# ESCAPO: ENTROPY-GUIDED SEGMENT-LEVEL CREDIT ASSIGNMENT FOR POLICY OPTIMIZATION IN VLA POST-TRAINING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Long-horizon vision-language-action (VLA) policies are increasingly post-trained with on-policy reinforcement learning using sparse terminal rewards. Critic-free methods such as Group Relative Policy Optimization (GRPO) derive a trajectory-level advantage from these outcomes, providing only coarse temporal credit: a failed rollout may contain useful intermediate behavior, yet every action receives the same advantage. We find that the policy itself provides a finer signal. Human-annotated rollout analysis shows that, after supervised fine-tuning (SFT), locally successful actions tend to have lower policy entropy than failed actions within the same trajectory, and that this separation becomes stronger during on-policy RL post-training. Based on this observation, we propose *Entropy-guided Segment-level Credit Assignment for Policy Optimization* (ESCAPO), which uses policy entropy to refine GRPO advantages without a critic, dense reward, teacher policy, or action-quality labels for training. In failed rollouts, reliable low-entropy segments receive increased credit while high-entropy segments are penalized; in successful rollouts, uncertain actions supported by successful outcomes receive additional positive credit. On LIBERO-10 and LIBERO-Plus-10, ESCAPO improves long-horizon learning over GRPO from both SFT and shared RL warm-up initializations. From SFT, it raises late-stage success from $46.9\%$ to $54.3\%$ and from $27.7\%$ to $31.4\%$, respectively.

## 1 INTRODUCTION

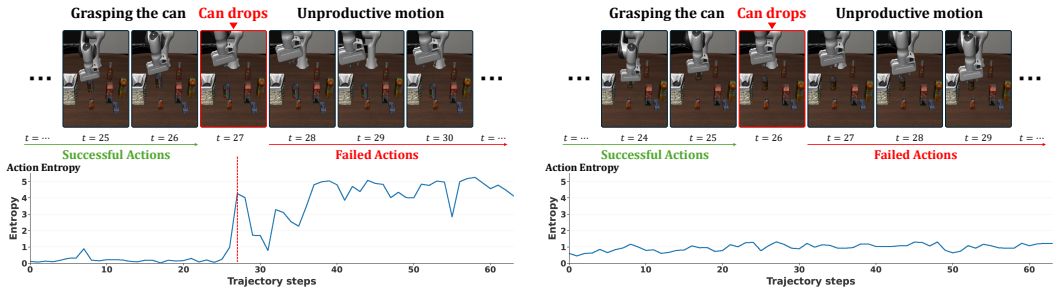
Vision-language-action (VLA) policies map language instructions and visual observations to robot actions (Zitkovich et al., 2023; Kim et al., 2025b; Black et al., 2025). Supervised fine-tuning (SFT) on expert demonstrations provides a strong initialization, but at deployment the policy encounters states induced by its own actions, where small errors can compound beyond the demonstration distribution (Ross & Bagnell, 2010). Online reinforcement learning (RL) can address this distribution shift by training on policy-generated rollouts with sparse terminal task rewards (Li et al., 2026; Lu et al., 2025; Zang et al., 2025).

A central difficulty is temporal credit assignment. A failed long-horizon rollout may still contain useful intermediate behavior—for example, a successful grasp and transport followed by a late placement error—but the terminal outcome does not reveal which actions were responsible. Critic-free methods such as Group Relative Policy Optimization (GRPO) avoid learning a value function, but with trajectory-level rewards they assign the same advantage to every action in a rollout. Consequently, a failed trajectory penalizes useful and erroneous behavior alike. Dense feedback or actor–critic methods can provide finer credit, but typically require additional supervision or value estimation (Zhong et al., 2026; Schulman et al., 2017).

Can the policy itself provide a finer within-trajectory signal? We find that policy entropy is already informative of local action quality. In human-annotated rollouts, locally successful actions tend to have lower entropy than failed actions at SFT initialization, and this discrimination strengthens during both GRPO and PPO post-training (Fig. 1). Entropy is not a perfect correctness indicator, so we do not treat it as a surrogate reward. Instead, we use it to refine how terminal outcomes assign credit within a trajectory.



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




(a) RL post-trained model (b) SFT model

Figure 1: **Policy entropy before and after RL post-training.** Locally successful actions tend to have lower entropy than failed actions. The distinction is stronger after RL (a) than after SFT alone (b).

Motivated by this observation, we introduce *Entropy-guided Segment-level Credit Assignment for Policy Optimization* (ESCAPO), which augments GRPO with outcome-conditioned entropy corrections. In failed rollouts, reliable low-entropy segments receive increased credit while high-entropy segments are penalized. In successful rollouts, uncertain actions supported by successful outcomes receive additional positive credit. ESCAPO modifies only the advantage construction and retains the clipped GRPO update, with terminal success as the only environment reward and no additional critic, dense reward model, teacher policy, action-quality labels for training, or global entropy bonus.

**Our contributions are threefold**: (i) We identify policy entropy as a within-trajectory cue for local action quality already at SFT initialization, and show through human-annotated rollout analysis that this discrimination strengthens during both GRPO and PPO post-training. (ii) We propose ESCAPO, which turns this entropy structure into outcome-conditioned, segment-level credit assignment by refining trajectory-level GRPO advantages differently for successful and failed rollouts. It retains sparse terminal rewards as the only environment supervision and requires no additional critic, dense reward model, teacher policy, or action-quality labels for training. (iii) We evaluate ESCAPO on LIBERO-10, LIBERO-Plus-10, and LIBERO-Spatial; it improves GRPO on the two long-horizon benchmarks from SFT initialization, with gains persisting when training continues from shared GRPO warm-up checkpoints.

## 2 Problem Formulation

A categorical VLA policy $\pi_\theta$ maps a language instruction $x$ and visual/proprioceptive observation $o_t$ to an action chunk $\mathbf{a}_t = (a_{t,c,d})_{c=1,\dots,C;d=1,\dots,D}$. Here, $t$ indexes policy decisions, $c$ positions within a chunk, and $d$ action dimensions; tokens belong to a discrete vocabulary $\mathcal{V}$ and are decoded in parallel. The policy is initialized by SFT on expert demonstrations. Online RL then optimizes expected terminal success $R_i = R(\tau_i) \in \{0, 1\}$ under policy-induced observations. Appendix A gives the policy factorization, SFT objective, and expanded PPO/GRPO formulations.

Unlike PPO (Schulman et al., 2017), which learns a critic, GRPO (Shao et al., 2024) computes advantages from a group $\mathcal{G}_x = \{\tau_i\}_{i=1}^G$ of $G > 1$ behavior-policy rollouts for each instruction $x$:

$$A_i^{\mathrm{GRPO}} = \frac{R_i - \bar{R}_x}{\sigma_x + \epsilon_{\mathrm{num}}}, \tag{1}$$

where $\bar{R}_x$ and $\sigma_x$ are the group reward mean and sample standard deviation, and $\epsilon_{\mathrm{num}} > 0$. This advantage is shared by every valid action token in a rollout. Mixed-outcome groups therefore penalize all tokens in a failed trajectory equally; uniform-outcome groups give zero advantages.

Let $\pi_{\theta_{\mathrm{old}}}$ be the behavior policy. The token-level probability ratio and clipped surrogate are

$$\rho_{i,t,c,d}(\theta) = \frac{\pi_\theta(a_{i,t,c,d} \mid o_{i,t}, x, c, d)}{\pi_{\theta_{\mathrm{old}}}(a_{i,t,c,d} \mid o_{i,t}, x, c, d)}, \tag{2}$$

$$s_\epsilon(\rho, A) = \min\{\rho A, \mathrm{clip}(\rho, 1 - \epsilon_{\mathrm{clip}}, 1 + \epsilon_{\mathrm{clip}})A\}, \tag{3}$$

where $\epsilon_{\mathrm{clip}}$ is the clipping threshold. The mask $m_{i,t,c}$ selects valid action positions, with $M_i = \sum_{t,c,d} m_{i,t,c}$ valid tokens. We use $\mathbb{E}_{\mathrm{old}}$ for expectations over $x \sim p_{\mathrm{task}}$ and groups sampled using

the behavior policy, with $p_{\text{task}}$ the task-instruction distribution. Our GRPO formulation has no value loss or additional KL penalty.

## 3 Entropy Dynamics During RL Post-Training

We study how policy entropy evolves from SFT initialization through on-policy RL post-training and how it relates to local action quality.

For each categorical action token, we compute the Shannon entropy of its predictive distribution:

$$H_{i,t,c,d} = -\sum_{v} p_{i,t,c,d}(v) \log p_{i,t,c,d}(v), \tag{4}$$

where $p_{i,t,c,d}(v)$ is the probability of token $v$ at chunk position $c$ and action dimension $d$. Averaging over action dimensions gives $H^{\text{act}}_{i,t,c} = \frac{1}{D}\sum_{d=1}^{D} H_{i,t,c,d}$. We then obtain the chunk-level entropy $H_{i,t} = \frac{1}{C}\sum_{c=1}^{C} H^{\text{act}}_{i,t,c}$, yielding a temporal entropy profile for each rollout. Intuitively, higher policy entropy indicates greater uncertainty in action prediction.

**Analysis setup.** We analyze failed on-policy rollouts ($R_i = 0$) from GRPO training on LIBERO-Long and LIBERO-Plus-Long, and PPO training on LIBERO-Long. The GRPO datasets contain 500 trajectories, while the PPO dataset contains 1,200. We manually annotate the local execution quality of each action chunk. Given the observation and instruction, a *successful action* correctly advances the task or maintains valid execution; a *failed action* does neither. These labels describe local behavior, so a failed rollout can still contain many successful actions.

At checkpoint $k$, let $\mathcal{I}_k$ contain the selected trajectories with both action classes. We report the mean within-trajectory entropy gap, $D_k = \frac{1}{|\mathcal{I}_k|}\sum_{i\in\mathcal{I}_k}(\bar{H}^{\text{fail}}_i - \bar{H}^{\text{succ}}_i)$, where $\bar{H}^{\text{fail}}_i$ and $\bar{H}^{\text{succ}}_i$ are the respective class means. A larger positive gap indicates greater mean entropy separation. We also compute ROC-AUC within each included trajectory using negative entropy and average with equal trajectory weights. ROC-AUC measures how often a successful action has lower entropy than a failed action drawn from the same trajectory, with ties counted as one half. Annotation and statistical details are provided in the appendix.

