PF-RL: PROGRESS FIELD REINFORCEMENT LEARNING VIA GOAL-CONDITIONED VALUE GEOMETRY FOR VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

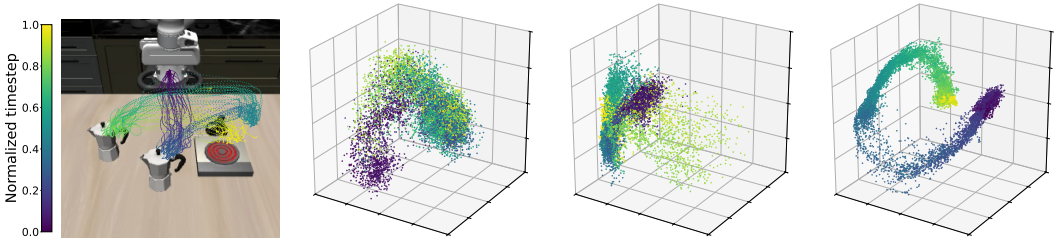
Reinforcement Fine-Tuning (RFT) has emerged as a promising paradigm for improving Vision-Language-Action (VLA) policies, yet sparse task-level outcomes provide limited credit for intermediate transitions, especially in long-horizon manipulation. A natural approach is to model intermediate task progress and use it as dense feedback for policy improvement. Despite their architectural differences, existing progress-aware methods commonly formulate task progress as an explicit scalar prediction, providing limited structure for modeling how intermediate observations relate to the task goal, which may hinder effective transition-level credit assignment. We introduce **Progress Field Reinforcement Learning (PF-RL)**, which learns a structured goal-conditioned progress representation over pretrained VLA features and converts it into dense credit for policy optimization. A lightweight shared Progress Field head maps current and goal representations into a compact progress space, where geometric distance induces goal-conditioned value, while complementary temporal and goal-structure objectives shape the learned geometry. Transition-level value changes naturally yield dense progress advantages, enabling fine-grained credit assignment for both offline policy improvement and online reinforcement fine-tuning. Extensive experiments on LIBERO, RoboTwin2.0, and real-world bimanual manipulation tasks show that PF-RL consistently improves policy performance over strong supervised fine-tuning, reinforcement fine-tuning, and progress-aware baselines.

1 INTRODUCTION

Vision-Language-Action (VLA) models have recently shown remarkable potential for general-purpose robotic manipulation, benefiting from large-scale multimodal pretraining (Team et al., 2026) and imitation learning (Kim et al., 2024; Intelligence et al., 2025b). Beyond supervised fine-tuning, Reinforcement Learning (RL) has emerged as a promising post-training paradigm for VLA models, enabling further policy improvement through interaction and task feedback (Lu et al., 2025; Chen et al., 2025c; Li et al., 2026; Kong et al., 2026; Lin et al., 2026). Despite these advances, current VLA-RL pipelines still rely heavily on sparse task-level outcomes, such as binary success or failure. This becomes particularly limiting in long-horizon manipulation, where a single terminal signal must provide learning feedback for many intermediate transitions that contribute differently to task completion. Such coarse trajectory-level supervision reveals whether an episode ultimately succeeds, but provides little guidance about which intermediate transitions advance the task and which lead to stagnation or regression. Consequently, credit assignment remains a key bottleneck for Reinforcement Fine-Tuning (RFT) of VLA policies.

To address the credit-assignment challenge, a natural solution is to model intermediate task progress and use it as dense feedback for policy improvement. Prior works have explored various forms of progress-aware guidance for robot learning (Ghasemipour et al., 2025; Zhai et al., 2025; Liang et al., 2026; Yan et al., 2026; Kim et al., 2026), demonstrating that denser intermediate feedback can substantially facilitate policy improvement under sparse task-level rewards. Despite their architectural differences, task progress is commonly represented as an explicit scalar prediction. While this formulation provides a simple and direct supervised learning target for capturing useful progress-related signals, it imposes limited structural constraints on how intermediate observations relate to

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) Task progression (b) Raw VLA space (c) Scalar prediction (d) Progress Field

Figure 1: Visualization of task progression and learned representations. The representation spaces are projected to three dimensions with PCA. (a) End-effector trajectories of *put both moka pots on the stove* task, with colors denoting normalized timesteps. (b) Raw VLA representations. (c) Representations from direct scalar progress prediction. (d) Our Progress Field representations.

the task goal, which may hinder effective transition-level credit assignment. As illustrated in Figure 1, although task execution exhibits clear temporal progression in the physical workspace, raw VLA representations remain highly entangled across stages, while representations learned with a direct scalar prediction head still exhibit substantial overlap. These observations motivate a more structured formulation of task progress over VLA representations, beyond direct scalar prediction.

Accordingly, we introduce **Progress Field Reinforcement Learning (PF-RL)**, which learns a structured goal-conditioned progress representation over pretrained VLA features and converts it into dense credit for policy optimization. Concretely, a lightweight shared **Progress Field** head maps current and goal VLA representations into a compact progress space, where their geometric distance induces the corresponding goal-conditioned value. The resulting goal-conditioned value geometry organizes observations according to their progress toward the task goal. We learn this structure through complementary temporal and goal-structure objectives. As illustrated in Figure 1d, the resulting representation exhibits a substantially clearer progress structure than both the raw VLA space and that learned with direct scalar prediction. Based on the learned Progress Field, transition-level changes in goal-conditioned value yield progress-aware advantages, providing fine-grained credit for whether each transition advances toward or deviates from the task goal. PF-RL then uses these advantages throughout offline-to-online reinforcement fine-tuning, providing dense progress-aware credit for both offline policy improvement and subsequent online adaptation. Experiments across simulated and real-world manipulation tasks demonstrate consistent improvements in policy performance and online adaptation across diverse settings.

Our main contributions are summarized as follows:

- We introduce **Progress Field**, a lightweight structured progress representation learned over pretrained VLA features, which organizes current and goal observations in a shared goal-conditioned space where geometric distance induces the corresponding value.
- We propose **PF-RL**, a VLA reinforcement fine-tuning framework that converts transition-level changes in the learned Progress Field into dense progress advantages, providing fine-grained credit for both offline policy improvement and online reinforcement fine-tuning.
- Extensive experiments on LIBERO, RoboTwin2.0, and real-world bimanual manipulation validate PF-RL and the benefits of its progress modeling over direct scalar prediction.

2 Related Work

Reinforcement Fine-Tuning (RFT) for VLA has recently emerged as an important post-training paradigm for VLA policies, enabling further improvement beyond imitation learning through interaction and task feedback. Recent studies have adapted classical policy optimization and preference-based objectives to VLA fine-tuning, including PPO (Lu et al., 2025), GRPO (Shao et al., 2024; Liu et al., 2026), and DPO (Rafailov et al., 2023; Hung et al., 2025), while open-source systems (Li et al., 2026; Yu et al., 2025) have facilitated large-scale VLA RFT and practical deployment. Building on these foundations, recent methods explore specialized mechanisms, including RL optimization for

flow-based VLAs (Chen et al., 2025a), self-referenced learning (Fei et al., 2025), world-model-based policy optimization (Zhu et al., 2025), and latent reasoning (Chen et al., 2026). However, many VLA reinforcement fine-tuning methods still rely on sparse rollout-level outcomes, limiting credit assignment across intermediate transitions. In contrast, PF-RL learns a structured goal-conditioned Progress Field over VLA representations and derives dense transition-level advantages for both offline and online policy optimization.

Progress Estimation has been widely explored as a means of providing dense learning signals for robot policy learning. Early representation-based approaches (Ma et al., 2022; 2023) pretrain goal-conditioned representations from large-scale human videos to derive dense rewards for downstream robot learning. More recent embodied VLA and RL methods leverage large-scale robotic data to learn progress-aware signals, such as steps-to-go estimates and progress changes (Ghasemipour et al., 2025; Zhai et al., 2025; Tan et al., 2025; Liang et al., 2026). Building on learned progress estimators, Yan et al. (2026) and Bai et al. (2025) use the resulting signals to guide downstream VLA policy improvement. Meanwhile, Kim et al. (2026) estimate progress directly from internal VLA representations using auxiliary prediction heads. Despite the architectural differences, many recent embodied approaches largely formulate progress as an explicit scalar prediction target, providing limited structural constraints on observation and goal relations and potentially restricting the utility of progress estimates for policy optimization. In contrast, PF-RL learns a goal-conditioned value geometry over VLA representations and derives dense progress advantages from transition-level value changes to guide policy optimization.

3 PRELIMINARY

Reinforcement Fine-Tuning of VLA Policies. The reinforcement fine-tuning of VLA policies can be formulated as a language-conditioned Partially Observable Markov Decision Process (POMDP) (Cassandra, 1998), characterized by $\mathcal{M} = \langle \mathcal{S}, \mathcal{A}, P, R_{\text{succ}}, \mathcal{I}, \mathcal{O}, \gamma, \rho_0 \rangle$, where $\mathcal{S}$ represents the underlying environment and robot state space, $\mathcal{A}$ represents the action space, $\mathcal{I}$ refers to the space of natural language task descriptions, $\mathcal{O}$ denotes the robot sensory observation space. $P : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ denotes the environment dynamics, $R_{\text{succ}} : \mathcal{S} \times \mathcal{I} \to \{0, 1\}$ is the binary reward function that detects whether the task is completed, $\gamma \in (0, 1]$ is the discount factor, and ρ_0 is the initial state distribution. A VLA policy π, parameterized as a multimodal policy $\pi_\theta(\mathbf{a}_{t:t+H-1}|o_t, l)$, feeds on observation o_t of current timestep t and language instruction $l \in \mathcal{I}$ to generate the action chunks $\mathbf{a}_{t:t+H-1} \in \mathcal{A}^H$ for the next H timesteps. The RL objective for VLA fine-tuning is to maximize the expected return under the POMDP: $\max_\theta \mathcal{J}(\theta) = \mathbb{E}_{\rho_0, P, \pi_\theta}[\sum_{t=0}^{\infty} \gamma^t R_{\text{succ}}(s_t, l)]$. Following policy-gradient methods (Williams, 1992; Silver et al., 2014), the policy can be optimized using an advantage-weighted surrogate objective:

$$\mathcal{L}_{\text{RFT}}(\theta) = -\mathbb{E}_{\pi_\theta}\left[\sum_t A_t \log \pi_\theta(\mathbf{a}_{t:t+H-1}|o_t, l)\right], \tag{1}$$

where A_t is the advantage estimator. The advantage estimator can be further obtained from value function estimation (Schulman et al., 2017), group-relative rollout comparison (Shao et al., 2024), or pairwise preference (Rafailov et al., 2023).

