

TGRPO: Fine-tuning Vision-Language-Action Model via Trajectory-wise Group Relative Policy Optimization

Zengjue Chen¹, Runliang Niu¹, He Kong¹, Qi Wang^{1*}, Qianli Xing¹, Zipei Fan¹

Abstract—Visual-Language-Action (VLA) models have demonstrated strong cross-scenario generalization capabilities in various robotic tasks through large-scale pre-training and task-specific fine-tuning. However, their training paradigm mainly relies on manually collected successful demonstrations, making it difficult to adapt to complex environments when encountering out-of-distribution (OOD) scenarios or execution biases. While Reinforcement Learning (RL) provides a closed-loop optimization framework via active trial-and-error mechanism, it suffers from sparse rewards, high variance, and unstable optimization in long-horizon robotic tasks. To address these limitations, we propose Trajectory-based Group Relative Policy Optimization (TGRPO), an online RL-based training framework for VLA models. TGRPO leverages task analysis generated by a large language model to automatically construct dense reward functions, providing fine-grained feedback to accelerate convergence and improve credit assignment. The core of our method is a group-based strategy that samples and normalizes multiple trajectories in parallel, reducing variance through relative comparison. By integrating trajectory-level and step-level advantage estimation, TGRPO captures both global and local optimization signals without relying on a value network. Experiments on four task categories of the LIBERO benchmark demonstrate that TGRPO achieves an average success rate of 80.7%, which is 4.2% higher than that of Supervised Fine-Tuning (SFT) and outperforms other representative RL-based post-training methods.

I. INTRODUCTION

Vision-Language-Action (VLA) [1], [3], [4], [12], [16], [17], [22], [39]–[50] models have gradually emerged as a crucial approach to achieve general robotic policies, exhibiting robust performance in robotic tasks across diverse scenario [6], [15], [30], [31]. Despite their successes, VLAs are trained solely on human-provided successful demonstrations and lacks the ability to learn from failures. As a result, it shows limited flexibility in real-world interactions, often remaining at the "action memorization" level rather than aligning actions with environmental context [32]. This not only restricts its capacity for autonomous exploration and self-correction, but also limits its performance in out-of-distribution (OOD) tasks and scenarios.

Reinforcement learning (RL) has been widely regarded as a natural and effective training paradigm that enables agents to optimize policies through interaction with the environment and to achieve autonomous behaviors in complex tasks [24]–[26], [28], [29]. A large body of work has successfully applied policy gradient methods, such as Proximal Policy Optimization (PPO) [9] and Soft Actor-Critic (SAC) [10], as

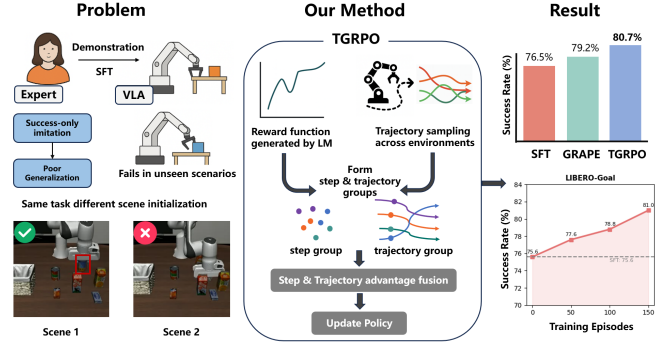


Fig. 1. Problem: VLA models trained by SFT generalize poorly and fail in unseen scenarios. Method: TGRPO uses LLM-generated rewards and step–trajectory group fusion for stable policy learning. Result: On the LIBERO benchmark, TGRPO outperforms SFT and GRAPE.

well as value-based approaches to tackle robotic tasks, including motion control, grasping, and manipulation, demonstrating impressive performance in both simulated and real-world environments. However, PPO requires simultaneously training a value function network (Critic) to estimate state values and balances bias and variance through Generalized Advantage Estimation (GAE) [33], which can hinder training efficiency and effectiveness. At the same time, reward signals in real-world robotic tasks are often highly sparse, frequently reduced to binary success/failure feedback at the episode level, making exploration and optimization challenging.

While researchers commonly adopt reward shaping or hand-crafted dense rewards to address this issue, such methods demand significant domain expertise and risk introducing reward bias, potentially diverting learning from the true task objective. Even with dense rewards, long-horizon robotic tasks pose further challenges: they involve multiple stages with diverse dynamics, where inconsistent reward scaling across subgoals can amplify gradient variance, slow convergence, destabilize training, and trap policies in suboptimal local minima. Agents may even overfit intermediate subgoals with disproportionately large rewards while neglecting overall task success. Thus, effectively balancing reward distributions across stages and reducing training variance remain key bottlenecks for scaling RL in robotic learning.

In this paper, we propose Trajectory-wise Group Relative Policy Optimization (TGRPO), a critic-free training algorithm for Vision-Language-Action (VLA) models. Specifically, TGRPO leverages a large language model (LLM) to extract task-relevant object information from the environment and combines it with end-effector pose data from demonstra-

¹ School of Artificial Intelligence, Jilin University, Changchun, China.

*Correspondence: qiwang@jlu.edu.cn

tions to construct a dense, multi-stage reward function. Using this reward, the VLA model controls robots in parallel environments to sample diverse trajectories. These trajectories are then organized into groups at both the trajectory level and the step level: entire trajectories are grouped to capture global performance, while steps at the same timestep across trajectories are grouped to capture local consistency. Relative advantages are computed within each group and fused across levels, integrating local and global optimization signals for stable policy updates. The overall problem setting, proposed method, and experimental results are illustrated in Fig. 1. Experimental results show that TGRPO achieves superior success rate against both SFT fine-tuning and RL-based approaches for VLA models, demonstrating its effectiveness and generalization capability.

