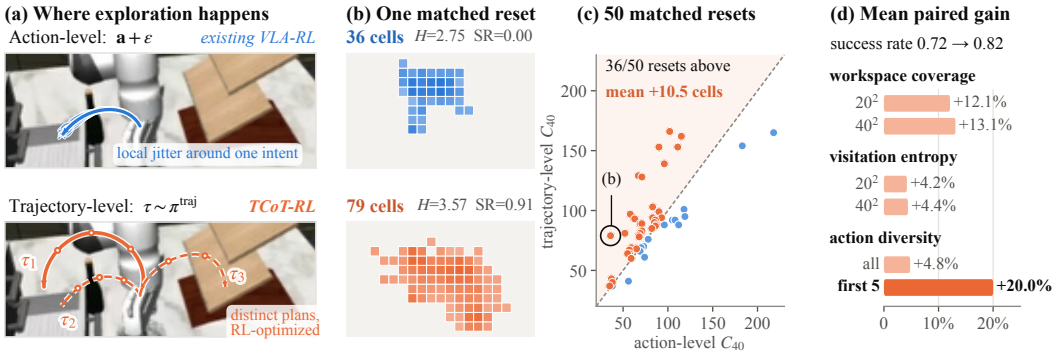

Figure 1: Action- vs. trajectory-level exploration. (a) Illustration on a LIBERO scene (overlays are schematic): existing VLA-RL perturbs actions around one intent, whereas TCoT-RL samples and optimizes alternative waypoint plans. (b) End-effector bird's-eye-view occupancy on one matched LIBERO-Plus reset (64 rollouts per mode); H: visitation entropy, SR: success rate. (c) Per-reset 40×40 coverage on 50 matched resets. (d) Mean paired gains (Sec. 4.3).

Yu et al., 2025; Huang et al., 2025; Chen et al., 2025b). However, these methods explore in the low-level action space by sampling continuous actions or action chunks. Such exploration is effective for refining local control, but it is inherently short-horizon and weakly structured. In long-horizon manipulation, failures often arise not from a single inaccurate motor command, but from an incorrect intermediate intent—for example, approaching the wrong object, selecting an unsuitable grasp direction, or following a spatial route that makes later execution difficult. Changing such an intent requires coordinated deviations over many steps in a high-dimensional continuous action space. Larger action noise or denser rewards can in principle induce them, but random perturbations find them inefficiently, and giving action-only RL our dense signals does not close the gap (App. E).

Fig. 1 illustrates this limitation: action-level exploration mostly varies one end-effector path, whereas sampling trajectory plans induces distinct routes and, on average over paired LIBERO-Plus resets, improves coverage, visitation entropy, early-stage action diversity, and success. These observations motivate an exploration space above individual actions but still grounded enough to guide continuous control. We argue that image-space end-effector trajectories provide such an interface. Compared with raw actions, it can express alternative approaches, intermediate subgoals, and multi-step spatial intent before committing to a specific action chunk. Compared with subgoal images or free-form language plans, 2D waypoints take only a few tokens, live in the pixel frame the VLM already perceives, can be labeled from demonstrations by an off-the-shelf point tracker without calibration, and admit cheap geometric feedback, making them practical for RL.

Based on this insight, we introduce **TCoT-RL**, a hierarchical RL framework in which a high-level trajectory policy autoregressively samples variable-length image-space waypoint sequences and a low-level flow-based action policy executes each sampled plan, so different samples represent alternative behavioral intents. Relative to trajectory chain-of-thought (TCoT) (Li et al., 2026b), which learns waypoint reasoning only by supervision, TCoT-RL adds adaptive-granularity waypoints, with fewer points for short motions and more for extended ones, and, more importantly, online RL that directly optimizes the trajectory distribution, turning plans into an exploration policy.

Crucially, TCoT-RL directly optimizes the distribution over trajectory hypotheses through online interaction, which raises a variance problem. A trajectory's task return is confounded by the stochasticity of its execution (action sampling and flow-matching noise), and sparse returns often assign identical values to many candidates from the same context, so the trajectory policy would receive high-variance, often degenerate group-relative advantages. To reduce this variance, the trajectory policy receives a dense reward that combines task progress with stage-aware endpoint grounding; the endpoint term depends only on the plan itself and does not require imitating an expert path. The action policy receives the task return with trajectory shaping, and the two levels are optimized with separate, level-specific group-relative advantages and clipped GRPO objectives.

Ablations show that the main gain emerges only when the trajectory distribution itself is optimized, not when trajectories are merely predicted. Our main contributions are threefold:

- We formulate trajectory reasoning as a directly optimizable exploration policy for VLA post-training. TCoT-RL samples high-level motion hypotheses and conditions action generation on them, enabling structured exploration over behavioral intent.
- We develop a joint trajectory–action RL framework in which a dense trajectory reward reduces the variance that low-level execution induces in trajectory-level advantages, with separate group-relative advantages and GRPO objectives for the two levels.
- We show that trajectory-policy optimization improves manipulation performance and robustness: TCoT-RL reaches a state-of-the-art **89.6%** on LIBERO-Plus, raises early-stage action diversity by 20%, and adds only 1.4% training cost, making it a practical VLA add-on.

2 Related Work

2.1 Reinforcement Learning for VLA Models

Reinforcement learning has recently emerged as an effective approach for improving VLA policies beyond supervised imitation (Guo et al., 2025; Chen et al., 2025b; Li et al., 2026a; Yu et al., 2025). iRe-VLA alternates online RL with supervised refinement, ConRFT uses consistency-based policy optimization (Guo et al., 2025; Chen et al., 2025b), and SimpleVLA-RL, RLinf-VLA, and SOP build scalable systems for parallel rollouts and distributed VLA training (Li et al., 2026a; Yu et al., 2025; Pan et al., 2026). Other approaches, such as GR-RL and RLDG, use reinforcement learning to improve precise or long-horizon manipulation and to generate higher-quality data for policy distillation (Li et al., 2025b; Xu et al., 2025). RIPT-VLA performs interactive post-training from sparse binary rewards (Tan et al., 2025), π_{RL} enables online RL for flow-based VLAs through a tractable stochastic denoising process (Chen et al., 2025a), SRPO builds progress-wise rewards from the policy's own successful rollouts in a latent world-model space (Fei et al., 2026a), and PRTS casts VLA pretraining as goal-conditioned contrastive RL (Zhang et al., 2026). Several works combine reinforcement learning with reasoning-oriented VLA architectures. VLA-R1 applies verifiable rewards and group-relative optimization to improve reasoning and action consistency, while ThinkAct reinforces latent visual planning before action generation (Liu et al., 2025; Huang et al., 2025). Hume introduces system-level reasoning into VLA decision making, and VLA-RL formulates robot interaction as a multi-step reasoning and action process optimized with environment feedback (Song et al., 2025; Ye et al., 2025). These methods show the value of RL for generalization and reasoning, but their exploration and policy optimization are still largely driven by action samples, rollout outcomes, latent variables, or system-level training signals. Like SRPO, TCoT-RL uses dense progress feedback, but applies it to a separately optimized trajectory policy rather than only to actions.

2.2 Embodied Reasoning for VLA Models

Embodied reasoning methods introduce intermediate representations between perception and action to improve spatial understanding and long-horizon manipulation, including textual reasoning over plans, objects, and gripper positions (Zawalski et al., 2024), predicted subgoal images (Zhao et al., 2025), and latent future states (Sun et al., 2026). Among them, trajectory-based representations provide a particularly grounded interface for connecting high-level intent with low-level control. RT-Trajectory conditions robot policies on rough trajectory sketches, enabling generalization to tasks and motions beyond the training distribution (Gu et al., 2024). TCoT introduces supervised trajectory chains of thought for manipulation and failure recovery (Li et al., 2026b), which TCoT-RL extends with adaptive waypoints and online RL. TraceVLA encodes state–action trajectories as visual traces to enhance spatiotemporal awareness (Zheng et al., 2024). HAMSTER adopts a hierarchical architecture in which a high-level VLM predicts a coarse end-effector path that guides a low-level control policy (Li et al., 2025a). MolmoAct generates editable image-space trajectory traces as mid-level spatial plans before predicting actions (Lee et al., 2025), and ACoT-VLA reasons through coarse reference trajectories that condition the action head (Zhong et al., 2026). These trajectories are effective intermediate conditions for action generation, but are learned by supervision or used as fixed conditions. TCoT-RL instead treats trajectory reasoning as a stochastic policy whose distribution is optimized with trajectory-level feedback during online interaction.

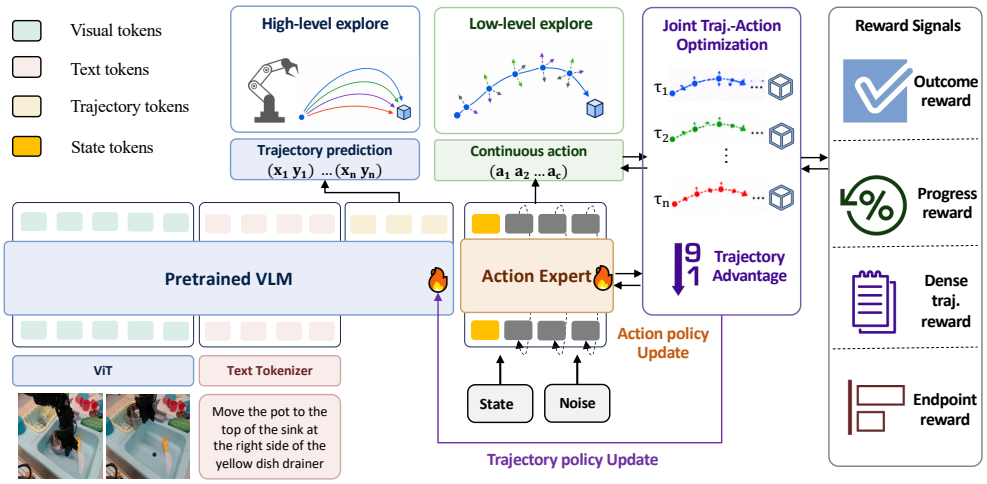

Figure 2: The VLM generates a trajectory-based chain-of-thought, representing a high-level motion hypothesis. The action expert then predicts continuous actions conditioned on the sampled trajectory, enabling low-level control. During post-training, TCoT-RL samples trajectory–action candidates, computes group-relative advantages, and jointly updates the trajectory and action policies.

3 Method

3.1 Problem Formulation and Overview

We consider language-conditioned robotic manipulation. At decision step t, the policy receives a visual observation o_t and a language instruction l, denoted by $c_t = (o_t, l)$, and predicts an action chunk $\mathbf{a}_t = (a_t, \ldots, a_{t+H-1})$. VLA policies often model $\mathbf{a}_t \sim \pi_\phi(\mathbf{a}_t \mid c_t)$, leaving the high-level motion intent implicit. TCoT-RL instead introduces an explicit trajectory policy that samples a variable-length sequence of image-space end-effector waypoints, $\tau_t = (\mathbf{q}_t^1, \ldots, \mathbf{q}_t^{N_t})$, $\mathbf{q}_t^i \in \mathbb{R}^2$. A trajectory-conditioned action policy then translates the sampled motion into continuous control:

$$\pi_{\theta,\phi}(\tau_t, \mathbf{a}_t \mid c_t) = \pi_\theta^{\text{traj}}(\tau_t \mid c_t)\, \pi_\phi^{\text{act}}(\mathbf{a}_t \mid c_t, \tau_t). \tag{1}$$

We instantiate TCoT-RL on top of $\pi_{0.5}$ (Black et al., 2025a). Training proceeds in two stages. First, trajectory-supervised fine-tuning initializes the trajectory policy with adaptive waypoint sequences and trains the action expert to execute trajectory-conditioned actions. Second, online reinforcement learning samples groups of trajectory–action candidates and jointly optimizes the trajectory and action policies with level-specific feedback and separate GRPO objectives.

3.2 Adaptive-Granularity Trajectory Policy

The trajectory policy represents high-level motion intent as a variable-length sequence of image-space end-effector waypoints. A fixed number of waypoints is inefficient for robotic manipulation: short or nearly stationary motions may be over-represented by redundant points, whereas spatially extended motions require finer structure to preserve their geometry. We therefore construct adaptive-granularity trajectory targets whose sequence length depends on the spatial extent of future motion.