4 METHODOLOGY

We introduce **Progress Field Reinforcement Learning (PF-RL)**, a VLA RFT framework that learns a structured goal-conditioned progress representation over pretrained VLA features and derives dense transition-level credit for policy optimization. Our central idea is to model task progress through our proposed Progress Field rather than through direct scalar prediction. For a fixed task goal, the Progress Field organizes observations in a shared progress space, where geometric distance to the goal induces the corresponding goal-conditioned value. PF-RL consists of three key components: (i) constructing a goal-conditioned Progress Field over VLA representations; (ii) learning its geometry to reflect temporal progress and goal structure; and (iii) deriving progress-aware advantages for offline and online policy optimization. An overview of PF-RL is shown in Figure 2.

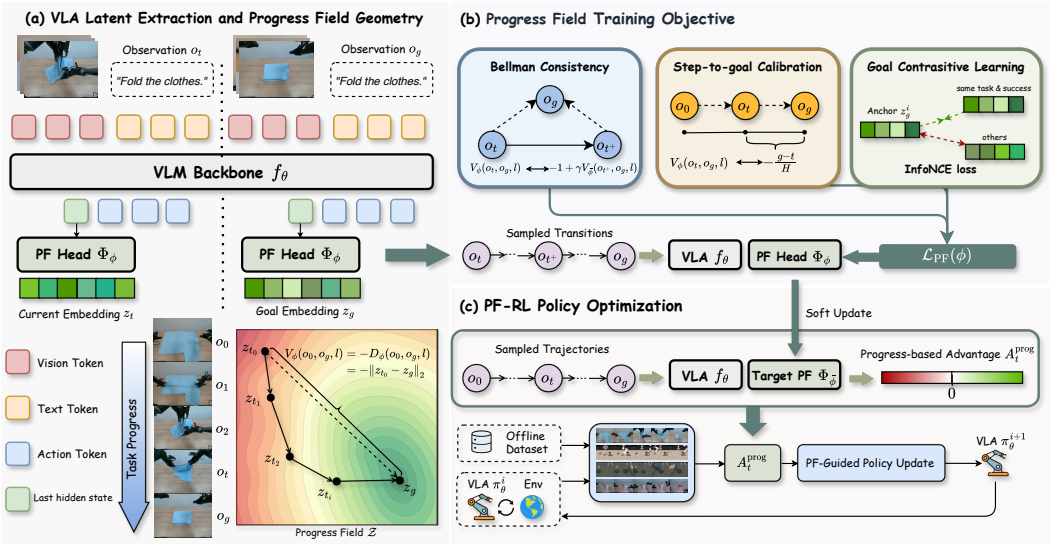


Figure 2: An illustrative diagram of our proposed *Progress Field Reinforcement Learning*.

4.1 Progress Field as Goal-Conditioned Value Geometry

VLA Latent Representation. Given the current observation o_t, goal observation o_g, and language instruction l, we extract their last-layer hidden representations from the VLA backbone:

$$h_t = f_\theta(o_t, l), \qquad h_g = f_\theta(o_g, l), \tag{2}$$

where f_θ denotes the VLA backbone and h_t and h_g are the corresponding end-of-input hidden representations before action prediction. We then map both representations into a compact progress space using a shared lightweight projection head Φ_ϕ parameterized by ϕ:

$$z_t = \Phi_\phi(h_t), \qquad z_g = \Phi_\phi(h_g). \tag{3}$$

Goal-Conditioned Value Geometry. Given the current and goal embeddings z_t and z_g, we define their distance and the corresponding goal-conditioned value as

$$D_\phi(o_t, o_g, l) = \|z_t - z_g\|_2, \qquad V_\phi(o_t, o_g, l) = -D_\phi(o_t, o_g, l). \tag{4}$$

Following Park et al. (2024), this parameterization assigns higher values to observations embedded closer to the task goal and lower values to those farther away. For a fixed goal o_g and instruction l, the resulting $V_\phi(\cdot, o_g, l)$ defines a goal-conditioned value geometry over observations, which we refer to as the **Progress Field**. We refer to the projection head Φ_ϕ as the Progress Field head. Task progress is therefore encoded through goal-relative geometry rather than predicted as a standalone scalar. However, this geometric parameterization alone does not ensure alignment with temporal task progress along robot trajectories. We next learn the Progress Field with objectives that impose temporal consistency and task-level goal structure.

4.2 Progress Field Learning

Let $\mathcal{D} = \{\tau_i\}_{i=1}^N$ denote the trajectory data used for Progress Field learning, where each trajectory $\tau_i = \{(o_t^i, \mathbf{a}_{t:t+H-1}^i)\}_{t=0}^{T_i}$ is associated with a language instruction l_i, and H denotes the action-chunk horizon. The dataset is initialized with offline demonstrations and continuously augmented with newly collected trajectories during online fine-tuning. During training, we construct tuples $(o_t, o_{t^+}, o_u, o_g, l)$. Here, o_t is the current observation, and o_{t^+} is the observation after executing one action chunk. Since each chunk executes H actions, $t^+ = t + H$. We take o_g as the terminal observation of the same trajectory and randomly sample o_u with $t^+ < u < g$ as an auxiliary subgoal. Both successful and failed trajectories are included in Progress Field training. For a failed trajectory, its terminal observation is treated as the observed endpoint of the trajectory rather than a task-completion goal. Further details on goal assignment are provided in Appendix B. Based on these tuples, we learn the Progress Field using complementary objectives that align its geometric distance with temporal progress while preserving task-level goal structure.

Bellman consistency. The first objective imposes temporal consistency on the Progress Field. Following Park et al. (2024), we use an expectile-weighted goal-conditioned TD objective:

$$\mathcal{L}^g_{\text{td}}(\phi) = \mathbb{E}_{(o_t, o_{t^+}, o_g, l)} \left[\ell^\tau_2 \left(-\mathbb{1}(o_t \neq o_g) + \gamma V_{\bar{\phi}}(o_{t^+}, o_g, l) - V_\phi(o_t, o_g, l) \right) \right], \tag{5}$$

where $\ell^\tau_2(x) = |\tau - \mathbb{1}(x < 0)|x^2$ is the expectile regression loss and $V_{\bar{\phi}}$ is computed using a slowly updated target projection with parameters $\bar{\phi}$. Using $\tau > 0.5$ places greater weight on positive TD residuals, biasing the learned value toward higher-value outcomes rather than the conditional mean. We set $\gamma = 1$ in practice, consistent with the undiscounted steps-to-goal interpretation of the learned value. This objective enforces temporal consistency of the goal-conditioned value, aligning its trajectory-wise changes with task progress.

Step-to-goal calibration. While Bellman bootstrapping enforces temporal consistency, we further calibrate Progress Field with the remaining temporal distance to the target. For a target indexed by g, we define $\hat{d}_{t,g} = \frac{g-t}{H}$ to measure the remaining number of action-chunk intervals. Since V_ϕ is parameterized as a negative distance, we optimize

$$\mathcal{L}^g_{\text{step}}(\phi) = \mathbb{E}_{(o_t, o_g, l)} \left[\left(V_\phi(o_t, o_g, l) + \hat{d}_{t,g} \right)^2 \right]. \tag{6}$$

This objective calibrates the geometric distance with the remaining steps to the goal, providing an explicit temporal scale for the Progress Field.

Goal Contrastive Learning. While the temporal objectives structure progress within individual trajectories, they do not explicitly distinguish goal representations across different tasks. We therefore apply an InfoNCE-style contrastive objective (Oord et al., 2018) over terminal goal embeddings to capture task-level goal structure. Let $e_g = \text{Normalize}(z_g)$. For anchor i, successful terminal observations from the same task form the positive set $\mathcal{P}(i)$, while terminal observations from failed trajectories of the same task and those from different tasks form the negative set $\mathcal{N}(i)$:

$$\mathcal{L}_{\text{con}}(\phi) = -\mathbb{E}_i \left[\log \frac{\sum_{j \in \mathcal{P}(i)} \exp\left(\text{sim}(e_{g_i}, e_{g_j})/\alpha \right)}{\sum_{j \in \mathcal{P}(i) \cup \mathcal{N}(i)} \exp\left(\text{sim}(e_{g_i}, e_{g_j})/\alpha \right)} \right], \tag{7}$$

where $\text{sim}(\cdot, \cdot)$ denotes cosine similarity and α is the contrastive temperature, encouraging same-task successful goals to cluster while separating failed and cross-task goals.