The main contributions are summarized as follows: (1) We introduce an online RL-based training framework for VLA models, addressing the fundamental limitation of training only from successful demonstrations and enabling VLA agents to actively explore and learn from interactions with the environment. This suits adaptive policy learning in dynamic robotic task scenario; (2) We design the TGRPO algorithm, a novel group-based policy optimization method integrating trajectory-level and step-level advantage estimation. Tailored to robot long-horizon multi-stage tasks, this integration fuses global and local optimization signals, avoiding single-level advantage estimation’s flaw in balancing global-task and local-action optimization; (3) We highlight the synergy of dense reward design and group-based optimization for robotic RL. Leveraging LLMs to parse tasks and generate stage-wise dense rewards, combined with the group-based strategy, TGRPO substantially improve the effectiveness of RL training for VLA models in complex robotic tasks.

II. RELATED WORK

Vision-Language-Action Models. Existing VLA models can be mainly categorized into hierarchical robotic models and end-to-end robotic models. Hierarchical models, such as Rekep [18], VoxPoser [14], Omnimanip [23] and Hi Robot [13], advocate dividing robotic models into high-level policies and low-level policies. The high-level policy handles understanding and reasoning about complex tasks, decomposing them into executable instructions for low-level policies, which then translate these instructions into corresponding robotic actions. In contrast, VLA model more commonly refers to end-to-end robotic models. Classic end-to-end VLA models like PaLM-E [17], RT-2 [16], and OpenVLA [1] typically use autoregressive large language models as their backbone, outputting discrete action tokens. OpenVLA employs SigLip [5] and DINOv2 [2] as its vision encoders and Llama2-7B [35] as its backbone. Through large-scale pre-training on the OXE open-source robotic dataset [6], the model gains the ability to comprehend both the current scene images and task-specific language instructions, subsequently generating corresponding action tokens as outputs. Recently, diffusion policy [37] is integrated into VLAs, such as π_0 [4], GO-1 [15], and RDT-1B [12]. The π_0

architecture builds on a pre-trained vision-language model (VLM) backbone (e.g., PaliGemma [36]) augmented with an "action expert" that uses flow matching to generate continuous action distributions for dexterous robotic control. Despite these promising performance, VLA models exhibit limitations when applied to out-of-domain tasks.

Reinforcement Learning for VLA Fine-Tuning. Due to the limitations of the aforementioned works, many studies have gradually turned their attention to using reinforcement learning to fine-tune VLA. A recent work proposed a VLA model fine-tuning method called ConRFT [21], which combines offline behavior cloning, Q-learning, and online consistency policy training, and introduces human intervention to improve exploration efficiency and safety. Another study introduced the iRe-VLA framework [20], which iterates between reinforcement learning and supervised learning to enhance the performance of the VLA model. In addition, GRAPE [19] was proposed, which implicitly models rewards for successful and failed trials at the trajectory level to improve the generalization ability of the VLA model for new tasks. At the same time, it splits complex operation tasks into several stages and uses key points generated by large vision-language models and customized spatio-temporal constraints to automatically guide preference modeling. Different from these works, our approach does not require excessive human intervention, nor does it need to accumulate a large amount of offline datasets. It focuses more on online training with the robot, and the training time is significantly reduced.

III. PROBLEM FORMULATION

In this paper, the policy optimization of VLA model π_θ is achieved by the online reinforcement learning. Unlike supervised fine-tuning (SFT), which relies solely on offline successful demonstrations, our work focuses on optimizing policies through direct interaction with the environment, thereby equipping VLA with the ability to learn from failures and self-correct. Formally, the task is modeled as a Markov decision process, defined by the tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \mathcal{V}, \mathcal{L})$, where \mathcal{S} is the state space and \mathcal{A} is the action space. \mathcal{P} is a transition function. \mathcal{R} is the reward function. \mathcal{V} is the observation space captured from a third-person camera. \mathcal{L} is a set of natural language instruction. In each timestep, the VLA model takes action $a \in \mathcal{A}$ based on the observation $v \in \mathcal{V}$ and corresponding $l \in \mathcal{L}$. Then, the VLA model will receive the reward r . In this paper, the reward is dense reward provided by an extra LLM, in which the reward function produces the reward at each timestep t based on the positions of key objects provided by the simulation environment and the robot’s end-effector pose, i.e., the robot’s state s_t . The overall objective of our policy is to maximize the cumulative trajectory reward $R = \sum_{t=1}^M r_t$, where M denotes the shortest length among multiple trajectories.

IV. METHODOLOGY

To better capture the characteristic of long-horizon robotic tasks, we propose Trajectory-wise Group Relative Policy Optimization (TGRPO), an online reinforcement learning