Trajectory target construction. For each demonstration, we extract a dense two-dimensional gripper trajectory $\mathcal{P} = \{\mathbf{p}_k\}_{k=1}^{T}$, where $\mathbf{p}_k \in \mathbb{R}^2$, using an off-the-shelf point tracker (Karaev et al., 2024). At decision step t, we consider the future motion segment $\mathcal{S}_t = \{\mathbf{p}_t, \mathbf{p}_{t+1}, \ldots, \mathbf{p}_{t+H_\tau-1}\}$, with horizon H_τ. The segment is truncated when it extends beyond the end of the demonstration.

Motion-adaptive waypoint sampling. We measure the spatial extent of the future motion using its cumulative path length: $D_t = \sum_{k=t}^{t+H_\tau-2} \|\mathbf{p}_{k+1} - \mathbf{p}_k\|_2$. We determine the number of retained waypoints N_t by linearly mapping D_t from $[D_{\min}, D_{\max}]$ to the integer range $[N_{\min}, N_{\max}]$.

We then uniformly sample N_t positions along the cumulative arc length of $\mathcal{S}_t$ and obtain their coordinates through linear interpolation: $\tau_t = (\mathbf{q}_t^1, \mathbf{q}_t^2, \ldots, \mathbf{q}_t^{N_t}), \quad \mathbf{q}_t^i \in \mathbb{R}^2$. Arc-length sampling makes trajectory granularity depend on spatial motion, reducing redundant waypoints during stationary phases while preserving the structure of large movements. Construction details are given in App. C.1, and Sec. 4.4 compares adaptive allocation with fixed 2- and 10-waypoint targets.

Trajectory tokenization and policy initialization. Each trajectory target is serialized into a structured token sequence, <think> $x_1\, y_1 \cdots x_{N_t}\, y_{N_t}$ </think>. During trajectory-supervised fine-tuning, the trajectory policy is trained autoregressively to predict this sequence, while the flow-based action expert learns to generate continuous actions conditioned on the trajectory tokens. The termination token allows the policy to produce variable-length trajectories at inference time rather than committing to a fixed waypoint count. During online post-training, the supervised target is no longer required: the trajectory policy autoregressively samples $\tau_t \sim \pi_\theta^{\mathrm{traj}}(\cdot \mid c_t)$, so both the waypoint sequence and its effective length can vary across candidates. These sampled trajectories serve as alternative high-level plans for the trajectory-level exploration described in Sec. 3.4.

3.3 Dense Trajectory Reward

Direct trajectory-policy optimization requires discriminative feedback across trajectory hypotheses, yet sparse task rewards often provide little within-group variation. We therefore introduce a dense trajectory reward that jointly evaluates task progress and stage-dependent spatial intent.

Task-progress reward. Let s_t denote the environment state and let $\Phi_l(s_t) \in [0, 1]$ be a task-conditioned progress potential defined from task-relevant geometry under instruction l. For example, in a pick-and-place task, the potential measures both the progress of the end effector toward the manipulated object and the progress of the object toward its destination. For articulated-object interactions, the potential is defined using the corresponding task-relevant entity.

In addition to continuous progress, we define a discrete stage index m_t that records semantically meaningful milestones, such as reaching the object, manipulating it, and completing the task. After executing the trajectory-conditioned action chunk, the progress reward is

$$r_t^{\mathrm{prog}} = \alpha\,[\Phi_l(s_{t+1}) - \Phi_l(s_t)] + \beta \max(0, m_{t+1} - m_t), \tag{2}$$

where α and β balance improvement and milestone completion (values in App. C.3). This potential-based reward credits incremental progress and thus differentiates sampled trajectories.

Stage-aware endpoint reward. The progress reward evaluates the consequence of executing a trajectory, but does not assess whether the sampled trajectory expresses an appropriate high-level intent. We therefore evaluate its final waypoint against a stage-dependent subgoal.

Given a trajectory $\tau_t = (\mathbf{q}_t^1, \ldots, \mathbf{q}_t^{N_t})$, let $\mathbf{q}_t^{N_t}$ denote its endpoint. We select a target $\mathbf{g}_t$ according to the current task stage. For example, in a pick-and-place task, the target corresponds to the manipulated object before it is reached and switches to the destination after the object has been acquired. For interaction tasks, it corresponds to the relevant handle, button, or articulated component. The target is projected into the same coordinate system as the trajectory, yielding $\mathbf{u}_t^*$. The endpoint reward is

$$r_t^{\mathrm{end}} = 1 - \mathrm{clip}(\|\mathbf{q}_t^{N_t} - \mathbf{u}_t^*\|_2 / \eta_{\mathrm{end}}, 0, 1), \tag{3}$$

where η_{end} controls the spatial tolerance. This soft reward assigns partial credit to approximately correct endpoints and therefore remains informative early in online training.

Combined trajectory reward. The trajectory-specific reward is

$$r_t^{\mathrm{traj}} = \lambda_{\mathrm{prog}} r_t^{\mathrm{prog}} + \lambda_{\mathrm{end}} r_t^{\mathrm{end}}. \tag{4}$$

The progress term measures whether trajectory-conditioned execution moves the environment toward the goal, while the endpoint term directly grounds the trajectory in the current task stage. The reward does not require the full trajectory to match an expert path, allowing the policy to explore alternative routes that preserve task-relevant intent. These task-state signals are used only for reward computation during online post-training and are not exposed to the policy as observations (App. J).

3.4 JOINT TRAJECTORY–ACTION OPTIMIZATION

Given the trajectory-specific feedback, we jointly optimize the high-level trajectory policy and action policy through online interaction. The key idea is to assign credit to the two policy levels according to their distinct roles: the trajectory is responsible for exploring high-level motion intent, while the action is responsible for executing the sampled intent through continuous control.

Grouped trajectory–action rollouts. For each observation–instruction context $c_t = (o_t, l)$, we sample a group of G trajectory–action candidates from the behavior policy:

$$\left\{\left(\tau_t^i, \mathbf{a}_t^i\right)\right\}_{i=1}^{G}, \qquad \tau_t^i \sim \pi_{\theta_{\text{old}}}^{\text{traj}}(\cdot \mid c_t), \quad \mathbf{a}_t^i \sim \pi_{\phi_{\text{old}}}^{\text{act}}(\cdot \mid c_t, \tau_t^i). \tag{5}$$

Each sampled trajectory represents a candidate high-level motion hypothesis, and the corresponding action chunk realizes that hypothesis in the environment. The resulting rollouts are evaluated by the task return R_i^{task} and the accumulated trajectory-specific return R_i^{traj}. It gives dense, discriminative feedback to trajectory hypotheses while keeping task-level supervision for actions.

Level-specific group-relative advantages. The two policy levels receive distinct learning signals. The action policy is evaluated by the task return augmented with trajectory shaping, whereas the trajectory policy is directly evaluated by the trajectory-specific reward: $J_i^{\text{act}} = R_i^{\text{task}} + \omega R_i^{\text{traj}}, \quad J_i^{\text{traj}} = R_i^{\text{traj}}$, where ω controls the contribution of trajectory shaping to action optimization. For each level $x \in \{\text{traj}, \text{act}\}$, we compute a group-relative advantage $\hat{A}_i^x$ by normalizing the corresponding score J_i^x across the G candidates sampled from the same context, following GRPO (Shao et al., 2024).

Separate GRPO objectives. The trajectory and action outputs are generated by different policy modules and are therefore optimized with separate importance ratios ρ_i^{traj} and ρ_i^{act}. For the flow-based action expert, we follow π_{RL} (Chen et al., 2025a) and compute ρ_i^{act} from the tractable Gaussian transition likelihoods of the stochastic denoising process.

For $x \in \{\text{act}, \text{traj}\}$, the clipped GRPO objective is written uniformly as

$$\mathcal{L}_x = -\mathbb{E}_i \left[\min\left(\rho_i^x \widehat{A}_i^x, \text{clip}(\rho_i^x, 1-\varepsilon_x, 1+\varepsilon_x)\widehat{A}_i^x\right)\right]. \tag{6}$$

The full reinforcement-learning objective is $\mathcal{L}_{\text{RL}} = c_{\text{act}}\mathcal{L}_{\text{act}} + c_{\text{traj}}\mathcal{L}_{\text{traj}}$, where c_{act} and c_{traj} balance low-level execution learning and high-level trajectory optimization.

Setting $c_{\text{traj}} = 0$ and freezing the trajectory policy reduces the method to trajectory-conditioned action-only RL. In contrast, $c_{\text{traj}} > 0$ directly changes the distribution over sampled motion hypotheses, so TCoT-RL optimizes not only how a trajectory is executed but also which trajectories are explored. Hyperparameters and importance-ratio diagnostics are given in Apps. C.4 and I.

4 EXPERIMENTS

We compare TCoT-RL with state-of-the-art methods on LIBERO (Liu et al., 2023) and LIBERO-Plus (Fei et al., 2026b), analyze its exploration behavior, and ablate its components.

4.1 EXPERIMENTAL SETUP

Benchmarks. We evaluate on LIBERO, whose four suites (SPATIAL, OBJECT, GOAL, LONG) measure standard manipulation performance, and on LIBERO-Plus, which adds seven distribution shifts (camera, robot, language, light, background, noise, and layout) to assess robustness.

We instantiate TCoT-RL on top of $\pi_{0.5}$ (Black et al., 2025a), with a pretrained vision–language backbone and a flow-based action expert. Training consists of trajectory-supervised initialization followed by online joint optimization of the trajectory and action policies. We compare against supervised VLAs (Kim et al., 2024; Bu et al., 2025; Black et al., 2025b; Pertsch et al., 2025; Kim et al., 2025; Sun et al., 2026; StarVLA Comm., 2026; Black et al., 2025a), ACoT-VLA (Zhong et al., 2026), and RL methods (Tan et al., 2025; Fei et al., 2026a; Zhang et al., 2026; Chen et al., 2025a; Yu et al., 2025). As some methods use LIBERO-Plus demonstrations for SFT, we report two

Method	RL	Camera	Robot	Language	Light	Background	Noise	Layout	Overall SR.
OpenVLA	-	0.8	3.5	23.0	8.1	34.8	15.2	28.5	15.6
UniVLA	-	1.8	46.2	69.6	69.0	81.0	21.2	31.9	43.9
π_0	-	13.8	6.0	58.8	85.0	81.4	79.0	68.9	53.6
π_0-FAST	-	65.1	21.6	61.0	73.2	73.2	74.4	68.8	61.6
OpenVLA-OFT	-	56.4	31.9	79.5	88.7	93.3	75.8	74.2	69.6
VLA-JEPA	-	63.3	67.1	85.4	95.6	93.6	66.3	85.1	79.5
StarVLA	-	52.5	64.3	86.2	97.8	98.1	80.2	79.1	79.7
$\pi_{0.5}$	-	75.8	79.4	83.3	95.5	95.0	89.6	87.0	85.7
ACoT-VLA*	-	72.6	**82.6**	**87.5**	97.7	96.5	87.8	88.1	86.6
RIPT-VLA	✓	55.2	31.2	77.6	88.4	91.6	73.5	74.2	68.4
SRPO*	✓	83.4	62.0	73.6	97.2	**97.7**	85.7	75.2	82.1
PRTS*	✓	56.7	56.1	87.3	96.4	95.8	72.3	82.9	76.5
pi-RL*	✓	93.6	64.4	81.4	**98.4**	96.1	**98.2**	88.0	87.8
TCoT-RL	✓	77.9	73.8	84.6	95.9	98.4	93.5	88.6	86.8
TCoT-RL*	✓	**94.3**	72.5	83.3	96.7	97.1	97.7	**89.7**	**89.6**

Table 1: Success rate (%) on LIBERO-Plus under seven OOD variations. An asterisk ($*$) indicates that LIBERO-Plus demonstrations are used during supervised fine-tuning. Bold and underlined values denote the best results among methods with and without LIBERO-Plus supervised data, respectively. Overall SR is aggregated across all evaluation tasks and therefore need not equal the unweighted mean of the seven variation-level rates. $\pi_{0.5}$ is the official result.

settings: *LIBERO-only SFT* and *LIBERO-Plus SFT*, the latter marked by *. Main results follow the official protocols (50 trials per LIBERO task; one for each of the 10,030 LIBERO-Plus variants); the TCoT-RL entries in Tables 1 and 3 share one checkpoint trained for 60 updates (App. B).

