OpenMOSS

SRPO: Self-Referential Policy Optimization for Vision-Language-Action Models

Senyu Fei[2,3,*] **Siyin Wang**[1,3,*] **Li Ji**[1,3] **Ao Li**[3] **Shiduo Zhang**[1] **Liming Liu**[3]
Jinlong Hou[3] **Jingjing Gong**[3,†] **Xianzhong Zhao**[2] **Xipeng Qiu**[1,2,†]

feisenyu@outlook.com,siyinwang20@fudan.edu.cn

[1]Fudan University [2]Tongji University [3]Shanghai Innovation Institute

https://github.com/sii-research/siiRL
https://huggingface.co/collections/Sylvest/srpo

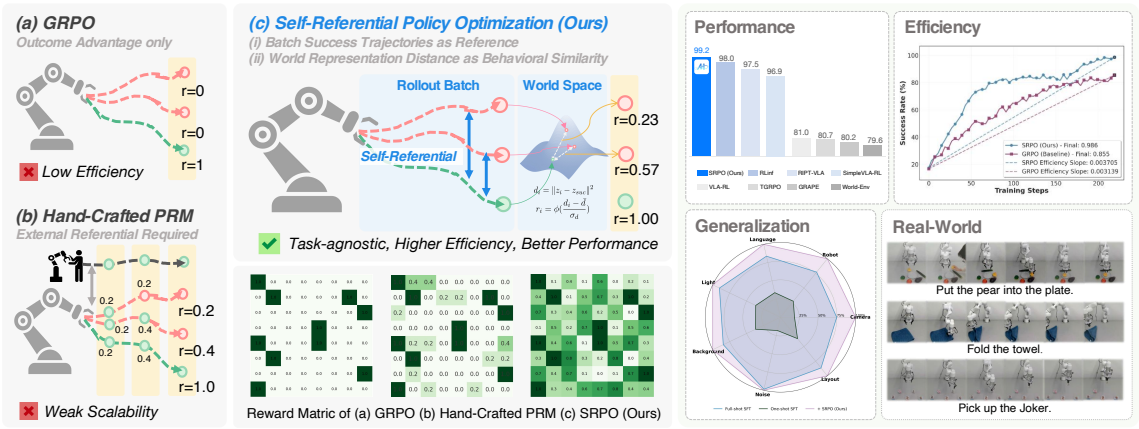


Figure 1: Overview of Self-Referential Policy Optimization (SRPO). Existing approaches for Vision-Language-Action (VLA) reinforcement learning face significant limitations: (a) methods like GRPO rely solely on sparse outcome rewards, providing limited learning signal, while (b) hand-crafted process reward modeling (PRM) requires costly external demonstrations and task-specific engineering. In contrast, our SRPO framework introduces a self-referential paradigm that leverages (i) in-batch successful trajectories and (ii) latent world representations to construct progress-wise rewards, enabling efficient utilization of failure trajectories. Extensive experimental evaluation demonstrates that SRPO achieves (1) state-of-the-art performance, (2) superior training efficiency, (3) stronger generalization capabilities, and (4) improved real-world performance.

Abstract

Vision-Language-Action (VLA) models excel in robotic manipulation but are constrained by their heavy reliance on expert demonstrations, leading to demonstration bias and limiting performance. Reinforcement learning (RL) is a vital post-training strategy to overcome these limits, yet current VLA-RL methods, including group-based optimization approaches, are crippled by severe reward sparsity. Relying on binary success indicators wastes valuable information in failed trajectories, resulting in low training efficiency. To solve this, we propose Self-Referential Policy Optimization (SRPO), a novel VLA-RL framework. SRPO eliminates the need for external demonstrations or manual reward engineering by leveraging the model's own successful trajectories, generated within the current training batch, as a self-reference. This allows us to assign a progress-wise reward to failed attempts. A core innovation is the use of latent world representations to measure behavioral progress robustly. Instead of relying on raw pixels or requiring domain-specific fine-tuning, we

*Equal Contribution.
†Corresponding Authors.

utilize the compressed, transferable encodings from a world model's latent space. These representations naturally capture progress patterns across environments, enabling accurate, generalized trajectory comparison. Empirical evaluations on the LIBERO benchmark demonstrate SRPO's efficiency and effectiveness. Starting from a supervised baseline with 48.9% success, SRPO achieves a new state-of-the-art success rate of 99.2% in just 200 RL steps, representing a 103% relative improvement without any extra supervision. Furthermore, SRPO shows substantial robustness, achieving a 167% performance improvement on the LIBERO-Plus benchmark.

1 Introduction

Vision-Language-Action (VLA) models (Zitkovich et al., 2023; Team et al., 2024; Kim et al., 2024; Black et al.) have demonstrated remarkable capabilities in robotic manipulation by leveraging large-scale pre-trained vision-language models. However, existing VLA systems rely heavily on expert demonstrations and tend to overfit small downstream datasets, resulting in a strong demonstration bias that prevents them from surpassing human performance (Fei et al., 2025a; Zhang et al., 2025). To address such limitations, many recent studies have shown that reinforcement learning (RL) is an effective post-training strategy that can substantially improve VLA performance, both in-distribution and out-of-distribution (Guo et al., 2025b; Liu et al., 2025; Lu et al., 2025). Among RL methods, group-based optimization approaches such as GRPO (Shao et al., 2024) have become particularly attractive, providing a simple yet highly effective learning paradigm for VLA fine-tuning (Li et al., 2025; Zang et al., 2025; Tan et al., 2025).

Despite the remarkable progress in GRPO, it still suffers from sparsity challenges in reward signals (Chan et al., 2024; Mu et al., 2024; Cao et al., 2024). This issue becomes particularly pronounced in VLA domains, as the computational cost of multi-turn trajectory rollouts is substantially higher and the inadequate utilization of information from failed trajectories significantly reduces training efficiency. While recent work has explored process supervision to provide denser feedback, these approaches typically depend on expert demonstrations or hand-crafted task decompositions to define intermediate progress milestones (Lu et al., 2025; Chen et al., 2025b), creating scalability limitations that contradict the goal of autonomous learning.

To address the reward sparsity challenge, we propose self-referential learning, where the model's own successful trajectories serve as reference standards to evaluate and guide failed attempts. While GRPO leverages outcome-only rewards for advantage estimation, our approach utilizes the entire trajectory batch more efficiently. This paradigm transforms the supervision problem from "how do we obtain expert labels" to "how do we extract progress-wise reward from our own successes".

A central challenge in this paradigm lies in quantitatively measuring the behavioral similarity between successful and failed trajectories to assess progress towards task completion. Since trajectories are represented solely by visual observations, we require a reward model capable of extracting global state information to measure progress. While world models offer a natural approach for modeling such dynamics (Assran et al., 2025; Ali et al., 2025), traditional pixel-level world models suffer from poor generalization across domains or typically require extensive task-specific fine-tuning (Xiao et al., 2025). We address this limitation by leveraging the latent representations learned by world models, which we find naturally capture transferable behavioral progress patterns across different environments. These latent world representations enable robust trajectory comparison without requiring precise environmental reconstruction or domain-specific training.

Building on these insights, we propose Self-Referential Policy Optimization (SRPO), which seamlessly integrates self-referential learning into the RL training structure. During each training iteration, we identify successful trajectories within the batch and use them as behavioral references through world model latent representations, while failed trajectories receive progress-wise rewards based on their behavioral alignment with successful patterns. We opt for trajectory-level rewards over fine-grained reward shaping as overly detailed hand-crafted dense signals can lead the policy to converge toward suboptimal solutions (Sutton, 2019; Levine, 2024).

Empirical evaluation on the LIBERO (Liu et al., 2023) benchmark demonstrates the transformative impact of our approach. Starting from a supervised fine-tuned baseline with just one demonstration per task and a 48.9% success rate, SRPO reaches 99.2% success in merely 200 reinforcement learning steps, representing 103% relative improvement that establishes new state-of-the-art performance. Importantly, this RL improvement requires no additional expert demonstrations or manual reward engineering. Our method also shows

substantial improvements in robustness on the LIBERO-Plus (Fei et al., 2025a) benchmark, with 167% performance improvement. Our proposed reward demonstrates its effectiveness in both simulation and real-world experiments.

Our contributions are threefold as follow:

1. We propose SRPO, a novel VLA reinforcement learning framework that mitigates reward sparsity by using model-generated successful trajectories to provide progress-wise rewards for failed attempts, eliminating reliance on expert demonstration or task-specific engineering.

2. We introduce a latent world representation-based progress-wise reward method that overcomes the generalization limitations and domain-specific training requirements of traditional pixel-level world models.

3. Experimental results demonstrate that our method achieves state-of-the-art performance on the LIBERO benchmark and exhibits strong generalization capabilities on LIBERO-Plus, while eliminating the need for additional supervision during RL training, establishing a new paradigm for autonomous VLA learning.

2 Related Work

2.1 Vision-Language-Action Models

The remarkable success of models trained on vast internet-scale datasets has demonstrated the power of scaling laws, motivating the field to increasingly focus on Vision-Language-Action (VLA) models for transferring these capabilities to embodied agents (Zitkovich et al., 2023; Kim et al., 2024; 2025; Black et al.; Pertsch et al., 2025; Wang et al., 2025b). Due to the limited generalization and domain adaptability of supervised post-training, the field is increasingly turning to Reinforcement Learning (RL) to encourage exploration and enhance overall robustness (Lu et al., 2025; Li et al., 2025; Liu et al., 2025). These studies have explored the distinct characteristics of RL compared to SFT, employing RL algorithms to train both autoregressive (Tan et al., 2025) and diffusion-based VLAs (Chen et al., 2025a), while also developing several integrated training-and-inference frameworks. However, these approaches often suffer from sparse reward signals, which prevent effective utilization of information from failure trajectories and limit training efficiency. To address this, we propose a novel VLA-RL framework that leverages dense, zero-shot transferable rewards to enable highly efficient policy learning.

2.2 Reinforcement Learning

The application of Reinforcement Learning (RL) to Large Language Models (LLMs) has evolved significantly (Jaech et al., 2024; Yang et al., 2025a; Guo et al., 2025a; Team et al., 2025; Comanici et al., 2025; Fei et al., 2025b). RLHF (Bai et al., 2022) successfully aligned models with human preferences using algorithms like PPO (Schulman et al., 2017). RLVR (Wen et al., 2025) replacing expensive human preferences with automatically verifiable rewards. GRPO (Shao et al., 2024) bypasses the need for a learned reward model by using programmatic rewards. Recent works have also attempted to enhance stability and efficiency in optimization (Yu et al., 2025; Wang et al., 2025a; Yang et al., 2025b). Since task success serves as a naturally verifiable reward, numerous works in the VLA domain have attempted to optimize policies using GRPO with binary (0/1) rewards (Li et al., 2025; Zang et al., 2025; Tan et al., 2025; Liu et al., 2025), while facing the challenge of sparse reward signals. As robotic trajectory rollouts are more time-consuming and resource-intensive, some works have begun to construct rewards using hand-crafted or task-specific priors (Chen et al., 2025b; Lu et al., 2025; Shu et al., 2025). However, such methods requiring expert demonstrations or prior information are difficult to scale in online RL settings that encourage autonomous learning. To this end, we introduce a self-referential approach leveraging task-agnostic latent world representation to provide progress-wise rewards, thus fully utilizing failure trajectory information to enhance learning efficiency.