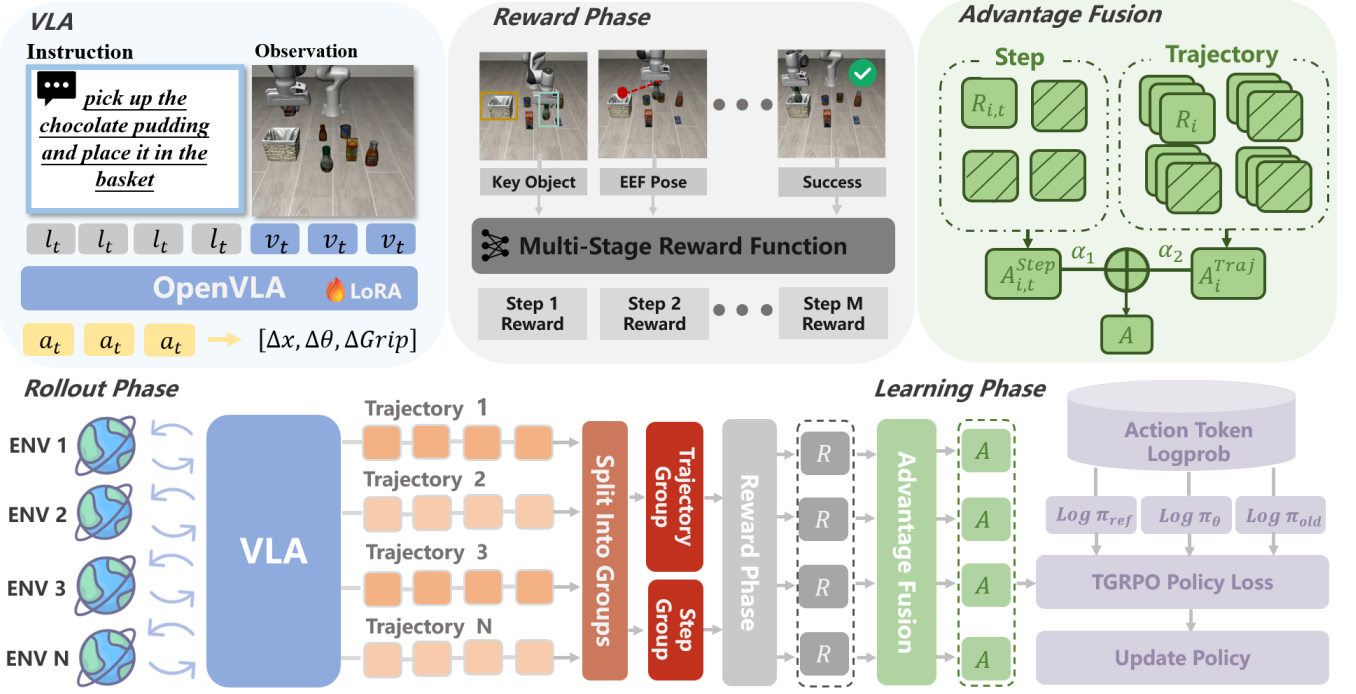


Fig. 2. **Overview of the proposed Trajectory-wise Group Relative Policy Optimization.** Given natural language instructions and multimodal observations, the OpenVLA model produces action tokens to control the robot. Trajectories sampled in parallel across environments are evaluated with a multi-stage reward function, grouped for step-level and trajectory-level advantage estimation, and fused to update the policy.

framework for post-training VLA fine-tuning. In concrete, TGRPO performs trajectory sampling across multiple environments initialized with identical states. During sampling, a multi-stage reward function designed by a large language model provides step-wise dense rewards. The collected trajectories are then organized into step-level and trajectory-level groups, where relative advantages are computed and fused to combine local and global optimization signals. The final loss is used to update the VLA policy effectively. Fig. 2 shows the framework of our TGRPO.

A. Multi-Stage Reward Design

In our TGRPO framework, step-level advantages are computed before being fused with trajectory-level estimates. Simply propagating a terminal binary reward to all steps neglects the heterogeneous contributions of different actions. For example, in a failed trajectory, early steps may still achieve several sub-goals. Thus, we introduce a multi-stage reward function that provides fine-grained signals

We leverage Claude 3.7 Sonnet to automatically decompose tasks into sub-stages from natural-language descriptions. For example, in the LIBERO-Object task of placing a tomato sauce bottle into a basket, the stages include approaching, grasping, moving, and placing. The LLM defines stage-wise rewards using spatial information of key objects and identifies the current stage at runtime. To further enhance efficiency, we add a shaping signal based on the end-effector pose: pre-collected successful poses are used as references, and the Euclidean distance to them provides additional guidance, encouraging alignment with expert demonstrations

and accelerating task completion.

We first define the positions of the key objects in the task as $P_{object} \in \mathbb{R}^3$, where the subscript *object* denotes the name of the task-relevant item. For example, in the task of placing a tomato sauce bottle into a basket, we have both P_{tomato} and P_{basket} . Next, we define the set of reference key poses obtained from successful demonstrations as $\{P_{pose}^1, P_{pose}^2, \dots, P_{pose}^j\}$, $P_{pose}^k \in \mathbb{R}^3$. And each $P_{pose}^k = (x, y, z)$ represents the 3D Cartesian position of the robot end-effector at a critical step of the task, and j denotes the total number of key poses. Based on the task description, together with the set of reference poses P_{pose}^k and the task-relevant object positions P_{object} , the LLM automatically generates a multi-stage reward function. Therefore, the reward for each step in the trajectory should be defined as

$$R_t = f_1(P_{object}(t), P_{pose}^k) + f_2(P_{pose}^k, s_t). \quad (1)$$

Here, $f_1(\cdot)$ and $f_2(\cdot)$, respectively denote the reward function based on key object information and the reward function based on key successful demonstration poses.

B. Trajectory-wise Group Relative Policy Optimization

With the multi-stage reward function, we are able to assign meaningful reward signals to each step within a trajectory, which facilitates subsequent policy optimization. Our optimization approach extends the group-based strategy by refining both the grouping procedure and the computation of advantages in the final loss. Specifically, we incorporate trajectory-level and step-level advantage estimation to better capture the structure of long-horizon robotic tasks, thereby

TABLE I
RESULTS ON LIBERO (MEAN SUCCESS RATE %, RANK). BEST RESULTS ARE INDICATED IN BOLD.