### 3.1 Entropy and Action Quality after SFT

SFT learns from expert demonstrations of valid task execution and encourages concentrated predictions around expert actions on familiar observations. Execution errors can move the policy away from the expert state distribution and lead to further errors (Ross et al., 2011). Prior work has observed higher predictive entropy on out-of-distribution inputs (Lakshminarayanan et al., 2017; Ovadia et al., 2019). Intuitively, execution within the expert distribution is more likely to involve successful actions with low entropy. A transition to out-of-distribution states may instead coincide with failed actions and rising entropy.

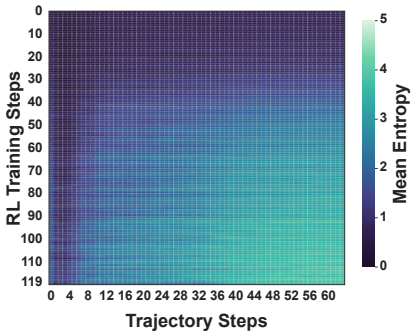
Figure 4 shows that SFT initialization yields mean AUC values of 0.82 on LIBERO-Long in both the GRPO and PPO datasets, and 0.73 on LIBERO-Plus-Long. Successful actions therefore tend to have lower entropy than failed actions even before RL post-training, although the two classes still overlap.

### 3.2 Evolution of Entropy Separation during RL

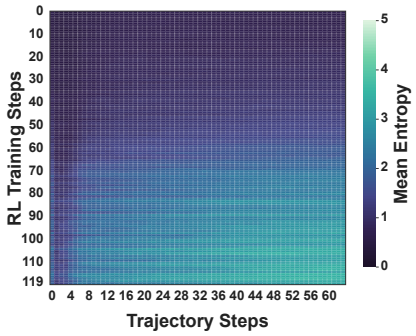
**An analytical view.** On-policy RL methods such as PPO and GRPO use reward-derived advantage estimates to encourage sampled actions with positive advantages and discourage those with negative advantages. Following the tabular softmax analysis of Cui et al. (Cui et al., 2025), consider a categorical token distribution $p_k(v)$ at a fixed policy input and token position. Let $A_k(v)$ denote the expected return from selecting token $v$ relative to the policy's average return at that input. For a sufficiently small vanilla policy-gradient update, the entropy change satisfies

$$H_{k+1} - H_k \approx -\eta\ \mathrm{Cov}_{v\sim p_k}\left(\log p_k(v),\, p_k(v)A_k(v)\right), \tag{5}$$

where $\eta > 0$ is the update step size. The covariance relates entropy change to the alignment between token probabilities and advantages. The chunk-level change follows by averaging over its action tokens.




(a) LIBERO-10




(b) LIBERO-Plus-10

Figure 3: **Temporal policy entropy during GRPO training.** Chunk-level entropy profiles in failed rollouts at different training checkpoints.

SFT produces concentrated predictions around expert actions, with relatively low policy entropy. Under policy-induced observations, some high-probability choices may nevertheless produce incorrect behavior. When these choices receive negative advantages, $p_k(v)A_k(v)$ becomes negative at tokens with relatively high $\log p_k(v)$. This can make the covariance in Eq. (5) negative and increase entropy: suppressing previously dominant choices makes the distribution less concentrated. Task-progressing behaviors can also receive positive advantages, which may help maintain concentrated predictions around successful behaviors. These effects suggest that on-policy RL can widen the mean entropy gap between successful and failed actions. The magnitude can depend on the advantage estimates and policy updates, motivating comparisons under both GRPO and PPO.

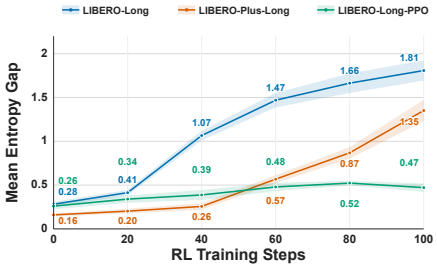



Figure 2: **Entropy gap during RL post-training.** Mean within-trajectory entropy gap in failed rollouts

**Growing entropy separation.** Figure 2 shows substantial growth in the trajectory-averaged entropy gap during GRPO training. From step 0 to 100, the gap increases from 0.28 to 1.81 on LIBERO-Long and from 0.16 to 1.35 on LIBERO-Plus-Long. PPO on LIBERO-Long shows a more modest increase, from 0.26 to 0.47, with a peak of 0.52 at step 80. Both algorithms therefore produce larger gaps at step 100 than at initialization, but the growth magnitudes differ substantially. The heatmaps in Fig. 3 show the corresponding temporal entropy profiles across chunk positions and training checkpoints.

**Discrimination of action quality.** Figure 4 reports trajectory-averaged ROC-AUC. Under GRPO, it increases from 0.82 to 0.93 on LIBERO-Long and from 0.73 to 0.89 on LIBERO-Plus-Long between steps 0 and 100. Under PPO on LIBERO-Long, it increases from 0.82 to 0.94. Discrimination is therefore stronger at step 100 than at SFT initialization in all three settings, although the improvement is not monotonic. For example, PPO reaches an AUC of 0.98 at step 80 before decreasing to 0.94. Its strong AUC improvement despite modest gap growth shows why the two metrics should be considered together: a larger difference in class means does not directly determine how consistently entropy ranks action quality. The appendix provides within-epoch pooled statistics and annotation sample sizes.

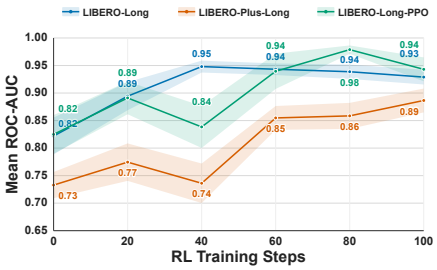



Figure 4: **Action-quality discrimination during on -policy RL post-training.** Mean within-trajectory ROC-AUC using negative policy entropy.

These results support using policy entropy to refine credit assignment within failed trajectories, while retaining reward-derived advantages as the underlying learning signal.

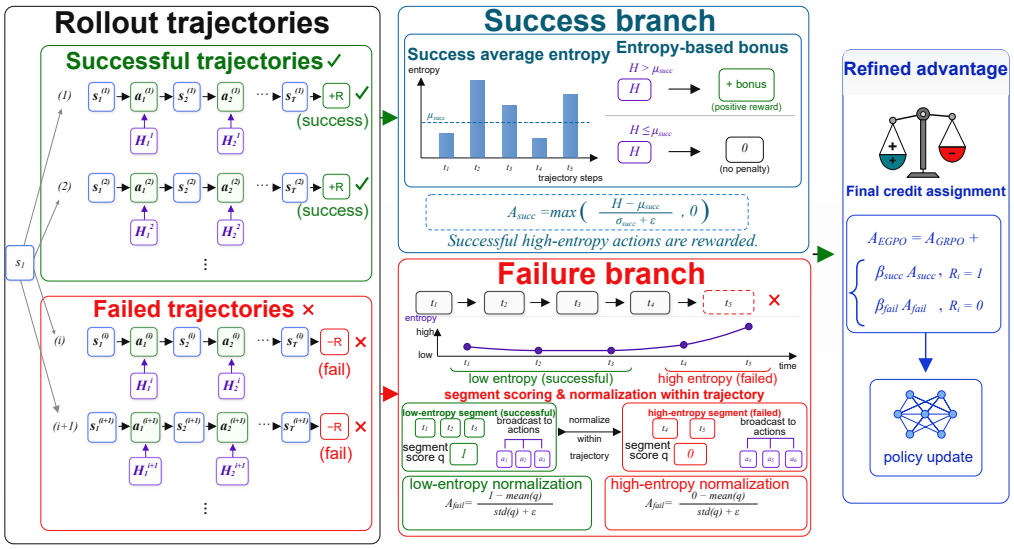



Figure 5: **ESCAPO post-training pipeline.** Failed rollouts receive entropy-guided segment-level advantage corrections. Successful rollouts receive additional positive credit for actions with above-baseline entropy. These corrections augment trajectory-level GRPO advantages for the clipped policy update.

# 4 Entropy-Guided Credit Assignment Policy Optimization

## 4.1 Entropy Modes in Trajectories

The analysis in Sec. 3 motivates refining GRPO's trajectory-level credit with policy entropy. ESCAPO augments $A_i^{\mathrm{GRPO}}$ with outcome-dependent corrections (Fig. 5). The failure branch uses chunk entropy $H_{i,t}$ to identify contiguous LOW and HIGH regions. The success branch uses action-position entropy $H_{i,t,c}^{\mathrm{act}}$ to reinforce high-entropy actions in successful rollouts.

## 4.2 Entropy-Guided Advantage Augmentation

**Failed trajectories.** A failed rollout can contain useful actions that receive the same negative GRPO advantage as its failed actions. To distinguish them across changing entropy scales, we define the trajectory-relative threshold $T_{H,i} = \mu_i - \alpha\sigma_i$, where $\mu_i$ and $\sigma_i$ are the mean and population standard deviation of valid-chunk entropy. Chunks below the threshold are LOW; the others are HIGH. Larger $\alpha > 0$ selects LOW chunks more conservatively. After bridging brief HIGH gaps, we retain LOW segments of length at least $\ell_{\min}$ and all HIGH segments. Short LOW segments receive zero correction. Since entropy separation can be weak early in training, the gate $g_i$ activates corrections only when validity and duration checks pass and the mean HIGH-minus-LOW entropy gap satisfies $\Delta H_i \geq \delta_H$. As shown by the gate-activation curves in Figure B.1, an increasing fraction of failed rollouts passes these checks during GRPO training. This indicates that more rollouts develop sustained LOW/HIGH regions with sufficient entropy separation. Otherwise, the original GRPO advantage is retained. Appendix C specifies these checks.

For the participating segments $\mathcal{S}_i = \{S_{i,1}, \ldots, S_{i,K_i}\}$, we assign $q_{i,j} = 1$ to LOW and $q_{i,j} = 0$ to HIGH. Let $\mu_i^{\mathrm{seg}}$ and $\sigma_i^{\mathrm{seg}}$ be their mean and population standard deviation, with each segment weighted equally. The normalized correction is broadcast to its member chunks:

$$A_{i,t}^{\mathrm{fail}} = \begin{cases} \dfrac{q_{i,j} - \mu_i^{\mathrm{seg}}}{\sigma_i^{\mathrm{seg}} + \epsilon_{\mathrm{num}}}, & g_i = 1,\ t \in S_{i,j}, \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

LOW segments receive positive corrections and HIGH segments negative corrections, shared across all action positions within each chunk.