Figure 2: Overview of the SRPO method. During policy rollout, both successful and failed trajectories are collected in the Rollout Reference Set. For each trajectory, we employ a world model pre-trained on large-scale robotics video data (Assran et al., 2025) as an encoder to extract latent world representations. Behavioral similarity is modeled as the L2 distance between trajectory embeddings in this space to yield progress-wise rewards. These rewards are subsequently used for advantage estimation and policy optimization under KL regularization.

3 Methods

3.1 Problem Formulation and Key Components

Given the current observation o_t at time step t, and a goal description l, the agent takes an action a_t according to a policy $\pi_\theta(a_t|o_t, l)$ parameterized by θ and conditioned on a goal description l. The agent then receives the next observation o_{t+1} from the environment, so on and so forth, until finally receiving a terminal observation o_T. If the agent successfully completes the task specified by the goal description l, it receives a positive reward; otherwise, it receives reward depending on how close it is to completing the task. The objective is to learn a policy that maximizes the reward received over time for a variety of goals $\mathbf{l}$ and the initial state of the environment $\mathbf{z}$.

The observation function is defined as $o_t \leftarrow O(z_t)$, where O is the observation function that maps the environment state z_t to the partial observation o_t received by the agent.

Action generation is modeled by the policy function $\pi_\theta(a_t|o_t, l)$ which is the θ parameterized probability distribution over actions given the current observation o_t, goal description l.

The environment function $z_{t+1} \sim E(\cdot|z_t, a_t)$ defines the stochastic transition from the current environment state z_t to the next environment state z_{t+1} given the action a_t taken by the agent.

Trajectory Rollout is the process of generating a sequence of observations and actions over time, starting from an initial environment state z_0. The trajectory $\{(z_0, o_0, a_0, z_1, o_1, a_1, \ldots, z_T, o_T)\}$ are generated by iteratively applying these functions:

$$o_t = O(z_t), \quad a_t \sim \pi_\theta(\cdot|o_t, l), \quad z_{t+1} \sim E(\cdot|z_t, a_t) \tag{1}$$

Note that the state of the environment z_t is not directly accessible to the agent; instead, the agent relies on the observation o_t derived from z_t through the observation function O.

The Reward function $R(z_{0:T}, l)$ evaluates how well the state trajectory is performing in achieving the goal specified by l. This reward is usually provided by the environment at the end of the episode, based on whether the goal described by l has been achieved. However, these rewards could be provided sparsely as 0 and 1, making it difficult for the agent to learn effective policies through traditional reinforcement learning methods. To address this, we adapt a world model to shape rewards $\hat{R}(o_{0:T}, \mathcal{S})$ for the failed trajectories to provide more informative feedback during training, as described in detail in Section 3.2.

3.2 World progress reward modeling

As have mentioned earlier, neither the states $z_{0:T}$ nor the reward function $R(z_{0:T}, l)$ are directly accessible to the agent. There is only the sparse terminal reward signal provided at the end of each episode. We propose a

world model-based task-agnostic reward shaping mechanism to provide progress-wise signals for unsuccessful rollout trajectories. The reward can be shaped as $\hat{R}(o_{0:T}, \mathcal{S})$, where $\mathcal{S} = \left\{ o^{(i)}_{0:T}; R(z^{(i)}_{0:T}, l) = 1,\ \forall i \right\}$ is the observation set of successful trajectories.

First, we encode the observations of the unsuccessful trajectory and the successful reference trajectories, with a world-model encoder $\mathcal{W}$,

$$h_i = \mathcal{W}(o^{(i)}_{0:T}). \tag{2}$$

We then cluster the representations of the successful trajectories using the DBSCAN algorithm (Ester et al., 1996) to obtain a set of representative centers. The distance to the closest center representation is then calculated; a smaller distance indicates a larger reward.

$$C = \text{DBSCAN}(\mathcal{S}) \tag{3}$$

$$d_i = \min(\left\{ \|h_i - h_j\|^2; h_j \in C \right\}) \tag{4}$$

Finally, the reward is calculated with

$$g_i = \begin{cases} 1.0 & \text{for succes trajectory} \\ \phi(\frac{d_i - \bar{d}}{\sigma_d}) & \text{for failed trajectory} \end{cases}, \tag{5}$$

where $\phi(\cdot)$ is an activation function transforming the output to the range $(0, 1)$, and $\bar{d}$ and σ_d are the mean and standard deviation, respectively, of the calculated distance for all failed trajectories.

3.3 Self-Referential Policy Optimization

We introduce *Self-Referential Policy Optimization* (SRPO), a novel policy optimization framework for VLA models that leverages latent world representations to enable **self-referential advantage estimation** on **trajectory level**.

Following GRPO, the probability ratio, advantage and regularization terms are

$$r_{i,t}(\theta) = \frac{\pi_\theta(a^{(i)}_t | o^{(i)}_t, l)}{\pi_{\theta_{old}}(a^{(i)}_t | o^{(i)}_t, l)}, \quad \hat{A}_i = \frac{g_i - \mu_g}{\sigma_g}, \quad \omega(\theta) = \beta D_{\text{KL}}\left(\pi_\theta \parallel \pi_{ref}\right). \tag{6}$$

The clipped surrogate objective is

$$\mathcal{L}^{\text{CLIP}}_{t,i}(\theta) = \min(r_{i,t}(\theta)\hat{A}_i, \text{clip}(r_{i,t}(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_i) \tag{7}$$

The overall SRPO objective, which we aim to maximize, is given by the expectation of the $\mathcal{L}_{\text{CLIP}}(\theta)$ function. This expectation is computed over time steps t and sample IDs i within the current training group, with an additional regularization term applied to the parameters θ.

$$\mathcal{L}_{\text{SRPO}}(\theta) = \mathbb{E}_{t,i}\mathcal{L}^{\text{CLIP}}_{t,i}(\theta) + \omega(\theta), \tag{8}$$

for simplicity, the expectation over different tasks l and initial state z_0 have been omitted. The group statistics are derived from world progress rewards as follows:

$$\mu_{\hat{R}} = \frac{1}{M}\sum_{j=1}^{M}\hat{R}_j,\ \sigma_{\hat{R}} = \sqrt{\frac{1}{M}\sum_{j=1}^{M}(\hat{R}_j - \mu_{\hat{R}})^2 + \epsilon}. \tag{9}$$

This self-referential approach enables the policy to learn from relative performance within trajectory groups, where rewards $\hat{R}_j$ are computed through our world progress reward model rather than traditional task-specific rewards. The KL divergence $D_{\text{KL}}(\pi_\theta \parallel \pi_{\text{ref}})$ term maintains policy stability.

Table 1: Performance comparison on LIBERO benchmark. We evaluate mainstream VLA foundation models and RL-based methods. OpenVLA* incorporates action chunking and parallel decoding on the basis of OpenVLA. Policy Input notation: T (Thirdview), I (Instrcution), P (Proprio), W (Wristimage), D (Depth). Our approach, built upon one-shot SFT, achieves state-of-the-art results on LIBERO benchmark, with ↑ indicating performance gains over the one-shot baseline.

Model	**Policy Input**	**LIBERO**				
		Spatial	Object	Goal	Long	Avg
OpenVLA (Kim et al., 2024)	T+I	84.7	88.4	79.2	53.7	76.5
Pi0+fast (Pertsch et al., 2025)	T+W+P+I	96.4	96.8	88.6	60.2	85.5
Pi0 (Black et al.)	T+W+P+I	96.8	98.8	95.8	85.2	94.2
SmolVLA (Shukor et al., 2025)	T+W+P+I	93.0	94.0	91.0	77.0	88.8
WorldVLA (Cen et al., 2025)	T+I	85.6	89.0	82.6	59.0	79.1
NORA (Hung et al., 2025)	T+I	92.2	95.4	89.4	74.6	87.9
CoT-VLA (Zhao et al., 2025)	T+I	87.5	91.6	87.6	69.0	81.1
UniVLA (Bu et al., 2025)	T+I	96.5	96.8	95.6	92.0	95.2
TraceVLA (Zheng et al., 2024)	T+I	84.6	85.2	75.1	54.1	74.8
MolmoAct (Lee et al., 2025)	T+I	87.0	95.4	87.6	77.2	86.6
ThinkAct (Huang et al., 2025)	T+I	88.3	91.4	87.1	70.9	84.4
GR00T N1 (Bjorck et al., 2025)	T+I	94.4	97.6	93.0	90.6	93.9
3D-CAVLA (Bhat et al., 2025)	T+W+P+D+I	98.2	99.8	98.2	96.1	98.1
OpenVLA-OFT (Kim et al., 2025)	T+W+P+I	96.2	98.3	96.2	90.7	95.3
OpenVLA*-Full	T+I	91.6	95.3	90.6	86.5	91.0
TGRPO (Chen et al., 2025b)	T+I	90.4	92.2	81.0	59.2	80.7
GRAPE (Zhang et al., 2024)	T+I	88.5	92.1	83.1	57.2	80.2
VLA-RL (Lu et al., 2025)	T+I	90.2	91.8	82.2	59.8	81.0
World-Env (Xiao et al., 2025)	T+I	87.6	86.6	86.4	57.8	79.6
SimpleVLA-RL (Li et al., 2025)	T+I	98.2	98.7	98.8	91.7	96.9
RIPT-VLA (Tan et al., 2025)	T+W+P+I	99.0	98.6	98.6	93.8	97.5
RLinf (Zang et al., 2025)	T+W+P+I	**99.4**	99.8	98.8	94.0	98.0
OpenVLA*-One	T+I	63.6	54.9	59.6	17.3	48.9
+ Offline SRPO	T+I	92.5	96.8	92.0	88.7	92.5
		↑28.9	↑41.9	↑32.4	↑71.4	↑43.6
+ Online SRPO	T+I	98.8	**100.0**	**99.4**	**98.6**	**99.2**
		↑35.2	↑45.1	↑39.8	↑81.3	↑50.3

4 Experiments

Our experimental design addresses several fundamental research questions: **RQ1:** Can SRPO achieve state-of-the-art performance on standard benchmarks? **RQ2:** Does the method possess strong generalization capabilities to adapt to unseen task scenarios? **RQ3:** Can the latent world modeling-based reward mechanism accurately assess task progress, and does it outperform both pixel-level approaches and conventional video encoding methods? **RQ4:** Can SRPO improve training efficiency? **RQ5:** Can SRPO motivate novel trajectory exploration? **RQ6:** Whether the reward shaping method can successfully transfer to real-world robotic systems without requiring domain-specific expert demonstrations?

4.1 Experimental Setup

Main experiments are conducted on the **LIBERO** (Liu et al., 2023) benchmark, specifically on Goal, Spatial, Object and Long suites, each of which contains 10 tasks for evaluating a distinct model capability. To evaluate generalization robustness, we employ the **LIBERO-Plus** (Fei et al., 2025a) benchmark that introduces seven dimensions of perturbations.

Table 2: Robustness Evaluation on LIBERO-Plus Benchmark. OpenVLA-OFT+ refers to the OpenVLA-OFT model that has been trained on the LIBERO-Plus dataset. Our method, SRPO, applied to a one-shot SFT policy, not only significantly outperforms its base model but also surpasses the full-shot SFT baseline in all 7 dimensions, demonstrating superior generalization capability. The ↑ indicates performance gains over the One-shot SFT base model.