Implementation details. Trajectory supervised fine-tuning uses AdamW with a learning rate of 5×10^{-5}, a global batch size of 256, and a trajectory-loss weight of 0.05; online RL starts from the step-10,000 checkpoint. Each RL update collects 256 candidate episodes in groups of $G = 8$ rollouts per task–reset context from 64 parallel environments, with trajectory sampling temperature 1.0 and top-k 50, four flow-SDE denoising steps with noise scale 0.5, clip range $[0.8, 1.2]$, actor learning rate 5×10^{-6}, and no KL penalty. The dense reward uses $\lambda_{\text{prog}} = 0.10$ and $\lambda_{\text{end}} = 0.05$. All training runs on four NVIDIA A100 GPUs; under a matched interaction budget, TCoT-RL adds only 1.4% wall-clock time over action-only RL (App. G). Full settings appear in App. C and Tables 4–5.

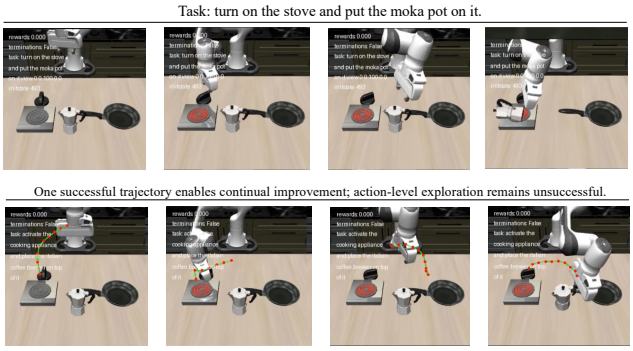

Figure 3: In this example, TCoT-RL continues to improve after discovering a successful rollout, whereas action-level exploration does not.

4.2 Comparison with State-of-the-Art Methods

Results on LIBERO-Plus. Table 1 compares TCoT-RL with representative supervised and RL-based VLA methods under seven distribution shifts. With LIBERO-Plus demonstrations available during supervised fine-tuning, TCoT-RL* achieves a state-of-the-art overall success rate of **89.6%**, outperforming the strongest competing RL baseline, pi-RL*, by **1.8 percentage points**. Importantly, TCoT-RL remains strong in the more restrictive *LIBERO-only SFT* setting, reaching **86.8%** without using LIBERO-Plus demonstrations during supervised training. These results indicate that the improvement is not solely due to in-distribution trajectory supervision, but also arises from online trajectory–action reinforcement learning. The gains are particularly pronounced under camera, background, visual-noise, and layout shifts, suggesting that trajectory-level exploration improves

robustness to substantial spatial and visual distribution changes. Robot-configuration shifts remain more challenging, indicating that embodiment changes may require stronger adaptation. Figure 3 illustrates how trajectory-level optimization can support improvement on a long-horizon task. The instruction requires two ordered subgoals—activating the stove and then transporting the moka pot to the burner. Once a successful trajectory candidate is discovered, its advantage reinforces the waypoint proposals and the action update refines their execution (more examples in App. K).

Results on LIBERO. Table 2 reports results on the standard LIBERO benchmark. TCoT-RL achieves an average success rate of 98.6%, exceeding its $\pi_{0.5}$ backbone by 1.7 percentage points. It obtains 99.8% success on SPATIAL and 97.6% on LONG, indicating that trajectory-conditioned execution remains effective on both standard spatial generalization and long-horizon manipulation. On this near-saturated benchmark, TCoT-RL cuts the average failure rate of $\pi_{0.5}$ by more than half, from 3.1% to 1.4% across the four suites.

Method	RL	Spatial	Object	Goal	Long	Avg.
OpenVLA	-	84.7	88.4	79.2	53.7	76.5
π_0-FAST	-	96.4	96.8	88.6	60.2	85.5
π_0	-	96.8	98.8	95.8	85.2	94.2
UniVLA	-	96.5	96.8	95.6	92.0	95.2
MemoryVLA	-	98.4	98.4	96.4	93.4	96.5
$\pi_{0.5}$	-	98.8	98.2	98.0	92.4	96.9
VLA-JEPA	-	96.2	99.6	97.2	95.8	97.2
X-VLA	-	98.2	98.6	97.8	97.6	98.1
RLinf-VLA	✓	99.4	**99.8**	**98.8**	94.0	98.1
TCoT-RL	✓	**99.8**	99.4	97.6	**97.6**	**98.6**

Table 2: Success rate (%) on the four LIBERO task suites.

4.3 TRAJECTORY SAMPLING EXPANDS POLICY SEARCH

Paired rollout protocol. To isolate the effect of the exploration mechanism, we compare trajectory-level and action-level exploration from the same SFT checkpoint, with identical instructions, resets, and rollout budgets. The action-level baseline explores only through action noise, whereas TCoT-RL first samples a trajectory hypothesis and then generates trajectory-conditioned actions. We evaluate 50 LIBERO-Plus resets with 64 rollouts per mode; Fig. 1(b) shows one such reset.

Exploration metrics. We measure state-space exploration by projecting executed end-effector paths onto the world XY plane and computing workspace coverage and visitation entropy at two grid resolutions. We also measure action diversity using the average pairwise ℓ_2 distance between full action sequences and between their first five actions. All metrics are paired by initial state.

Results. As shown in Fig. 1(c,d) and Fig. 14, trajectory-level exploration consistently expands policy search beyond action-level stochasticity. It increases workspace coverage by 12.1% and 13.1% at 20×20 and 40×40 resolutions, respectively, and improves visitation entropy by 4.2% and 4.4%. The corresponding absolute gain at 40×40 is 10.5 visited cells and 0.148 in entropy. Trajectory sampling also increases action diversity by 4.8% over complete rollouts and, more notably, by 20.0% over the first five actions. The much larger early-stage gain indicates that sampled trajectories alter the initial behavioral intent rather than merely accumulating local control noise. This broader exploration is accompanied by an increase in success rate from 0.72 to 0.82. App. L further reports trajectory-space diversity statistics and a high-coverage matched example.

4.4 ABLATION STUDIES

We conduct controlled ablations to disentangle the effects of trajectory supervision, trajectory-conditioned execution, trajectory reward shaping, and direct trajectory-policy optimization. All online RL variants are initialized from the same TCoT-SFT checkpoint, i.e., TCoT-style supervised trajectory reasoning (Li et al., 2026b) with adaptive waypoints, and use the same rollout budget and evaluation protocol. As shown in Table 3, trajectory-supervised initialization improves success from 83.9% to 85.5%, showing that explicit trajectory prediction provides a stronger starting policy. Action-only RL further reaches 85.8%, but its 0.3-point gain over TCoT-SFT is limited, indicating that optimizing low-level actions alone provides only modest additional gains.

Training configuration	Traj. output	Traj. reward	Act. update	Traj. update	Avg. SR.
Action-only SFT	–	–	–	–	83.9
TCoT-SFT	✓	–	–	–	85.5
Action-only RL	–	–	✓	–	85.8
Trajectory-conditioned RL	✓	–	✓	–	85.5
Reward-shaped Action RL	✓	✓	✓	–	84.8
Joint TCoT-RL	✓	✓	✓	✓	**86.8**

Table 3: Component ablation on LIBERO-Plus. "Act./Traj. update": whether the action/traj. policy is optimized online; Action-only SFT is $\pi_{0.5}$ fine-tune without traj. supervision.

Sampling trajectories without optimizing them does not help: trajectory-conditioned RL obtains 85.5%, identical to TCoT-SFT. Trajectory sampling still diversifies rollouts, but the frozen trajectory policy cannot reinforce the hypotheses that succeed, and the outcome variation they cause enters the action advantages as noise that the action policy cannot control, consistent with its slightly lower score. Adding the trajectory reward to the action update alone reduces it further to 84.8%, since the reward scores trajectory hypotheses while its gradient reaches only the action policy.

Joint TCoT-RL achieves the best average success rate of 86.8%, improving over TCoT-SFT by 1.3 points, action-only RL by 1.0 point, and reward-shaped action RL by 2.0 points. The contrast between reward-shaped action RL and Joint TCoT-RL is particularly important: the same trajectory reward becomes beneficial only when it is used to directly update the trajectory policy. The gain therefore comes from optimizing the trajectory distribution itself, rather than from predicting trajectories, conditioning actions on them, or using trajectory rewards as action shaping. Apps. E–I further cover dense action-only rewards, late-stage learning, trajectory interventions, and a second training seed.

Adaptive waypoint allocation. Fig. 4 compares adaptive allocation with fixed 2- and 10-waypoint targets resampled from the same tracks and trained under a matched budget (App. D). Mean online success over all 24 updates is nearly identical, but the dynamics differ: around update 10 the fixed variants are still improving, whereas adaptive allocation temporarily drops. This dip plausibly reflects the policy learning to exploit variable-length trajectories, whose length is itself sampled and optimized online. Once it does, adaptive allocation pulls ahead and attains the highest final-four-update success (82.8% vs. 80.2% for Fixed-10 and 79.7% for Fixed-2). On 128 LIBERO-Plus-10 evaluation resets, it reaches 81.3% task success versus 78.9% (Fixed-2) and 75.8% (Fixed-10), and the gap widens for end-state success (75.8% vs. 73.4% and 67.2%). Notably, denser fixed trajectories are not better: Fixed-10 trails Fixed-2 on both evaluation metrics, consistent with redundant waypoints on short or stationary motions adding sampling noise without useful structure, whereas two waypoints cannot describe spatially extended motions.

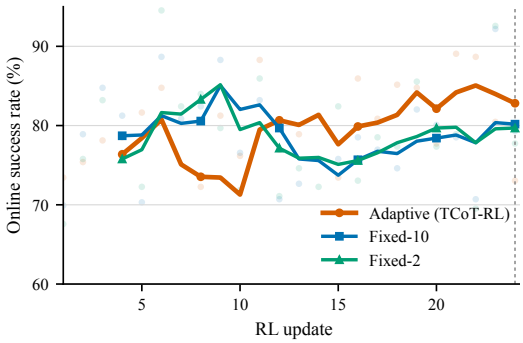

Figure 4: Adaptive vs. fixed waypoint allocation: online success over 24 RL updates (dots: per-update rate, lines: four-update trailing mean).

5 Conclusion

We presented TCoT-RL, which treats trajectory reasoning as an explicit, RL-optimized exploration policy for VLA post-training. By sampling compact, image-grounded motion hypotheses instead of perturbing low-level actions, it changes behavioral intent rather than adding control noise, reaching a state-of-the-art 89.6% on LIBERO-Plus and 98.6% on LIBERO with broader state coverage and more diverse early-stage behavior, at only 1.4% extra training cost. Ablations show that trajectory rewards help only when they directly optimize the trajectory policy, making the trajectory distribution itself the key object to optimize, rather than its downstream action execution.

Limitations and future work. Our experiments focus on single-arm manipulation in simulation, with image-space waypoints from a single camera view. Promising directions include real-robot post-training, 3D or multi-view trajectory representations, learned or VLM-based progress rewards, and extending trajectory-level exploration to contact-rich, bimanual, and mobile manipulation.

GENERATIVE AI USAGE DISCLOSURE

Generative AI tools were used to assist with manuscript editing and presentation, including improving clarity, refining figures, and giving feedback on experimental design and result interpretation. All research ideas, methodological decisions, implementations, experiments, and reported results were reviewed and verified by the authors. The authors take full responsibility for the final content.

REFERENCES

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. GR00T N1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

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

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025b. doi: 10.15607/RSS.2025.XXI.010.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, et al. RT-1: robotics transformer for real-world control at scale. In *Robotics: Science and Systems*, 2023.

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. UniVLA: Learning to act anywhere with task-centric latent actions. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.014.

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, 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. *arXiv preprint arXiv:2510.25889*, 2025a.

Yuhui Chen, Shuai Tian, Shugao Liu, Yingting Zhou, Haoran Li, and Dongbin Zhao. Conrft: A reinforced fine-tuning method for vla models via consistency policy. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025b. doi: 10.15607/RSS.2025.XXI.019.

Senyu Fei, Siyin Wang, Li Ji, Ao Li, Shiduo Zhang, Liming Liu, Jinlong Hou, Jingjing Gong, Xianzhong Zhao, and Xipeng Qiu. SRPO: Self-referential policy optimization for vision-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6718–6728, June 2026a.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, Jinlan Fu, Jingjing Gong, and Xipeng Qiu. LIBERO-Plus: A progressive robustness benchmark for vision-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 38574–38583, June 2026b.