**Successful trajectories.** Following GTPO (Tan et al., 2026), we reinforce high-entropy actions in successful rollouts to encourage exploration supported by task success. Let $\mu_{\mathrm{succ}}$ and $\sigma_{\mathrm{succ}}$ be the

entropy mean and standard deviation over all valid action positions in successful trajectories from the current rollout epoch. We define

$$A^{\text{succ}}_{i,t,c} = s_{\text{aux}} \left[ \frac{H^{\text{act}}_{i,t,c} - \mu_{\text{succ}}}{\sigma_{\text{succ}} + \epsilon_{\text{num}}} \right]_+, \tag{7}$$

where $[z]_+ = \max(z, 0)$ and $s_{\text{aux}} = \sigma_{\text{target}}/(\sigma_{\text{aux}} + \epsilon_{\text{num}})$. Here, $\sigma_{\text{aux}}$ is the standard deviation of the unscaled corrections over the same valid positions, including zeros. Only above-baseline positions receive positive credit, rescaled to the target standard deviation $\sigma_{\text{target}}$. This branch is skipped when no successful trajectories are available. Appendix C provides normalization details.

**Unified policy update.** We combine the original advantage and the outcome-dependent corrections:

$$A^{\text{ESCAPO}}_{i,t,c} = A^{\text{GRPO}}_{i} + \begin{cases} \beta_{\text{succ}} A^{\text{succ}}_{i,t,c}, & R_i = 1, \\ \beta_{\text{fail}} A^{\text{fail}}_{i,t}, & R_i = 0. \end{cases} \tag{8}$$

The coefficients independently control the two branches. LOW advantages shift upward and HIGH advantages downward; their final signs depend on both the original advantage and the correction. Combined advantages are not renormalized within trajectories. Using the policy ratio and clipped surrogate in Eqs. (2) and (3), we optimize

$$\mathcal{L}_{\text{ESCAPO}}(\theta) = -\mathbb{E}_{\text{old}} \left[ \frac{1}{G} \sum_{i=1}^{G} \frac{1}{M_i} \sum_{t,c,d} m_{i,t,c}\, s_\epsilon\big(\rho_{i,t,c,d}(\theta), A^{\text{ESCAPO}}_{i,t,c}\big) \right]. \tag{9}$$

Each position's advantage is shared by its $D$ action tokens. Entropy statistics and corrections are detached and fixed during policy updates. Terminal success remains the only environment reward; ESCAPO requires no human action-quality labels, additional critic, or learned reward model. Algorithm C.1 in Appendix C summarizes the procedure.

## 5 Experiments

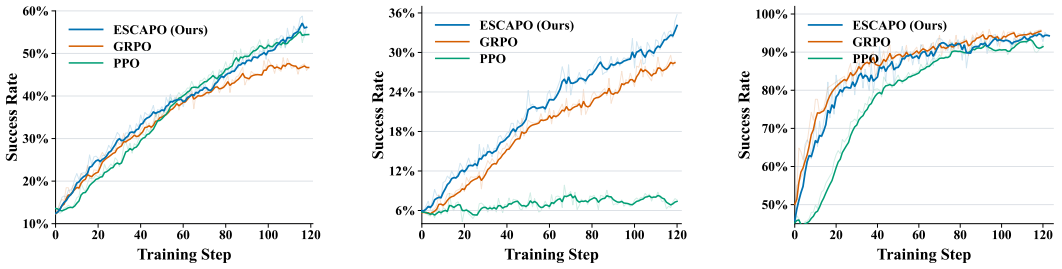
We evaluate five questions: (i) whether ESCAPO improves long-horizon VLA post-training from SFT, including under distribution shift; (ii) whether the gains persist when training continues from a shared GRPO warm-up checkpoint; (iii) whether it improves training efficiency; (iv) whether it yields gains in a real-robot evaluation; and (v) how individual design choices and hyperparameter settings affect its performance.

### 5.1 Experimental Setup

**Benchmarks and baselines.** ESCAPO reads credit from token-level policy entropy, so we adopt a categorical OpenVLA-OFT policy whose discrete action tokens have tractable predictive distributions (Kim et al., 2025a). We evaluate on three suites chosen to probe the regime targeted by the method (Liu et al., 2023; Fei et al., 2026): LIBERO-10, ten long-horizon tasks; LIBERO-Plus-10, the same tasks with visual, physical, and semantic perturbations; and LIBERO-Spatial, shorter tasks emphasizing spatial relations, which serves as a horizon-control setting. Within each suite, every SFT-initialized method starts from the same policy trained with one expert demonstration per task. We compare ESCAPO with GRPO (Shao et al., 2024) and, on LIBERO-10, PPO (Schulman et al., 2017), using binary terminal rewards. Appendix D.1 gives the optimization details.

**Metrics.** We adopt four complementary scores. Normalized AUC is the area under the success-rate curve divided by the comparison interval; it captures learning progress over training rather than only final performance. Late-15% SR averages success over the final 15% of that interval and measures late-stage performance. Peak SR, used after warm-up, is the highest recorded success rate. Target-crossing is the first iteration at which a trailing three-iteration moving average reaches a chosen success rate (Appendix D.3).

### 5.2 Main Results




(a) LIBERO-10 (b) LIBERO-Plus-10 (c) LIBERO-Spatial

Figure 6: **RL post-training from SFT initialization.** Success-rate curves on (a) LIBERO-10, (b) LIBERO-Plus-10, and (c) LIBERO-Spatial. Within each benchmark, all methods start from the same SFT policy trained with one expert demonstration per task.

We compare methods from the same SFT initialization within each benchmark (Table 1, Fig. 6). On LIBERO-10, ESCAPO improves both normalized AUC and late-stage success over GRPO, with the latter increasing from $46.90\%$ to $54.30\%$. It also slightly exceeds PPO on both metrics. PPO uses four update epochs per iteration and takes approximately twice as long as GRPO per iteration in our runs, whereas ESCAPO uses one epoch and requires no value-function training. On LIBERO-Plus-10, ESCAPO raises late-stage success from $27.66\%$ to $31.38\%$, retaining its benefit under environmental perturbations. PPO, using one update epoch on this benchmark, reaches only $7.70\%$. The perturbations may make value estimation harder to stabilize with this limited update budget. Errors in value estimates could then produce noisy advantages and disrupt policy learning, suggesting that PPO may need additional optimization in this setting. On LIBERO-Spatial, ESCAPO remains close to GRPO, with late-stage success of $93.71\%$ versus $94.73\%$, while PPO reaches $91.82\%$ with one optimization epoch.

Table 1: **Performance from SFT initialization.** Bold indicates the best available result within each benchmark.

| | LIBERO-10 | | LIBERO-Plus-10 | | LIBERO-Spatial | |
|---|---|---|---|---|---|---|
| Method | AUC↑ | Late SR↑ | AUC↑ | Late SR↑ | AUC↑ | Late SR↑ |
| GRPO | 35.84% | 46.90% | 18.65% | 27.66% | **87.67%** | **94.73%** |
| PPO | 37.27% | 53.42% | 7.01% | 7.70% | 79.46% | 91.82% |
| **ESCAPO (Ours)** | **38.42%** | **54.30%** | **21.66%** | **31.38%** | 85.70% | 93.71% |

The performance advantage becomes more apparent as training proceeds (Fig. 6). This pattern is consistent with the entropy dynamics discussed in Sec. 3: entropy-based action-quality discrimination is already present after SFT and strengthens during GRPO training. As this distinction becomes clearer, entropy can provide a more informative signal for refining trajectory-level credit.

On LIBERO-Spatial, ESCAPO performs similarly to GRPO but achieves slightly lower late-stage success. One possible explanation is that shorter failed trajectories contain fewer successful action chunks that can benefit from credit refinement. This limits the useful behavior that the failure correction can preserve. Meanwhile, the additional negative advantages assigned to HIGH segments may strengthen penalties without a comparable benefit from protecting successful actions. These effects may offset the gains from finer-grained credit assignment on short trajectories. GRPO also already reaches nearly $95\%$ late-stage success, leaving limited performance headroom. The long-horizon suites therefore provide the more direct test of our proposed credit-assignment mechanism.

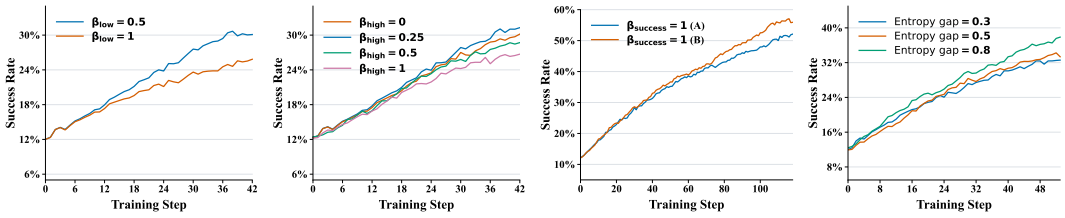
### 5.3 Continued Training after RL Warm-Up

Section 3 shows that policy entropy becomes more informative about local action quality during on-policy RL. This motivates a complementary question: can a GRPO-trained policy benefit further from entropy-guided credit assignment once this distinction has become more pronounced? We therefore continue GRPO and ESCAPO from the same GRPO checkpoint under matched continuation budgets.

Table 2: **Continued training from a shared RL warm-up checkpoint.** Both methods start from the same warm-up initialization.

| Benchmark | Method | Norm. AUC↑ | Late-15% SR↑ | Peak SR↑ |
|---|---|---|---|---|
| LIBERO-10 | GRPO | 90.09% | 90.43% | 92.29% |
| | ESCAPO | **91.45%** | **92.70%** | **95.21%** |
| LIBERO-Plus-10 | GRPO | 52.59% | 58.90% | 60.74% |
| | ESCAPO | **53.17%** | **62.18%** | **64.65%** |

ESCAPO improves over continued GRPO on both long-horizon suites as shown in Table 2. On LIBERO-10, ESCAPO improves learning progress and late-stage success even with GRPO already above $90\%$, reaching a peak of $95.21\%$. Gains also appear on LIBERO-Plus-10, where late-stage and peak success rise by three to four percentage points. The entropy analysis in Sec. 3 offers a possible




(a) $\beta_{\text{low}}$ (b) $\beta_{\text{high}}$ (c) $\beta_{\text{succ}}$ (d) $\delta_H$

Figure 7: **Ablations of entropy-guided credit assignment.** (a) LOW correction weight. (b) HIGH penalty weight. (c) Failure branch disabled (0) or enabled (1), with $\beta_{\text{succ}} = 1$. (d) Entropy-gap threshold for failure correction.

explanation: GRPO develops a clearer entropy–quality association, including under environmental perturbations, which ESCAPO can use to assign credit within failed rollouts. Appendix D.2 reports the learning curves and target-crossing results.