Method	LIBERO-Spatial		LIBERO-Object		LIBERO-Goal		LIBERO-Long		Average	
	Success Rate	Rank	Success Rate	Rank	Success Rate	Rank	Success Rate	Rank	Success Rate	Rank
Octo	77.6	5	84.9	5	82.9	2	50.3	5	73.9	4.25
OpenVLA-SFT	84.7	3	88.4	4	79.2	5	51.1	4	76.5	4
OpenVLA-DPO	84.2	4	88.6	3	79.5	4	52.6	3	76.2	3.5
GRAPE	88.5	2	92.1	2	83.1	1	57.2	2	80.2	1.75
TGRPO (Ours)	90.4	1	92.2	1	81	3	59.2	1	80.7	1.5

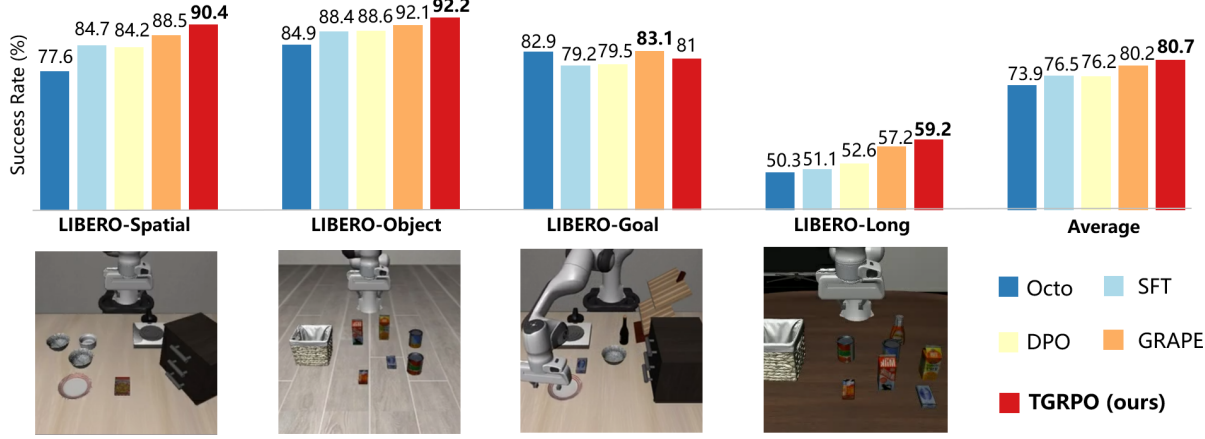


Fig. 3. **Success rates of different methods on the LIBERO benchmark suites.** Our proposed TGRPO consistently outperforms baseline methods (Octo, SFT, DPO, GRAPE) across all task categories (Spatial, Object, Goal, Long), achieving the highest average success rate of 80.7%.

improving credit assignment and enhancing policy stability in sequential decision-making. Suppose we have N trajectories, each of length M . The reward at step t of trajectory i is denoted as $R_{i,t}$. At this point, the cumulative sum of step rewards within each trajectory represents its total reward: $R_i = \sum_{t=1}^M R_{i,t}$. We group together the steps at the same timestep t from all N trajectories, which allows us to compute the step-level relative advantage as:

$$A_{i,t} = \frac{R_{i,t} - \frac{1}{N} \sum_{i=1}^N R_{i,t}}{\sqrt{\frac{1}{N-1} \sum_{i=1}^N (R_{i,t} - \frac{1}{N} \sum_{i=1}^N R_{i,t})^2}}. \quad (2)$$

Next, we treat the entire set of N trajectories as a single group to compute the relative advantage of each trajectory within the group. Formally, the trajectory-level relative advantage for trajectory i is defined as:

$$A_i = \frac{R_i - \frac{1}{N} \sum_{i=1}^N R_i}{\sqrt{\frac{1}{N-1} \sum_{i=1}^N (R_i - \frac{1}{N} \sum_{i=1}^N R_i)^2}}. \quad (3)$$

Combining the two and weighting them with hyperparameters α_1 and α_2 , we obtain the final relative advantage for each step within every trajectory. Thus, the final relative advantage for step t in trajectory i is given as follows:

$$Adv_{i,t} = \alpha_1 A_{i,t} + \alpha_2 A_i, \quad (4)$$

In our experiments, we empirically set $\alpha_1 = 0.3$ and $\alpha_2 = 0.7$ based on hyperparameter tuning. Finally, we incorporate

this relative advantage into a policy optimization objective. We define the importance sampling ratio between the updated policy π_θ and the old policy $\pi_{\theta_{\text{old}}}$ as:

$$\rho_{i,t} = \frac{\pi_\theta(a_{i,t} | s_{i,t})}{\pi_{\theta_{\text{old}}}(a_{i,t} | s_{i,t})}. \quad (5)$$

The optimization objective follows a clipped surrogate formulation similar to GRPO, incorporating our dual-level relative advantage:

$$\begin{aligned} \mathcal{J}_{\text{TGRPO}}(\theta) = & \mathbb{E}[q \sim P(Q), \{o_i\}_{i=1}^N \sim \pi_{\theta_{\text{old}}}(O | q)] \\ & \cdot \frac{1}{N} \sum_{i=1}^N \frac{1}{|o_i|} \sum_{t=1}^{|o_i|} \left\{ \min \left[\rho_{i,t} Adv_{i,t}, \right. \right. \\ & \left. \left. \text{clip}(\rho_{i,t}, 1 - \epsilon, 1 + \epsilon) Adv_{i,t} \right] \right. \\ & \left. - \beta D_{\text{KL}}[\pi_\theta \| \pi_{\text{ref}}] \right\}. \end{aligned} \quad (6)$$

Here, o denotes a sampled trajectory, while q represents the current task environment, which directly encompasses the sequence of visual observations and the task instruction as input. For the KL divergence term, we adopt an unbiased

estimator:

$$D_{\text{KL}}[\pi_{\theta} \parallel \pi_{\text{ref}}] = \frac{\pi_{\text{ref}}(a_{i,t} \mid s_{i,t})}{\pi_{\theta}(a_{i,t} \mid s_{i,t})} - \log \frac{\pi_{\text{ref}}(a_{i,t} \mid s_{i,t})}{\pi_{\theta}(a_{i,t} \mid s_{i,t})} - 1. \quad (7)$$

In summary, the TGRPO algorithm optimizes the policy by maximizing Eq. (6), thereby driving the continual improvement of the VLA model’s capabilities.