Jiayuan Gu, Sean Kirmani, Paul Wohlhart, Yao Lu, Montserrat Gonzalez Arenas, Kanishka Rao, Wenhao Yu, Chuyuan Fu, Keerthana Gopalakrishnan, Zhuo Xu, et al. Rt-trajectory: Robotic task generalization via hindsight trajectory sketches. In *International Conference on Learning Representations*, 2024.

Yanjiang Guo, Jianke Zhang, Xiaoyu Chen, Xiang Ji, Yen-Jen Wang, Yucheng Hu, and Jianyu Chen. Improving vision-language-action model with online reinforcement learning. In *2025 IEEE International Conference on Robotics and Automation*, pp. 15665–15672. IEEE, 2025. doi: 10.1109/ICRA55743.2025.11127299.

Chi-Pin Huang, Yueh-Hua Wu, Min-Hung Chen, Yu-Chiang Frank Wang, and Fu-En Yang. Thinkact: Vision-language-action reasoning via reinforced visual latent planning. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. In *European Conference on Computer Vision*, 2024.

Kim, Moo Jin, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P. Foster, Pannag R. Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. In *8th Annual Conference on Robot Learning*, 2024.

Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, Winson Han, Wilbert Pumacay, Angelica Wu, Rose Hendrix, Karen Farley, Eli VanderBilt, Ali Farhadi, Dieter Fox, and Ranjay Krishna. MolmoAct: Action reasoning models that can reason in space. *arXiv preprint arXiv:2508.07917*, 2025.

Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaohui Yang, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, Dehui Wang, Dingxiang Luo, Yuchen Fan, Youbang Sun, Jia Zeng, Jiangmiao Pang, Shanghang Zhang, Yu Wang, Yao Mu, Bowen Zhou, and Ning Ding. Simplevla-rl: Scaling vla training via reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026a.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, Xiaofan Wang, Bei Liu, Jianlong Fu, Jianmin Bao, Dong Chen, Yuanchun Shi, Jiaolong Yang, and Baining Guo. CogACT: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation. *arXiv preprint arXiv:2411.19650*, 2024.

Xiang Li, Ya-Li Li, Yuan Wang, Huaqiang Wang, and Shengjin Wang. TCoT: Trajectory chain-of-thoughts for robotic manipulation with failure recovery in vision-language-action model. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(8):6486–6494, 2026b. doi: 10.1609/aaai.v40i8.37577.

Yi Li, Yuquan Deng, Jesse Zhang, Joel Jang, Marius Memmel, Raymond Yu, Caelan Reed Garrett, Fabio Ramos, Dieter Fox, Anqi Li, Abhishek Gupta, and Ankit Goyal. HAMSTER: Hierarchical action models for open-world robot manipulation. In *The Thirteenth International Conference on Learning Representations*, 2025a.

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. *arXiv preprint arXiv:2512.01801*, 2025b.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 44776–44791, 2023.

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. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Mingjie Pan, Siyuan Feng, Qinglin Zhang, Xinchen Li, Jianheng Song, Chendi Qu, Yi Wang, Chuankang Li, Ziyu Xiong, Zhi Chen, Yi Liu, and Jianlan Luo. Sop: A scalable online post-training system for vision-language-action models. *arXiv preprint arXiv:2601.03044*, 2026.

Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. FAST: Efficient action tokenization for vision-language-action models. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.012.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Haoming Song, Delin Qu, Yuanqi Yao, Qizhi Chen, Qi Lv, Yiwen Tang, Modi Shi, Guanghui Ren, Maoqing Yao, Bin Zhao, Dong Wang, and Xuelong Li. Hume: Introducing system-2 thinking in visual-language-action model. *arXiv preprint arXiv:2505.21432*, 2025.

StarVLA Comm. StarVLA: A lego-like codebase for vision-language-action model developing. *arXiv preprint arXiv:2604.05014*, 2026.

Jingwen Sun, Wenyao Zhang, Zekun Qi, Shaojie Ren, Zezhi Liu, Hanxin Zhu, Guangzhong Sun, Xin Jin, and Zhibo Chen. VLA-JEPA: Enhancing vision-language-action model with latent world model. *arXiv preprint arXiv:2602.10098*, 2026.

Shuhan Tan, Kairan Dou, Yue Zhao, and Philipp Krähenbühl. Interactive post-training for vision-language-action models. *arXiv preprint arXiv:2505.17016*, 2025.

Charles Xu, Qiyang Li, Jianlan Luo, and Sergey Levine. Rldg: Robotic generalist policy distillation via reinforcement learning. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.028.

Angen Ye, Zeyu Zhang, Boyuan Wang, Xiaofeng Wang, Dapeng Zhang, and Zheng Zhu. Vla-r1: Enhancing reasoning in vision-language-action models. *arXiv preprint arXiv:2510.01623*, 2025.

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, Xiangyuan Wang, Xu Fu, 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. *arXiv preprint arXiv:2509.15965*, 2025.

Michał Zawalski, William Chen, Karl Pertsch, Oier Mees, Chelsea Finn, and Sergey Levine. Robotic control via embodied chain-of-thought reasoning. In *Conference on Robot Learning*, 2024.

Yang Zhang, Jiangyuan Zhao, Chenyou Fan, Fangzheng Yan, Tian Li, Haitong Tang, Sen Fu, Xuan'er Wu, Qizhen Weng, Weinan Zhang, Xiu Li, Chi Zhang, Chenjia Bai, and Xuelong Li. PRTS: A primitive reasoning and tasking system via contrastive representations. *arXiv preprint arXiv:2604.27472*, 2026. doi: 10.48550/arXiv.2604.27472.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. Cot-vla: Visual chain-of-thought reasoning for vision-language-action models. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025.

Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. *arXiv preprint arXiv:2412.10345*, 2024.

Linqing Zhong, Yi Liu, Yifei Wei, Ziyu Xiong, Maoqing Yao, Si Liu, and Guanghui Ren. ACoT-VLA: Action chain-of-thought for vision-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8152–8162, June 2026.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of the 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 2023.

A Appendix Overview

This appendix provides implementation details and controlled analyses that complement the main paper. We first describe trajectory-target construction, trajectory-supervised initialization, dense reward design, and online optimization. We then conduct a series of controlled studies to examine the main design choices and potential confounding factors in TCoT-RL.

Specifically, we compare adaptive waypoint allocation with fixed trajectory horizons, construct reward- and interaction-matched action baselines, analyze learning dynamics and computational cost, and intervene on the trajectory input to test whether the action expert functionally depends on it. We further report independent training runs and policy-ratio diagnostics, evaluate robustness to imperfect training-time reward labels, and provide qualitative and trajectory-space analyses of the resulting behaviors.

Together, these studies test whether the gains of TCoT-RL arise from explicit trajectory-policy optimization rather than denser reward alone, whether the hierarchical policy incurs substantial additional cost, whether trajectory conditioning affects executed behavior, and whether the optimization remains stable under realistic reward noise.

B Controlled Evaluation Protocol

Evaluation protocols. All results reported in the main paper follow the official benchmark evaluation protocols: 50 trials per task on LIBERO, and one trial for each of the 10,030 task variants on LIBERO-Plus. The TCoT-RL entries in Tables 1 and 3 are obtained from the same checkpoint after 60 online-RL updates. To reduce the substantial cost of repeated controlled studies, most appendix ablations instead use 24 online-RL updates and a smaller fixed evaluation set of 128 episodes, consisting of 32 predetermined resets from each of four LIBERO-Plus suites. The waypoint-allocation study (App. D) uses 128 fixed resets from the harder LIBERO-Plus-10 suite, and the repeated-run analysis (App. I) uses 1,024 episodes (256 per suite). Because of the shorter training and the different evaluation sets, absolute success rates in the appendix are not directly comparable with those in the main paper. These reduced evaluations are used only for matched comparisons among controlled variants and are not used to replace the official evaluation results reported in the main paper. Unless otherwise stated, all compared methods within a controlled study are evaluated on exactly the same task/reset list.

Matched training conditions. Unless stated otherwise, controlled experiments start from the same trajectory-supervised checkpoint and use seed 42, 24 online-RL updates, and 256 candidate rollouts per update. Each method therefore executes 6,144 candidate rollouts and 2,949,120 simulator steps under the same interaction budget. Training uses four NVIDIA A100 GPUs.

Uncertainty estimates. For task-success rates on the 128-episode controlled evaluations, we report 95% Wilson score intervals computed from episode-level outcomes. These intervals capture finite-evaluation uncertainty but not variation across training seeds. Accordingly, single-seed comparisons in the appendix are interpreted as controlled point-estimate evidence rather than definitive estimates of training variance. Continuous paired exploration metrics use paired bootstrap intervals where explicitly stated.

C Implementation Details

C.1 Adaptive Trajectory Targets

We construct trajectory supervision from the agent-view demonstration stream. A vision–language detector localizes the gripper center in the first frame, and CoTracker (Karaev et al., 2024) propagates the point throughout the episode. Tracking is re-initialized when visibility falls below 0.5 to reduce drift under prolonged occlusion.

At decision step t, we consider the next $H_\tau = 50$ frames, or the remaining suffix near episode termination. Let D_t denote the cumulative image-space arc length over this window. For non-stationary motion, we use a clipped linear mapping from $D_t \in [48, 96]$ pixels to $N_t \in [2, 10]$

waypoints and uniformly interpolate the selected points in arc-length space. Thus, short motions receive compact targets, whereas spatially extended motions retain finer geometric detail.

For storage, each annotation is padded to a 10×2 array at the original 256×256 resolution by repeating the final valid point. Before tokenization, coordinates are rescaled to 224×224, rounded to integer pixel locations, and consecutive duplicates are removed. Only the valid waypoints are serialized as

$$\texttt{<think>}\ x_1\, y_1\, \cdots\, x_N\, y_N\ \texttt{</think>},$$

so storage padding never becomes supervised output. For visually stationary segments whose maximum displacement is at most one pixel, we use an empty trajectory body rather than assigning arbitrary waypoint coordinates.

C.2 Trajectory-Supervised Initialization

Each supervised example contains the agent view, wrist view, proprioceptive state, language instruction, a 10-step expert action chunk, and the adaptive trajectory target. We optimize the original $\pi_0.5$ flow-matching objective together with trajectory-token cross-entropy:

$$\mathcal{L}_{\mathrm{SFT}} = \mathcal{L}_{\mathrm{flow}} + 0.05\, \mathcal{L}_{\mathrm{traj}}.$$

Table 4 summarizes the training configuration. Although supervised training can continue beyond 10,000 steps, all controlled RL comparisons start from the same fixed step-10,000 checkpoint. No method selects a different initialization checkpoint, avoiding checkpoint-selection as a confounding factor.

Setting	Value	Setting	Value
Backbone	$\pi_{0.5}$	Global batch	256
Action horizon	10	Optimizer	AdamW
Future horizon	50 frames	β_1, β_2	$0.9, 0.95$
Waypoint count	2–10	Peak/final LR	5×10^{-5}
Trajectory weight	0.05	Warmup	10,000 steps
Gradient norm	1.0	Weight decay	10^{-10}
RL initialization	step 10,000	Image size	224×224

Table 4: Trajectory-supervised initialization. The LR schedule is configured with a 10,000-step warmup and equal peak and terminal learning rates.

C.3 Dense Reward Construction

We instantiate the trajectory reward in Sec. 3.3 using task-relevant simulator geometry during online training. For placement predicates, end-effector–object and object–target distances are converted to clipped scores using normalization radii of 0.40 m and 0.50 m, respectively. The task-conditioned potential is

$$\Phi_l(s) = 0.35\, s_{\mathrm{reach}} + 0.65\, s_{\mathrm{place}}.$$

Reaching, placement, and completion stages use thresholds of 0.70, 0.65, and 0.85. Actuation tasks analogously use distance to the corresponding fixture. For tasks with multiple predicates, potentials are averaged and endpoint selection prioritizes the least advanced subgoal.

The progress reward is

$$r_t^{\mathrm{prog}} = \mathrm{clip}\left(4[\Phi_l(s_{t+1}) - \Phi_l(s_t)] + 0.25 \max(0, m_{t+1} - m_t), -1, 1\right). \tag{7}$$

For the endpoint term, the target is the manipulated object before acquisition and the destination afterward; for actuation tasks, it is the relevant fixture. The 3D target is projected into the transformed 224×224 agent view, and the final predicted waypoint is scored as

$$r_t^{\mathrm{end}} = 1 - \mathrm{clip}\left(\frac{\|\mathbf{q}_t^{N_t} - \mathbf{u}_t^*\|_2}{120}, 0, 1\right). \tag{8}$$

The final trajectory reward is

$$r_t^{\text{traj}} = 0.10\, r_t^{\text{prog}} + 0.05\, r_t^{\text{end}}.$$

No additional format, smoothness, start-point, or action-alignment reward is used in the reported model. Simulator geometry is used only to construct training-time rewards and is never exposed as policy input; Sec. J examines this separation and robustness to imperfect reward labels.