### 5.4 Training Efficiency

We measure how many optimization iterations each method needs to reach selected success-rate targets, using the same SFT-initialized runs as in Sec. 5.2. On the two long-horizon suites, ESCAPO reaches both selected targets in fewer iterations than GRPO (Table 3). The reduction is modest on LIBERO-10 at the lower target, where PPO reaches the target first, but is larger on LIBERO-Plus-10, where ESCAPO requires roughly one-fifth to three-tenths fewer iterations than GRPO. The same speedup does not appear on LIBERO-Spatial, matching the weaker short-horizon result in Sec. 5.2. These comparisons count optimization iterations, not wall-clock time: on LIBERO-10, PPO uses four update epochs per iteration, versus one for GRPO, and each iteration takes about twice as long.

Table 3: **Iterations to target success rate from SFT initialization.** Iterations to first reach target success rates using a trailing three-iteration moving average. Bold indicates the fewest iterations. Reductions are relative to GRPO.

| Benchmark | Target SR | GRPO↓ | PPO↓ | ESCAPO↓ | Reduction |
|---|---|---|---|---|---|
| LIBERO-10 | 40% | 68 | **60** | 66 | 2.9% |
| | 45% | 98† | **74** | 78 | 20.4% |
| LIBERO-Plus-10 | 20% | 61 | – | **49** | 19.7% |
| | 25% | 94 | – | **67** | 28.7% |
| LIBERO-Spatial | 85% | **33** | 61 | 41 | -24.2% |
| | 90% | **61** | 76 | 62 | -1.6% |

### 5.5 Ablation Studies

We examine the LOW and HIGH correction weights, the contribution of the failure branch, and the entropy-gap threshold (Table 4, Fig. 7). For LOW segments, increasing $\beta_{\text{low}}$ from 0.5 to 1 reduces normalized AUC from 0.22 to 0.20 and late-stage success from 30.10% to 25.20%. A larger positive correction is therefore not always beneficial. Although it protects useful actions in failed rollouts, it can outweigh the negative GRPO advantage and turn the total advantage positive, actively reinforcing those actions. An excessive correction may give LOW segments too much influence relative to actions in successful trajectories. The smaller weight limits this effect while retaining the protection of useful behavior. For HIGH segments, $\beta_{\text{high}} = 0.25$ achieves the best late-stage success of 30.68%, compared with 29.38% when the additional penalty is disabled. Without this penalty, HIGH segments retain their original GRPO advantages. A moderate penalty strengthens their credit distinction from LOW segments, but increasing the weight to 0.5 and 1 lowers success to 28.22% and 26.20%. Stronger negative corrections may suppress sampled actions too aggressively, making 0.25 a better trade-off in this comparison. For the branch ablation, we fix $\beta_{\text{succ}} = 1$; settings 0 and 1 denote disabling and enabling the failure branch, respectively. It raises normalized AUC from 0.36 to 0.38 and late-stage success from 50.26% to 55.36%. The branches serve complementary purposes: failure-side credit favors LOW segments, which often represent familiar, task-progressing behavior, while the success branch reinforces high-entropy choices in rollouts validated by task success. The latter can

Table 4: **Ablations of entropy-guided credit assignment.** Bold indicates the best result within each group.

| Ablation | Setting | Normalized AUC↑ | Late-15% SR↑ |
|---|---|---|---|
| $\beta_{\text{low}}$ | 0.5 | **0.22** | **30.10%** |
| | 1.0 | 0.20 | 25.20% |
| $\beta_{\text{high}}$ | 0 | 0.22 | 29.38% |
| | 0.25 | **0.23** | **30.68%** |
| | 0.5 | 0.21 | 28.22% |
| | 1.0 | 0.21 | 26.20% |
| $\beta_{\text{succ}} = 1$ | 0 | 0.36 | 50.26% |
| | 1 | **0.38** | **55.36%** |
| Threshold $\delta_H$ | 0.3 | 0.25 | 32.24% |
| | 0.5 | 0.25 | 33.25% |
| | 0.8 | **0.27** | **36.67%** |

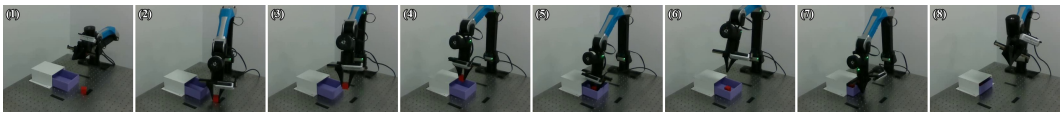


Figure 8: **Real-robot task execution.** The ARX5 places the red cube inside the open drawer and then closes the drawer.

help preserve exploration that might otherwise be limited by favoring familiar behaviors. With the success branch active, the observed gain shows the additional benefit of refining credit within failed trajectories. Finally, the strictest tested gate threshold, $\delta_H = 0.8$, achieves normalized AUC of $0.27$ and late-stage success of $36.67\%$, compared with success rates of $32.24\%$ and $33.25\%$ for thresholds $0.3$ and $0.5$. A small threshold admits trajectories with weak LOW–HIGH separation, where credit differences may be driven by minor entropy fluctuations. A higher threshold preserves the original GRPO advantage until the separation is clearer. This more conservative activation rule performs better over the evaluated interval.

### 5.6 Real-robot transfer.

We evaluate sim-to-real transfer on an ARX5 manipulator using a two-stage task: placing a block inside an open drawer and then closing the drawer (Fig. 8). After 70 training steps in simulation, ESCAPO succeeds in 4 of 10 real-robot trials (**40%**), compared with 2 of 10 trials (**20%**) for GRPO. This preliminary evaluation shows a higher observed success rate for ESCAPO in physical execution. The low absolute success rates may partly reflect residual camera-alignment errors and differences between simulated and real control. The setup is detailed in Appendix D.4.

## 6 Limitations

ESCAPO currently uses fixed correction weights and an entropy-gap threshold selected empirically. Their best values may vary with the policy and task, motivating future work on adaptive selection. Entropy is also an imperfect cue for local action quality: supplemental analyses suggest that part of its association with action quality may reflect progress through a task. Finally, our real-robot results are preliminary, covering one task with ten trials per method. We will present evaluations on additional real-robot tasks to better assess transfer.

## 7 Related Works

VLA post-training spans offline-to-online RL and online optimization with sparse outcomes or richer feedback (Chen et al., 2025b; Huang et al., 2025; Tan et al., 2025; Li et al., 2026; Lu et al., 2025; Chen et al., 2025c). Entropy has informed exploration (Haarnoja et al., 2018), fine-grained credit assignment in LLMs (Cui et al., 2025; Tan et al., 2026; Zhao et al., 2026), and VLA uncertainty analysis (Yang et al., 2026; Liang et al., 2026). ESCAPO uses outcome-conditioned entropy and temporal segments to refine trajectory-level GRPO advantages from terminal rewards. Appendix E discusses these connections in more detail.

## 8 Conclusion

ESCAPO uses policy entropy to refine trajectory-level credit with outcome-conditioned corrections. It assigns segment-level corrections to failed rollouts and reinforces above-baseline entropy actions in successful rollouts, without additional critics, dense rewards, or human training annotations. Our training-dynamics analysis motivates this refinement, and SFT-initialized experiments show improved long-horizon performance over GRPO. The gains do not extend to LIBERO-Spatial, and iteration-based comparisons do not establish wall-clock efficiency. Broader benchmarks, multi-seed evaluation, and further ablations remain important future work.

### AI USE STATEMENT

We used generative AI tools to assist with writing and polishing the language of the manuscript, and to identify and retrieve related work. The authors reviewed and edited all AI-assisted text, verified AI-suggested references, and take full responsibility for the final content.

### ETHICS STATEMENT

This work studies reinforcement-learning post-training of VLA policies for robotic manipulation. Human action-quality annotations are used only for analysis and evaluation, not for policy optimization. Performance on simulation benchmarks or controlled real-robot experiments does not establish reliability in unrestricted deployment. However, physical robot experiments and deployment require task-specific safety measures, appropriate human oversight, and mechanisms to interrupt unsafe behavior, particularly when failures could harm people or damage property.

### REPRODUCIBILITY STATEMENT

Section 5.1 describes the policy initialization, benchmarks, and evaluation setting. Appendix D provides additional experimental protocols and results. Section 4 presents the method. Appendix C contains Algorithm C.1 with the complete procedure and gives further segmentation and advantage-construction details. The entropy analysis uses human annotations only for evaluation, not during RL updates. We have also attached an zip file of the code in Supplementary Material to support reproducibility.

## REFERENCES


Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. $\pi_0$: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.010.

Kang Chen, Zhihao Liu, Tonghe Zhang, Zhen Guo, Si Xu, Hao Lin, Hongzhi Zang, Quanlu Zhang, Zhaofei Yu, Guoliang Fan, et al. $\pi$rl: Online rl fine-tuning for flow-based vision-language-action models. *arXiv preprint arXiv:2510.25889*, 8, 2025a.

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

Zengjue Chen, Runliang Niu, He Kong, Qi Wang, Qianli Xing, and Zipei Fan. Tgrpo :fine-tuning vision-language-action model via trajectory-wise group relative policy optimization, 2025c. URL https://arxiv.org/abs/2506.08440.

Ganqu Cui, Yuchen Zhang, Jiacheng Chen, Lifan Yuan, Zhi Wang, Yuxin Zuo, Haozhan Li, Yuchen Fan, Huayu Chen, Weize Chen, Zhiyuan Liu, Hao Peng, Lei Bai, Wanli Ouyang, Yu Cheng, Bowen Zhou, and Ning Ding. The entropy mechanism of reinforcement learning for reasoning language models. *arXiv preprint arXiv:2505.22617*, 2025.

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

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 1861–1870. PMLR, 2018.

Dongchi Huang, Zhirui Fang, Tianle Zhang, Yihang Li, Lin Zhao, and Chunhe Xia. CO-RFT: Efficient fine-tuning of vision-language-action models through chunked offline reinforcement learning. *arXiv preprint arXiv:2508.02219*, 2025.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. In *Proceedings of Robotics: Science and Systems*, 2025a. doi: 10.15607/RSS.2025.XXI.017.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P. Foster, Pannag R. Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025b.

Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, volume 30, 2017. URL https://papers.nips.cc/paper/2017/hash/9ef2ed4b7fd2c810847ffa5fa85bce38-Abstract.html.

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

Yuanchang Liang, Xiaobo Wang, Kai Wang, Shuo Wang, Xiaojiang Peng, Haoyu Chen, David Kim Huat Chua, and Prahlad Vadakkepat. Adaptive action chunking at inference-time for vision-language-action models. *arXiv preprint arXiv:2604.04161*, 2026.