C. VLA Post-training Framework in Simulation

After improving the computation of relative advantages and deriving the corresponding optimization objective, we integrated these components into a complete online reinforcement learning framework for VLA post-training in simulation.

First, our overall framework trains a VLA model for a single task using reinforcement learning across multiple environments initialized with identical states. In this setup, the VLA executes the same task in parallel environments, sampling actions step by step until either one environment completes the task or all environments reach the maximum number of steps. During sampling, we incorporate the multi-stage reward function designed by the LLM described earlier, where each environment’s observations provide the necessary object positions and robot state information required for reward computation. Once the trajectories terminate simultaneously, they all share the same length, which facilitates consistent grouping for subsequent processing.

After collecting multiple trajectories, they are organized into a trajectory-level group, where relative advantages are computed within the group according to Eq. (3), yielding trajectory-level relative advantages. Similarly, since all trajectories terminate at the same timestep (ensuring that every step can be grouped), we extract step-level data across trajectories (e.g., rewards and log probabilities of actions), and group together steps at the same timestep to form step-level groups. Step-level relative advantages are then computed using Eq. (2).

Finally, the trajectory-level and step-level advantages are combined with weights as defined in Eq. (4), and the resulting objective is optimized using the loss function in Eq. (6) to update the model parameters.

V. EXPERIMENTS

A. Tasks and Experimental Setup

In our experiments, we adopt OpenVLA [1] as the base VLA model and fine-tune it with LoRA during reinforcement learning training. All reinforcement learning updates are performed using the AdamW optimizer with a fixed learning rate of 1×10^{-5} . Consistent with the experimental setup in [19], we employ the LIBERO [8] robotics simulator as our testbed. LIBERO provides a clean, Python-style API that grants direct access to object states, robot joint angles, and other scene information, enabling us to conveniently construct dense and informative reward functions. LIBERO is a benchmark for lifelong robot learning comprising four task suites: spatial,

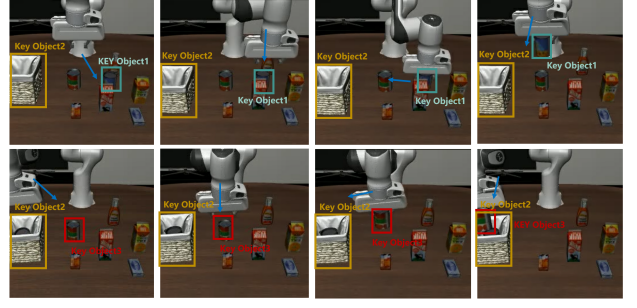


Fig. 4. **An example from LIBERO-Long.** A successful trajectory trained with TGRPO. At each sampling step, rewards are assigned based on the states of key objects and the robot. The illustrated task is "put both the alphabet soup and the tomato sauce in the basket", where the key objects are alphabet soup, tomato sauce, and basket.

object, goal, and long, targeting generalization across spatial locations, object categories, task goals, and long-horizon complexity, respectively. Each suite contains 10 tasks. During training and evaluation, we sample one task at a time, run it in four parallel environments, and evaluate performance over 50 test episodes with average success rates reported per task and per suite. All experiments are conducted on a single NVIDIA A100 GPU.

B. TGRPO Evaluation on LIBERO

The performance of TGRPO on LIBERO. TGRPO is a fine-tuning approach for VLA models. Accordingly, for baseline selection, we first include methods that also follow the fine-tuning paradigm, such as SFT, DPO, and GRAPE. In addition, to further highlight the effectiveness of our method, we incorporate several independent models that are not based on post-training as additional baselines. As shown in Table 1 and Fig. 3, our proposed TGRPO model achieves a significantly higher average success rate across the four LIBERO task suites compared to all baselines, demonstrating its superior performance in fine-tuning VLAs on new tasks. Specifically, TGRPO outperforms standard SFT fine-tuning by 4.2% and also surpasses other reinforcement learning-based methods such as DPO and GRAPE. Notably, on the LIBERO-Long suite that involves long-horizon complex tasks, TGRPO achieves an average success rate that is 8.1% higher than SFT. This result highlights that step-wise reward assignment and integration of trajectory- and step-level signals allow TGRPO to better enable VLA models to explore and learn effectively in challenging long-horizon scenarios.

Ablation study. Table 2 presents ablation results on the LIBERO-Object suite. Compared with the SFT baseline, our TGRPO framework achieves the highest success rate of 92.2%, demonstrating the benefit of integrating both trajectory-level and step-level advantages. Removing trajectory-level advantages leads to a noticeable performance drop, highlighting their importance for capturing global task progress across trajectories. Similarly, removing step-level advantages results in an even larger decrease, indicating that fine-grained step-wise feedback is critical for effective credit

TABLE II
ABLATION STUDIES ON LIBERO-GOAL (SUCCESS RATE %). BEST RESULTS ARE INDICATED IN BOLD.