C.4 Online Optimization and Evaluation

Table 5 gives the shared online settings. The main results use 60 online-RL updates, whereas all appendix studies, including the repeated-run analysis, use 24 updates. A group repeats one task/reset context eight times; 64 parallel environments and four rollout epochs yield 32 groups, or 256 candidate episodes, per update. Delayed trajectory feedback is associated with the trajectory that generated the corresponding action chunk before computing its group-relative advantage.

Setting	Value	Setting	Value
Group size	8	Candidates/update	256
Parallel envs	64	Rollout epochs	4
Episode horizon	480 steps	Action chunk	10×7
Denoising steps	4	Flow-SDE noise	0.5
Trajectory temp.	1.0	Top-k	50
Update epochs	4	Clip range	$[0.8, 1.2]$
Actor LR	5×10^{-6}	KL coefficient	0
Global/micro batch	1024/128	Gradient norm	1.0
Adam β_1, β_2	$0.9, 0.95$	Adam ϵ	10^{-5}
Weight decay	0.01	Log-ratio guard	$[-60, 60]$

Table 5: Online post-training settings shared by the main-result run and the controlled runs. Both action and trajectory loss coefficients are 1.0.

The main results follow the official protocols (50 trials per LIBERO task; one trial for each of the 10,030 LIBERO-Plus variants). The repeated-run evaluation uses 1,024 episodes (256 per suite); the lower-cost controlled studies use 128 fixed episodes (32 per suite, or 128 LIBERO-Plus-10 resets for App. D). An episode is successful if the environment predicate is reached at least once within 480 simulator steps. All compared checkpoints use identical task/reset lists and are evaluated sequentially to avoid host-memory coupling. Evaluation and online training disable video saving; qualitative figures are constructed only from separately designated visualization rollouts.

D Adaptive Waypoint Allocation versus Fixed Horizons

TCoT-RL represents future motion with an adaptive number of waypoints: short or nearly stationary motions use compact trajectories, whereas spatially extended motions receive finer resolution. We test whether this adaptive allocation is beneficial compared with fixed-length trajectory representations.

Matched comparison. We construct two controls from the same tracked demonstration trajectories: *Fixed-2* uniformly resamples each valid trajectory to two waypoints, whereas *Fixed-10* resamples it to ten waypoints. Thus, the underlying motion tracks remain unchanged and only the trajectory granularity differs. All three variants use the same base initialization, demonstration set, 10,000-step supervised-training budget, online-RL configuration, seed, and interaction budget. We compare them after 24 RL updates (6,144 candidate rollouts) on the same 128 fixed evaluation episodes from the LIBERO-Plus-10 suite. This suite is harder than the four-suite set used in the other controlled studies, so the absolute success rates here are lower.

Results. As shown in Table 6 and Fig. 4, adaptive allocation achieves the highest evaluation success rate at **81.25%**, compared with 78.91% for Fixed-2 and 75.78% for Fixed-10. It also obtains the highest final-four-update online success (82.81%), and the same ordering is observed for end-state success.

Method	Online mean	Final-4 mean	Eval. success (95% Wilson CI)	End-state success (95% Wilson CI)	Adaptive gain (pp)
Adaptive (TCoT-RL)	**79.23**	**82.81**	**81.25 [73.62, 87.06]**	**75.78 [67.68, 82.38]**	–
Fixed-10	78.87	80.18	75.78 [67.68, 82.38]	67.19 [58.66, 74.72]	+5.47
Fixed-2	78.53	79.69	78.91 [71.05, 85.08]	73.44 [65.18, 80.32]	+2.34

Table 6: Matched waypoint-allocation comparison on the LIBERO-Plus-10 suite (%). All methods use the same 24-update interaction budget and the same 128 deterministic evaluation resets drawn from LIBERO-Plus-10. Because this suite is harder than the four-suite 128-episode set used in Apps. E, H, and J, absolute success rates are not comparable across these studies. "Adaptive gain" is Adaptive minus the corresponding fixed-horizon method. Intervals are 95% Wilson score intervals computed from 128 episode-level success outcomes.

These results favor adapting trajectory granularity to motion extent rather than using a globally fixed waypoint budget. Fixed-10 can spend unnecessary capacity on short or stationary motions, whereas Fixed-2 may under-represent spatially extended motion. Adaptive allocation avoids this manual horizon choice and provides the strongest overall learning and evaluation profile. Because this study uses a single training seed and the Wilson intervals overlap, we interpret the differences as descriptive evidence rather than a statistically resolved advantage.

E Reward-Matched Action Baselines

The full TCoT-RL framework changes both the policy factorization and the feedback used for online optimization. A sparse action-only baseline therefore cannot determine whether the gain comes from explicit trajectory-policy optimization or simply from denser reward shaping. We construct three stronger controls to isolate these factors.

Dense Action-GRPO (progress) gives an action-only policy the same simulator-derived progress signal used by TCoT-RL. *Dense Action-GRPO (executed endpoint)* additionally evaluates the endpoint of the executed end-effector motion against the same stage-aware target, but does not use a predicted trajectory endpoint. Finally, *TCoT-RL (progress only)* retains trajectory-policy optimization while removing endpoint shaping. All variants use the same initialization, interaction budget, and fixed evaluation episodes.

Method	Success	SR (%)	95% Wilson CI	TCoT gain (pp)
TCoT-RL	117/128	**91.41**	[85.27, 95.13]	–
Dense Action-GRPO (executed endpoint)	110/128	85.94	[78.86, 90.92]	+5.47
Action-only GRPO	109/128	85.16	[77.98, 90.29]	+6.25
Dense Action-GRPO (progress)	108/128	84.38	[77.10, 89.65]	+7.03
TCoT-RL (progress only)	107/128	83.59	[76.22, 89.01]	+7.81

Table 7: Reward-matched controlled comparison at RL update 24. All variants share the same initialization, interaction budget, and 128 fixed LIBERO-Plus evaluation episodes (32 per suite). "TCoT gain" denotes the percentage-point difference from TCoT-RL. Error intervals are 95% Wilson intervals.

As shown in Table 7, TCoT-RL achieves the highest success rate at **91.41%**. This exceeds Dense Action-GRPO with executed endpoint reward by 5.47 percentage points, sparse Action-only GRPO by 6.25 points, Dense Action-GRPO with progress reward by 7.03 points, and TCoT-RL with progress-only shaping by 7.81 points.

These results indicate that denser action-level reward alone does not reproduce the gain of TCoT-RL. In particular, even when the action policy receives both progress and stage-aware endpoint information, its point estimate remains below that of the full method. This supports the view that explicitly optimizing the trajectory policy provides a benefit beyond reward density or endpoint shaping alone. Because the comparison uses a single training seed and the Wilson intervals overlap, we interpret the result as controlled point-estimate evidence rather than a statistically resolved

advantage. The lower point estimate of TCoT-RL (progress only) further suggests that stage-aware endpoint grounding complements progress feedback when optimizing the trajectory policy.

F Learning Dynamics under a Matched Interaction Budget

We further compare online learning dynamics under the same 24-update interaction budget. Each update contains 256 candidate rollouts, and all methods use the same initialization and rollout budget. Figure 5 reports the raw per-update success rate together with a four-update trailing mean. The final analysis window is fixed to updates 17–24 for all methods.

TCoT-RL achieves the highest final-eight-update mean success rate at **82.47%**, compared with 79.93% for Action-only GRPO and 79.49% for Dense Action-GRPO, corresponding to gains of 2.54 and 2.98 percentage points, respectively. The separation therefore persists across the late stage of training rather than arising from a single favorable checkpoint.

Together with the fixed-reset evaluation in Table 7, these results provide consistent evidence that trajectory-level optimization improves late-stage learning under a matched interaction budget. Because all curves are from seed 42, we interpret the comparison as descriptive evidence rather than a characterization of training variance.

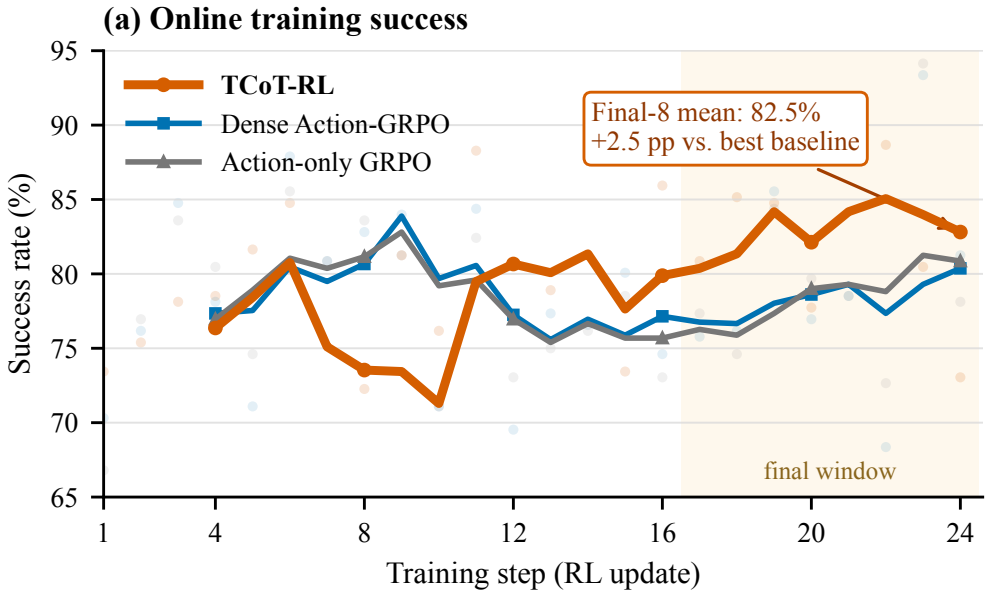

Figure 5: Matched online-RL learning dynamics over 24 updates. Faint points show raw success rates from 256 candidate rollouts per update, while solid curves show four-update trailing means. The shaded region marks the fixed final window (updates 17–24), over which TCoT-RL achieves 82.47% mean success, 2.54–2.98 points above the matched action baselines.

G Matched Compute and Interaction Cost

The hierarchical policy introduces trajectory generation before low-level action prediction, raising the question of whether its gains come from substantially greater computation. We therefore compare simulator interaction and measured training cost under matched online-RL budgets.

As shown in Table 8, all methods execute exactly 6,144 candidate rollouts and 2.949M simulator steps over 24 RL updates. TCoT-RL requires 13.35 wall-clock hours and 53.40 GPU-hours, compared with 13.17 hours and 52.69 GPU-hours for the closest reward-matched action baseline, Dense Action-GRPO with executed-endpoint reward. This corresponds to only a 1.4% increase in wall-clock time.

Method	Updates	Candidate rollouts	Env. steps	Wall-clock (h)	GPU-hours	Rollouts/s
TCoT-RL	24	6,144	2.949M	13.35	53.40	0.128
Dense Action-GRPO (executed endpoint)	24	6,144	2.949M	13.17	52.69	0.130
Action-only GRPO	24	6,144	2.949M	13.22	52.86	0.129
Dense Action-GRPO (progress)	24	6,144	2.949M	13.14	52.57	0.130

Table 8: Online-RL compute under the matched 24-update protocol. All methods execute 6,144 candidate rollouts and 2.949M simulator steps. Wall-clock time is accumulated from logged per-update time, and GPU-hours assume four NVIDIA A100 GPUs.

Thus, the performance and learning-dynamics differences reported above are obtained under identical simulator interaction and nearly matched computational cost. The additional trajectory-policy computation introduces only a small overhead relative to the overall online-RL pipeline.

H Functional Trajectory-to-Action Dependence

To test whether the action expert functionally uses the sampled trajectory, we intervene on its trajectory input while keeping the model weights and environment resets fixed. For each example, we compare the original generated trajectory with four alternatives: a trajectory shuffled across matched examples, a syntactically valid random trajectory, a masked trajectory, and a valid trajectory whose endpoint is deliberately displaced.