Auxiliary subgoal shaping. To better exploit the available geometric structure in long-horizon trajectories, we reuse later observations as auxiliary subgoals. For the auxiliary subgoal o_u, we apply the Bellman-consistency and step-to-target objectives, while the terminal-goal branch additionally includes goal contrastive learning. The overall Progress Field objective is

$$\mathcal{L}_{\text{PF}}(\phi) = \underbrace{\mathcal{L}^g_{\text{td}} + \mathcal{L}^g_{\text{step}} + \mathcal{L}_{\text{con}}}_{\mathcal{L}_{\text{goal}}} + \lambda_{\text{sub}} \underbrace{\left(\mathcal{L}^u_{\text{td}} + \mathcal{L}^u_{\text{step}} \right)}_{\mathcal{L}_{\text{sub}}}. \tag{8}$$

After each update, the target parameters $\bar{\phi}$ are softly updated from ϕ to stabilize TD bootstrapping. Meanwhile, gradients from $\mathcal{L}_{\text{PF}}$ are stopped at the VLA representation h, so the progress-aware geometry is learned through the Progress Field head Φ_ϕ while preserving the underlying VLA representation for further policy optimization.

4.3 PF-RL Policy Optimization

Based on the learned Progress Field, we define a progress-aware advantage from transition-level changes in the goal-conditioned value. Specifically, for a transition $o_t \to o_{t^+}$, we compute

$$A^{\text{prog}}_t = -\mathbb{1}(o_t \neq o_g) + \gamma V_{\bar{\phi}}(o_{t^+}, o_g, l) - V_{\bar{\phi}}(o_t, o_g, l). \tag{9}$$

This advantage provides fine-grained credit by evaluating progress of each transition toward the goal relative to the learned temporal scale. For successful trajectories, the terminal observation naturally serves as the goal reference. For failed trajectories, the terminal observation does not represent task completion. During online fine-tuning, we therefore replace its goal embedding with a per-task goal prototype only when computing A^{prog}_t. The prototype is maintained through an exponential moving average of successful goal embeddings from the same task. This provides a success-based reference for evaluating progress on failed online trajectories. PF-RL then uses A^{prog}_t for offline policy improvement and subsequent online reinforcement fine-tuning, as described below.

Offline Optimization. In the offline stage, PF-RL reweights the native VLA imitation objective using A_t^{prog} following the principle of advantage-weighted regression (Peng et al., 2019):

$$\mathcal{L}_{\text{off}}(\theta) = \mathbb{E}_{(o_t, \mathbf{a}_{t:t+H-1}, l)\sim\mathcal{D}} \left[\exp\left(\frac{A_t^{\text{prog}}}{\beta} \right) \ell_{\text{VLA}}\left(\theta; o_t, l, \mathbf{a}_{t:t+H-1}\right) \right], \tag{10}$$

where β controls the sharpness of advantage weighting and ℓ_{VLA} denotes the backbone-specific imitation loss. In simulation, ℓ_{VLA} is the token-level cross-entropy loss of OpenVLA-OFT, whereas in real-world experiments, it is the flow-matching objective of $\pi_{0.5}$. This weighting favors transitions with higher progress advantages.

Online Fine-Tuning. In simulation, sparse terminal rewards provide rollout-level task feedback, while A_t^{prog} provides dense progress-aware advantages for intermediate transitions. For a group of G rollouts with trajectory-level task rewards $\{R_i\}_{i=1}^G$, we compute the GRPO (Shao et al., 2024) advantage by group-wise normalization and combine it with the transition-level progress advantage:

$$A_i^{\text{GRPO}} = \frac{R_i - \text{mean}(\{R_j\}_{j=1}^G)}{\text{std}(\{R_j\}_{j=1}^G) + \varepsilon}, \qquad A_{i,t}^{\text{sim}} = A_i^{\text{GRPO}} + \lambda_{\text{prog}} A_{i,t}^{\text{prog}}, \tag{11}$$

where λ_{prog} controls the contribution of the progress advantage. We then optimize the policy with the clipped GRPO objective. Defining the policy ratio as $r_{i,t}(\theta) = \frac{\pi_\theta(\mathbf{a}_{t:t+H-1}|o_t,l)}{\pi_{\theta_{\text{old}}}(\mathbf{a}_{t:t+H-1}|o_t,l)}$, we optimize

$$\mathcal{L}_{\text{GRPO}}(\theta) = -\mathbb{E}\left[\min\left(r_{i,t}(\theta) A_{i,t}^{\text{sim}},\ \text{clip}\left(r_{i,t}(\theta), 1-\epsilon, 1+\epsilon\right) A_{i,t}^{\text{sim}} \right) \right]. \tag{12}$$

For real-robot fine-tuning, where interaction is costly and online data are limited, we use A_t^{prog} to reweight the flow-matching objective of $\pi_{0.5}$ according to Eq. 10. This conservative update emphasizes transitions with higher progress advantages while retaining the native policy objective.

Notably, Progress Field learning is interleaved with both offline and online policy optimization rather than performed as a separate pre-training stage. This keeps the learned geometry synchronized with evolving VLA representations while allowing newly collected trajectories to continuously refine the Progress Field. Overall, PF-RL leverages the learned Progress Field to provide fine-grained credit throughout VLA policy optimization. Detailed pseudocode is provided in Appendix A.

5 Experiments

We evaluate PF-RL on simulated and real-world robotic manipulation tasks. Our experiments aim to answer the following questions: (1) Can PF-RL effectively improve VLA reinforcement fine-tuning across simulation and real-world settings? (Section 5.1 and Section 5.2) (2) Does the Progress Field formulation provide more effective progress modeling than direct scalar prediction? (Section 5.3 and Appendix D) (3) How do the key design choices of PF-RL affect policy performance? (Section 5.3 and Appendix E) (4) Does the learned Progress Field capture structured goal-conditioned value geometry that reflects task progress? (Section 5.4, Appendix F, and Appendix H)

5.1 Simulation Experiments.

Experimental Setup. We evaluate PF-RL on two simulated manipulation benchmarks, LIBERO and RoboTwin2.0. LIBERO (Liu et al., 2023) comprises the Spatial, Object, Goal, and Long suites, with 10 tasks per suite and 40 tasks in total. For each task, we sample 30 expert demonstrations from the official dataset for offline training. RoboTwin2.0 (Chen et al., 2025b) features bimanual manipulation tasks with complex and contact-rich interactions. We construct the offline training data following the protocol in Li et al. (2026). Compared with LIBERO, where recent VLA methods already achieve strong performance, RoboTwin2.0 offers a more challenging benchmark for evaluating policy improvement under complex manipulation dynamics. For all simulation experiments, we initialize PF-RL from the OpenVLA-OFT checkpoint provided by Li et al. (2026). We use RLinf-GRPO as our primary controlled RFT baseline on LIBERO. Both methods start from the same OpenVLA-OFT checkpoint and are trained on the same offline data for the same number of optimization steps, with PF-RL applying its offline objective and RLinf-GRPO using standard SFT. The resulting policies are then fine-tuned online using their respective objectives under the same training budget. We additionally report published results of SFT, RFT, and progress-aware VLA methods for reference. More details are provided in Appendix B.1.

Table 1: Success Rate (%) comparison on LIBERO Liu et al. (2023). The best results are in **bold**, while ± denotes standard deviation over 3 independent training seeds. † denotes baselines reproduced under the same offline-to-online budgets; other baseline results are from the original papers.

Models	Spatial	Object	Goal	Long	Average
SFT baselines					
π_0 (Black et al., 2024)	96.8	98.8	95.8	85.2	94.2
$\pi_{0.5}$ (Intelligence et al., 2025b)	98.8	98.2	98.0	92.4	96.9
OpenVLA (Kim et al., 2024)	84.7	88.4	79.2	53.7	76.5
OpenVLA-OFT (Kim et al., 2025)	97.6	98.4	97.9	94.5	97.1
RFT baselines					
GRAPE (Zhang et al., 2024)	88.5	92.1	83.1	57.2	80.2
TGRPO (Chen et al., 2025c)	90.4	92.2	81.0	59.2	80.7
VLA-RL (Lu et al., 2025)	90.2	91.8	82.2	59.8	81.0
RLinf-GRPO† (Yu et al., 2025)	97.6±0.4	98.7±0.6	95.2±0.7	96.8±0.5	97.1
Progress-aware baselines					
ProgressVLA (Yan et al., 2026)	88.2	96.4	87.2	66.2	84.5
ProgVLA (Kim et al., 2026)	87.6	96.0	92.0	88.6	91.1
EVOLVE-VLA (Bai et al., 2025)	95.4	97.4	95.8	94.4	95.8
OpenVLA-OFT + PF-RL					
PF-RL (offline)	87.4±0.8	77.8±0.5	84.6±1.2	76.8±1.4	81.7
+ online	**99.2**±0.7	**99.5**±0.3	**98.8**±1.0	**99.0**±0.5	**99.1**

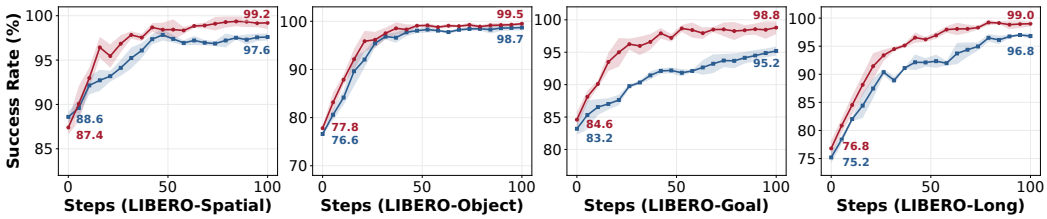


Figure 3: Online fine-tuning curves of **PF-RL** and **RLinf-GRPO** across the LIBERO task suites.