Model	**Perturbation Dimensions**							**Total**
	Camera	Robot-Init	Language	Light	Background	Noise	Layout	
Zero-Shot								
Pi0	13.8	6.0	58.8	85.0	81.4	79.0	68.9	53.6
Pi0+fast	65.1	21.6	61.0	73.2	73.2	74.4	68.8	61.6
UniVLA	1.8	46.2	69.6	69.0	81.0	21.2	31.9	42.9
WorldVLA	0.1	27.9	41.6	43.7	17.1	10.9	38.0	25.0
OpenVLA	0.8	3.5	23.0	8.1	34.8	15.2	28.5	15.6
OpenVLA-OFT	56.4	31.9	79.5	88.7	93.3	75.8	74.2	69.6
OpenVLA-OFT_w	10.4	38.7	70.5	76.8	93.6	49.9	69.9	55.8
OpenVLA-OFT_m	55.6	21.7	81.0	92.7	91.0	78.6	68.7	67.9
NORA-long	2.2	37.0	65.1	45.7	58.6	12.8	62.1	39.0
RIPT-VLA	55.2	31.2	77.6	88.4	91.6	73.5	74.2	68.4
OpenVLA*-Full	12.8	39.4	68.5	63.4	75.0	34.8	62.7	51.1
OpenVLA*-One	3.2	14.0	27.6	25.7	32.7	6.4	26.4	19.4
+ Online SRPO	17.1	51.0	81.8	70.4	88.9	35.3	72.4	59.6
	↑13.9	↑37.0	↑54.2	↑44.7	↑56.2	↑28.9	↑46.0	↑40.2
With Augmented Data								
OpenVLA-OFT+	92.8	30.3	85.8	94.9	93.9	89.3	77.6	79.5
OpenVLA*-Full	69.4	49.6	66.3	88.2	88.5	78.7	70.3	73.0
OpenVLA*-One	12.8	23.0	30.0	42.0	49.6	23.3	34.5	30.7
+ Online SRPO	83.4	62.0	73.6	97.2	97.7	85.7	75.2	82.1
	↑70.6	↑39.0	↑43.6	↑55.2	↑48.1	↑62.4	↑40.7	↑51.4

Models. For simulation experiments, we utilize a modified version of OpenVLA enhanced with action chunking and parallel decoding, which we refer to as **OpenVLA*** in the following sections. OpenVLA*-One refers to base model with one-traj-per-task SFT, while OpenVLA*-Full refers to full-shot SFT. More details can be found in Appendix G.1.

Baseline Methods. Our baseline selection includes: **SimpleVLA-RL** (Li et al., 2025), **RIPT-VLA** (Tan et al., 2025) and **RLinf** (Zang et al., 2025) trained with GRPO, **VLA-RL** (Lu et al., 2025) with PPO, **GRAPE** (Zhang et al., 2024) with trajectory-wise DPO, **TGRPO** (Chen et al., 2025b) with task-specific progress reward, **World-Env** (Xiao et al., 2025) utilizing world model as simulator. We also provide some imitation learning methods (Kim et al., 2024; Pertsch et al., 2025; Black et al.; Shukor et al., 2025; Cen et al., 2025; Hung et al., 2025; Zhao et al., 2025; Bu et al., 2025; Zheng et al., 2024; Lee et al., 2025; Huang et al., 2025; Bjorck et al., 2025; Bhat et al., 2025; Kim et al., 2025) as reference, although they have different information inputs or pre-training data.

Implementation Details. In the scope of this work, we consider the observation o_t to be the third view image, and the l to be a language description of the goal. The environment is a physics simulator that takes in the current state z_t (including object positions, velocities, etc.) and the action a_t (e.g., end-effector commands) to produce the next state z_{t+1}. The architecture of our policy is based on a pre-trained Visual-Language-Action model, which integrates visual inputs and language descriptions to inform the agent's actions.

We leverage SiiRL (Wang et al., 2025c) to develop our training framework. Our pipeline begins with supervised fine-tuning using a single demonstration per task from the official checkpoint, followed by our SRPO method for online reinforcement learning post-training. To obtain shared latent world representations, we employ a large-scale video-pretrained latent world model, V-JEPA 2 (Assran et al., 2025). Training details and compute reports are provided in the Appendix F.

4.2 Main Results

Main results are summarized in Table 1, demonstrating that our approach achieves state-of-the-art performance in terms of the overall average success rate across all four suites. The key findings are as follows:

(1) Effectiveness of Online Post-Training: While the initial one-shot SFT baseline shows modest performance, SRPO significantly improves upon it, validating the efficacy of our online post-training paradigm in enhancing policy performance through environment interaction.

(2) Superiority in Reward Design: Our method outperforms RL-based methods like SimpleVLA-RL and RLinf, which rely on sparse outcome rewards, highlighting the importance of process rewards that provide denser supervisory signals. Furthermore, it surpasses approaches using manually designed process rewards like TGRPO and demonstrates that self-reference-based rewards, derived from pre-trained world models, offer a more effective learning signal than task-specific alternatives that require heuristic stage partitioning and reward tuning.

(3) SOTA Performance with Limited Inputs: Despite using only third-person visual observations and language instructions as input, our method outperforms several baselines that utilize additional modalities such as multiple camera views, proprioception (Black et al.; Pertsch et al., 2025; Shukor et al., 2025; Kim et al., 2025) and 3D data (Bhat et al., 2025). This demonstrates that SRPO can achieve state-of-the-art performance relying solely on visual inputs, highlighting its effectiveness and practical applicability.

4.3 Generalization Performance

To assess SRPO's generalization, we validate two key aspects: (1) whether SRPO training in LIBERO environments improves generalization over SFT, and (2) whether SRPO is more robust to perturbations than training on the pre-collected LIBERO-Plus dataset. The following key findings are summarized from Table 2:

(1) **SRPO Overcomes the Diversity Limitation of One-Shot SFT.** While the one-shot SFT baseline exhibits weak generalization due to limited trajectory diversity, SRPO post-training enables it to surpass even the full-shot SFT policy. This is attributed to SRPO's online interaction, which explores a broader range of trajectories than those contained in the static full-shot dataset. A detailed analysis of this trajectory diversity advantage is provided in Section 5.3.

(2) **SRPO also Surpasses Full-Shot SFT in Perturbed Data Settings.** When trained in perturbed environments, our method not only exceeds the performance of the full-shot SFT baseline but also outperforms OpenVLA-OFT (utilizes additional wrist image and proprioception) trained with full-shot SFT. This result underscores that the trajectory diversity afforded by SRPO plays a major role in improving generalization, compensating for and exceeding the advantage of more complex input modalities.

5 Analysis

5.1 Can Latent World Representation Improve Reward Shaping?

We compare our approach against two alternative reward formulations: (1) pixel-level progress reward (Wen et al., 2025), which computes similarity based on pixel-wise features, and (2) ImageBind-based progress reward, derived from the general-purpose vision embeddings model ImageBind (Girdhar et al., 2023).

Qualitative Visualization. We evaluate our progress reward approach on several tasks (e.g., placing a cup in a microwave, tidying a desk). As shown in Figure 3, our method produces smoother, more monotonic progress curves that better reflect task structure, especially for long-horizon tasks with repeated sub-tasks. In contrast, pixel-level and ImageBind-based methods generate fluctuating, non-monotonic progress signals. This improvement comes from embedding entire trajectories into a latent space that captures physical regularities, allowing for more accurate estimation of task progress. Further reward curves and success/failure trajectory examples are provided in Appendix B.

Table 3: Progress Reward Benchmark Results. Our method achieved a better level than the baseline in all 5 indicators.

Method	SC	Mono	MMD	JS	SMD
Pixel-level	0.125	0.498	0.274	0.548	2.100
ImageBind	0.957	0.837	0.356	0.408	18.111
SRPO (Ours)	**0.998**	**0.992**	**0.615**	**0.572**	**188.799**

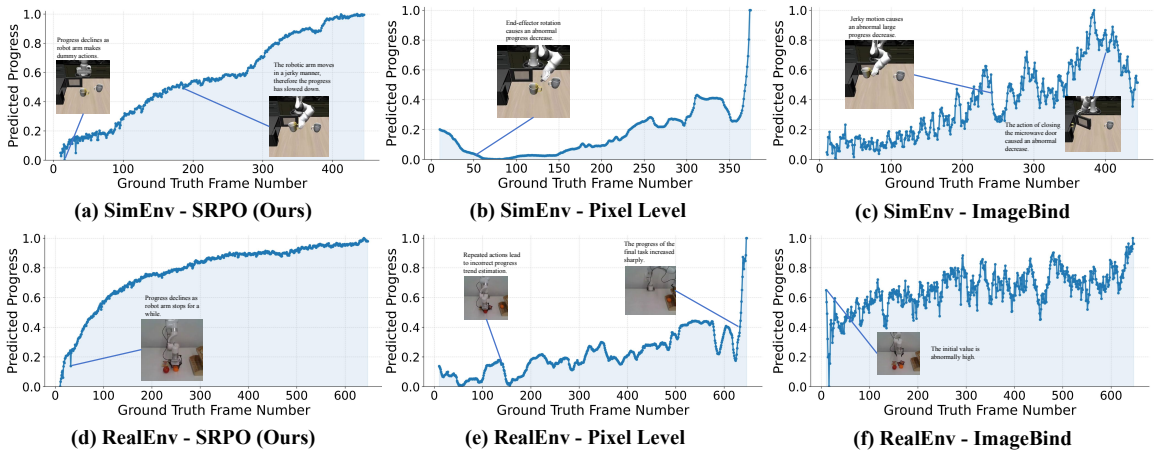


Figure 3: Comparison of progress estimation methods in simulated (a-c) and real-world (d-f) environments. Our SRPO reward (a,d) provides monotonic and physically plausible progress estimation. Pixel-level rewards (b,e) show sensitivity to perceptual changes, while ImageBind rewards (c,f) exhibit erratic trends from jerky motions.

Quantitative Comparison. We carefully selected 700 success trajectories progressing steadily towards the goal and 300 failure trajectories from multiple tasks in both simulation and real-world environments. We establish five metrics for evaluating progress rewards: *Spearman Correlation (SC)* and *Monotonicity (Mono)* are computed exclusively on success trajectories to measure the positive correlation with frame numbers and the trajectory's monotonicity, respectively; *Maximum Mean Discrepancy (MMD)*, *JS Divergence (JS)*, and *Standardized Mean Difference (SMD)* are statistical measures that quantify the distinguishability between success and failure trajectories. For all five metrics, a higher value indicates better performance. As shown in Table 3, our method provides more reasonable progress rewards for intermediate failure states and better distinguishes between success and failure trajectories, thereby more effectively utilizing the failure trajectories. The detailed formulations of our bench are provided in Appendix A.

Training Effectiveness. As shown in Figure 4, the pixel-level method exhibits notably slow convergence, suggesting that its reward signal provides insufficient guidance for effective policy optimization. While ImageBind demonstrates faster initial learning compared to the pixel-level approach, its performance plateaus around 85% without further improvement, indicating that the suboptimal progress estimation hinders the model's ability to extract useful information from failure trajectories, ultimately causing training stagnation. We argue that the *pixel-level* method is fundamentally limited because it relies solely on the final frame and exhibits sensitivity to minor pixel changes, and a general visual model like *ImageBind* lacks the capability of understanding robotics-relevant physical concepts. Our experimental results confirm that effective progress estimation for VLAs requires rewards grounded in *world progress* rather than simple perceptual similarity.