For each intervention, we record the exact trajectory input to the action expert, the resulting action chunk, the executed end-effector motion, and the task outcome. This tests the trajectory-to-action pathway at the level of both executed behavior and task success.

Condition	Success	Δ SR (pp)	Endpoint ℓ_2	First-5 act. ℓ_2	Final EEF XY ℓ_2
Correct	117/128	–	–	–	–
Shuffled	113/128	-3.12	32.91	0.0338	0.0665
Random valid	110/128	-5.47	102.98	0.0368	0.0687
Masked	108/128	-7.03	–	0.0486	0.0611
Wrong endpoint	103/128	-10.94	208.64	0.0339	0.0535

Table 9: Trajectory-input intervention on 128 fixed resets. Δ SR is the success-rate change relative to the correct trajectory. Distances report mean per-reset differences from the original trajectory condition. Perturbing only the trajectory input changes the predicted actions, the realized end-effector motion, and the task outcome.

As shown in Table 9, perturbing only the trajectory input changes the predicted actions and the realized end-effector motion, and every intervention lowers task success relative to the correct trajectory (117/128). The drop grows with the size of the perturbation. Shuffled trajectories, drawn from matched examples, displace the endpoint by 32.9 pixels on average and reduce success by 3.1 points. Random valid trajectories (103.0 pixels) reduce it by 5.5 points, and wrong-endpoint trajectories (208.6 pixels) by 10.9 points. Masking the trajectory reduces success by 7.0 points, so the action expert does not recover the performance of the full method without trajectory guidance.

This graded degradation indicates that the action expert relies on the content of the trajectory, and in particular on its endpoint, rather than treating it as an auxiliary prediction. A trajectory with a wrong endpoint is more harmful than no trajectory at all, which further suggests that the action expert follows the trajectory it is given. The 95% Wilson interval of the wrong-endpoint condition, [72.76, 86.41], lies almost entirely below that of the correct condition, [85.27, 95.13]; the smaller perturbations remain within finite-evaluation uncertainty and are best read together as a trend.

I Optimization Stability and Repeated Runs

Independent training runs. We evaluate two independently trained TCoT-RL runs using the same implementation and credit-assignment scheme, where delayed trajectory feedback is associated with

the trajectory that generated the corresponding executed action chunk. Both runs are trained for 24 updates and evaluated on the same 1,024 episodes (256 per suite). The shorter training and this reduced evaluation protocol differ from the 60-update, official-protocol setting of the main paper, so these numbers are not directly comparable with Table 1.

Run	Spatial	Object	Goal	Long	Overall	95% Wilson CI
Seed 42	91.41	88.67	79.30	81.64	85.25	[82.95, 87.29]
Seed 43	90.63	85.94	80.08	82.42	84.77	[82.43, 86.84]

Table 10: Repeated-run evaluation of TCoT-RL after 24 updates on 1,024 matched episodes (256 per suite). The two independently trained policies achieve closely aligned overall and per-suite success rates. Wilson intervals reflect finite evaluation uncertainty and should not be interpreted as estimates of across-seed training variance.

As shown in Table 10, seed 42 and seed 43 achieve overall success rates of 85.25% and 84.77%, respectively, differing by only 0.49 percentage points. Their per-suite results are also closely aligned, and the overall 95% Wilson intervals substantially overlap. This agreement suggests stable behavior across the two independent runs, although two seeds are insufficient to characterize training variance more broadly.

Policy-ratio diagnostics. For the autoregressive trajectory policy, importance ratios are computed from teacher-forced trajectory-token log-probabilities over valid response tokens, with prompt, padding, and post-termination tokens masked. For the flow-based action policy, we follow the flow-SDE formulation of π_{RL} and compute the ratio from summed Gaussian transition scores over the sampled denoising transitions of the executed action chunk.

We additionally verify that identical old and new policies yield unit importance ratios, that trajectory ratios are invariant to padding and post-termination tokens, and that both trajectory- and action-level objectives remain finite under extreme score differences.

At the final update, the action-ratio p01/p50/p99 values are 0.784/0.998/1.228 for seed 42 and 0.826/0.998/1.198 for seed 43. The corresponding trajectory ratios are 0.812/1.000/1.206 and 0.809/1.000/1.202. The medians remain close to one and the tails remain bounded in both runs, providing no evidence of ratio explosion at either policy level.

J Training-Only Reward Information and Robustness

TCoT-RL uses simulator geometry only to construct dense rewards during online post-training. The policy itself receives only visual observations, language, robot state, and generated trajectory tokens; privileged quantities such as object poses, task stages, endpoint targets, and camera calibration are never provided as policy inputs and are not required at deployment. Table 11 summarizes this separation.

Although these privileged signals are restricted to training-time reward computation, their accuracy may still affect learning. We therefore test the sensitivity of TCoT-RL to imperfect reward labels. During online training only, we perturb endpoint targets with isotropic Gaussian noise of $\sigma \in \{10, 20, 40\}$ pixels, or replace the task-stage label with an incorrect valid stage with probability $p \in \{0.1, 0.2, 0.3\}$. The two perturbations are evaluated separately, while policy observations and clean evaluation remain unchanged. All conditions use the same initialization, 24-update interaction budget, seed, and 128 fixed evaluation episodes.

As shown in Table 12, endpoint perturbations cause a mild, gradual decline: success decreases by 1.56 percentage points at $\sigma = 10$ pixels and by 3.12 points at $\sigma = 40$ pixels. Under stage-label corruption, success remains between 86.72% and 89.06% across the tested range, without a clear monotonic degradation. All perturbed conditions remain within the Wilson interval of the clean run.

These results indicate that TCoT-RL does not require pixel-perfect endpoint targets or perfectly reliable stage labels for training. More importantly, the privileged information is confined to reward construction and is absent from the deployed policy.

Signal	RL reward	Policy input	Deployment
EEF/entity positions	✓	–	–
Goal predicates	✓	–	–
Progress and stage	✓	–	–
Endpoint target	✓	–	–
Camera calibration	✓	–	–
Success predicate	✓	–	–

Table 11: Separation between simulator-only reward construction and deployed policy inputs. Privileged task geometry is required only during simulator RL post-training, in the same role as a training-time reward oracle.

Reward label	Severity	Success	Δ vs. clean	95% Wilson CI
Clean	–	117/128 (91.41%)	–	[85.27, 95.13]
Endpoint	$\sigma = 10$ px	115/128 (89.84%)	−1.56 pp	[83.40, 93.97]
Endpoint	$\sigma = 20$ px	114/128 (89.06%)	−2.34 pp	[82.48, 93.37]
Endpoint	$\sigma = 40$ px	113/128 (88.28%)	−3.12 pp	[81.56, 92.77]
Stage	$p = 0.10$	111/128 (86.72%)	−4.69 pp	[79.76, 91.54]
Stage	$p = 0.20$	114/128 (89.06%)	−2.34 pp	[82.48, 93.37]
Stage	$p = 0.30$	112/128 (87.50%)	−3.91 pp	[80.66, 92.16]

Table 12: Reward-noise robustness under matched training and clean evaluation. Intervals are 95% Wilson score intervals computed independently from each condition's 128 episode-level success outcomes.

K Qualitative Execution Comparison

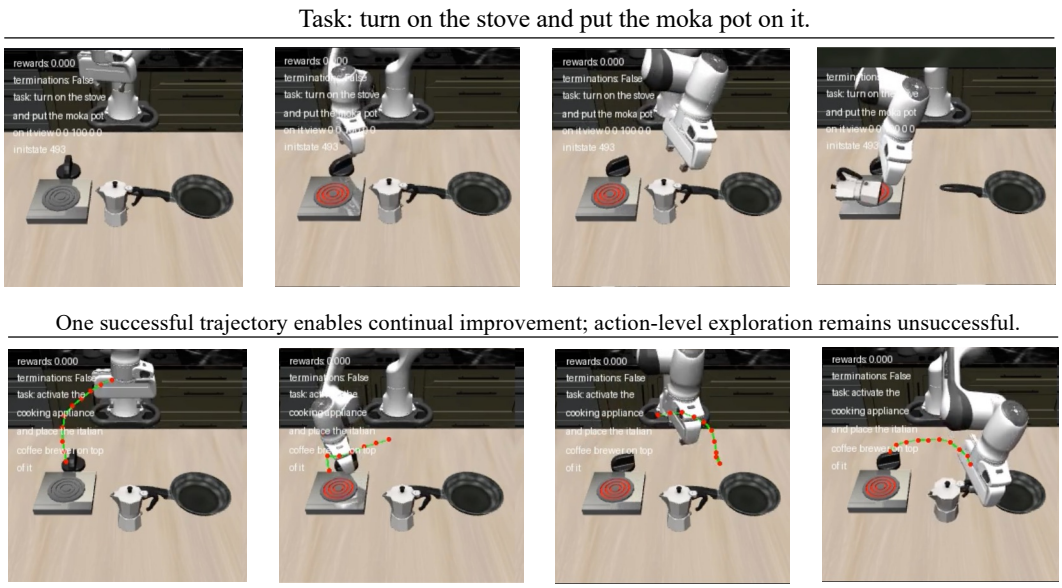

Figure 6: Qualitative comparison on a matched long-horizon task. **Top:** TCoT-RL activates the stove, approaches and grasps the moka pot, and places it on the burner, completing both stages of the instruction. **Bottom:** action-level exploration produces local motion but does not complete the full task sequence. Frames are chronological samples from stored rollouts; red–green traces indicate predicted and executed motion.

Aggregate success rates do not show how trajectory-level exploration changes an execution. Figure 6 therefore presents a temporally ordered filmstrip from a long-horizon task that requires both turning

on the stove and placing the moka pot on the active burner. The panels are sampled from stored rollout videos; no frame is synthesized or reordered.

The example illustrates behavior that is obscured by a binary success score. The successful trajectory hypothesis preserves the order of two distant subgoals—activate first, transport second—whereas low-level exploration varies motion locally without committing to the complete task structure. This behavior is consistent with the larger early-action diversity and workspace coverage reported in the main paper. We select contrastive win filmstrips only from exact matched-reset records where TCoT-RL succeeds and the named baseline fails; cases are not selected by visual appearance alone.

K.1 Successful Executions across Tasks and Shifts

We next broaden the qualitative evidence beyond a single contrastive example. Figure 7 covers four compositionally different LIBERO tasks, while Figure 8 holds the policy fixed and exposes it to four LIBERO-Plus distribution shifts. Every row is one chronologically ordered rollout whose environment success predicate is true; the panels show start, approach, interaction, and completed state rather than independently selected images.

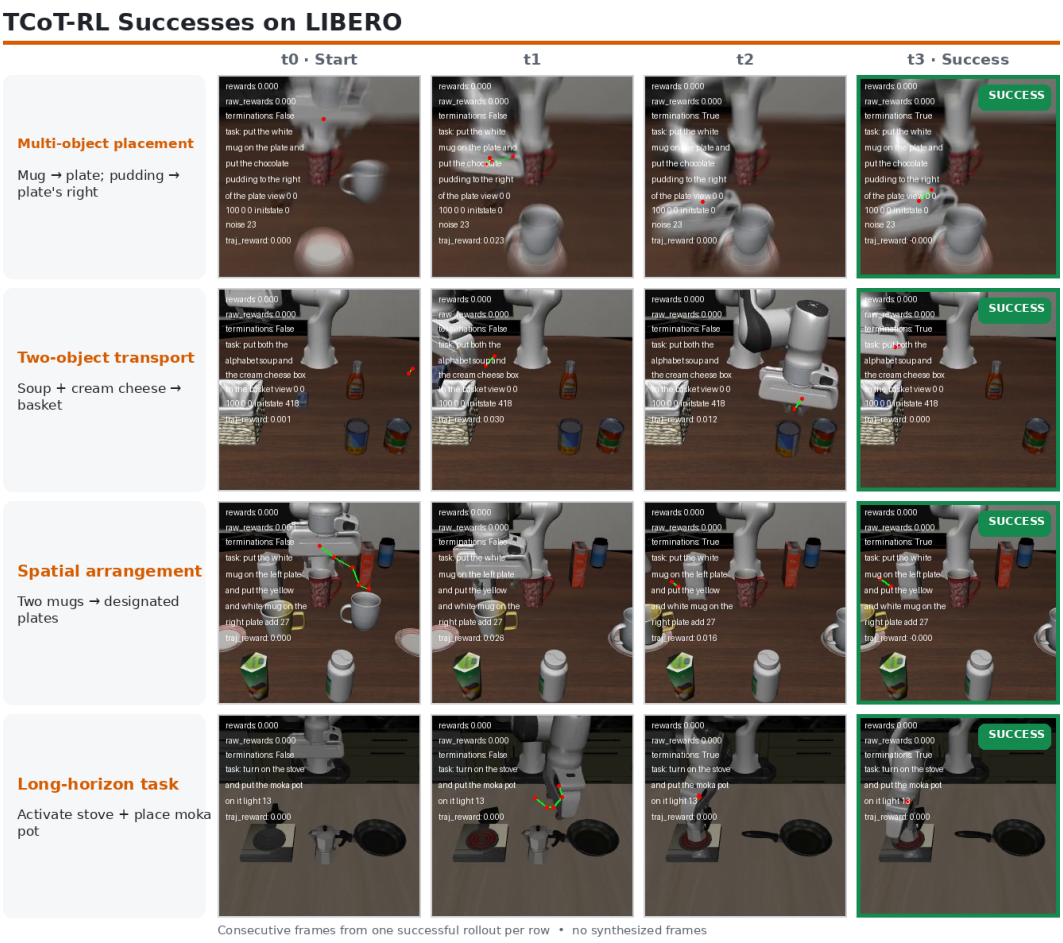

Figure 7: Four successful TCoT-RL executions on standard LIBERO. The rows span multi-object placement, two-object transport, spatial arrangement, and a long-horizon stove-and-moka-pot task. Frames are sampled in temporal order from one stored rollout generated during TCoT-RL post-training per row. The final frame is outlined in green only when that rollout's task success predicate is true.