Results Analysis. Table 1 shows that PF-RL achieves an average success rate of 99.1% on LIBERO after online fine-tuning. Compared with RLinf-GRPO under matched offline-to-online training budgets, PF-RL improves performance across all four suites, with gains of 2.0 percentage points on average and 2.2 percentage points on LIBERO-Long. The improvement on LIBERO-Long suggests that dense progress feedback can facilitate policy learning in temporally extended manipulation tasks. Figure 3 further shows faster convergence and higher final success rates, supporting the effectiveness of progress-aware feedback for policy optimization. PF-RL also achieves a higher reported average success rate than the progress-aware baselines. Meanwhile, Table 2 shows consistent improvements over RLinf-GRPO across all four evaluated tasks on RoboTwin2.0, providing further evidence of its effectiveness in challenging bimanual manipulation.

5.2 Real-World Experiments

Experimental Setup. To evaluate PF-RL in real-world settings, we conduct experiments on the Agilex Cobot Magic dual-arm platform. We consider four challenging manipulation tasks: *Fold Clothes*, *Stack all Plates*, *Put all Blocks into Bucket*, and *Fill Vase and Arrange Flowers*. We train *Fold Clothes* separately in a single-task setting, while the latter three tasks are jointly trained in a multi-task setting. For all real-world experiments, we initialize PF-RL from the official JAX checkpoint of $\pi_{0.5}$ (Intelligence et al., 2025b), and then perform two iterations of real-world reinforcement fine-tuning following Intelligence et al. (2025a), with 20 rollouts collected per task in each iteration. To improve the safety and efficiency of real-world exploration, we adopt human-in-the-loop intervention (Luo et al., 2025), allowing corrective intervention during rollout collection. More detailed settings are provided in Appendix B.2.

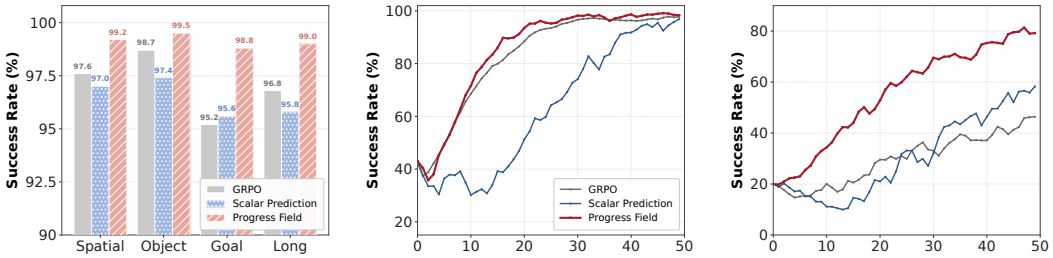
Table 2: Success Rate (%) comparison on RoboTwin2.0 Chen et al. (2025b).

Method	Beat Block Hammer	Pick Dual Bottles	Stack Bowls Two	Place Empty Cup	Avg.
π_0 (SFT) (Black et al., 2024)	59	50	53	60	56
RDT (SFT) (Liu et al., 2024)	22	18	42	42	31
OpenVLA-OFT (Kim et al., 2025)	13	11	82	81	47
RLinf-GRPO† (Yu et al., 2025)	82	70	71	94	79
OpenVLA-OFT+PF-RL (offline→online)	25→**91**	19→**93**	53→**86**	64→**96**	**92**

Table 3: Real-world comparison under the original setting and three visual shifts: unseen objects (-Obj), background (-BG), and lighting (-L). All baselines are reproduced under the same protocol and evaluated over 20 rollouts per setting. Detailed settings are provided in Appendix C.

Methods	**Fold Clothes**				**Stack all Plates**			
	Original	-Obj	-BG	-L	Original	-Obj	-BG	-L
$\pi_{0.5}$ (Intelligence et al., 2025b)	75	65 (-10%)	70 (-5%)	55 (-20%)	70	65 (-5%)	55 (-15%)	65 (-5%)
Evo-RL (Contributors, 2026)	90	70 (-20%)	80 (-10%)	65 (-25%)	90	85 (-5%)	80 (-10%)	85 (-5%)
$\pi_{0.5}$+PF-RL (offline→online)	80→**95**	90 (-5%)	90 (-5%)	85 (-10%)	85→**95**	95 (-0%)	90 (-5%)	90 (-5%)
Methods	**Put all Blocks into Bucket**				**Fill Vase and Arrange Flowers**			
	Original	-Obj	-BG	-L	Original	-Obj	-BG	-L
$\pi_{0.5}$ (Intelligence et al., 2025b)	70	55 (-15%)	60 (-10%)	35 (-35%)	50	40 (-10%)	35 (-15%)	30 (-20%)
Evo-RL (Contributors, 2026)	85	80 (-5%)	75 (-10%)	65 (-20%)	75	60 (-15%)	55 (-20%)	65 (-10%)
$\pi_{0.5}$+PF-RL (offline→online)	80→**90**	80 (-10%)	85 (-5%)	75 (-15%)	65→**85**	75 (-10%)	75 (-10%)	80 (-5%)

Results Analysis. As shown in Table 3, PF-RL consistently improves policy performance across all evaluated real-world tasks. Under the Original setting, offline PF-RL already improves upon the $\pi_{0.5}$ policy, while further adaptation with newly collected real-robot trajectories yields substantial additional gains and outperforms online Evo-RL. Meanwhile, the resulting PF-RL policy also maintains strong performance under the evaluated unseen-object, background, and lighting shifts. Overall, these results demonstrate the effectiveness of progress-aware feedback for both offline policy improvement and real-world online adaptation.



(a) LIBERO Final Performance (b) LIBERO-Goal Task5 (c) LIBERO-Long Task0

Figure 4: Scalar prediction vs. Progress Field. (a) Final performance on four LIBERO suites. (b–c) Online adaptation on held-out tasks, with progress models trained on the remaining nine tasks.

5.3 Ablation Study

Effect of the Progress Field Formulation. Figure 4a shows that Progress Field outperforms direct scalar prediction across all four LIBERO suites under the same training pipeline. Table 4 further shows that this advantage persists when both methods omit contrastive learning, and that Progress Field outperforms the MLP readout with the same projection architecture and training objectives. To assess cross-task transfer, we train the progress models offline on the remaining nine tasks for each held-out task. The results demonstrate that Progress Field enables stronger online adaptation on held-out tasks, suggesting better cross-task transfer. Further analysis is provided in Appendix D.

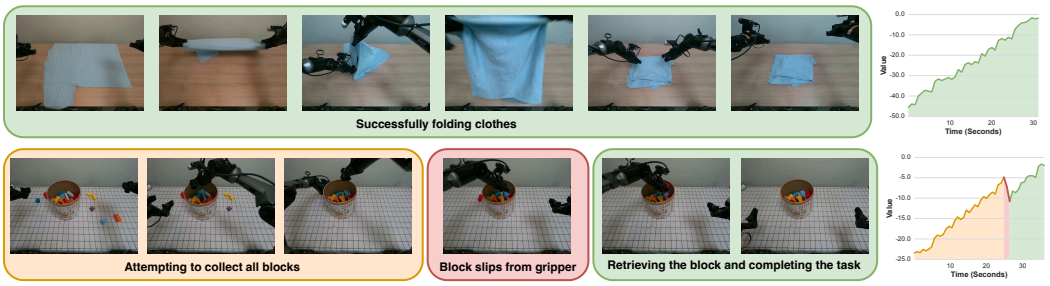


Figure 6: Qualitative visualization of goal-conditioned values induced by the learned Progress Field during real-world task execution. The top shows successful cloth folding, while the bottom illustrates putting all blocks into bucket with an intermediate grasp failure followed by successful recovery. More visualizations are provided in Appendix H.

Effect of Progress Field Training Objectives. As shown in Figure 5, the complete Progress Field training objective consistently achieves the strongest performance across all four LIBERO suites, with detailed numerical results reported in Table 5. Removing the Bellman consistency loss causes the most pronounced degradation on LIBERO-Long, highlighting its importance for propagating goal-conditioned values over long-horizon trajectories. Meanwhile, removing the step-to-goal objective leads to broad performance drops by weakening the temporal calibration of the learned geometry, while removing the goal-contrastive objective mainly reduces task-level goal discrimination. Moreover, the HILP-inspired TD-only metric variant as shown in Table 5 also underperforms the full objective. Together, these results show that the three objectives provide complementary temporal, metric, and task-level constraints for learning a well-structured goal-conditioned value geometry. Detailed scores are reported in Appendix E.2.

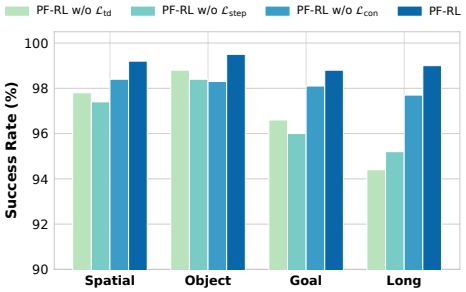


Figure 5: Ablation of Progress Field training objectives on the four LIBERO suites.

5.4 Visualization

Figure 6 further visualizes the goal-conditioned value induced by the learned Progress Field along representative real-world executions. During successful cloth folding, the estimated value exhibits a clear overall increase as the manipulation progresses toward the goal. More notably, in the *Put all Blocks into Bucket* task, the value first increases with task progress, drops sharply when the red block slips from the gripper, and rises again after the robot recovers and successfully completes the task. Such non-monotonic changes indicate that the learned value responds to current events rather than simply increasing with elapsed time. These observations show that the learned goal-conditioned value responds consistently to task advancement, execution degradation, and subsequent recovery, supporting the effectiveness of the Progress Field for progress-aware advantage estimation.