	Task0	Task1	Task2	Task3	Task4	Task5	Task6	Task7	Task8	Task9	Avg.
Method	SR	SR	SR	SR	SR	SR	SR	SR	SR	SR	SR
SFT	86	76	90	74	92	92	98	92	92	92	88.4
TGRPO w/o Trajectory-level Adv.	88	56	86	60	92	82	92	92	92	60	80.2
TGRPO w/o Step-level Adv.	78	78	98	58	94	82	96	96	92	96	86.8
TGRPO (Ours)	88	82	98	76	98	94	98	98	94	96	92.2

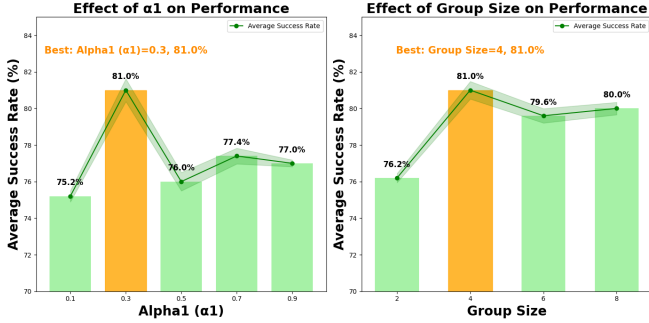


Fig. 5. **Hyperparameter analysis of TGRPO on the LIBERO-Goal suite.** We evaluate the average success rate across different settings of the weighting coefficients (α_1, α_2), which balance step-level and trajectory-level advantages. The best performance is achieved at (0.3, 0.7), yielding an average success rate of 81.0%.

assignment in long-horizon tasks. These results prove that the dual-level advantage design is essential for the stability and effectiveness of TGRPO.

Case Study. To further validate the effectiveness of TGRPO, we present a case study on the LIBERO-Long suite. The selected task, put both the alphabet soup and the tomato sauce in the basket, requires the robot to complete multiple sub-goals in sequence, making it a representative long-horizon challenge task. As shown in Fig. 4, the environment provides key object states (alphabet soup, tomato sauce, and basket) together with the robot’s state, which are used by our multi-stage reward function to assign step-wise feedback. This enables the policy to recognize partial progress such as approaching or grasping the correct object. By integrating both trajectory-level and step-level advantages, TGRPO effectively combines global task progress with fine-grained corrective signals.

C. Hyperparameter Analysis of Advantage Weights

In TGRPO, the final relative advantage is defined as a weighted combination of step-level and trajectory-level advantages, parameterized by coefficients α_1 and α_2 with the constraint $\alpha_1 + \alpha_2 = 1$. To examine the impact of these weights, we conducted a set of hyperparameter analysis experiments on the LIBERO-Goal suite, which consists of long-horizon manipulation tasks. The results are showed in Fig. 5. We find that the performance of TGRPO is sensitive to the choice of α_1 . Specifically, setting $\alpha_1 = 0.3$ and $\alpha_2 = 0.7$ yields the highest average success rate of 81.0%, clearly outperforming other configurations. When α_1 is too

small (e.g., 0.1), the model underutilizes step-level advantages and fails to exploit fine-grained reward feedback, leading to weaker credit assignment. On the other hand, when α_1 is too large, trajectory-level signals are overshadowed, reducing stability in long-horizon training. These results highlight that for long-horizon tasks in LIBERO-Goal, trajectory-level advantages serve as the dominant factor in stabilizing policy learning, while step-level advantages complement them by providing detailed guidance. A balanced configuration, $\alpha_1 = 0.3$ and $\alpha_2 = 0.7$, proves to be the most effective setting for maximizing the performance of TGRPO.

D. Impact of Group Size on Policy Performance

The group size N is the number of parallel trajectories sampled and aggregated per update in our framework. Larger groups may reduce variance and stabilize normalization at the cost of increased training time, whereas smaller groups can lead to noisier estimates and potentially degrade performance. To investigate the impact of group size in TGRPO, we evaluate $N \in \{2, 4, 6, 8\}$ on the LIBERO-Goal suite. As shown in Fig. 5, setting group size too small ($N = 2$) leads to poor overall performance, with an average success rate of only 76.2%. This suggests that insufficient trajectories per group fail to provide reliable relative comparisons, resulting in unstable optimization. When the group size increases to $N \geq 4$, the performance differences across configurations become relatively minor. Among these, $N = 4$, achieves the best overall performance with an average success rate of 81.0%, slightly outperforming larger group sizes while requiring fewer trajectories. This finding indicates that a moderate group size $N = 4$ strikes the best balance between accuracy and efficiency, achieving stable performance while reducing training cost and improving the overall efficiency of VLA post-training.

VI. CONCLUSION

In this work, we propose TGRPO, an online RL framework for fine-tuning Vision-Language-Action models that overcomes the limitations of supervised learning by enabling failure-aware policy improvement through environmental interaction. TGRPO uses LLM to automatically generate dense multi-stage rewards and introduces a group-based dual-level advantage estimation mechanism—combining trajectory-level and step-level signals—to enhance credit assignment and training stability without a value network. Experiments on LIBERO show TGRPO achieves an 80.7% average success rate, outperforming SFT and prior RL methods.

Ablations validate the necessity of both advantage levels, and hyperparameter studies confirm the optimal weighting for long-horizon tasks. TGRPO offers a scalable and efficient solution for adaptive VLA fine-tuning, reducing reliance on human engineering. Future work will extend TGRPO to real-world and multi-task settings.