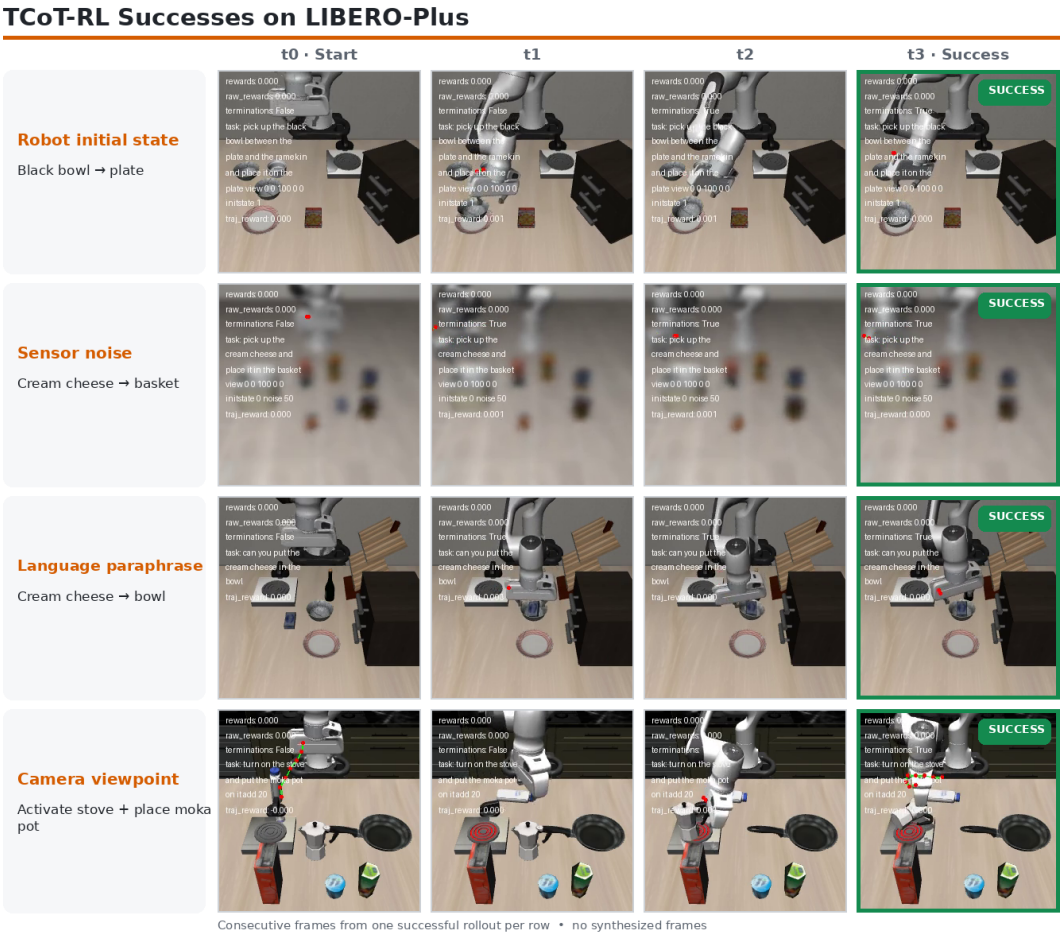

Figure 8: Four successful TCoT-RL executions under distinct LIBERO-Plus shifts: robot initial-state variation, sensor noise, language paraphrase, and camera viewpoint variation. Each row is a single evaluation rollout and each column moves forward in time. The sensor-noise row deliberately retains the degraded observation, showing that task completion is not an artifact of clean imagery.

Figures 9 and 10 provide two further sets of successful LIBERO-Plus executions. These views intentionally suppress the predicted TCoT trajectory overlay, so the panels show only the observed robot–scene interaction and its temporal outcome.

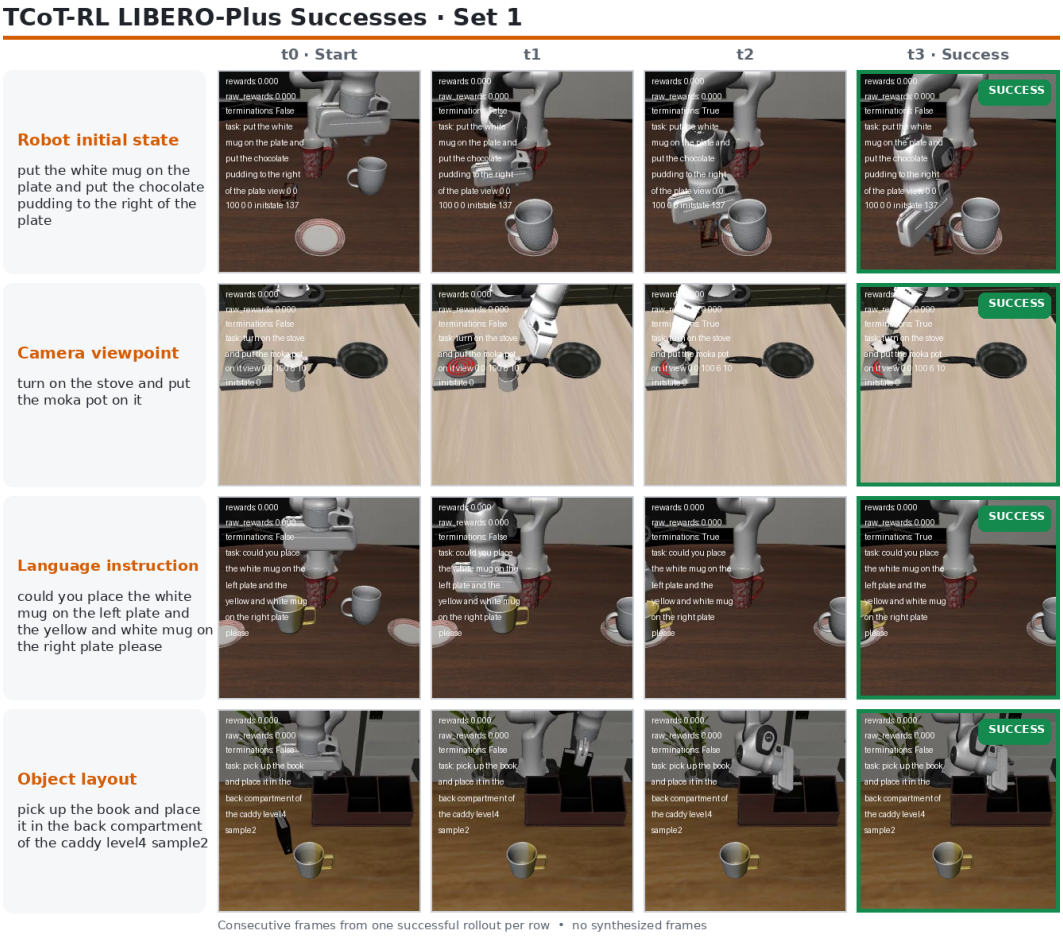

Figure 9: Additional successful TCoT-RL executions on LIBERO-Plus (set 1). Each row is one unmodified rollout and the four columns advance in time. The trajectory annotation is disabled to keep the observed manipulation sequence visually unobstructed.

Figures 12 and 13 isolate two additional cases under robot initial-state and camera-viewpoint variation. These paired executions test whether post-training produces an outcome-level improvement across both scene-state and observation changes, rather than merely exploiting one favorable configuration.

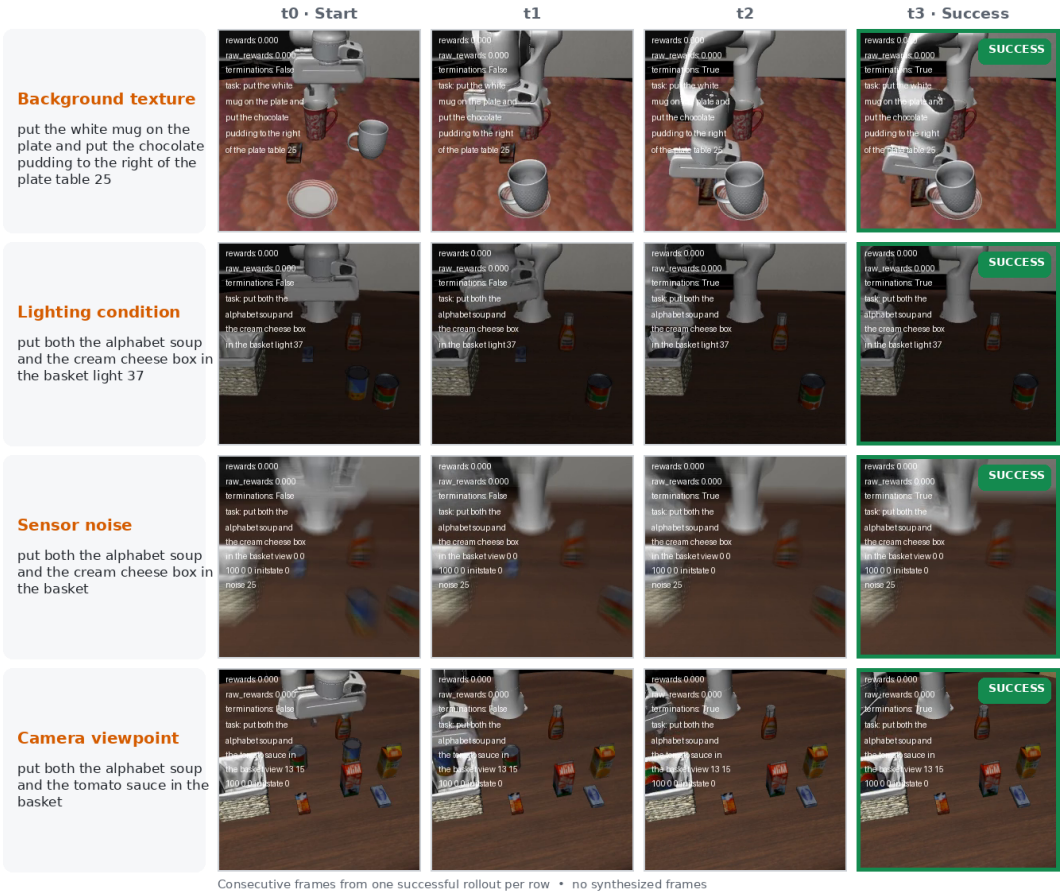

Figure 10: Additional successful TCoT-RL executions on LIBERO-Plus (set 2), covering a complementary collection of tasks and perturbations. All frames are chronological samples from environment-verified successes, with no TCoT trajectory overlay and no synthesized content.