6 Conclusion

In this work, we study the credit-assignment challenge in VLA reinforcement fine-tuning, where sparse task-level outcomes provide limited guidance for intermediate transitions. To address this problem, we propose PF-RL, which learns a lightweight structured progress representation over pretrained VLA features and derives dense progress advantages from the resulting goal-conditioned value geometry for both offline and online policy optimization. Extensive experiments on LIBERO, RoboTwin2.0, and real-world bimanual manipulation tasks demonstrate consistent policy improvements, while controlled comparisons and representation analyses further support the benefits of Progress Field over direct scalar prediction. Overall, our results highlight goal-conditioned value geometry as an effective formulation for learning progress-aware credit in VLA RFT.

AI USE STATEMENT

In this work, we used generative AI tools to help refine the conceptual framework of PF-RL, articulate and refine the research idea initially developed by the authors, provide feedback on research methodology and experimental design, assist with method implementation and translation, and support the analysis and discussion of experimental results. We did not use generative AI tools to generate synthetic datasets, formulate mathematical claims, provide critical ingredients for proving mathematical claims, assist in the writing of proofs, or clean and reformat datasets. Qualitative and thematic data analysis was not applicable to this work. We have reviewed all AI-assisted work, including carefully assessing suggestions on research methodology and experimental design, verifying that the AI-assisted analysis of experimental results was supported by the underlying data, and inspecting AI-assisted code for correctness and consistency with the described method. We take full responsibility for all AI-assisted content in this work.

REPRODUCIBILITY STATEMENT

We provide comprehensive details to facilitate the reproduction of our results. The complete formulation of PF-RL and its training procedures are presented in Section 4 and Appendix A, including the construction and sampling of training tuples, the Progress Field training objective, and the offline and online policy optimization pipelines. All experimental settings, model hyperparameters, and hardware specifications are reported in Appendix B. Our simulation implementation builds on RLinf, while our real-robot implementation uses OpenPI as the underlying VLA codebase. For human-in-the-loop data collection during real-world online RL, we build on the open-source Evo-RL. Together, the mathematical formulations, implementation details, and experimental configurations provided in the paper and appendices specify the complete experimental protocol of this work.

REFERENCES

Zechen Bai, Chen Gao, and Mike Zheng Shou. Evolve-vla: Test-time training from environment feedback for vision-language-action models. *arXiv preprint arXiv:2512.14666*, 2025.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. pi0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Anthony R Cassandra. A survey of pomdp applications. In *Working notes of AAAI 1998 fall symposium on planning with partially observable Markov decision processes*, volume 1724, 1998.

Hao Chen, Jiaming Liu, Zhonghao Yan, Nuowei Han, Renrui Zhang, Chenyang Gu, Jialin Gao, Ziyu Guo, Siyuan Qian, Yinxi Wang, et al. Last-r1: Reinforcing robotic manipulation via adaptive physical latent reasoning. *arXiv preprint arXiv:2604.28192*, 2026.

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

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025b.

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. *arXiv preprint arXiv:2506.08440*, 2025c.

Evo-RL Contributors. Evo-rl: Towards iterative policy improvement in real-world offline rl. `https://github.com/MINT-SJTU/Evo-RL`, 2026.

Senyu Fei, Siyin Wang, Li Ji, Ao Li, Shiduo Zhang, Liming Liu, Jinlong Hou, Jingjing Gong, Xianzhong Zhao, and Xipeng Qiu. Srpo: Self-referential policy optimization for vision-language-action models. *arXiv preprint arXiv:2511.15605*, 2025.

Seyed Kamyar Seyed Ghasemipour, Ayzaan Wahid, Jonathan Tompson, Pannag Sanketi, and Igor Mordatch. Self-improving embodied foundation models. *arXiv preprint arXiv:2509.15155*, 2025.

Chia-Yu Hung, Navonil Majumder, Haoyuan Deng, Liu Renhang, Yankang Ang, Amir Zadeh, Chuan Li, Dorien Herremans, Ziwei Wang, and Soujanya Poria. Nora-1.5: A vision-language-action model trained using world model-and action-based preference rewards. *arXiv preprint arXiv:2511.14659*, 2025.

Physical Intelligence, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Kevin Black, Ken Conley, Grace Connors, James Darpinian, Karan Dhabalia, Jared DiCarlo, et al. $\pi^*_{0.6}$: A vla that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025a.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025b.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. In *Robotics: Science and Systems (RSS)*, 2025.

Seungsu Kim, Jinyoung Choi, Seungmin Baek, and Jean-Michel Renders. Progvla: Progress-aware robot manipulation skill learning. *arXiv preprint arXiv:2605.28231*, 2026.

Yilun Kong, Yunpeng Qing, Guozheng Ma, Haoyu Wang, Li Shen, Zhi Hou, and Dacheng Tao. Extoken: Structured exploration for efficient vision-language-action reinforcement fine-tuning. *arXiv preprint arXiv:2607.12931*, 2026.

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

Anthony Liang, Yigit Korkmaz, Jiahui Zhang, Minyoung Hwang, Abrar Anwar, Sidhant Kaushik, Aditya Shah, Alex S Huang, Luke Zettlemoyer, Dieter Fox, et al. Robometer: Scaling general-purpose robotic reward models via trajectory comparisons. *arXiv preprint arXiv:2603.02115*, 2026.

Sixu Lin, Yunpeng Qing, Litao Liu, Ming Zhou, Ruixing Jin, Xiaoyi Fan, and Guiliang Liu. Dygro-vla: Cross-task scaling of vision-language-action models via dynamic grouped residual optimization. *arXiv preprint arXiv:2605.17486*, 2026.

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. *Advances in Neural Information Processing Systems*, 38:97121–97151, 2026.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. *arXiv preprint arXiv:2410.07864*, 2024.

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.

Jianlan Luo, Charles Xu, Jeffrey Wu, and Sergey Levine. Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning. *Science Robotics*, 10(105):eads5033, 2025.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. Liv: Language-image representations and rewards for robotic control. In *International Conference on Machine Learning*, pp. 23301–23320. PMLR, 2023.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.

Seohong Park, Tobias Kreiman, and Sergey Levine. Foundation policies with hilbert representations. *arXiv preprint arXiv:2402.15567*, 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.

Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in neural information processing systems*, 36:53728–53741, 2023.

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.

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, pp. 387–395. Pmlr, 2014.

Huajie Tan, Sixiang Chen, Yijie Xu, Zixiao Wang, Yuheng Ji, Cheng Chi, Yaoxu Lyu, Zhongxia Zhao, Xiansheng Chen, Peterson Co, et al. Robo-dopamine: General process reward modeling for high-precision robotic manipulation. *arXiv preprint arXiv:2512.23703*, 2025.

Brain Team, Ziyang Gong, Haoming Gu, Zehang Luo, Tianyi Zhang, Tao Tao, Yixiao Chi, Zhe Liu, Lingsi Zhu, Jingyuan Liu, et al. Ace-brain-0.5: A unified embodied foundational model for physical agentic ai. *arXiv preprint arXiv:2607.04426*, 2026.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256, 1992.

Hongyu Yan, Qiwei Li, Jiaolong Yang, and Yadong Mu. Progressvla: Progress-guided diffusion policy for vision-language robotic manipulation. *arXiv preprint arXiv:2603.27670*, 2026.

Chao Yu, Yuanqing Wang, Zhen Guo, Hao Lin, Si Xu, Hongzhi Zang, Quanlu Zhang, Yongji Wu, Chunyang Zhu, Junhao Hu, et al. Rlinf: Flexible and efficient large-scale reinforcement learning via macro-to-micro flow transformation. *arXiv preprint arXiv:2509.15965*, 2025.

Shaopeng Zhai, Qi Zhang, Tianyi Zhang, Fuxian Huang, Haoran Zhang, Ming Zhou, Shengzhe Zhang, Litao Liu, Sixu Lin, and Jiangmiao Pang. A vision-language-action-critic model for robotic real-world reinforcement learning. *arXiv preprint arXiv:2509.15937*, 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.

Fangqi Zhu, Zhengyang Yan, Zicong Hong, Quanxin Shou, Xiao Ma, and Song Guo. Wmpo: World model-based policy optimization for vision-language-action models. *arXiv preprint arXiv:2511.09515*, 2025.

A PSEUDOCODE

To provide a concise overview of PF-RL, we summarize its training procedure in Algorithm 1.