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

Guanxing Lu, Wenkai Guo, Chubin Zhang, Yuheng Zhou, Haonan Jiang, Zifeng Gao, Yansong Tang, and Ziwei Wang. VLA-RL: Towards masterful and general robotic manipulation with scalable reinforcement learning. *arXiv preprint arXiv:2505.18719*, 2025.

Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, D. Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, volume 32, 2019. URL https://proceedings.neurips.cc/paper/2019/hash/8558cb408c1d76621371888657d2eb1d-Abstract.html.

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

Stephane Ross and Drew Bagnell. Efficient reductions for imitation learning. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pp. 661–668. PMLR, 2010.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635. PMLR, 2011.

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, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Hongze Tan, Zihan Wang, Jianfei Pan, Jinghao Lin, Hao Wang, Yifan Wu, Tao Chen, Zhihang Zheng, Zhihao Tang, and Haihua Yang. Gtpo and grpo-s: Token and sequence-level reward shaping with policy entropy, 2026. URL https://arxiv.org/abs/2508.04349.

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

Jiabing Yang, Yixiang Chen, Yuan Xu, Peiyan Li, Xiangnan Wu, Zichen Wen, Bowen Fang, Tao Yu, Zhengbo Zhang, Yingda Li, Kai Wang, Jing Liu, Nianfeng Liu, Yan Huang, and Liang Wang. UAOR: Uncertainty-aware observation reinjection for vision-language-action models. *arXiv preprint arXiv:2602.18020*, 2026.

Hongzhi Zang, Mingjie Wei, Si Xu, Yongji Wu, Zhen Guo, Yuanqing Wang, Hao Lin, Liangzhi Shi, Yuqing Xie, Zhexuan Xu, Zhihao Liu, Kang Chen, Wenhao Tang, Quanlu Zhang, Weinan Zhang, Chao Yu, and Yu Wang. RLinf-VLA: A unified and efficient framework for VLA+RL training. *arXiv preprint arXiv:2510.06710*, 2025.

Haotian Zhao, Songlin Zhou, Yuxin Zhang, Stephen S.-T. Yau, Wenyu Zhang, Lun Tian, Tianshu Zhu, Yifeng Huang, Yucheng Zeng, Jingnan Gu, Daxiang Dong, and Jianmin Wu. AEM: Adaptive entropy modulation for multi-turn agentic reinforcement learning. *arXiv preprint arXiv:2605.00425*, 2026.

Zhide Zhong, Haodong Yan, Junfeng Li, Junjie He, Tianran Zhang, and Haoang Li. VLA-OPD: Bridging offline SFT and online RL for vision-language-action models via on-policy distillation. *arXiv preprint arXiv:2603.26666*, 2026.

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

## APPENDIX CONTENTS


**A Additional Preliminaries** **14**

A.1 Categorical VLA Policy . . . . . 14

A.2 Supervised Fine-Tuning . . . . . 14

A.3 On-policy RL Post-training . . . . . 14

**B Training Entropy Dynamics** **15**

B.1 Gate Activation during Training . . . . . 15

B.2 Failed Initialization Cases . . . . . 16

**C ESCAPO Implementation Details** **18**

C.1 Entropy Granularity . . . . . 18

C.2 Adaptive Segmentation and Reliability Gating . . . . . 18

C.3 Relative Segment Advantages . . . . . 19

C.4 Success-Branch Normalization . . . . . 19

C.5 Policy Optimization . . . . . 19

**D Additional Experimental Results and Protocols** **20**

D.1 Additional Experimental Setup . . . . . 20

D.2 Additional Warm-Up Results . . . . . 20

D.3 Target-Crossing Protocol and Additional Discussion . . . . . 20

D.4 Simulation and Real-Robot Setup . . . . . 22

D.5 Implementation and Computational Resources . . . . . 22

**E Extended Related Works** **23**

# A Additional Preliminaries

## A.1 Categorical VLA Policy

For the parallel-decoding policy considered here, each action token $a_{t,c,d}$ takes a value from a discrete vocabulary $\mathcal{V}$. The action-chunk distribution factorizes as

$$\pi_\theta(\mathbf{a}_t \mid o_t, x) = \prod_{c=1}^{C} \prod_{d=1}^{D} \pi_\theta(a_{t,c,d} \mid o_t, x, c, d). \tag{A.1}$$

All tokens are predicted from the same visual and proprioceptive observation $o_t$ and instruction $x$; sampled tokens are not autoregressively fed back within a chunk. The index $t$ denotes policy decisions, $c$ action positions within a chunk, and $d$ action dimensions.

## A.2 Supervised Fine-Tuning

Before online RL post-training, the policy is initialized by supervised fine-tuning on expert demonstrations $\mathcal{D}_E$. For an expert trajectory $\tau_E$, let $m^E_{t,c} \in \{0,1\}$ indicate whether an action position is valid. The number of valid action tokens is $M_E = \sum_{t,c,d} m^E_{t,c}$. The token-level SFT objective is

$$\mathcal{L}_{\mathrm{SFT}}(\theta) = -\mathbb{E}_{\tau_E \sim \mathcal{D}_E} \left[ \frac{1}{M_E} \sum_{t,c,d} m^E_{t,c} \log \pi_\theta(a^E_{t,c,d} \mid o^E_t, x_E, c, d) \right], \tag{A.2}$$

where $x_E$ is the instruction associated with $\tau_E$. SFT learns from expert actions and observations. During deployment, the policy instead encounters observations induced by its own actions, which may extend beyond the demonstration distribution.

## A.3 On-policy RL Post-training

On-policy RL post-training collects trajectories through environment interaction and directly optimizes task outcomes. A rollout with $T_i$ policy decisions is denoted by $\tau_i = (x, o_{i,0}, \mathbf{a}_{i,0}, \ldots, \mathbf{a}_{i,T_i-1}, o_{i,T_i})$. The environment provides a terminal binary reward

$$R_i = R(\tau_i) \in \{0,1\}, \tag{A.3}$$

where $R_i = 1$ indicates task success. The objective is

$$\max_\theta \ \mathbb{E}_{x \sim p_{\mathrm{task}},\, \tau \sim \pi_\theta(\cdot|x)}[R(\tau)], \tag{A.4}$$

where $p_{\mathrm{task}}$ is the task-instruction distribution. Trajectory sampling also includes environment initialization and transitions.

**PPO** (Schulman et al., 2017) learns a value function $V_\phi$ and uses generalized advantage estimation to compute $A^{\mathrm{PPO}}_t$. Its policy objective is

$$L^{\mathrm{PPO}}(\theta) = \mathbb{E}_t \left[ \min \left\{ r_t(\theta) A^{\mathrm{PPO}}_t, \mathrm{clip}(r_t(\theta), 1-\epsilon_{\mathrm{clip}}, 1+\epsilon_{\mathrm{clip}})\, A^{\mathrm{PPO}}_t \right\} \right], \tag{A.5}$$

where $r_t(\theta) = \pi_\theta(\mathbf{a}_t \mid o_t, x) / \pi_{\theta_{\mathrm{old}}}(\mathbf{a}_t \mid o_t, x)$ is the joint-action policy ratio, and $\mathbb{E}_t$ denotes an empirical average over sampled decision steps. We use the same clipping-threshold notation $\epsilon_{\mathrm{clip}}$ as in Eq. (3). PPO obtains time-dependent advantages through its critic, but requires additional value learning.

**GRPO** (Shao et al., 2024) constructs advantages from a group of rollout rewards without learning a value function. For each instruction $x$, the behavior policy samples $\mathcal{G}_x = \{\tau_i\}_{i=1}^G$, where $G > 1$. The group reward statistics used in Eq. (1) are

$$\bar{R}_x = \frac{1}{G} \sum_{i=1}^{G} R_i, \qquad \sigma_x = \sqrt{\frac{1}{G-1} \sum_{i=1}^{G} (R_i - \bar{R}_x)^2}. \tag{A.6}$$

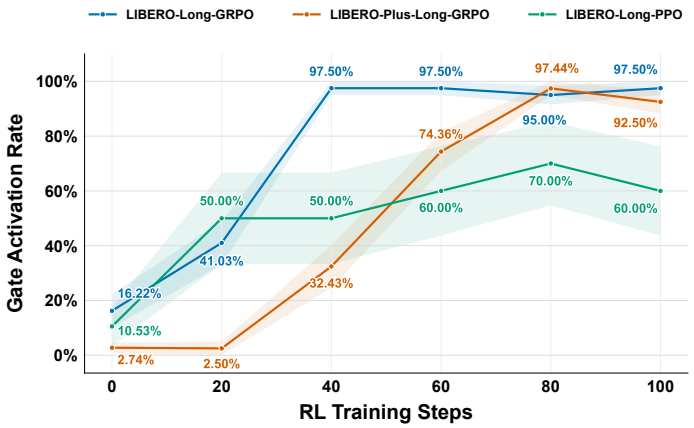



Figure B.1: **Entropy-gate activation during training.** Fraction of recorded failed rollouts that satisfy the failure-branch gating criteria at selected training checkpoints.

The scalar advantage $A_i^{\mathrm{GRPO}}$ is assigned to every valid action token in trajectory $i$. Using the token-level ratio in Eq. (2) and clipped surrogate in Eq. (3), the GRPO loss considered here is

$$\mathcal{L}_{\mathrm{GRPO}}(\theta) = -\mathbb{E}_{\substack{x\sim p_{\mathrm{task}} \\ \mathcal{G}_x\sim\pi_{\theta_{\mathrm{old}}}(\cdot|x)}} \left[ \frac{1}{G}\sum_{i=1}^{G} \frac{1}{M_i} \sum_{t,c,d} m_{i,t,c}\, s_\epsilon\big(\rho_{i,t,c,d}(\theta), A_i^{\mathrm{GRPO}}\big) \right]. \tag{A.7}$$

Here, $m_{i,t,c}$ masks invalid action positions and $M_i = \sum_{t,c,d} m_{i,t,c}$ counts valid action tokens. This formulation includes neither a value loss nor an additional KL penalty. In mixed-outcome groups, every valid token in a failed trajectory receives the same negative advantage regardless of local action quality. When all rollout rewards agree, all GRPO advantages are zero. This uniform temporal assignment motivates ESCAPO.

# B Training Entropy Dynamics

## B.1 Gate Activation during Training

We examine when the trajectory gate would activate the failure correction during training. We apply the gate to the annotated failed rollouts collected during GRPO training on LIBERO-Long and LIBERO-Plus-Long.

At checkpoint $k$, let $\mathcal{F}_k$ contain all analyzed failed rollout records. The activation rate is $r_k = |\mathcal{F}_k|^{-1} \sum_{i\in\mathcal{F}_k} g_i$. Its denominator includes all failed records, including those with only one action-quality label. Human labels are not used to determine $g_i$.