REFERENCES

- [1] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi *et al.*, “Open-vla: An open-source vision-language-action model,” *arXiv preprint arXiv:2406.09246*, 2024.
- [2] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby *et al.*, “Dinov2: Learning robust visual features without supervision,” *arXiv preprint arXiv:2304.07193*, 2023.
- [3] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu *et al.*, “Rt-1: Robotics transformer for real-world control at scale,” *arXiv preprint arXiv:2212.06817*, 2022.
- [4] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter *et al.*, “ π_0 : A vision-language-action flow model for general robot control,” *arXiv preprint arXiv:2410.24164*, 2024.
- [5] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer, “Sigmoid loss for language image pre-training,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2023, pp. 11 975–11 986.
- [6] A. O’Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain *et al.*, “Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0,” in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 6892–6903.
- [7] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen *et al.*, “Lora: Low-rank adaptation of large language models,” *ICLR*, vol. 1, no. 2, p. 3, 2022.
- [8] B. Liu, Y. Zhu, C. Gao, Y. Feng, Q. Liu, Y. Zhu, and P. Stone, “Libero: Benchmarking knowledge transfer for lifelong robot learning,” *arXiv preprint arXiv:2306.03310*, 2023.
- [9] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.
- [10] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel *et al.*, “Soft actor-critic algorithms and applications,” *arXiv preprint arXiv:1812.05905*, 2018.
- [11] Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. Li, Y. Wu *et al.*, “Deepseekmath: Pushing the limits of mathematical reasoning in open language models,” *arXiv preprint arXiv:2402.03300*, 2024.
- [12] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu, “Rdt-1b: a diffusion foundation model for bimanual manipulation,” *arXiv preprint arXiv:2410.07864*, 2024.
- [13] L. X. Shi, B. Ichter, M. Equi, L. Ke, K. Pertsch, Q. Vuong, J. Tanner, A. Walling, H. Wang, N. Fusai *et al.*, “Hi robot: Open-ended instruction following with hierarchical vision-language-action models,” *arXiv preprint arXiv:2502.19417*, 2025.
- [14] W. Huang, C. Wang, R. Zhang, Y. Li, J. Wu, and L. Fei-Fei, “Voxposer: Composable 3d value maps for robotic manipulation with language models,” *arXiv preprint arXiv:2307.05973*, 2023.
- [15] Q. Bu, J. Cai, L. Chen, X. Cui, Y. Ding, S. Feng, S. Gao, X. He, X. Huang, S. Jiang *et al.*, “Agibot world colosseum: A large-scale manipulation platform for scalable and intelligent embodied systems,” *arXiv preprint arXiv:2503.06669*, 2025.
- [16] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choremanski, T. Ding, D. Driess, A. Dubey, C. Finn *et al.*, “Rt-2: Vision-language-action models transfer web knowledge to robotic control,” *arXiv preprint arXiv:2307.15818*, 2023.
- [17] D. Driess, F. Xia, M. S. M. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. H. Vuong, T. Yu, W. Huang, Y. Chebotar, P. Sermanet, D. Duckworth, S. Levine, V. Vanhoucke, K. Hausman, M. Toussaint, K. Greff, A. Zeng, I. Mordatch, and P. R. Florence, “Palm-e: An embodied multimodal language model,” in *International Conference on Machine Learning*, 2023. [Online]. Available: <https://api.semanticscholar.org/CorpusID:257364842>
- [18] W. Huang, C. Wang, Y. Li, R. Zhang, and L. Fei-Fei, “Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation,” *arXiv preprint arXiv:2409.01652*, 2024.
- [19] Z. Zhang, K. Zheng, Z. Chen, J. Jang, Y. Li, S. Han, C. Wang, M. Ding, D. Fox, and H. Yao, “Grape: Generalizing robot policy via preference alignment,” *arXiv preprint arXiv:2411.19309*, 2024.
- [20] Y. Guo, J. Zhang, X. Chen, X. Ji, Y.-J. Wang, Y. Hu, and J. Chen, “Improving vision-language-action model with online reinforcement learning,” *arXiv preprint arXiv:2501.16664*, 2025.
- [21] Y. Chen, S. Tian, S. Liu, Y. Zhou, H. Li, and D. Zhao, “Conrft: A reinforced fine-tuning method for vla models via consistency policy,” *arXiv preprint arXiv:2502.05450*, 2025.
- [22] Q. Bu, Y. Yang, J. Cai, S. Gao, G. Ren, M. Yao, P. Luo, and H. Li, “Univla: Learning to act anywhere with task-centric latent actions,” *arXiv preprint arXiv:2505.06111*, 2025.
- [23] M. Pan, J. Zhang, T. Wu, Y. Zhao, W. Gao, and H. Dong, “Omnimanip: Towards general robotic manipulation via object-centric interaction primitives as spatial constraints,” *arXiv preprint arXiv:2501.03841*, 2025.
- [24] J. Luo, C. Xu, J. Wu, and S. Levine, “Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning,” *arXiv preprint arXiv:2410.21845*, 2024.
- [25] S. Huang, Z. Zhang, T. Liang, Y. Xu, Z. Kou, C. Lu, G. Xu, Z. Xue, and H. Xu, “Mentor: Mixture-of-experts network with task-oriented perturbation for visual reinforcement learning,” *arXiv preprint arXiv:2410.14972*, 2024.
- [26] J. Zhang, N. Gireesh, J. Wang, X. Fang, C. Xu, W. Chen, L. Dai, and H. Wang, “Gamma: Graspability-aware mobile manipulation policy learning based on online grasping pose fusion,” in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 1399–1405.
- [27] R. Rafailov, A. Sharma, E. Mitchell, C. D. Manning, S. Ermon, and C. Finn, “Direct preference optimization: Your language model is secretly a reward model,” *Advances in Neural Information Processing Systems*, vol. 36, pp. 53 728–53 741, 2023.
- [28] S. McIlvanna, N. N. Minh, Y. Sun, M. Van, and W. Naeem, “Reinforcement learning-enhanced control barrier functions for robot manipulators,” *arXiv preprint arXiv:2211.11391*, 2022.
- [29] S. A. Mehta, S. Habibian, and D. P. Losey, “Waypoint-based reinforcement learning for robot manipulation tasks,” in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024, pp. 541–548.
- [30] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis *et al.*, “Droid: A large-scale in-the-wild robot manipulation dataset,” *arXiv preprint arXiv:2403.12945*, 2024.
- [31] H.-S. Fang, H. Fang, Z. Tang, J. Liu, C. Wang, J. Wang, H. Zhu, and C. Lu, “Rh20t: A comprehensive robotic dataset for learning diverse skills in one-shot,” *arXiv preprint arXiv:2307.00595*, 2023.
- [32] T. Chu, Y. Zhai, J. Yang, S. Tong, S. Xie, D. Schuurmans, Q. V. Le, S. Levine, and Y. Ma, “Sft memorizes, rl generalizes: A comparative study of foundation model post-trainings,” *arXiv preprint arXiv:2501.17161*, 2025.
- [33] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, “High-dimensional continuous control using generalized advantage estimation,” *arXiv preprint arXiv:1506.02438*, 2015.
- [34] D. Guo, D. Yang, H. Zhang, J. Song, R. Zhang, R. Xu, Q. Zhu, S. Ma, P. Wang, X. Bi *et al.*, “Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning,” *arXiv preprint arXiv:2501.12948*, 2025.
- [35] H. Touvron, L. Martin, K. Stone, P. Albert, A. Almahairi, Y. Babaei, N. Bashlykov, S. Batra, P. Bhargava, S. Bhosale *et al.*, “Llama 2: Open foundation and fine-tuned chat models,” *arXiv preprint arXiv:2307.09288*, 2023.
- [36] L. Beyer, A. Steiner, A. S. Pinto, A. Kolesnikov, X. Wang, D. Salz, M. Neumann, I. Alabdulmohsin, M. Tschannen, E. Bugliarello *et al.*, “Paligemma: A versatile 3b vlm for transfer,” *arXiv preprint arXiv:2407.07726*, 2024.
- [37] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, “Diffusion policy: Visuomotor policy learning via action diffusion,” *The International Journal of Robotics Research*, p. 02783649241273668, 2023.
- [38] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, “Asynchronous methods for deep reinforcement learning,” in *International conference on machine learning*. PmlR, 2016, pp. 1928–1937.