Algorithm 1 Progress Field Reinforcement Learning (PF-RL)

Input: offline dataset $\mathcal{D}$, VLA policy π_θ, projection head Φ_ϕ, offline training steps N_{off}, online iterations N_{on}.
Initialize: target projection $\bar{\phi} \leftarrow \phi$; randomly initialize task-level goal prototypes.

```
# PF-RL Offline Optimization
for i = 1 to N_off do
    Sample (o_t, a_{t:t+H-1}, o_{t+}, o_u, o_g, l) from D_offline.
    Extract VLA representations h ← sg[f_{θ_VLA}(o, l)].
    Update Φ_φ with L_PF according to Eq. 8.
    Soft-update φ̄ from φ; EMA-update goal prototypes with successful terminal embeddings.
    Compute transition-level A_t^prog with the Progress Field according to Eq. 9.
    Update π_θ with the progress-weighted objective in Eq. 10.
end for
# PF-RL Online Fine-Tuning
for i = 1 to N_on do
    Collect online rollout dataset D_online^(i) with the current policy π_θ.
    Sample (o_t, a_{t:t+H-1}, o_{t+}, o_u, o_g, l) from D_online^(i).
    Extract VLA representations h ← sg[f_{θ_VLA}(o, l)].
    Update Φ_φ with L_PF according to Eq. 8.
    Soft-update φ̄ from φ; EMA-update goal prototypes with successful terminal embeddings.
    Compute A_t^prog by Eq. 9, using goal prototypes for failed trajectories.
    if simulation then
        Combine A^GRPO and A^prog according to Eq. 11, and update π_θ according to Eq. 12.
    else
        Update π_θ with the progress-weighted objective in Eq. 10.
    end if
end for
return optimized VLA policy π_θ.
```

B EXPERIMENTAL DETAILS

We use a shared residual MLP head Φ_ϕ with a hidden dimension of 512 to project final-layer VLA representations of current and goal observations into 32-dimensional embeddings. Progress Field learning and policy optimization are interleaved throughout offline and online training, updating only the projection head and VLA policy parameters, respectively. The target head $\Phi_{\bar{\phi}}$ is softly updated after each Progress Field update with a coefficient of 0.005. For Progress Field training, g denotes the terminal timestep for both successful and failed trajectories, such that o_g corresponds to the final observation of the trajectory. The auxiliary observation o_u is sampled from the same trajectory with $t^+ < u < g$. Accordingly, the calibration targets $(g-t)/H$ and $(u-t)/H$ encode observed temporal separations within a trajectory. For progress-advantage computation, we maintain a task-level goal prototype for each task. Each prototype is randomly initialized and updated via an exponential moving average of successful terminal embeddings throughout offline and online training. At each optimization step, a prototype is updated only if the current batch contains successful terminal samples from the corresponding task; otherwise, it remains unchanged. As successful expert trajectories are available during offline training, the prototypes are progressively anchored to task-completion states before online fine-tuning. The main LIBERO comparison in Table 1 and Figure 3 is conducted with three independent training seeds, and the full PF-RL results are reused in the ablation studies. Due to computational constraints, other experiments use a single training seed unless otherwise specified. We report the checkpoint obtained at the end of each prescribed training stage, without validation-based checkpoint selection.

B.1 Simulation Settings

For simulation experiments, we instantiate PF-RL with OpenVLA-OFT (Kim et al., 2025) as the VLA backbone. For Progress Field learning, we use an expectile coefficient of $\tau = 0.7$, a contrastive temperature of $\alpha = 0.05$, and an auxiliary subgoal weight of $\lambda_{\text{sub}} = 1.0$. During offline optimization, we use the advantage-weighted objective in Eq. 10 with an advantage temperature of $\beta = 0.5$ for 30K steps. During online reinforcement fine-tuning, we combine the group-relative advantage with the Progress Field advantage using $\lambda_{\text{prog}} = 1.0$ and optimize the policy with the clipped GRPO objective. We perform 100 optimization epochs, with each epoch comprising 64 groups and a group size of 8. The same rollout data are also used to continuously update the Progress Field head, allowing its value geometry to incorporate states encountered during online exploration. The Progress Field head and VLA policy are both optimized using AdamW with a micro-batch size of 32. For LIBERO, we use a learning rate of 2×10^{-5} and an effective batch size of 16,384. For RoboTwin2.0, we use a learning rate of 2×10^{-4} and an effective batch size of 1,024. For offline training, we use 30 expert demonstrations per task on LIBERO and 100 per task on RoboTwin2.0, with the latter collected using the official data collection pipeline. For a fair online comparison, PF-RL and our reproduced RLinf-GRPO baseline are initialized from the same OpenVLA-OFT checkpoint provided by Li et al. (2026) and trained on the same offline data for 30K steps, using the PF-RL offline objective and standard SFT, respectively. Online fine-tuning then starts from the resulting policies under the same training budget. Following Li et al. (2026), we evaluate each LIBERO task on 50 held-out test scenarios and each RoboTwin2.0 task on 100 held-out test scenarios. All simulation experiments are conducted on a cluster of 8 A800 GPUs.

B.2 Real-World Settings

For real-world experiments, we instantiate PF-RL with $\pi_{0.5}$ (Intelligence et al., 2025b) as the VLA backbone and deploy it on the Agilex Cobot Magic dual-arm platform. The platform is equipped with two Agilex PIPER robot arms and three RealSense D435 cameras providing one top-down view and two wrist views. We evaluate PF-RL on four manipulation tasks: *Fold Clothes*, *Stack all Plates*, *Put all Blocks into Bucket*, and *Fill Vase and Arrange Flowers*. We treat *Fold Clothes* as a single-task setting, while the latter three tasks are jointly trained under a multi-task setting. For offline training, we collect 200 expert demonstrations for *Fold Clothes* and 100 demonstrations per task for the multi-task setting. We first perform offline PF-RL using the advantage-weighted objective in Eq. 10 with an advantage temperature of $\beta = 1.5$, optimizing the policy for 20K steps with a batch size of 64. Compared with simulation, we use a larger β to reduce aggressive advantage reweighting and encourage more conservative policy updates in the more complex real-world setting. Starting from the resulting offline policy, we further perform real-world online PF-RL. To improve the safety and efficiency of real-world exploration, we employ human-in-the-loop intervention (Contributors, 2026) during online interaction. A human operator monitors policy execution and temporarily takes control when the robot enters an undesirable state, fails to make progress, or requires correction for recovery. Control is returned to the policy once the robot reaches a recoverable state. The intervention segments are retained as part of the collected trajectories and used for subsequent policy and Progress Field updates. Interventions are triggered as needed during policy execution. We conduct two RL iterations and collect 20 rollouts per task in each iteration, following the setting of $\pi^*_{0.6}$ (Intelligence et al., 2025a). The newly collected trajectories are then evaluated using the Progress Field advantage and used to update the policy and the Progress Field head. All other Progress Field architecture and training hyperparameters follow the simulation setting. All real-world training is conducted on a cluster of 8 A800 GPUs, while policy inference is performed on a single A800 GPU.

C Real-World Tasks and Visual Variations

Figure 7 provides an overview of the four real-world manipulation tasks and the visual variations used for evaluation. The tasks cover diverse long-horizon manipulation behaviors, including cloth folding, plate stacking, multi-block collection, and vase arrangement. For each task, we evaluate the policy in the Original setting and under three visual variations while keeping the task objective unchanged. Specifically, (-Obj) replaces the task-relevant objects with alternative instances, (-BG) changes the visual background of the workspace, and (-L) introduces substantial lighting variations.

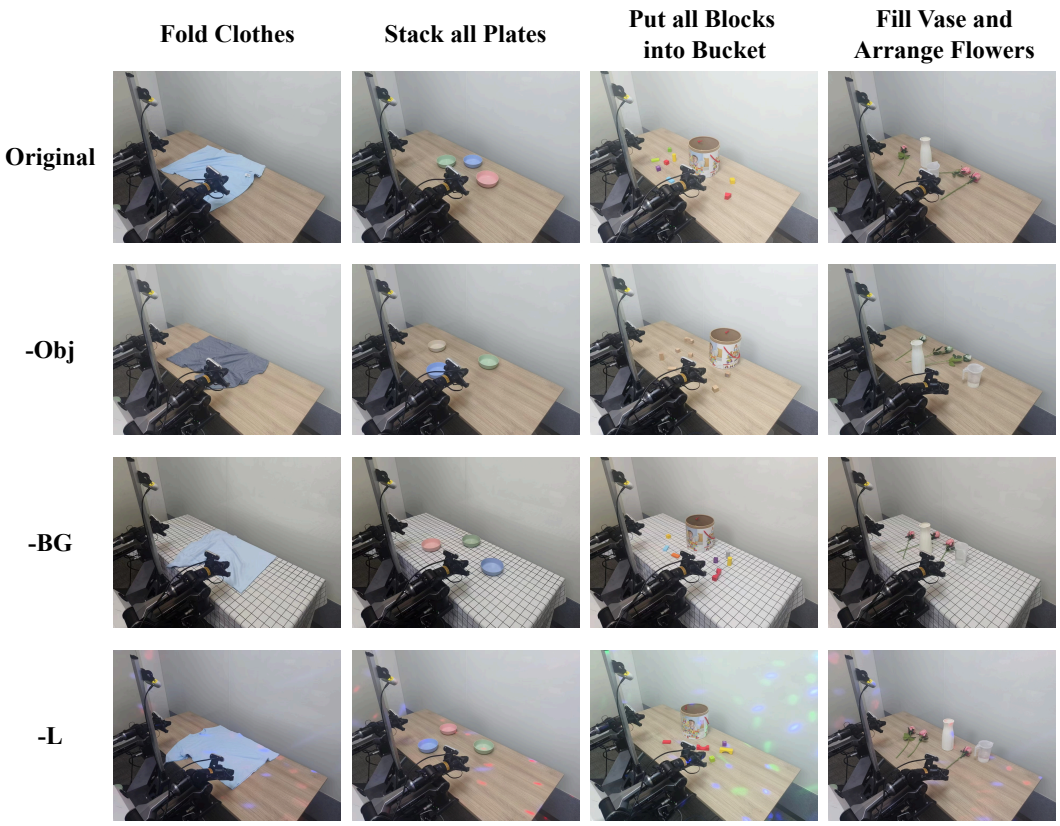


Figure 7: Overview of real-world tasks and visual variations. Columns show the four manipulation tasks: *Fold Clothes*, *Stack all Plates*, *Put all Blocks into Bucket*, and *Fill Vase and Arrange Flowers*. Rows correspond to the original environment settings and three visual variations involving object (-Obj), background (-BG), and lighting (-L) changes.

Since these visual variations are not observed during training, they provide a direct evaluation of the robustness of the VLA policy to out-of-distribution visual changes.