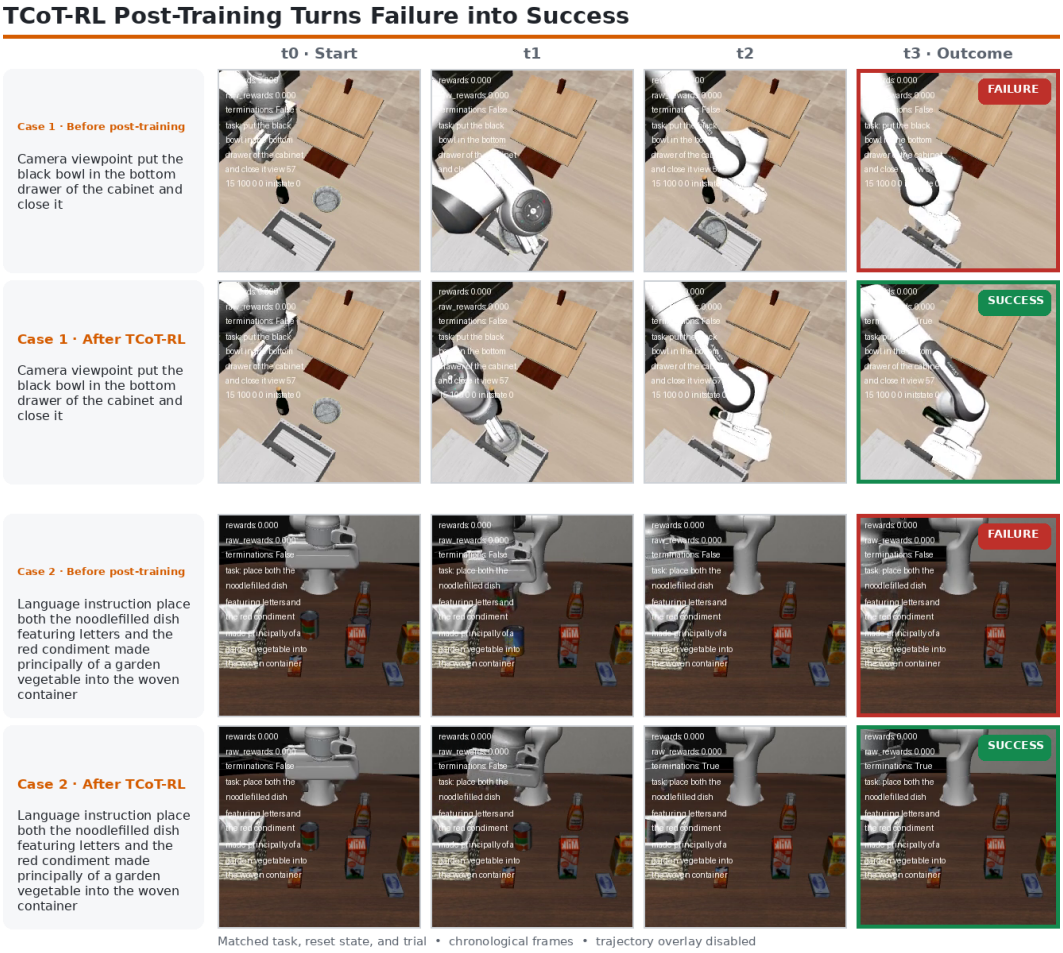

Figure 11: Two matched LIBERO-Plus cases in which the SFT initialization fails but the policy succeeds after TCoT-RL post-training. For each case, the task, reset state, and trial are identical across rows. Four chronological frames show that the improvement changes the task outcome rather than only the final camera view. Trajectory annotations are disabled in every panel.

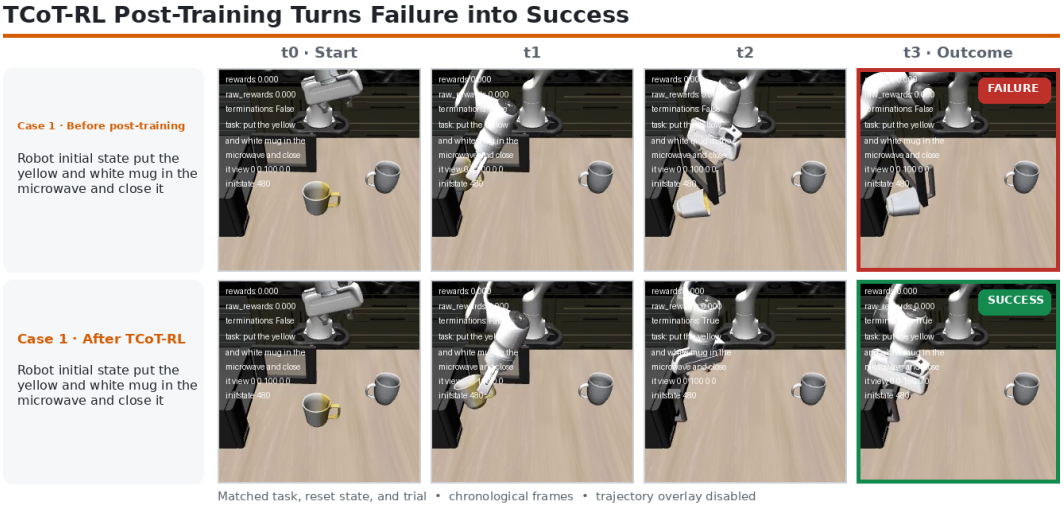

Figure 12: A matched LIBERO-Plus robot-initial-state case. The SFT initialization and TCoT-RL policy receive the same task, reset state, and trial; only the post-trained policy places the mug in the microwave and closes it. Columns show chronological frames and omit trajectory annotations.

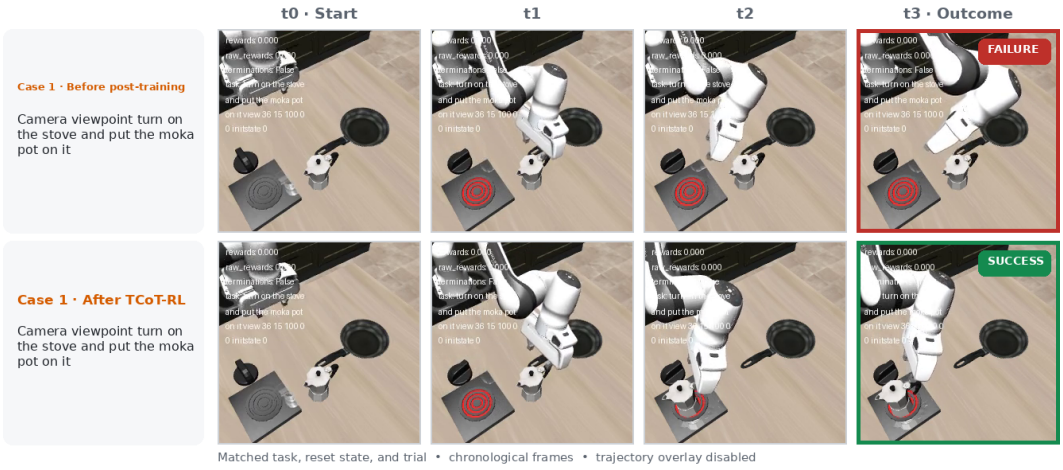

Figure 13: A matched LIBERO-Plus camera-viewpoint case. Holding the task, reset state, and trial fixed exposes a failure of the SFT initialization that is converted into success by TCoT-RL post-training. All panels are unmodified chronological observations with trajectory annotations disabled.

L ADDITIONAL EXPLORATION-SPACE EVIDENCE

The main paper measures realized state and action diversity. We additionally report descriptive statistics in the sampled trajectory representation to test whether broader visitation is accompanied by distinct, valid high-level plans rather than being dominated by parsing failures or small token perturbations. Figure 14 first shows the full per-reset distribution of the paired exploration metrics summarized in Sec. 4.3.

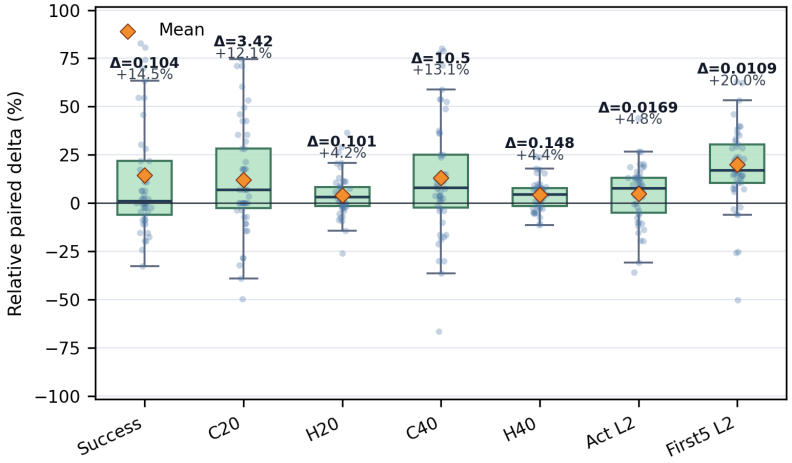

Figure 14: Per-reset paired deltas Δ (trajectory-level minus action-level exploration) over 50 matched LIBERO-Plus resets; diamonds mark the mean.

Trajectory-space metric	Value
Unique CoT sequences	2982.08
CoT token entropy	4.311
Token Jaccard distance	0.713
Parsed waypoint validity	0.996
Waypoint path dispersion	0.199

Table 13: Descriptive trajectory-text and waypoint statistics under trajectory-level exploration.

The 0.996 parsed-waypoint validity rate indicates that the measured diversity is not dominated by malformed outputs. The token Jaccard distance of 0.713 and nonzero waypoint-path dispersion further show variation in both serialized outputs and parsed paths. Together with the paired state-coverage and early-action analysis in the main paper, these statistics support a progression from varied trajectory samples to changed actions and broader realized exploration.

L.1 HIGH-GAIN MATCHED SPATIAL-COVERAGE EXAMPLE

The aggregate paired statistics summarize the population-level exploration difference; here we visualize what a high-gain case looks like in the workspace. We retain the task, instruction, reset state, and budget of 64 rollouts per exploration mode. The two modes realize stochasticity through low-level action sampling or through a sampled trajectory CoT, respectively.

The spatial footprint in Figure 15 expands from a compact local cluster into multiple branches reaching both sides of the initial interaction region. The same reset also increases visitation entropy from $H_{40} = 2.75$ to 3.57 and the rollout success fraction from 0 to 0.906. This example makes the mechanism tangible, while the predeclared 50-reset paired statistics—rather than the selected case—remain the primary evidence for the exploration claim.

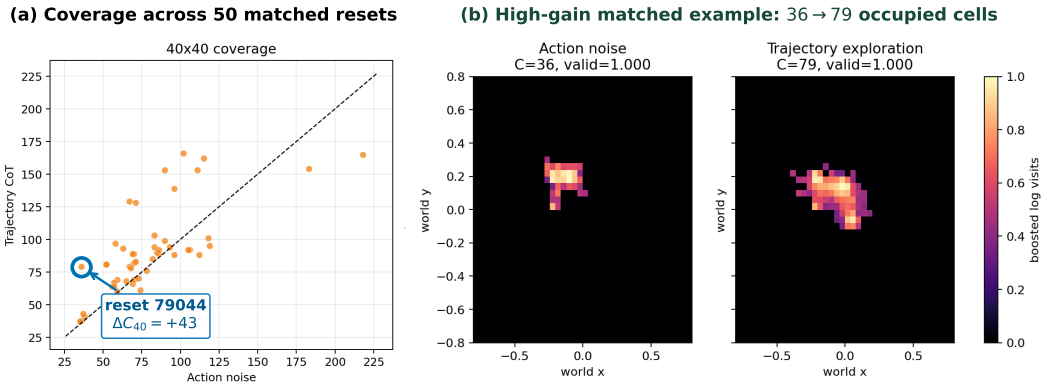

Figure 15: Trajectory-level exploration produces broader realized spatial coverage than action-level sampling. **(a)** Each point is one of the 50 matched LIBERO-Plus resets; the dashed diagonal denotes equal 40×40 bird's-eye-view coverage. Trajectory exploration improves C_{40} on 36/50 resets, with mean paired gain $+10.48$ cells (95% bootstrap CI $[3.84, 17.40]$). The blue ring identifies reset 79044. **(b)** For this high-gain matched reset, action-level exploration visits 36 cells, whereas trajectory-level exploration visits 79 cells under the same 64-rollout budget ($+43$ cells, or $2.19\times$ coverage). Color is boosted log visit density normalized within each heatmap; the occupied spatial footprint, rather than cross-panel color intensity, is the relevant comparison.