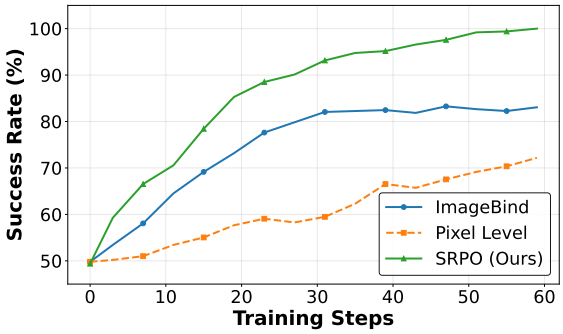
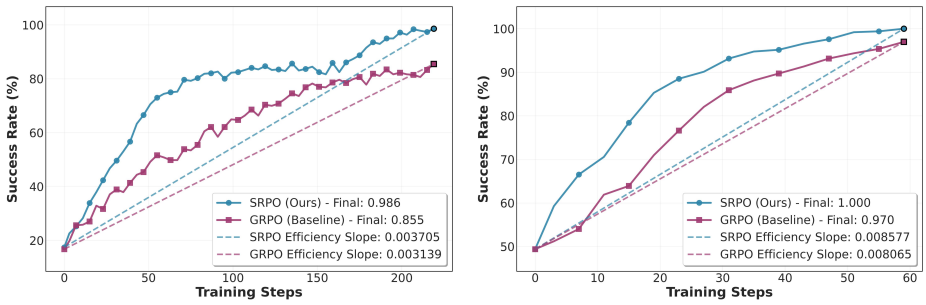


Figure 4: Training performance comparison using different progress reward formulations. Our SRPO-based reward enables stable and efficient learning, consistently outperforming both baselines.

5.2 Can SRPO Improve Training Efficiency?

Our results across the four suites are obtained using 79 steps (Spatial), 59 steps (Object), 103 steps (Goal), and 219 steps (Long), demonstrating significant advantages over SFT which requires tens of thousands of steps. Meanwhile, we compare the efficiency differences between SRPO and GRPO, as shown in Figure 5. The results clearly indicate that SRPO achieves a steeper efficiency slope than GRPO, especially for long-horizon

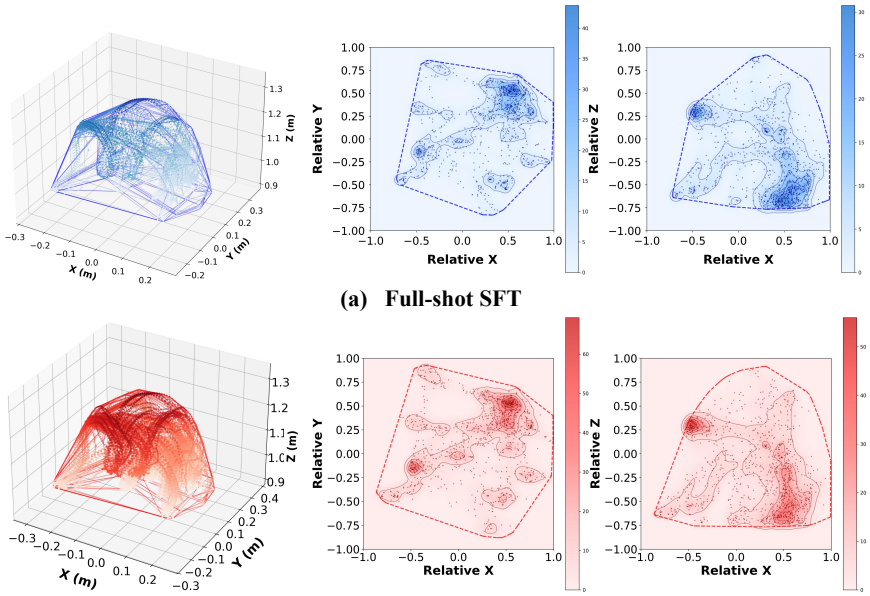


(a) LIBERO-Long **(b) LIBERO-Object**

Figure 5: Training efficiency comparison between SRPO and GRPO: (a) LIBERO-Long, (b) LIBERO-Object.

tasks. Unlike GRPO, which essentially discards unsuccessful episodes, our self-referential framework can extract valuable learning signals from near-successful trajectories by recognizing and rewarding the productive segments within them, thereby guiding the policy more efficiently towards the final goal.

5.3 Can SRPO Motivate Novel Trajectory Exploration?



(a) Full-shot SFT

(b) One-shot SFT + SRPO

Figure 6: Action space comparison of end-effector trajectories between (a) full-shot supervised fine-tuning (SFT) and (b) SRPO online reinforcement learning (RL) policies.

A fundamental limitation of imitation learning is its inherent constraint to the state-action distribution present in the demonstration data. To quantitatively assess whether SRPO enables the policy to transcend these limitations and explore beyond the demonstrator's domain, we analyze the diversity of actions generated by the policy in *LIBERO-Spatial* task suite. We performed rollouts on this suite using both the full-shot SFT weights and the RL fine-tuned weights, executing 10 trajectory episodes per task, with end-effector positions utilized as actions, which is visualize in Figure 6.

The results demonstrate that the online RL-trained policy exhibits two key advantages over the SFT baseline in terms of action distribution: (1) it explores previously unreachable regions; and (2) it generates more dispersed trajectories, indicating the model's propensity for spatial exploration rather than merely fitting to specific demonstration paths.

Furthermore, as illustrated in Figure 7, even when the policy is initially exposed to only a single successful demonstration trajectory, the subsequent online RL fine-tuning enables it to discover novel strategies beyond the provided example. This exploration is evident not only in the diversity of spatial paths taken to approach the object but also in the variety of grasping positions it discovers.

This finding underscores a key strength of our method: the ability to autonomously acquire and leverage novel knowledge, in terms of both motor skills and affordance understanding, that was not present in the original, limited demonstration data, thereby significantly enhancing the policy's robustness and generalization.

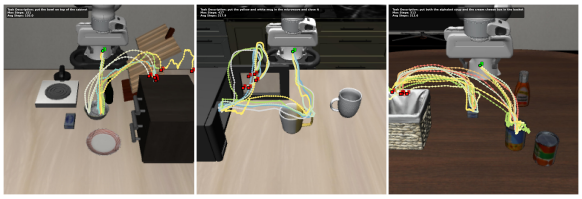


Figure 7: Visualization of end-effector trajectories across three tasks (from left to right): *put the bowl on top of the cabinet*, *put the yellow and white mug in the microwave and close it*, and *put both the alphabet soup and the cream cheese box in the basket*.

5.4 Can world progress rewarding modeling extend to real-world?

In this section, we investigate whether our proposed latent world progress rewarding method can effectively generalize to real-world robotic tasks. To this end, we conduct experiments with offline RL approach on five real-world manipulation tasks performed on an X-ARM 7 robot: *Put apple into the plate*, *Put pear into the plate*, *Folding towels*, *Cleaning whiteboard* and *Select Poker*.

For each task, we compare two state-of-the-art vision-language-action (VLA) policy backbones: π_0 (a diffusion-based VLA model) and π_0-FAST (an autoregressive VLA enhanced with frequency-space tokenization). Both models are compared against standard supervised fine-tuning (SFT) baselines.

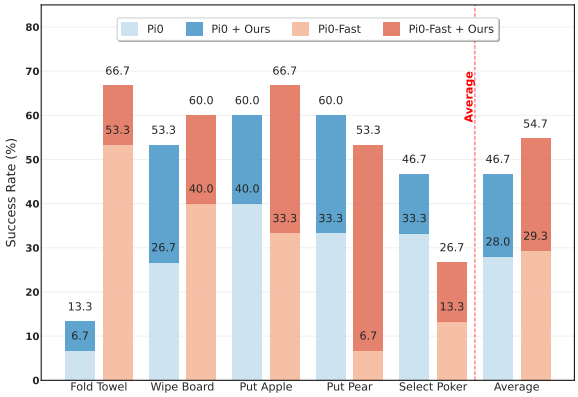


Figure 8: Real-world task success rates comparing supervised fine-tuning (SFT) baselines against our offline RL approach. Our method consistently improves performance across both diffusion-based (π_0) and autoregressive (π_0-FAST) VLA policies.

Both the diffusion-based π_0 and autoregressive π_0-FAST models demonstrate substantial performance improvements when enhanced with our reward shaping method, with average gains of $+66.8\%$ and $+86.7\%$, respectively. The most significant improvements are observed on tasks involving object placement and manipulation, underscoring the method's efficacy in adapting to perceptual variations. These consistent and substantial improvements across diverse tasks and policy architectures confirm that progress-aware reward modeling effectively transfers to real-world robotic manipulation, enabling more efficient policy optimization through better credit assignment and progression-aware weighting.

6 Conclusion

In this paper, we introduce Self-Referential Policy Optimization (SRPO), a novel VLA reinforcement learning approach leveraging task-agnostic latent world representation to provide progress-wise reward. By using self-referential learning, we eliminate reliance on expert demonstration or task-specific engineering, and develop an efficient VLA RL framework that can fully utilize the information provided by the failure trajectories, achieving significant improvements in both performance and efficiency compared to the existing methods. Further analysis demonstrates that our method improves reward shaping, motivates novel trajectory exploration, and the world progress reward modeling can be effectively applied to real-world robotic tasks. This work establishes a novel paradigm for efficient VLA reinforcement learning by demonstrating that pre-trained world model latent representations provide a powerful, generic substrate for progress assessment.

References

Arslan Ali, Junjie Bai, Maciej Bala, Yogesh Balaji, Aaron Blakeman, Tiffany Cai, Jiaxin Cao, Tianshi Cao, Elizabeth Cha, Yu-Wei Chao, et al. World simulation with video foundation models for physical ai. *arXiv preprint arXiv:2511.00062*, 2025.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Yuntao Bai, Andy Jones, Kamal Ndousse, Amanda Askell, Anna Chen, Nova DasSarma, Dawn Drain, Stanislav Fort, Deep Ganguli, Tom Henighan, et al. Training a helpful and harmless assistant with reinforcement learning from human feedback. *arXiv preprint arXiv:2204.05862*, 2022.

Vineet Bhat, Yu-Hsiang Lan, Prashanth Krishnamurthy, Ramesh Karri, and Farshad Khorrami. 3d cavla: Leveraging depth and 3d context to generalize vision language action models for unseen tasks. *arXiv preprint arXiv:2505.05800*, 2025.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π0: A vision-language-action flow model for general robot control. corr, abs/2410.24164, 2024. doi: 10.48550. *arXiv preprint ARXIV.2410.24164*.

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. *arXiv preprint arXiv:2505.06111*, 2025.

Meng Cao, Lei Shu, Lei Yu, Yun Zhu, Nevan Wichers, Yinxiao Liu, and Lei Meng. Enhancing reinforcement learning with dense rewards from language model critic. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 9119–9138, 2024.

Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, et al. Worldvla: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025.

Alex J Chan, Hao Sun, Samuel Holt, and Mihaela Van Der Schaar. Dense reward for free in reinforcement learning from human feedback. *arXiv preprint arXiv:2402.00782*, 2024.

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, Quanlu Zhang, Zhaofei Yu, Guoliang Fan, et al. π-rl: Online rl fine-tuning for flow-based vision-language-action models. *arXiv preprint arXiv:2510.25889*, 2025a.

Zengjue Chen, Runliang Niu, He Kong, and Qi Wang. Tgrpo :fine-tuning vision-language-action model via trajectory-wise group relative policy optimization. *ArXiv*, abs/2506.08440, 2025b. URL https://api.semanticscholar.org/CorpusID:279260912.

Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *kdd*, volume 96, pp. 226–231, 1996.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, et al. Libero-plus: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025a.