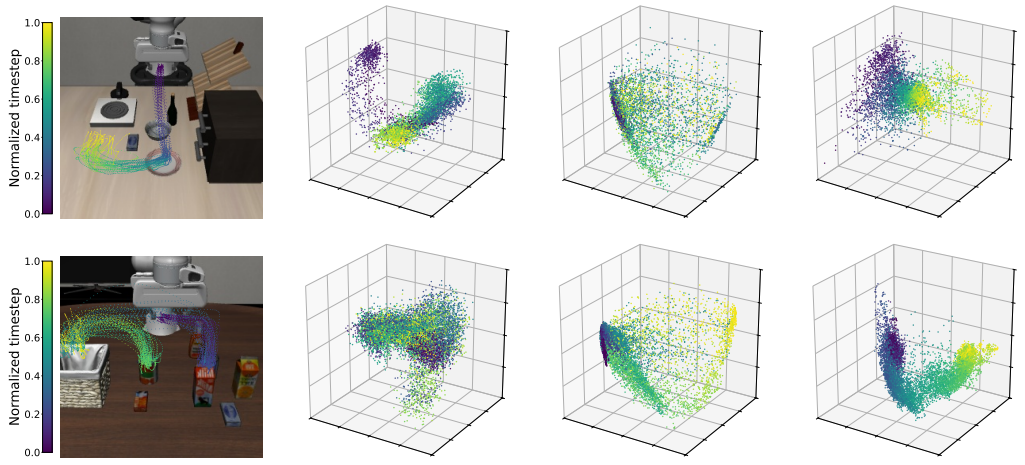
D Additional Analysis of Progress Modeling

D.1 Experimental Setup for Progress Modeling Comparison

For the direct scalar prediction baseline in Section 5.3, we replace the Progress Field with a lightweight prediction head that directly regresses a goal-conditioned scalar value from the current and goal VLA representations. The scalar baseline uses the Bellman-consistency and step-to-goal calibration objectives without goal contrastive learning, while the VLA backbone, policy optimization procedure, and training budget remain unchanged. Thus, the primary difference lies in whether progress is modeled through direct scalar prediction or goal-conditioned value geometry. For the held-out-task experiments in Figure 4, both methods start from the same pretrained SFT VLA policy from Li et al. (2026). We train the corresponding progress head on nine tasks from each LIBERO suite, while excluding the evaluated task from progress-model training. Since the held-out task is excluded from prior progress-model training, no task-specific goal reference is available beforehand. We therefore initialize it from successful rollouts collected by the policy at the start of online fine-tuning. For direct scalar prediction, we average the terminal VLA representations to obtain the goal input, whereas for Progress Field, we average the corresponding projected embeddings to construct the goal prototype. During online adaptation, the corresponding progress head continues to be updated following the standard PF-RL training procedure.

Table 4: Additional comparison with scalar prediction variants on LIBERO.

Variant	Value Estimation	$\mathcal{L}_{\text{con}}$	Spatial	Object	Goal	Long	Avg.
Direct scalar prediction	$\Phi_{\text{scalar}}(h_t, h_g)$	×	97.0	97.4	95.6	95.8	96.5
PF-RL w/o $\mathcal{L}_{\text{con}}$	$-\|z_t - z_g\|_2$	×	98.4	98.4	98.0	97.6	98.1
MLP readout variant	$\Phi_{\text{scalar}}(z_t, z_g)$	✓	98.8	98.2	94.4	95.2	96.7
PF-RL	$-\|z_t - z_g\|_2$	✓	**99.2**	**99.5**	**98.8**	**99.0**	**99.1**



(a) Task progression (b) Raw VLA space (c) Scalar prediction (d) Progress Field

Figure 8: Representation visualization on held-out tasks. The top row shows LIBERO-Goal Task 5, *push the plate to the front of the stove*, while the bottom row shows LIBERO-Long Task 0, *put both the alphabet soup and the tomato sauce in the basket*. (a) Expert end-effector trajectories, (b) Raw VLA representations, (c) Intermediate representations of the direct scalar prediction head, and (d) Projected embeddings learned by Progress Field.

D.2 Additional Comparison with Scalar Prediction Variants

To further evaluate our geometric value parameterization, we introduce an MLP readout variant that replaces the distance-based value computation with a scalar prediction head $\Phi_{\text{scalar}}(z_t, z_g)$ feeding on the projected embeddings z_t and z_g, while retaining the projection architecture and training objectives of Progress Field. The projection head and scalar readout are jointly trained, with the same goal contrastive loss applied to the goal Progress Field embeddings. Table 4 compares this variant with direct scalar prediction from VLA latent representation, PF-RL without $\mathcal{L}_{\text{con}}$, and full PF-RL. All of the variants share the same reinforcement fine-tuning pipeline, with differences limited to the value parameterization and the inclusion of goal contrastive learning. PF-RL outperforms both scalar prediction variants across all four suites. Even without $\mathcal{L}_{\text{con}}$, PF-RL outperforms direct scalar prediction. These results support the effectiveness of geometric value parameterization, with further gains from goal contrastive learning.

D.3 Representation Visualization on Held-out Tasks

To further evaluate the learned progress representations, Figure 8 visualizes representations along expert trajectories from the same held-out tasks, using the progress models before online adaptation. Although direct scalar prediction captures some progress-related structure, its representations remain relatively dispersed with substantial overlap across trajectory stages. In contrast, the Progress Field exhibits a clearer and more coherent progression from early to late stages, despite these tasks being excluded from progress-model training. This representation-level evidence further suggests that the learned goal-conditioned geometry captures progress structure that transfers more effectively across related tasks.

Table 5: Results of the Progress Field training objective ablation on LIBERO.

Method	Spatial	Object	Goal	Long	Avg.
PF-RL w/o $\mathcal{L}_{\text{td}}$	97.8	98.8	96.6	94.4	96.9
PF-RL w/o $\mathcal{L}_{\text{step}}$	97.4	98.4	96.0	95.2	96.8
PF-RL w/o $\mathcal{L}_{\text{con}}$	98.4	98.4	98.0	97.6	98.1
PF-RL w/o $\mathcal{L}_{\text{step}}$ and $\mathcal{L}_{\text{con}}$ (TD-only)	97.6	99.0	95.4	95.6	96.9
PF-RL	**99.2**	**99.5**	**98.8**	**99.0**	**99.1**

E Additional Ablation Results

E.1 Auxiliary Subgoal Training

Effect of Auxiliary Subgoal Training. Figure 9 compares PF-RL with vanilla GRPO and a variant trained without the auxiliary subgoal objective $\mathcal{L}_{\text{sub}}$ on LIBERO-Long. The results show that PF-RL outperforms both vanilla GRPO and the terminal-goal-only variant during training on LIBERO-Long. Even without $\mathcal{L}_{\text{sub}}$, PF-RL still improves over GRPO, demonstrating that goal-conditioned progress guidance alone provides an effective learning signal for long-horizon VLA policy optimization. However, this variant underperforms the full PF-RL method and exhibits more pronounced fluctuations throughout training. In contrast, introducing intermediate observations as auxiliary subgoals more fully exploits the intermediate structure of long-horizon rollout trajectories, thereby accelerating convergence and yielding a higher success rate under the same training budget. These results suggest that auxiliary subgoal training better exploits intermediate structure in long-horizon rollout trajectories, enabling further policy improvement beyond terminal-goal supervision alone.

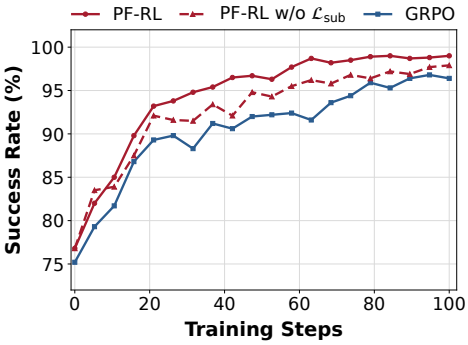


Figure 9: Auxiliary subgoal ablation.

E.2 Progress Field Training Objectives.

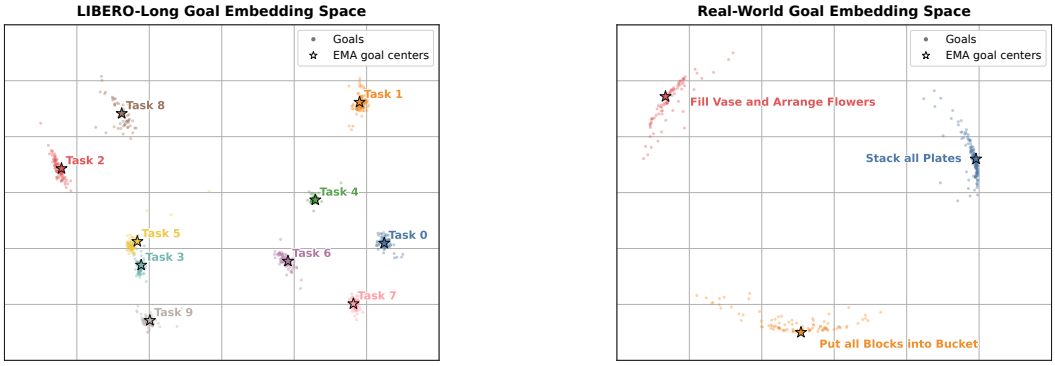
Table 5 summarizes the Progress Field objective ablations corresponding to Figure 5. The full objective performs best across all four LIBERO suites, with removing $\mathcal{L}_{\text{td}}$ causing the largest degradation on LIBERO-Long. The TD-only metric variant adopts HILP-style value learning (Park et al., 2024) within the same PF-RL pipeline, retaining both terminal-goal and auxiliary-subgoal TD losses. Its lower performance, particularly on LIBERO-Goal and LIBERO-Long, supports the additional benefit of step-to-goal calibration and goal contrastive learning beyond metric TD learning alone. These results support combining Bellman consistency, step-to-goal calibration, and goal contrastive learning for effective progress-aware policy optimization.