We use the trajectory-relative threshold $T_{H,i} = \mu_i - 0.8\sigma_i$ and bridge internal HIGH intervals of at most three chunks. The gate requires an entropy-region gap of at least 0.5 and a longest run of at least four chunks for each mode, together with the validity checks described in the method. This region gap compares HIGH and LOW entropy regions. It differs from the label-based gap, which compares failed and successful actions.

As shown in Figure B.1, the activation rate on LIBERO-Long increases from 16.2% at SFT initialization to 79.5% at step 30. It remains between 95.0% and 100.0% from steps 40 to 110. On LIBERO-Plus-Long, activation is at most 2.7% through step 20 and reaches 90.0%–97.4% between steps 70 and 110.

These results show that the gate would restrict additional failure credit to a small fraction of early rollouts and enable it more broadly as sustained entropy separation develops.

Table B.1: **Initial policy entropy on 400 shared clean observations.** Values are reported in nats.

| Initialization | Temperature 1.6 | Temperature 1.0 |
|---|---|---|
| Failed High-Entropy | 3.03 | 1.75 |
| Failed Low-Entropy | 0.63 | 0.33 |
| Main SFT | 0.45 | 0.25 |

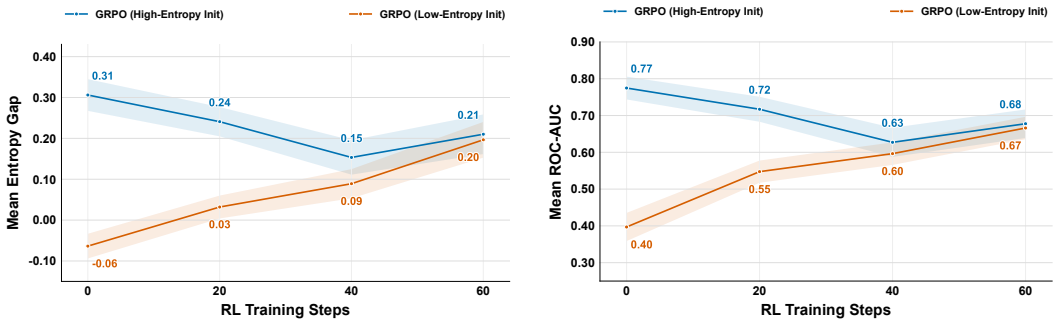
### B.2 Failed Initialization Cases

We examine two additional SFT initializations: *Failed High-Entropy*, obtained after limited SFT, and *Failed Low-Entropy*, obtained after prolonged SFT. Both exhibit weaker entropy-based action-quality discrimination during RL than the main initialization, which we denote *Main SFT*.

**Initial policy entropy.** We first compare the three models on the same 400 clean, unperturbed observations. Table B.1 reports their mean policy entropy at temperatures 1.6 and 1.0. At temperature 1.6, Failed High-Entropy has a mean entropy of 3.03, compared with 0.63 for Failed Low-Entropy and 0.45 for Main SFT. The ordering remains the same at temperature 1.0. These measurements provide a common reference for comparing initial prediction confidence.

**Data and annotations.** For each failed initialization, we analyze 40 failed rollout records at each of steps 0, 20, 40, and 60. Each record contains 64 action chunks, giving 160 records and 10,240 chunks per initialization. A successful action advances the task or maintains valid execution; a failed action does neither. Local labels are obtained through AI visual annotation. They have not been individually verified by human annotators and are analyzed separately from the human-annotated results in the main text.

Entropy and labels are aligned by rollout identifier and chunk index. We compute the entropy gap and ROC-AUC within each rollout containing both action classes, then average with equal rollout weights. Negative entropy scores successful actions. All eligible records are included, regardless of gate activation. One record at step 0 in Failed High-Entropy contains only successful actions and is excluded. All other initialization–checkpoint combinations contain 40 eligible records.




(a) Mean entropy gap (b) Mean ROC-AUC

Figure B.2: **Entropy dynamics under alternative SFT initializations.** Mean within-trajectory entropy gap and ROC-AUC for the two supplemental initialization runs. Metrics use AI-annotated action chunks from failed rollouts.

**Failed High-Entropy.** This initialization has diffuse predictions after limited SFT. Its initial within-trajectory entropy gap is 0.31, with ROC-AUC 0.77. Thus, entropy already provides some action-quality information despite the high initial entropy.

During RL, entropy decreases for both action classes, with a larger decrease for failed actions. As shown in Fig. B.2, the gap falls to 0.21 and ROC-AUC to 0.68 at step 60. Both metrics partially recover after step 40, but remain below their initial values. In this run, increasing prediction confidence during RL is accompanied by weaker alignment between entropy and action quality. This

Table B.2: **Policy entropy under input perturbations.** We compare the two failed initializations with Main SFT. OOD AUROC is reported for Main SFT, using higher entropy to identify perturbed inputs.

| | Mean policy entropy | | | |
|---|---|---|---|---|
| Input condition | Failed Low-Entropy | Failed High-Entropy | Main SFT | OOD AUROC (Main SFT) |
| Clean input | 0.63 | 3.03 | 0.45 | – |
| Gaussian noise $\sigma = 32$ | 0.63 | 3.20 | 0.58 | 0.70 |
| Gaussian noise $\sigma = 64$ | 0.53 | 3.57 | 0.69 | 0.83 |
| Image occlusion | 0.58 | 3.26 | 0.52 | 0.61 |
| Different task instruction | 0.63 | 3.04 | 0.51 | 0.61 |
| Random image | 0.41 | 3.87 | 0.72 | 0.87 |

suggests that the confidence developed during limited SFT provides an insufficient foundation for the stronger discrimination observed in the main run.

**Failed Low-Entropy.** After prolonged SFT, this initialization exhibits concentrated predictions, including at failed-action positions. At step 0, failed actions have lower mean entropy than successful actions. The resulting entropy gap is $-0.06$, and ROC-AUC is 0.40.

RL gradually corrects this reversed association. Entropy increases for both action classes, with a larger increase for failed actions. The gap reaches 0.20 and ROC-AUC reaches 0.67 at step 60. Both metrics improve across the four sampled checkpoints, although the resulting discrimination remains weaker than in the main run.

**Input sensitivity and the main initialization.** The perturbation results in Table B.2 help characterize these different initializations. We discuss the probe results at temperature 1.6. Failed Low-Entropy responds weakly to moderate image noise. Under stronger noise with $\sigma = 64$, its mean entropy decreases from 0.63 to 0.53. Random images reduce it further to 0.41. The policy therefore retains concentrated predictions even when visual inputs are severely corrupted.

Main SFT combines lower clean-input entropy with a clearer response to these perturbations. Its entropy increases from 0.45 to 0.69 under noise with $\sigma = 64$ and to 0.72 for random images. The corresponding OOD AUROC values are 0.83 and 0.87. Here, OOD AUROC measures whether higher entropy distinguishes perturbed inputs from clean inputs. Failed High-Entropy also responds to these perturbations, despite its weaker action-quality discrimination during subsequent RL.

As discussed in Sec. 3, Main SFT already associates lower entropy with successful actions. This association becomes stronger during GRPO post-training.

Together, these cases suggest that initialization quality depends on both learned confidence and sensitivity to unfamiliar inputs. Failed High-Entropy starts with diffuse predictions, and its action-quality discrimination weakens during RL. Failed Low-Entropy retains confidence under severe input perturbations and develops only partial discrimination during continued training. Main SFT combines confident predictions on clean observations with increasing entropy under perturbations, alongside stronger entropy–quality alignment during RL.

**Interpretation.** These cases show that absolute entropy alone does not determine its usefulness as an action-quality signal. The high-entropy run loses some of its initial discrimination, whereas the low-entropy run gradually develops a positive association from an initially reversed one. They complement the main results by showing that the direction and strength of entropy dynamics vary across initializations and training runs.

Execution stage may also contribute to the observed associations. In these supplemental datasets, using chunk position alone to score failed actions yields mean ROC-AUC values of approximately 0.92–0.99. For the low-entropy run, entropy-based discrimination does not consistently improve when comparisons are restricted to the same 16-chunk window. The full-trajectory results therefore describe an association that can also reflect task progression.

**Algorithm C.1** ESCAPO

**Require:** Initialized policy $\pi_\theta$, task distribution $p_{\text{task}}$, group size $G$; $\alpha$, $\delta_H$, $\ell_{\min}$, $\sigma_{\text{target}}$, $\beta_{\text{succ}}$, $\beta_{\text{fail}}$, $\epsilon_{\text{clip}}$, $\epsilon_{\text{num}}$

**while** interaction budget is not exhausted **do**
  Set $\theta_{\text{old}} \leftarrow \theta$ and sample tasks $x \sim p_{\text{task}}$
  Collect $G$ on-policy rollouts per task using $\pi_{\theta_{\text{old}}}$ and obtain terminal outcomes $R_i$
  Compute $A_i^{\text{GRPO}}$ using Eq. (1)
  Compute behavior-policy action-position entropy $H^{\text{act}}_{i,t,c}$ and chunk entropy $H_{i,t}$
  **if** successful trajectories are available **then**
    Compute epoch-wide successful-position statistics and $A^{\text{succ}}_{i,t,c}$ using Eq. (7)
  **end if**
  **for all** rollouts $\tau_i$ **do**
    **if** $R_i = 0$ **then**
      Compute $T_{H,i}$; classify LOW/HIGH chunks and bridge HIGH intervals of at most three chunks
      Apply validity and duration checks and the separation gate $g_i$; form participating segments $\mathcal{S}_i$
      Compute $A^{\text{fail}}_{i,t}$ using Eq. (6)
    **end if**
    Form $A^{\text{ESCAPO}}_{i,t,c}$ using Eq. (8)
  **end for**
  Detach and fix all augmented advantages; update $\pi_\theta$ using Eq. (9)
**end while**

# C ESCAPO Implementation Details

## C.1 Entropy Granularity

All entropy values are computed under the behavior policy $\pi_{\theta_{\text{old}}}$ using the token entropy in Eq. (4). For clarity, the action-position and chunk averages from Sec. 3 are

$$H^{\text{act}}_{i,t,c} = \frac{1}{D}\sum_{d=1}^{D} H_{i,t,c,d}, \tag{C.1}$$

$$H_{i,t} = \frac{1}{C}\sum_{c=1}^{C} H^{\text{act}}_{i,t,c}. \tag{C.2}$$

The failure branch segments the chunk-level sequence $H_{i,t}$. Its corrections are shared across all action positions within a chunk. The success branch instead uses $H^{\text{act}}_{i,t,c}$ for position-specific credit. Both branches share each position's advantage across its $D$ action tokens.