Zhaoye Fei, Li Ji, Siyin Wang, Junhao Shi, Jingjing Gong, and Xipeng Qiu. Unleashing embodied task planning ability in llms via reinforcement learning. *ArXiv*, abs/2506.23127, 2025b.

Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. Imagebind: One embedding space to bind them all. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15180–15190, 2023.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025a.

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. *arXiv preprint arXiv:2501.16664*, 2025b.

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. *arXiv preprint arXiv:2507.16815*, 2025.

Chia-Yu Hung, Qi Sun, Pengfei Hong, Amir Zadeh, Chuan Li, U Tan, Navonil Majumder, Soujanya Poria, et al. Nora: A small open-sourced generalist vision language action model for embodied tasks. *arXiv preprint arXiv:2504.19854*, 2025.

Aaron Jaech, Adam Kalai, Adam Lerer, Adam Richardson, Ahmed El-Kishky, Aiden Low, Alec Helyar, Aleksander Madry, Alex Beutel, Alex Carney, et al. Openai o1 system card. *arXiv preprint arXiv:2412.16720*, 2024.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, 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. *ArXiv*, abs/2406.09246, 2024. URL https://api.semanticscholar.org/CorpusID:270440391.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *ArXiv*, abs/2502.19645, 2025. URL https://api.semanticscholar.org/CorpusID:276647709.

Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, et al. Molmoact: Action reasoning models that can reason in space. *arXiv preprint arXiv:2508.07917*, 2025.

Sergey Levine. Sporks of agi, 2024. URL https://sergeylevine.substack.com/p/sporks-of-agi. Substack blog post.

Hao-Si 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. *ArXiv*, abs/2509.09674, 2025. URL https://api.semanticscholar.org/CorpusID:281252318.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36: 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. *arXiv preprint arXiv:2505.19789*, 2025.

Guanxing Lu, Wenkai Guo, Chubin Zhang, Yuheng Zhou, Hao Jiang, Zifeng Gao, Yansong Tang, and Ziwei Wang. Vla-rl: Towards masterful and general robotic manipulation with scalable reinforcement learning. *ArXiv*, abs/2505.18719, 2025. URL https://api.semanticscholar.org/CorpusID:278904856.

Tongzhou Mu, Minghua Liu, and Hao Su. Drs: Learning reusable dense rewards for multi-stage tasks. *arXiv preprint arXiv:2404.16779*, 2024.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

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. *ArXiv*, abs/2501.09747, 2025. URL https://api.semanticscholar.org/CorpusID:275570494.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

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

Junyang Shu, Zhiwei Lin, and Yongtao Wang. Rftf: Reinforcement fine-tuning for embodied agents with temporal feedback. *arXiv preprint arXiv:2505.19767*, 2025.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, et al. Smolvla: A vision-language-action model for affordable and efficient robotics. *arXiv preprint arXiv:2506.01844*, 2025.

Rich Sutton. The bitter lesson. http://www.incompleteideas.net/IncIdeas/BitterLesson.html, 2019. Accessed: 2025-11-12.

Shuhan Tan, Kairan Dou, Yue Zhao, and Philipp Krähenbühl. Interactive post-training for vision-language-action models. *ArXiv*, abs/2505.17016, 2025. URL https://api.semanticscholar.org/CorpusID:278789484.

Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Shenzhi Wang, Le Yu, Chang Gao, Chujie Zheng, Shixuan Liu, Rui Lu, Kai Dang, Xionghui Chen, Jianxin Yang, Zhenru Zhang, et al. Beyond the 80/20 rule: High-entropy minority tokens drive effective reinforcement learning for llm reasoning. *arXiv preprint arXiv:2506.01939*, 2025a.

Siyin Wang, Jinlan Fu, Feihong Liu, Xinzhe He, Huangxuan Wu, Junhao Shi, Kexin Huang, Zhaoye Fei, Jingjing Gong, Zuxuan Wu, Yugang Jiang, See-Kiong Ng, Tat-Seng Chua, and Xipeng Qiu. Roboomni: Proactive robot manipulation in omni-modal context. *ArXiv*, abs/2510.23763, 2025b.

Zhixin Wang, Tianyi Zhou, Liming Liu, Ao Li, Jiarui Hu, Dian Yang, Jinlong Hou, Siyuan Feng, Yuan Cheng, and Yuan Qi. Distflow: A fully distributed rl framework for scalable and efficient llm post-training, 2025c. URL https://arxiv.org/abs/2507.13833.

Xumeng Wen, Zihan Liu, Shun Zheng, Shengyu Ye, Zhirong Wu, Yang Wang, Zhijian Xu, Xiao Liang, Junjie Li, Ziming Miao, et al. Reinforcement learning with verifiable rewards implicitly incentivizes correct reasoning in base llms. *arXiv preprint arXiv:2506.14245*, 2025.

Junjin Xiao, Yandan Yang, Xinyuan Chang, Ronghan Chen, Feng Xiong, Mu Xu, Wei-Shi Zheng, and Qing Zhang. World-env: Leveraging world model as a virtual environment for vla post-training. *arXiv preprint arXiv:2509.24948*, 2025.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025a.

Shihui Yang, Chengfeng Dou, Peidong Guo, Kai Lu, Qiang Ju, Fei Deng, and Rihui Xin. Dcpo: Dynamic clipping policy optimization. *arXiv preprint arXiv:2509.02333*, 2025b.

Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *arXiv preprint arXiv:2503.14476*, 2025.

Hongzhi Zang, Mingjie Wei, Si Xu, Yongji Wu, Zhen Guo, Yuanqing Wang, Hao Lin, Liangzhi Shi, Yuqing Xie, Zhexuan Xu, et al. Rlinf-vla: A unified and efficient framework for vla+ rl training. *arXiv preprint arXiv:2510.06710*, 2025.

Shiduo Zhang, Zhe Xu, Peiju Liu, Xiaopeng Yu, Yuan Li, Qinghui Gao, Zhaoye Fei, Zhangyue Yin, Zuxuan Wu, Yu-Gang Jiang, et al. Vlabench: A large-scale benchmark for language-conditioned robotics manipulation with long-horizon reasoning tasks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11142–11152, 2025.

Zijian Zhang, Kaiyuan Zheng, Zhaorun Chen, Joel Jang, Yi Li, Siwei Han, Chaoqi Wang, Mingyu Ding, Dieter Fox, and Huaxiu Yao. Grape: Generalizing robot policy via preference alignment. *arXiv preprint arXiv:2411.19309*, 2024.

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*, pp. 1702–1713, 2025.

Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daum'e, Andrey Kolobov, Furong Huang, and Jianwei Yang. Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. *ArXiv*, abs/2412.10345, 2024. URL https://api.semanticscholar.org/CorpusID:274762779.

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 T. 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 Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Conference on Robot Learning, CoRL 2023, 6-9 November 2023, Atlanta, GA, USA*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

A Progress Reward Benchmark

Based on the carefully curated multi-task dataset of 700 human-annotated successful trajectories and 300 failure trajectories across diverse task domains, we propose a practical evaluation framework to assess the quality of progress reward functions. All successful trajectories are manually selected to exhibit approximately linear growth characteristics, meaning they demonstrate consistent forward progression without significant regressions or backtracking (e.g., no instances of dropping objects and having to retrieve them). To better test the model's generalization performance, the selected trajectories contain various perturbations including camera viewpoint changes, lighting variations, compounding objects, sensor noise and background distractions. The benchmark focuses on five core metrics that capture the essential properties of effective progress signals across multiple tasks and perturbation scenarios.

A.1 Core Evaluation Metrics

- **Temporal Correlation**: Measures the correlation between progress values and frame numbers across all tasks using Spearman's rank correlation coefficient:

$$\rho = \frac{1}{N}\sum_{k=1}^{N} \frac{\sum_{i=1}^{T_k}(x_i^{(k)} - \bar{x}^{(k)})(y_i^{(k)} - \bar{y}^{(k)})}{\sqrt{\sum_{i=1}^{T_k}(x_i^{(k)} - \bar{x}^{(k)})^2 \sum_{i=1}^{T_k}(y_i^{(k)} - \bar{y}^{(k)})^2}}, \tag{10}$$

  where N is the number of tasks, T_k is the trajectory length for task k, $x_i^{(k)}$ represents frame numbers and $y_i^{(k)}$ represents progress values. Higher absolute values indicate stronger monotonic relationships.

- **Temporal Monotonicity**: Quantifies the average percentage of steps where progress increases across all tasks:

$$M_{\text{mono}} = \frac{1}{N}\sum_{k=1}^{N} \frac{1}{T_k - 1}\sum_{t=1}^{T_k-1} \mathbb{I}(r_{t+1}^{(k)} > r_t^{(k)}), \tag{11}$$

  where $\mathbb{I}$ is the indicator function and $r_t^{(k)}$ is the progress at step t for task k. Values closer to 100 indicate stronger monotonic progression.

- **Distribution Separation (MMD)**: Evaluates the average separation between success and failure trajectories across tasks using Maximum Mean Discrepancy:

$$\text{MMD} = \frac{1}{N}\sum_{k=1}^{N} \left\| \frac{1}{n_k}\sum_{i=1}^{n_k} \phi(R_{s,k}^{(i)}) - \frac{1}{m_k}\sum_{j=1}^{m_k} \phi(R_{f,k}^{(j)}) \right\|_{\mathcal{H}}^2, \tag{12}$$

  where $R_{s,k}$ and $R_{f,k}$ represent final progress values for success and failure trajectories in task k, n_k and m_k are the respective sample sizes, and ϕ is the feature map in reproducing kernel Hilbert space $\mathcal{H}$.

- **Jensen-Shannon Divergence**: Measures the average distributional divergence between success and failure trajectories across tasks:

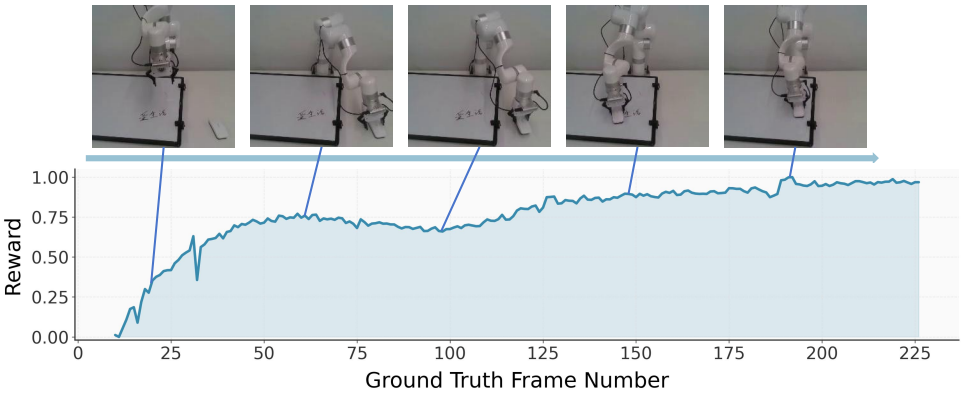
$$\text{JSD} = \frac{1}{N}\sum_{k=1}^{N} \left[\frac{1}{2} D_{\text{KL}}(P_{\text{success}}^{(k)} \| M^{(k)}) + \frac{1}{2} D_{\text{KL}}(P_{\text{failure}}^{(k)} \| M^{(k)}) \right], \tag{13}$$

  where $M^{(k)} = \frac{1}{2}(P_{\text{success}}^{(k)} + P_{\text{failure}}^{(k)})$ is the mixture distribution for task k, and D_{KL} denotes the Kullback-Leibler divergence.