E.3 Analysis of Progress-Aware Advantage

We further analyze the Progress Field advantage in online policy optimization. Recall that PF-RL combines the trajectory-level group-relative advantage with the transition-level progress advantage as $A^{\text{sim}}_{i,t} = A^{\text{GRPO}}_i + \lambda_{\text{prog}} A^{\text{prog}}_{i,t}$. We evaluate PF-RL across a range of λ_{prog} values on all four LIBERO suites. As shown in Table 6, PF-RL consistently outperforms the original GRPO baseline across a broad range of λ_{prog}, validating the benefit of incorporating progress-aware credit into online policy optimization. Notably, $\lambda_{\text{prog}} = 1.0$ yields the strongest average performance across the four suites, suggesting that appropriately combining trajectory-level group-relative advantages with transition-level progress advantages benefits online policy optimization. We further remove A^{GRPO} and optimize VLA using only A^{prog}. Using A^{prog} alone remains competitive, while adding A^{GRPO} further improves performance through complementary trajectory-level feedback.

Table 6: Analysis of the progress advantage on LIBERO.

Setting	Spatial	Object	Goal	Long	Avg.
GRPO ($\lambda_{\text{prog}} = 0$)	97.6	98.7	95.2	96.8	97.1
PF-RL ($\lambda_{\text{prog}} = 0.1$)	98.8	98.8	98.0	97.2	98.2
PF-RL ($\lambda_{\text{prog}} = 0.5$)	98.6	**99.8**	97.6	98.2	98.6
PF-RL ($\lambda_{\text{prog}} = 1.0$)	**99.2**	**99.5**	**98.8**	99.0	**99.1**
PF-RL ($\lambda_{\text{prog}} = 2.0$)	99.0	99.4	98.0	**99.2**	98.9
PF-RL ($\lambda_{\text{prog}} = 5.0$)	98.8	99.2	96.2	97.2	97.9
PF-RL (A^{prog} only)	97.8	99.2	97.4	96.4	97.7



(a) LIBERO-Long

(b) Real-world tasks

Figure 10: PCA visualization of goal embeddings from successful trajectories, colored by task.

F Visualization of Goal Embedding Space

To further examine the structure learned in the goal embedding space, we visualize goal embeddings from all tasks within the LIBERO-Long suite and the real-world multi-task settings. Specifically, we collect the goal embeddings of successful trajectories for each task and project them into two dimensions using PCA, with different tasks shown in different colors. As shown in Figure 10, goal embeddings from the same task form compact clusters, while those from different tasks are clearly separated, indicating that the learned Progress Field preserves task-specific goal semantics. We further mark the corresponding per-task goal prototypes maintained in the goal embedding table with stars. These prototypes lie close to their respective task clusters, suggesting that the prototype provides a representative reference for evaluating task progress, particularly for failed trajectories where the terminal observation does not correspond to task completion.

G Analysis of Progress Advantages

We further examine the progress advantages induced by the learned Progress Field under different execution behaviors. Specifically, we manually annotate real-world rollout segments as *progress*, *regression*, or *stagnation*, according to whether the robot advances toward the goal, loses progress, or struggles to advance. For each of the four tasks, we collect 10 segments per category and average A_t^{prog} over action chunks within each segment. Figure 11 shows the highest median advantages for progress, followed by stagnation and regression, across all tasks. Progress segments may receive near-zero or negative advantages because the signal evaluates progress relative to the learned temporal scale. These results indicate that progress advantages faithfully reflect local task progress, supporting their use as reliable feedback for policy optimization.

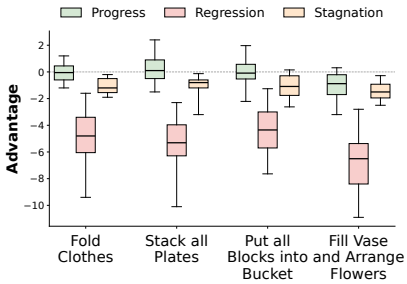


Figure 11: Progress advantage distributions across execution behaviors.



Figure 12: Successful rollout trajectories and corresponding goal-conditioned values.

H PROGRESS FIELD VISUALIZATIONS

Figure 12 and Figure 13 provide visualizations of the inferred trajectories as well as the corresponding goal-conditioned values induced by the Progress Field. Figure 12 shows successful executions across simulation tasks and four real-world tasks with different variations. Despite their different manipulation behaviors and execution horizons, the estimated values exhibit a consistent overall increase as the robot progresses toward task completion, approaching higher values as the goal is reached. Figure 13 further examines trajectories that encounter intermediate execution failures. In simulation, when the robot struggles to make progress toward the final goal, the estimated progress value plateaus or even declines over subsequent timesteps. Across different failure modes, including unsuccessful grasps and object slippage, the estimated value first decreases or plateaus, and increases again once the robot resumes progress toward the goal. This indicates that the learned value tracks task progress rather than merely elapsed execution time. These visualizations show that the Progress Field captures both successful task advancement and temporary execution degradation, providing meaningful transition-level signals for progress-aware policy optimization.

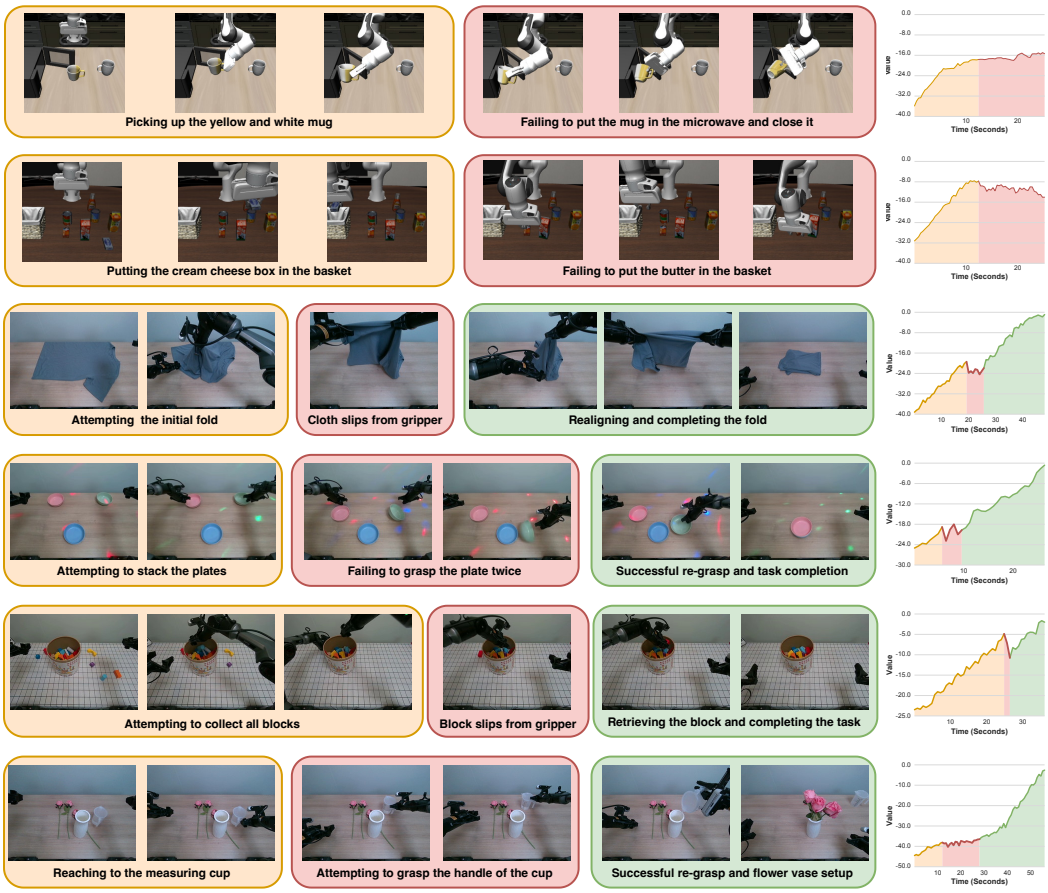


Figure 13: Rollout trajectories with intermediate failures and corresponding values.

I Computational Overhead

To ensure a fair comparison, we measure the overhead of PF-RL against the offline SFT and online GRPO baselines on a cluster of 8 A800 GPUs, 30K offline optimization steps following Li et al. (2026), and 100 online optimization epochs. As shown in Figure 14, offline training time increases from 8.3 to 11.3 hours, mainly due to additional encoding of goal observations, despite stopping Progress Field gradients at the VLA representations. In contrast, online fine-tuning increases only from 27.6 to 28.7 hours. A simple caching strategy largely enables this low overhead: we store the VLA latent representations during online rollout collection and reuse them for subsequent progress advantage computation, avoiding additional forward passes through the VLA backbone. The remaining online overhead mainly comes from updating the lightweight Progress Field head.

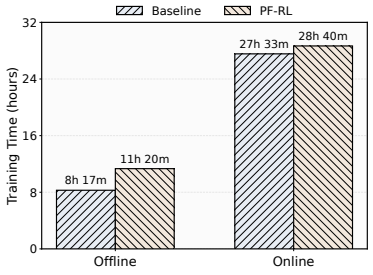


Figure 14: Training time of baselines and PF-RL.

J Limitations

PF-RL relies on successful trajectories in the offline dataset to construct initial task-level goal references. Although failed trajectories also contribute to Progress Field learning, an offline dataset containing only failed trajectories would not provide the successful terminal embeddings needed to establish this table. Extending PF-RL to such failure-only offline settings remains future work.