## C.2 Adaptive Segmentation and Reliability Gating

Let $\mathcal{T}_i$ contain the valid executed chunk positions of a failed trajectory and let $N_i = |\mathcal{T}_i|$. Its trajectory-level entropy statistics are

$$\mu_i = \frac{1}{N_i}\sum_{t\in\mathcal{T}_i} H_{i,t}, \qquad \sigma_i = \sqrt{\frac{1}{N_i}\sum_{t\in\mathcal{T}_i}(H_{i,t}-\mu_i)^2}. \tag{C.3}$$

Using the relative threshold $T_{H,i} = \mu_i - \alpha\sigma_i$ defined in Sec. 4.2, we initially classify valid chunks as

$$z_{i,t} = \begin{cases} \text{LOW}, & H_{i,t} < T_{H,i}, \\ \text{HIGH}, & H_{i,t} \geq T_{H,i}. \end{cases} \tag{C.4}$$

The relative threshold accommodates entropy-scale differences across trajectories and training stages; larger $\alpha > 0$ selects LOW regions more conservatively. To reduce fragmentation caused by brief fluctuations, HIGH intervals of at most three chunks are merged into the surrounding LOW segments.

After bridging, let $\mathcal{T}_i^{\text{LOW}}$ and $\mathcal{T}_i^{\text{HIGH}}$ denote the LOW and HIGH positions. Their mean entropy separation is

$$\Delta H_i = \frac{\sum_{t\in\mathcal{T}_i^{\text{HIGH}}} H_{i,t}}{|\mathcal{T}_i^{\text{HIGH}}|} - \frac{\sum_{t\in\mathcal{T}_i^{\text{LOW}}} H_{i,t}}{|\mathcal{T}_i^{\text{LOW}}|}. \tag{C.5}$$

For failed trajectories that pass basic validity and segment-duration checks, the trajectory-level gate is

$$g_i = \begin{cases} 1, & \Delta H_i \geq \delta_H, \\ 0, & \text{otherwise.} \end{cases} \tag{C.6}$$

Trajectories failing these checks receive $g_i = 0$. The threshold $\delta_H$ sets the minimum required separation. This gate avoids relying on weak entropy separation early in RL post-training. When $g_i = 0$, the trajectory retains its original GRPO advantage.

### C.3 Relative Segment Advantages

For an enabled failed trajectory, let $\ell_{\min}$ denote the minimum retained LOW-segment duration in chunks. LOW runs shorter than $\ell_{\min}$ are excluded and receive zero additional credit; they are not relabeled as HIGH. The participating segment set is

$$\mathcal{S}_i = \{S_{i,1}, \ldots, S_{i,K_i}\}, \tag{C.7}$$

containing the remaining LOW segments and all HIGH segments. This duration-based selection does not use a first-segment or trajectory-position prior. We score participating segments as

$$q_{i,j} = \begin{cases} 1, & S_{i,j} \text{ is LOW}, \\ 0, & S_{i,j} \text{ is HIGH}. \end{cases} \tag{C.8}$$

Their within-trajectory statistics are

$$\mu_i^{\text{seg}} = \frac{1}{K_i} \sum_{j=1}^{K_i} q_{i,j}, \qquad \sigma_i^{\text{seg}} = \sqrt{\frac{1}{K_i} \sum_{j=1}^{K_i} (q_{i,j} - \mu_i^{\text{seg}})^2}. \tag{C.9}$$

These statistics weight segments equally, irrespective of duration. The normalized segment advantage is

$$a_{i,j}^{\text{seg}} = \frac{q_{i,j} - \mu_i^{\text{seg}}}{\sigma_i^{\text{seg}} + \epsilon_{\text{num}}}. \tag{C.10}$$

Equation (6) broadcasts $a_{i,j}^{\text{seg}}$ to chunks $t \in S_{i,j}$ when $g_i = 1$, and assigns zero elsewhere. Participating LOW segments receive positive corrections, HIGH segments receive negative corrections, and excluded positions receive zero additional advantage. The correction is shared across all action positions in a chunk.

### C.4 Success-Branch Normalization

Let $\mu_{\text{succ}}$ and $\sigma_{\text{succ}}$ denote the mean and standard deviation of $H_{i,t,c}^{\text{act}}$ over all valid action positions in successful trajectories from the current rollout epoch. The raw additional advantage is

$$\widetilde{A}_{i,t,c}^{\text{succ}} = \left[ \frac{H_{i,t,c}^{\text{act}} - \mu_{\text{succ}}}{\sigma_{\text{succ}} + \epsilon_{\text{num}}} \right]_+, \tag{C.11}$$

where $[z]_+ = \max(z, 0)$. The standard deviation $\sigma_{\text{aux}}$ is computed over these raw advantages at *all* valid positions in successful trajectories, including positions with zero raw advantage. Equation (7) applies the scale $s_{\text{aux}} = \sigma_{\text{target}}/(\sigma_{\text{aux}} + \epsilon_{\text{num}})$, so that $A_{i,t,c}^{\text{succ}} = s_{\text{aux}} \widetilde{A}_{i,t,c}^{\text{succ}}$. Only positions above the successful-rollout entropy baseline receive positive additional credit, encouraging exploration supported by successful outcomes. When there are no successful trajectories, this branch is skipped.

### C.5 Policy Optimization

The unified advantage in Eq. (8) retains the original trajectory-level GRPO term and independently scales the success and failure corrections by $\beta_{\text{succ}}$ and $\beta_{\text{fail}}$. The failure gate is already included in Eq. (6). For enabled failed trajectories, LOW advantages shift upward and HIGH advantages shift downward; the combined sign depends on both the original advantage and the correction. Combined

advantages are not renormalized within trajectories. Expanding the clipped surrogate in Eq. (9) gives

$$\mathcal{L}_{\text{ESCAPO}}(\theta) = -\mathbb{E}_{\text{old}}\Bigg[\frac{1}{G}\sum_{i=1}^{G}\frac{1}{M_i}\sum_{t,c,d} m_{i,t,c} \times \min\Big\{\rho_{i,t,c,d}(\theta)A_{i,t,c}^{\text{ESCAPO}}, \text{clip}(\rho_{i,t,c,d}(\theta), 1-\epsilon_{\text{clip}}, 1+\epsilon_{\text{clip}})\, A_{i,t,c}^{\text{ESCAPO}}\Big\}\Bigg]. \tag{C.12}$$

The expectation is over instructions $x \sim p_{\text{task}}$ and rollout groups sampled using $\pi_{\theta_{\text{old}}}$. The mask $m_{i,t,c}$ selects valid action positions, and $M_i = \sum_{t,c,d} m_{i,t,c}$ counts valid action tokens. Entropy statistics, segmentation, and additional advantages are detached and fixed during policy updates. Entropy thus affects advantage construction rather than acting as a differentiable entropy regularizer. No additional critic or learned reward model is introduced.

# D Additional Experimental Results and Protocols

## D.1 Additional Experimental Setup

The main experiments start directly from SFT policies trained with one expert demonstration per task, for ten demonstrations in total. We use the publicly available LIBERO-10[1] and LIBERO-Spatial[2] SFT checkpoints. Within each benchmark, all compared methods share the same SFT initialization and use terminal binary task rewards. The complementary warm-up study compares continued GRPO training with ESCAPO from a shared GRPO checkpoint under matched continuation budgets. The main RL training hyperparameters are summarized in Table D.1.

**Baseline optimization settings.** GRPO uses one update epoch on all three benchmarks. PPO uses four update epochs on LIBERO-10 to improve optimization stability; its training iterations take approximately twice as long as GRPO's in this configuration. Under the available computational budget, PPO uses one update epoch on LIBERO-Plus-10 and LIBERO-Spatial. Table 1.

**Performance metrics.** Normalized AUC is computed by trapezoidal integration over the available iteration indices, divided by the interval length. Late-15% SR is the arithmetic mean of the available success-rate values in the final 15% of that interval. For SFT-initialized experiments, the common comparison interval is iterations 0–118 on LIBERO-10 and 0–119 on LIBERO-Plus-10 and LIBERO-Spatial. For warm-up experiments, Peak SR denotes the highest recorded success rate over training.

## D.2 Additional Warm-Up Results

The main warm-up results are reported in Sec. 5.3 and Table 2. Figure D.1 provides the corresponding learning curves from the shared GRPO checkpoint.

**Training efficiency after warm-up.** We report the first target crossing of the three-step moving-average success rate. Table D.2 shows that ESCAPO reaches 91% success on LIBERO-10 at iteration 25, compared with iteration 60 for GRPO, requiring 58.3% fewer training iterations. On LIBERO-Plus-10, it reaches 60% success at iteration 71 instead of 89, corresponding to a 20.2% reduction. These comparisons measure training iterations rather than wall-clock time.

## D.3 Target-Crossing Protocol and Additional Discussion

For SFT-initialized experiments, we smooth success rates with a trailing three-iteration moving average:

$$\widetilde{S}(n) = \frac{S(n-2) + S(n-1) + S(n)}{3}. \tag{D.1}$$

[1] https://huggingface.co/Haozhan72/Openvla-oft-SFT-libero10-traj1

[2] https://huggingface.co/Haozhan72/Openvla-oft-SFT-libero-spatial-traj1

Table D.1: **Key hyperparameters for on-policy RL post-training.** Dashes denote settings not used by the corresponding method.

| Parameter | GRPO | ESCAPO | PPO |
|---|---|---|---|
| Parallel training environments | 64 | 64 | 64 |
| Rollout epochs per iteration | 16 | 16 | 16 |
| Episode horizon | 512 actions | 512 actions | 512 actions |
| Actor learning rate | $2 \times 10^{-5}$ | $2 \times 10^{-5}$ | $2 \times 10^{-5}$ |
| Critic learning rate | – | – | $3 \times 10^{-3}$ |
| Global / micro-batch size | 16,384 / 32 | 16,384 / 32 | 16,384 / 32 |
| Training temperature | 1.6 | 1.6 | 1.6 |
| Rollout group size | 8 | 8 | 1 |
| Update epochs (LIBERO-10) | 1 | 1 | 4 |
| Update epochs (LIBERO-Plus-10 / Spatial) | 1 | 1 | 1 |
| GAE $(\gamma, \lambda)$ | – | – | $(0.99, 0.95)$ |
| Success weight $\beta_{\text{succ}}$ | – | 1.0 | – |
| LOW weight $\beta_{\text{low}}$ | – | 0.5 | – |
| HIGH weight $\beta_{\text{high}}$ | – | 0.25 | – |
| Entropy-gap threshold $\delta_H$ | – | 0.5 | – |
| Threshold multiplier $\alpha$ | – | 0.8 | – |

Table D.2: **Training efficiency after RL warm-up.** $N_\tau$ is the first crossing of the target by the three-step moving-average success rate. Reductions are relative to GRPO.