- [39] D. Qu, H. Song, Q. Chen, Y. Yao, X. Ye, Y. Ding, Z. Wang, J. Gu, B. Zhao, D. Wang *et al.*, “Spatialvla: Exploring spatial representations for visual-language-action model,” *arXiv preprint arXiv:2501.15830*, 2025.
- [40] M. A. Patratskiy, A. K. Kovalev, and A. I. Panov, “Spatial traces: Enhancing vla models with spatial-temporal understanding,” *arXiv preprint arXiv:2508.09032*, 2025.
- [41] Q. Li, Y. Liang, Z. Wang, L. Luo, X. Chen, M. Liao, F. Wei, Y. Deng, S. Xu, Y. Zhang *et al.*, “Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation,” *arXiv preprint arXiv:2411.19650*, 2024.
- [42] M. Shukor, D. Aubakirova, F. Capuano, P. Kooijmans, S. Palma, A. Zouitine, M. Aractingi, C. Pascal, M. Russi, A. Marafioti *et al.*, “Smolvla: A vision-language-action model for affordable and efficient robotics,” *arXiv preprint arXiv:2506.01844*, 2025.
- [43] M. J. Kim, C. Finn, and P. Liang, “Fine-tuning vision-language-action models: Optimizing speed and success,” *arXiv preprint arXiv:2502.19645*, 2025.
- [44] H. Wang, C. Xiong, R. Wang, and X. Chen, “Bitvla: 1-bit vision-language-action models for robotics manipulation,” *arXiv preprint arXiv:2506.07530*, 2025.
- [45] H. Shi, B. Xie, Y. Liu, L. Sun, F. Liu, T. Wang, E. Zhou, H. Fan, X. Zhang, and G. Huang, “Memoryvla: Perceptual-cognitive memory in vision-language-action models for robotic manipulation,” *arXiv preprint arXiv:2508.19236*, 2025.
- [46] T. Lin, G. Li, Y. Zhong, Y. Zou, and B. Zhao, “Evo-0: Vision-language-action model with implicit spatial understanding,” *arXiv preprint arXiv:2507.00416*, 2025.
- [47] Z. Liang, Y. Li, T. Yang, C. Wu, S. Mao, L. Pei, X. Yang, J. Pang, Y. Mu, and P. Luo, “Discrete diffusion vla: Bringing discrete diffusion to action decoding in vision-language-action policies,” *arXiv preprint arXiv:2508.20072*, 2025.
- [48] Q. Lv, W. Kong, H. Li, J. Zeng, Z. Qiu, D. Qu, H. Song, Q. Chen, X. Deng, and J. Pang, “F1: A vision-language-action model bridging understanding and generation to actions,” *arXiv preprint arXiv:2509.06951*, 2025.
- [49] Y. Zhang, C. Wang, O. Lu, Y. Zhao, Y. Ge, Z. Sun, X. Li, C. Zhang, C. Bai, and X. Li, “Align-then-steer: Adapting the vision-language action models through unified latent guidance,” *arXiv preprint arXiv:2509.02055*, 2025.
- [50] H. Huang, M. Cen, K. Tan, X. Quan, G. Huang, and H. Zhang, “Graphcot-vla: A 3d spatial-aware reasoning vision-language-action model for robotic manipulation with ambiguous instructions,” *arXiv preprint arXiv:2508.07650*, 2025.