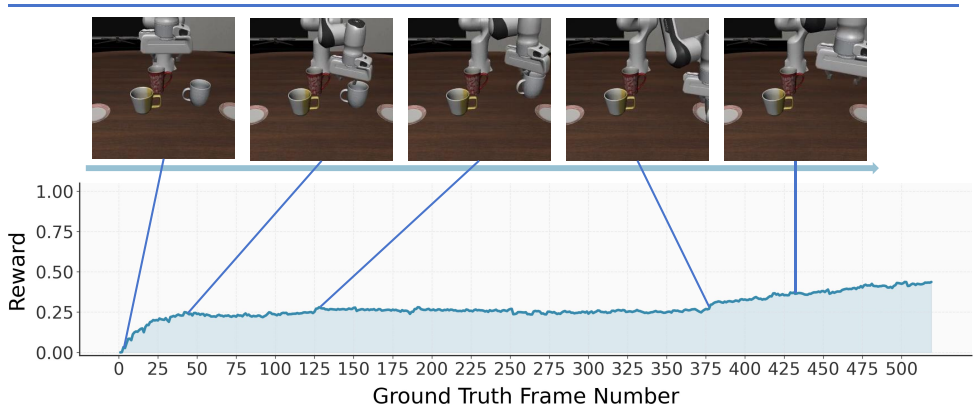
- **Standardized Mean Difference**: Measures the average separation effect size between success and failure trajectories across tasks:

$$\text{SMD} = \frac{1}{N}\sum_{k=1}^{N} \frac{\mu_{\text{success}}^{(k)} - \mu_{\text{failure}}^{(k)}}{\sigma_{\text{pooled}}^{(k)}}, \tag{14}$$

  where $\mu_{\text{success}}^{(k)}$ and $\mu_{\text{failure}}^{(k)}$ are the means for task k, and $\sigma_{\text{pooled}}^{(k)} = \sqrt{\frac{(n_k-1)(\sigma_{\text{success}}^{(k)})^2 + (m_k-1)(\sigma_{\text{failure}}^{(k)})^2}{n_k+m_k-2}}$ is the pooled standard deviation for task k.



(a) Success Trajectory. Task: Wipe the board.



(b) Faliure Trajectory. Task: Put the white mug on the left plate and put the yellow and white mug on the right plate.

Figure 9: Visualization of reward signals between successful and failure trajectories. (a) A success trajectory (task: wipe the board) exhibits a reward curve that rises smoothly; a slight dip in reward corresponds to a pause during grasping. (b) A failure trajectory (task: put mugs on plates) shows a stagnant reward signal, failing to reach a high value, as the model fails to locate the second mug.

A.2 Evaluating Progress Reward Quality

A high-quality progress reward function should exhibit several key characteristics that can be assessed through the following complementary metrics:

- **Temporal Consistency**: The progress values should demonstrate strong *temporal correlation* (ρ) with monotonically increasing patterns (*temporal monotonicity* M_{mono}). Effective progress rewards should show consistent improvement over time, with correlation values approaching 1.0 and monotonicity scores near 100% indicating smooth, predictable progression toward task completion.
- **Distribution Discriminability**: Successful and failed trajectories should be well-separated in the progress reward space. This is quantified through multiple complementary measures:
  - *Maximum Mean Discrepancy (MMD)*: Captures distributional differences in reproducing kernel Hilbert spaces, with larger values indicating better separation between success and failure trajectories.
  - *Jensen-Shannon Divergence (JSD)*: Measures the information-theoretic divergence between success and failure distributions, where values closer to $\ln 2$ suggest maximal discriminability.
  - *Standardized Mean Difference (SMD)*: Provides an effect size measure for the separation between success and failure means, with larger absolute values indicating stronger differentiation.

The combination of these metrics provides a comprehensive framework for evaluating progress reward quality, emphasizing both the internal consistency within successful trajectories and the external discriminability between success and failure outcomes.

B Reward Analysis Details

Reward construction for success traj. To ensure a fair and standardized comparison, we adopt the following normalized procedure for progress reward computation: For SRPO and ImageBind, we compute video embeddings using a cumulative sliding window approach: starting from frames 0-10, we progressively extend the window through frames 1-11, 1-12, and so forth, until the final window spanning frames 1 to the penultimate frame, which generates a sequence of embeddings representing cumulative visual context up to each time step. Then, we compute their L2 distances to the embedding of the entire video sequence, and normalize the progress reward by assigning a value of 0 to the frame with the maximum distance and 1 to the frame with the minimum distance. For the pixel-level method (RLVR), we calculate the L1 distance between each frame from frame 10 to the penultimate frame and the final frame, applying the same normalization scheme.

Reward construction for failure traj. For failure trajectories, we use the cluster centers of successful trajectories as reference points. We compute the minimum L2 distance from each failure trajectory segment to these success cluster centers, then normalize the progress reward by assigning a value of 0 to the segment with the maximum distance and 1 to the segment with zero distance (closest to success patterns).

Reward Curve Analysis. We randomly selected some successful and failure trajectories, plotting frame number-progress reward curves to compare our approach with two baseline methods. As shown in Figure 13 14 15, our analysis reveals that pixel-level rewards fail to properly evaluate long-horizon tasks with multiple sub-tasks and tend to exhibit sharp progress increases only in the final few frames. While general visual encoders like ImageBind can capture trajectory-level features, these features lack physical intuition, resulting in oscillatory progress rewards that are non-smooth and frequently display incorrect sudden spikes or drops—characteristics particularly unfriendly for reinforcement learning. In contrast, our reward method demonstrates more reasonable and stable progress estimation. We also provide detailed reward curves for two trajectories as visualizations in Figure 9.

C Ablations

C.1 Ablation on Self-Referential Mechanism

A core design of our method is its self-referential nature, which assesses progress by comparing within the current policy's own rollouts. We hypothesize that relying on a fixed set of external expert trajectories could constrain the policy's exploration and potentially lead to convergence to suboptimal local minima. To validate this, we ablate the self-referential mechanism by replacing the within-batch successful trajectories with a fixed set of 50 pre-selected expert trajectories per task for progress computation. As illustrated in Figure 10, the ablated variant initially trains slightly slower than our full SRPO method, yet still outperforms GRPO significantly. However, its performance plateaus in later stages, requiring nearly 1.4 times the training steps yet still yielding suboptimal results. This suggests two key insights: (1) The *progress-wise* reward, even when computed from external trajectories, provides a more effective learning signal than sparse binary rewards, explaining the initial efficiency gain over GRPO, while the introduction of external information eventually allows GRPO, which requires no such information, to catch up it in performance. (2) The fixed external references ulti-

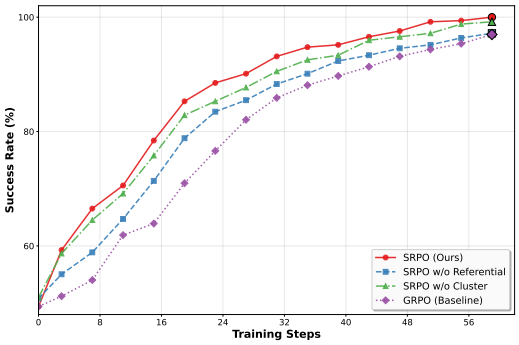


Figure 10: Ablation study on Object suite. We compare our SRPO method against its ablated variants. Removing the referential component (w/o Referential) leads to significant performance drop, while removing the clustering component (w/o Cluster) slows down convergence.

mately limit the policy's capacity for open-ended exploration. As the policy evolves, these static trajectories fail to provide nuanced, step-by-step progress assessments for the diverse rollouts generated by the policy, leading to a performance ceiling lower than that achieved by our self-referential approach.

C.2 Ablation on Success Clustering

We introduce clustering of successful trajectories to compute progress rewards based on two primary motivations. First, a task can often be accomplished through multiple distinct strategies (e.g., placing object A before B, or vice versa). A failure trajectory should be compared to the nearest *successful strategy* rather than an arbitrary one. Second, using the centroid of a cluster, rather than the single nearest successful trajectory, provides a more robust distance measure. Individual success trajectories might contain sub-optimal or noisy segments (e.g., a gripper moving momentarily away from an object before successfully grasping it). If a failure trajectory shares a similar initial deviation, measuring the distance solely to this specific, noisy success trajectory could yield an inaccurately high progress score. Using a cluster centroid mitigates this by representing a more prototypical and cleaner version of a success strategy.

To substantiate these points, we conduct an ablation where progress is computed using the distance to the single nearest success trajectory instead of the cluster centroid. The results are presented in Figure 10. We observe that this variant's initial learning efficiency is comparable to our full SRPO method. However, its performance gains diminish significantly as training progresses. We attribute this to the following: in early training stages, the repertoire of successful strategies is limited, and the number of success trajectories is small, thus diminishing the advantage of clustering. In later stages, as successful exploration diversifies and the number of success trajectories grows substantially, the ability of clustering to distill prototypical strategies and provide robust progress signals becomes crucial, leading to the increasing performance gap observed in our results.

D Hyperparameter Analysis

In Equation (5), we employ an activation function to map the reward trajectory into the range (0, 1). In our implementation, we use a sigmoid function and precede it with a scaling coefficient α to control the trade-off between progress awareness and outcome correctness in our reward function. We now conduct a comprehensive analysis of this hyperparameter α. The reward design provides full credit (1.0) for correct answers and scales the progress reward by α otherwise. We evaluate five different values of α: 0, 0.3, 0.5, 0.8, and 1.0.

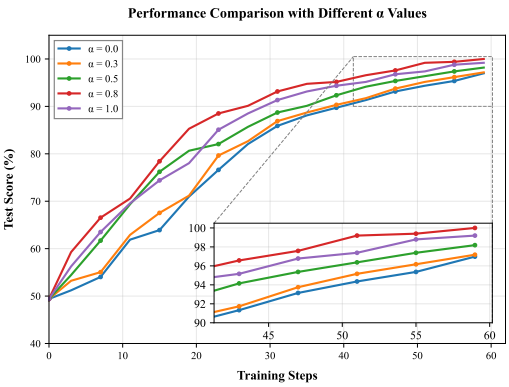


Figure 11: Performance comparison with different α values in the reward function. The results demonstrate that $\alpha = 0.8$ achieves the best performance, followed by $\alpha = 1.0$, $\alpha = 0.5$, $\alpha = 0.3$, and $\alpha = 0$ in descending order. This validates the importance of balancing progress awareness with outcome correctness in our reward design.

The experimental results reveal a clear performance hierarchy: $0 < 0.3 < 0.5 < 1.0 < 0.8$. This pattern provides several important insights:

$\alpha = 0$ **(no progress reward)**: Performs worst, confirming that purely outcome-based rewards are insufficient for complex tasks requiring sequential reasoning.

$\alpha = 0.3$ and 0.5: Show gradual improvement, indicating that even small progress rewards enhance learning efficiency.

$\alpha = 1.0$ **(equal weighting)**: Performs better than lower values but suboptimally, suggesting that over-emphasizing progress may distract from the final objective.

$\alpha = 0.8$ **(optimal)**: Achieves the best performance, demonstrating that a strong but not exclusive focus on progress rewards (80% of maximum) provides the ideal balance for guiding the learning process while maintaining focus on task completion.

This analysis validates our reward design principle: properly weighted progress awareness ($\alpha = 0.8$) significantly outperforms both purely outcome-based rewards ($\alpha = 0$) and excessively progress-focused rewards ($\alpha = 1.0$). The optimal α value enables the agent to benefit from intermediate guidance while remaining oriented toward the ultimate task objective.