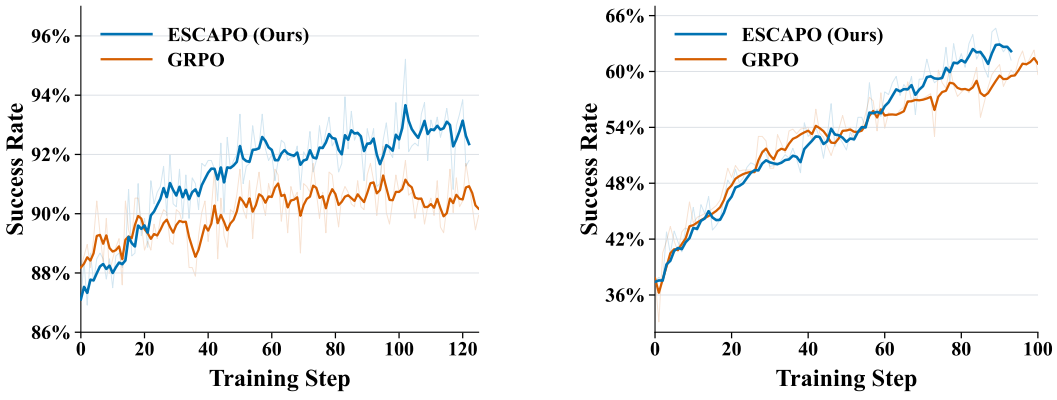
| Benchmark | Target SR | GRPO $N_\tau \downarrow$ | ESCAPO $N_\tau \downarrow$ | Reduction |
|---|---|---|---|---|
| LIBERO-10 | 91% | 60 | **25** | 58.3% |
| LIBERO-Plus-10 | 60% | 89 | **71** | 20.2% |

Let $\mathcal{I}_b$ contain comparison iterations for benchmark $b$ at which all available methods have complete three-iteration windows. The first crossing of target $\tau$ on this shared set is

$$N_\tau = \min\{n \in \mathcal{I}_b : \widetilde{S}(n) \geq \tau\}. \tag{D.2}$$

We use the same comparison intervals as Table 1 and report two targets per benchmark. Missing records can prevent the detection of earlier crossings. The GRPO crossing of 45% success on LIBERO-10 uses supplied imputed values at iterations 96–98, as marked in Table 3.

Table 3 shows that ESCAPO reaches both selected targets earlier than GRPO on the two long-horizon benchmarks. On LIBERO-10, it reaches 40% success at iteration 66, compared with iteration 68 for GRPO. On LIBERO-Plus-10, it reaches 20% success at iteration 49 instead of 61, and




(a) LIBERO-10 (b) LIBERO-Plus-10

Figure D.1: **Continued training after RL warm-up.** Success-rate curves on (a) LIBERO-10 and (b) LIBERO-Plus-10. Within each benchmark, GRPO and ESCAPO continue training from the same GRPO warm-up checkpoint.

25% at iteration 67 instead of 94. These correspond to 19.7% and 28.7% fewer iterations. Together with its higher normalized AUC, these results support improved learning progress over GRPO on long-horizon tasks.

On LIBERO-10, PPO reaches the selected targets in fewer iterations than ESCAPO: 60 versus 66 for 40% success, and 74 versus 78 for 45% success. This earlier target attainment coexists with the higher normalized AUC and late-stage success rate achieved by ESCAPO in Table 1. On LIBERO-Spatial, ESCAPO reaches 85% success at iteration 41, compared with 33 for GRPO, and reaches 90% at iteration 62, compared with 61. Thus, iteration savings do not extend to every benchmark or target.

### D.4 Simulation and Real-Robot Setup

**Task and simulation.** We build a customized MuJoCo and robosuite environment for an ARX5 six-DoF manipulator with a two-finger gripper. The robot must place a 25 mm red cube inside an open drawer and then close it as shown in Figure 8. The simulated drawer matches the geometry of the 3D-printed hardware. At each reset, the cube position is perturbed by $\pm 10$ mm along both horizontal axes, with randomized yaw. The drawer remains fixed and initially open. Simulation success requires the released cube to remain inside the open drawer for five control cycles, followed by drawer closure within 2 mm for ten cycles. Success receives a terminal reward of 5, with no dense rewards.

**Policy and training.** We initialize OpenVLA-OFT 7B using 50 successful demonstrations, equally divided between nominal and visually randomized scenes. GRPO and ESCAPO start from the same SFT checkpoint and undergo 70 on-policy RL training steps in simulation. Both methods share the environment, sampling, optimization, and domain randomization settings.

Training uses four NVIDIA A100-80GB GPUs with FSDP, BF16 precision, and gradient checkpointing. Each iteration collects 256 trajectories from 64 parallel environments over four rollout epochs. Within each GRPO group, eight rollouts share the same initial state and visual randomization parameters. Only groups containing both successful and failed rollouts contribute to policy updates. ESCAPO augments the GRPO advantage using behavior-policy entropy recorded during rollout collection. Table D.3 lists the key training and credit-assignment parameters. Simulation evaluation and real deployment use deterministic actions, with no policy updates performed on the real robot.

**Visual domain randomization.** All RL training and simulation-evaluation episodes use visual domain randomization. We perturb external-camera position and orientation by $\pm 5$ cm per axis and $\pm 2^\circ$, and wrist-camera position and orientation by $\pm 1$ cm and $\pm 1^\circ$. Focal length, material appearance, and lighting are also randomized, while the cube and drawer retain their semantic colors. Appearance is resampled every 4 s; physical parameters are not randomized.

**Real-robot platform and observations.** The real platform uses an ARX5, a PLA-printed drawer, and two Intel RealSense D435 cameras. Both domains use the same image cropping and resizing: $640 \times 480$ images are cropped and resized to $256 \times 256$, followed by the model's standard $224 \times 224$ preprocessing. The policy receives external and wrist images together with a fixed instruction.

**Action execution.** The policy predicts eight-action chunks executed at 20 Hz. Each action specifies a relative end-effector translation, an axis-angle rotation, and a gripper command. Pose increments use the robot-base frame, with limits of 0.05 m and 0.5 rad per component. Simulation uses an operational-space pose controller with 500 Hz physics. On the real robot, relative pose increments are converted into absolute end-effector targets. A 200 Hz controller executes these targets through inverse kinematics and joint-position control with torque feedforward.

### D.5 Implementation and Computational Resources

We implement GRPO, PPO, and ESCAPO using the RLinf framework (Zang et al., 2025). Each RL training run uses four NVIDIA A100 GPUs or four NVIDIA A800 GPUs, with 80 GB memory per GPU. OpenVLA-OFT 7B is trained with Fully Sharded Data Parallel (FSDP) and gradient

Table D.3: **Configuration for sim-to-real experiments.** Both methods are trained in simulation and then deployed on the real robot. Dashes denote settings not used by GRPO.

| Parameter | GRPO | ESCAPO |
|---|---|---|
| Base policy | OpenVLA-OFT 7B | |
| Training GPUs | 4×A100-80GB | |
| RL training steps | 70 | |
| Parallel environments | 64 | |
| GRPO group size | 8 | |
| Rollout epochs per iteration | 4 | |
| Training / evaluation horizon | 896 / 1,024 atomic actions | |
| Action chunk length / control frequency | 8 / 20 Hz | |
| Actor learning rate | $2 \times 10^{-5}$ | |
| Weight decay | 0.01 | |
| Global / micro-batch size | 2,048 / 16 | |
| Training sampling temperature | 1.6 | |
| Evaluation and deployment | Deterministic | |
| Terminal success reward | 5 | |
| Policy ratio clipping range | $[0.80, 1.28]$ | |
| KL / global entropy coefficient | 0 / 0 | |
| Visual / physical domain randomization | Enabled / Disabled | |
| Training precision | BF16 | |
| Actor / environment seed | 1,234 / 42 | |
| Success correction $\beta_{\text{succ}}$ | – | 1.0 |
| LOW correction $\beta_{\text{low}}$ | – | 0.5 |
| HIGH correction $\beta_{\text{high}}$ | – | 0.25 |
| Threshold multiplier $\alpha$ | – | 0.8 |
| Entropy-gap threshold $\delta_H$ | – | 1.0 |
| Minimum valid chunks | – | 8 |
| Minimum LOW / HIGH run length | – | 4 / 4 chunks |
| Minimum reliable LOW length | – | 4 chunks |
| Maximum bridged HIGH length | – | 3 chunks |

checkpointing. GRPO and ESCAPO use one optimization epoch per iteration. PPO uses four on LIBERO-10 and one on LIBERO-Plus-10 and LIBERO-Spatial. Matched rollout budgets therefore do not imply equal optimization costs.

# E Extended Related Works

**Vision-Language-Action Models.** Vision-Language-Action (VLA) models transfer vision-language knowledge to robotic control. Two prominent action-generation paradigms are discrete-token autoregressive prediction and flow-matching-based continuous generation (Kim et al., 2025b; Pertsch et al., 2025; Black et al., 2025). Autoregressive VLAs expose tractable action-token probabilities and log-probabilities, making them naturally compatible with PPO- and GRPO-style optimization. In contrast, flow-matching VLAs generate continuous action chunks without discretization and typically require specialized RL methods because standard formulations do not directly provide tractable action log-likelihoods (Chen et al., 2025a). Our work focuses on autoregressive VLAs.

**Reinforcement Learning for VLA Post-Training.** Reinforcement learning enables VLA policies to improve beyond supervised demonstrations through task feedback. Prior work has explored offline RL and offline-to-online adaptation for data-efficient post-training (Chen et al., 2025b; Huang et al., 2025), while recent studies increasingly use interactive online RL with sparse outcome rewards, learned process rewards, or multi-stage feedback (Tan et al., 2025; Li et al., 2026; Lu et al., 2025; Chen et al., 2025c). Temporal credit assignment under sparse trajectory-level supervision, however, remains underexplored.

**Entropy-Guided Credit Assignment.** Policy entropy has traditionally been used for exploration in RL (Haarnoja et al., 2018). Recent LLM post-training studies use entropy to analyze training

dynamics and assign fine-grained credit to tokens or decisions (Cui et al., 2025; Tan et al., 2026), with related ideas extended to multi-turn agents under sparse outcome rewards (Zhao et al., 2026). In VLA models, action entropy has mainly been studied for uncertainty estimation, adaptive inference, and training diagnostics (Lu et al., 2025; Yang et al., 2026; Liang et al., 2026). Its use for temporal credit assignment in online VLA RL remains underexplored. Our method uses entropy and segment structure to redistribute trajectory-level advantages.