E What if we use a pixel-level world model for reward shaping?

Another promising avenue for reward shaping involves leveraging pixel-level world models to generate reference trajectories based on language instructions or action priors. Since these priors are already inputs to the policy or RL algorithm, this approach does not strictly constitute introducing external information. In this section, we use Cosmos-Predict2 (Ali et al., 2025) as a case study to elucidate why this paradigm, while intuitive, often falls short in practice.

Zero-Shot Generation Yields Unsatisfactory Results. We experimented with the large-scale Cosmos-Predict2-14B model in a zero-shot setting on several tasks from the LIBERO benchmark. Using the language instruction as conditioning signals, the model was tasked with generating a reference video trajectory. As shown in Figure 16, the generated videos suffer from poor scene consistency. Furthermore, although task-specific supervised fine-tuning (SFT) may mitigate these inconsistencies and improve generation quality, this high-cost approach, which relies heavily on extensive expert demonstrations, is clearly inferior to the reward modeling paradigm proposed by SRPO, which is based on a latent world representation and proves to be more cost-effective and generalizable.

F Training Details

F.1 Supervised Fine-Tuning (SFT) Stage

Our one-shot SFT phase builds upon the OpenVLA (with Action Chunking and Parallel Decoding) checkpoint. The training was conducted on $8 \times$ A100 GPUs. Key training configurations are as follows:

- **Optimizer:** AdamW
- **Learning Rate:** 5×10^{-4}
- **Batch Size:** 8
- **Maximum Training Steps:** 150,005
- **Learning Rate Decay:** Applied after 100,000 steps
- **LoRA Rank:** 32
- **Image Augmentation:** Enabled
- **Input Modalities:** Text instruction and single image observation
- **Action Chunking:** 8 action chunks

F.2 SRPO Post-Training Stage

The SRPO reinforcement learning phase is initialized from the one-shot SFT model. Key hyperparameters include:

- **Algorithm:** SRPO (based on GRPO advantage estimation)
- **Learning Rate:** 5×10^{-6}
- **Samples Per Group:** 8
- **Batch Size:** 64 ($\times 8$, training), 496 (validation)
- **Mini-batch Size:** 128
- **Progress Reward Weight:** 0.8
- **Number of Trials per Task:** 50

- **Action Configuration:** 7 action tokens, 8 action chunks
- **Context Length:** 512 tokens (prompt), 128 tokens (response)
- **Trajectory Mini-batch Size:** 16
- **Log Probability Batch Size:** 8 (for both rollout and reference model)
- **FP16 Inference:** Enabled for the video embedding model
- **Model Offloading:** Enabled to optimize GPU memory usage

F.3 Model Details

This modified architecture retains the Llama 2 backbone for generating discrete action tokens, in contrast to the continuous action heads employed in OpenVLA-OFT. While this design choice may potentially sacrifice some level of action precision compared to continuous output methods, it provides the crucial advantage of enabling direct access to action log-probabilities essential for policy gradient methods in reinforcement learning.

G Real-world Experiment Details

G.1 Detailed Setup

In our physical robot experiments, due to safety concerns associated with online exploration and the significant time cost of manual resets, we adopted an offline reinforcement learning (RL) paradigm. Specifically, our technical approach integrates the Advantage-Weighted Regression (AWR) strategy (Peng et al., 2019) with SRPO's self-referential progress-wise reward and advantage mechanism. We first collect demonstration data and store it in a trajectory buffer. The expected cumulative reward (or value) $R_{i,t}$ for the i-th trajectory at step t is computed as in (2)–(5). We then define the incremental progress at step t as $D_{i,t} = R_{i,t} - R_{i,t-1}$. Following the SRPO framework, the advantage function is calculated as:

$$A_{i,t} = \frac{D_{i,t} - \mu}{\sigma}, \tag{15}$$

where μ and σ are the mean and standard deviation of $D_{i,t}$ computed across all trajectories. Our real-world task policy is trained from scratch via RL starting from pretrained weights, with the baseline for comparison being the SFT model.

The pick-and-place tasks include *put apple into the plate* and *put pear into the plate*. To test robust object identification, we place multiple fruits on the table initially, requiring the model to distinguish target objects from distractors. During task execution, we randomly swap positions between target and distractor objects to assess the model's capacity for rapid adaptation.

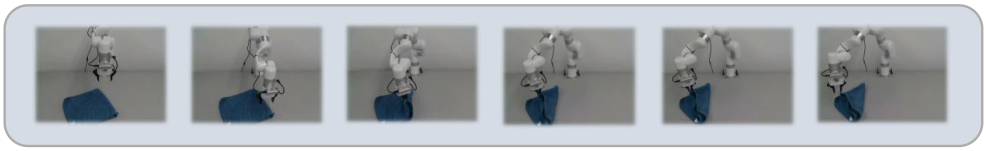
To specifically demonstrate the advantages of dense progress-aware rewards in complex manipulation, we incorporate *folding towels* involving deformable object manipulation and *cleaning whiteboard* requiring coordinated surface contact. Furthermore, to evaluate semantic understanding capabilities, we include the *select poker* task where the model must identify the specified card from these five playing cards: Joker, Jack of Spades, King of Club, Jack of Spades, 10 of Spades.

Our real-world experimental results demonstrate significant performance improvements across all five manipulation tasks when applying progress-aware value weighting. As shown in Figure 8, both VLA policy backbones exhibit substantial gains over their SFT counterparts.

Table 4: Progress Reward Benchmark results on real-robot datasets. Our method maintains strong performance across all tasks and metrics, demonstrating robust generalization to diverse real-world manipulation tasks.

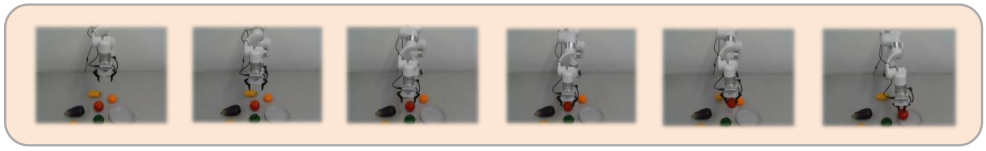
Task	SC	Mono	MMD	JS	SMD
Put Apple	0.987	0.975	0.589	0.562	165.3
Put Pear	0.991	0.982	0.601	0.578	172.8
Fold Towel	0.984	0.968	0.572	0.549	158.6
Wipe Board	0.993	0.986	0.624	0.591	181.2
Select Poker	0.989	0.979	0.595	0.569	169.5
Average	0.989	0.978	0.596	0.570	169.5

(a) Put the pear into the plate.

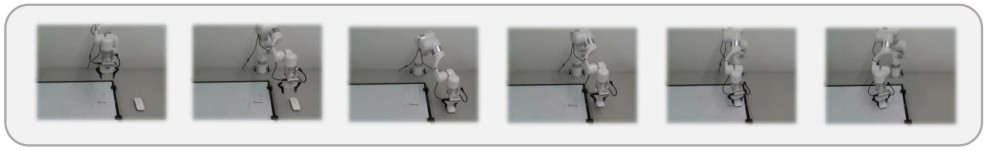


(b) Fold the towel.

(c) Pick up the Joker.



(d) Put the apple into the plate.



(e) Wipe the board.

Figure 12: Success cases of the real-world experiment.

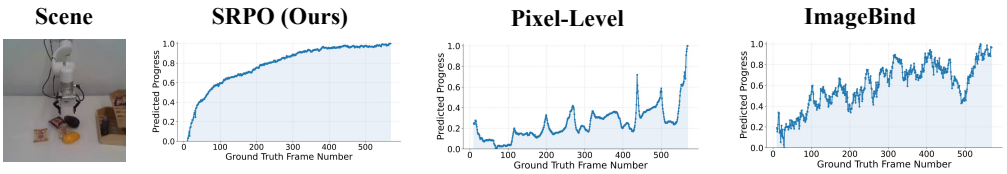
G.2 Rewarding Validation

To quantitatively validate the generalization capability of our progress reward model to real-world scenarios, we evaluate our pre-trained progress reward function on five real-robot manipulation tasks, each containing 30 successful trajectories and 20 failure trajectories. The comprehensive Progress Reward Benchmark results across all tasks demonstrate consistent performance in real-world settings:

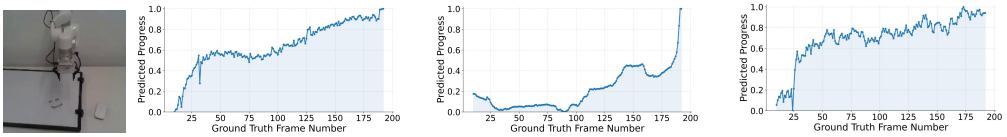
As shown in Table 4, the consistently high Progress Reward Quality scores across all five real-robot tasks confirm that our latent progress reward modeling effectively captures task progression dynamics in real-world robotic manipulation. With near-perfect Spearman correlation and monotonicity, coupled with strong distribution separation in MMD and SMD, our method demonstrates robust generalization despite domain shifts from simulation to reality.

G.3 Case Study

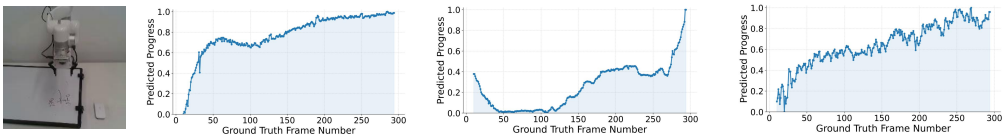
As shown in Figure 12, our real-world case studies demonstrate the robustness and generalization of our approach across diverse scenarios: in *put the pear into the plate*, the policy dynamically replans when the target is moved mid-execution; in *fold the towel*, it reliably handles deformable object manipulation; and in *pick up the Joker card*, it maintains precise semantic understanding to identify the target among multiple options. These results validate the adaptability and task-aware capability of our method in challenging real environments.



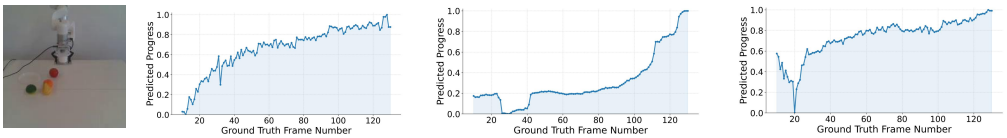
(a) Manage the table. The robot smoothly approaches the target.



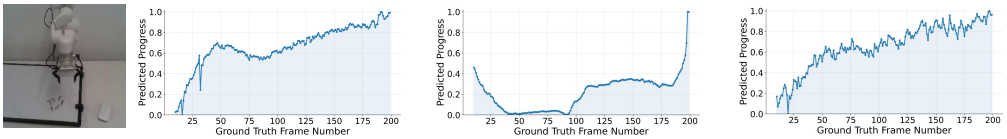
(b) Wipe the board. Pausing during grasping can lead to a decrease in progress.



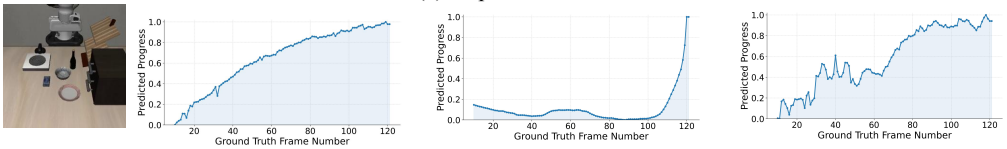
(c) Wipe the board.



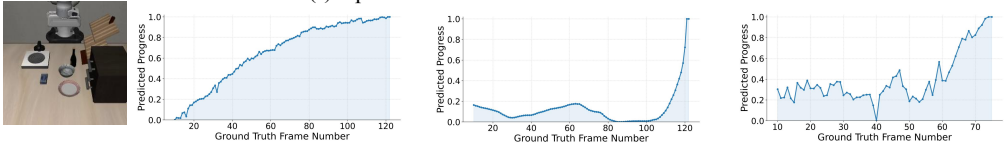
(d) Put the apple into the plate.



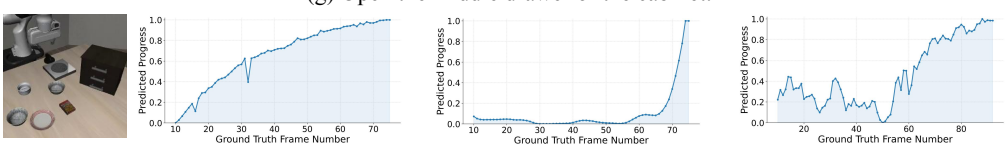
(e) Wipe the board.



(f) Open the middle drawer of the cabinet.

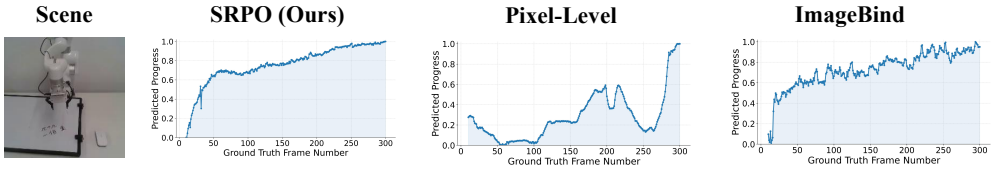


(g) Open the middle drawer of the cabinet.

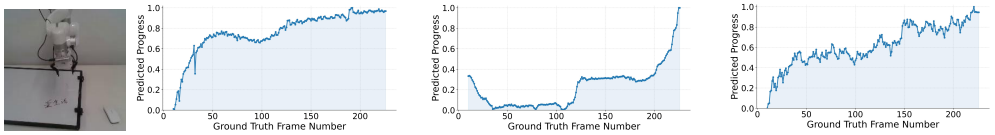


(h) Pick up the black bowl from table center and place it on top of the cabinet.

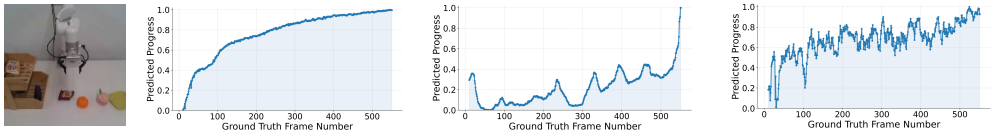
Figure 13: Reward curves of different methods on successful trajectories (Part One).



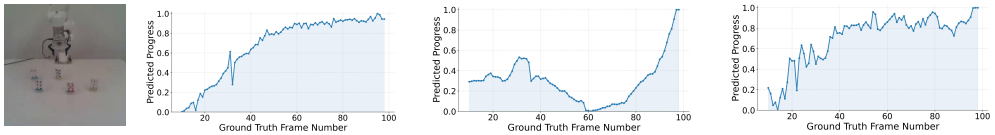
(a) Wipe the board. During moving the eraser, there was a process of dropping and picking it up again.



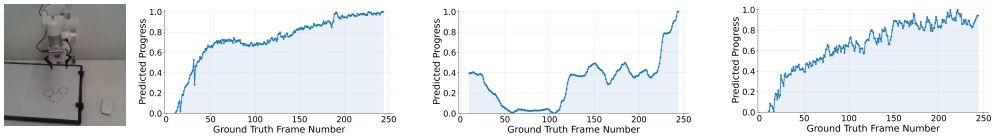
(b) Wipe the board. We found that pausing during grasping can lead to a decrease in progress, as the model might consider this action to be unrelated to the final goal.



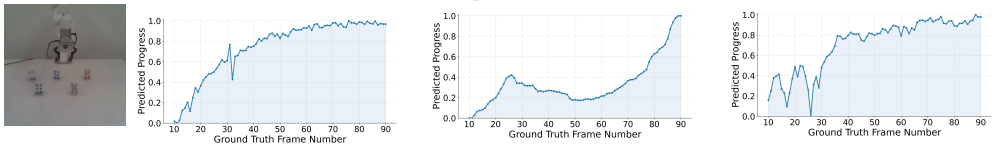
(c) Manage the table. The robot smoothly approaches the target.



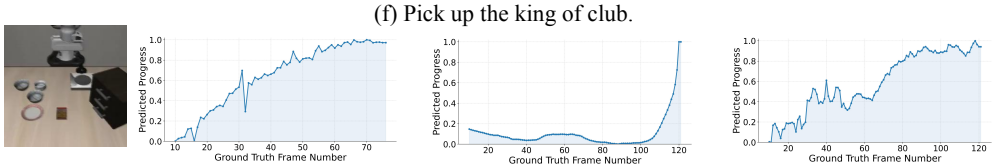
(d) Pick up the joker.



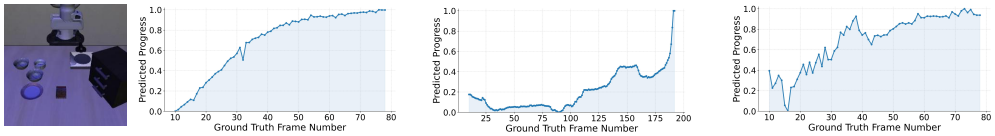
(e) Wipe the board.



(f) Pick up the king of club.

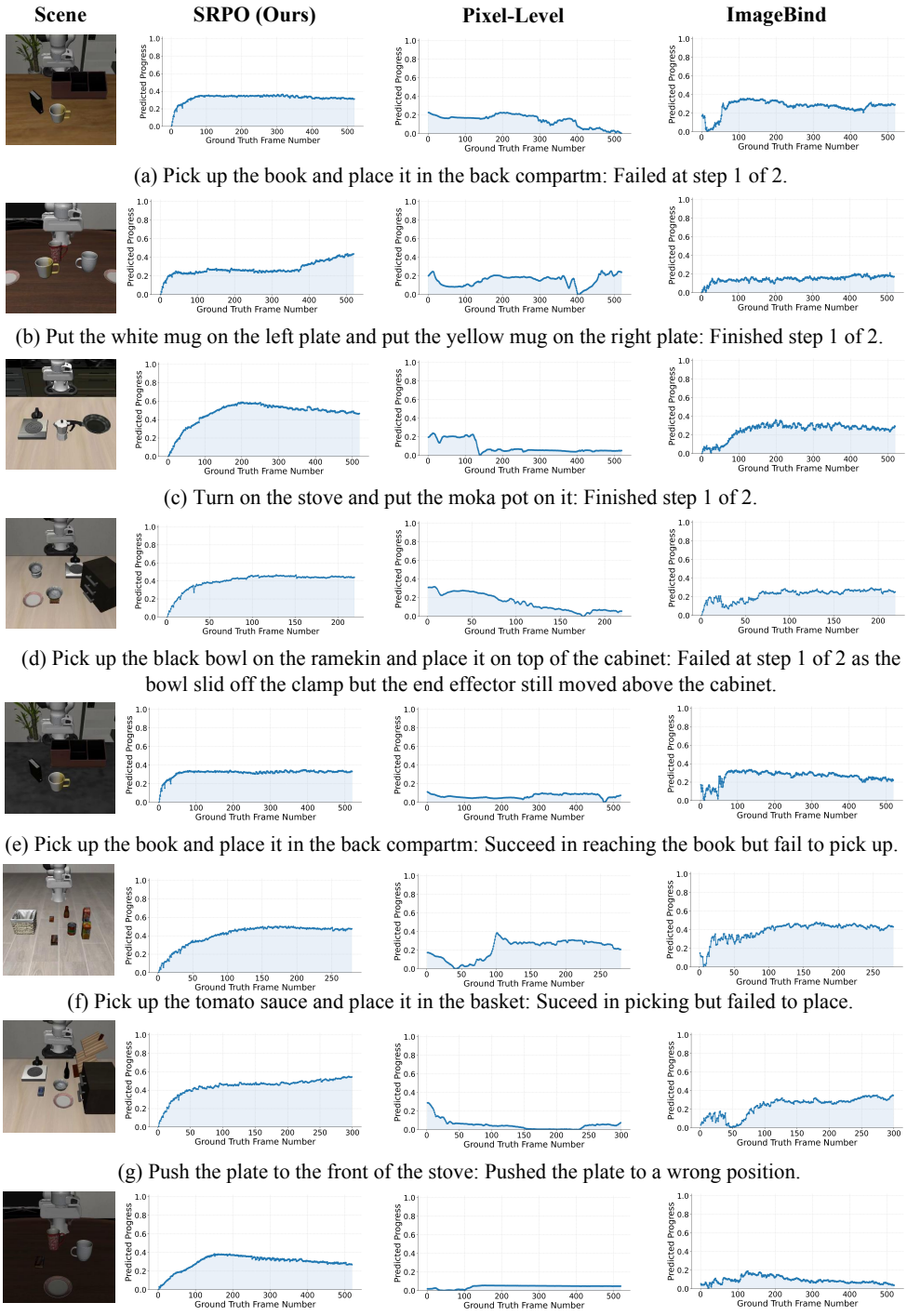


(g) Open the middle drawer of the cabinet + Sensor Noise



(h) Pick up the black bowl from table center and place it on top of the cabinet + Light Perturbation

Figure 14: Reward curves of different methods on successful trajectories (Part Two).



(a) Pick up the book and place it in the back compartm: Failed at step 1 of 2.

(b) Put the white mug on the left plate and put the yellow mug on the right plate: Finished step 1 of 2.

(c) Turn on the stove and put the moka pot on it: Finished step 1 of 2.

(d) Pick up the black bowl on the ramekin and place it on top of the cabinet: Failed at step 1 of 2 as the bowl slid off the clamp but the end effector still moved above the cabinet.

(e) Pick up the book and place it in the back compartm: Succeed in reaching the book but fail to pick up.

(f) Pick up the tomato sauce and place it in the basket: Suceed in picking but failed to place.

(g) Push the plate to the front of the stove: Pushed the plate to a wrong position.

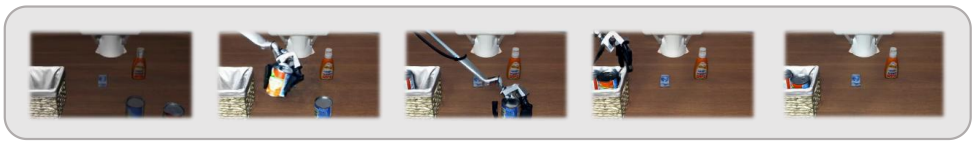
(h) Put the white mug on the plate and put the chocolate pudding to the right of the plate: mug has been put on the proper position but was knocked over when being released.

Figure 15: Reward curves of different methods on failure trajectories.

(a) Pick up the book and place it in the back compartment of the caddy.

(b) Put both moka pots on the stove.

(c) Put the black bowl in the bottom drawer of the cabinet and close it.



(d) Put both the alphabet soup and the cream cheese box in the basket.

(e) Put the yellow and white mug in the microwave and close it.

(f) Manage the table.

(g) Wipe the board.

Figure 16: Trajectories generated by Cosmos-Predict2 (Ali et al., 2025).