Vision-Language-Action Jump-Starting for Reinforcement Learning Robotic Agents

Author Names Omitted for Anonymous Review.

***Abstract*—Reinforcement learning (RL) enables high-frequency, closed-loop control for robotic manipulation, but scaling to long-horizon tasks with sparse or imperfect rewards remains difficult due to inefficient exploration and poor credit assignment. Vision-Language-Action (VLA) models leverage large-scale multimodal pretraining to provide generalist, task-level reasoning, but current limitations hinder their direct use in fast and precise manipulation. In this paper, we propose *Vision-Language-Action Jump-Starting (VLAJS)*, a method that bridges sparse VLA guidance with on-policy RL to improve exploration and learning efficiency. VLAJS treats VLAs as transient sources of high-level action suggestions that bias early exploration and improve credit assignment, while preserving the high-frequency, state-based control of RL. Our approach augments Proximal Policy Optimization (PPO) with a directional action-consistency regularization that softly aligns the RL agent's actions with VLA guidance during early training, without enforcing strict imitation, requiring demonstrations, or relying on continuous teacher queries. We evaluate VLAJS on six challenging manipulation tasks in simulation, and validate a subset on a real Franka Panda robot. VLAJS consistently outperforms PPO and distillation-style baselines in sample efficiency, reducing required environment interactions by over 50% in several tasks.**

I. Introduction

Reinforcement Learning (RL) provides a powerful framework for learning closed-loop control policies directly from interaction [40]. In robotics, RL enables high-frequency, state-based controllers that exploit rich proprioceptive and geometric feedback, enabling precise motor behaviors and online adaptation. These properties make RL particularly attractive for real-world manipulation, where robustness to disturbances, tight feedback loops, and reliability are critical [24, 47].

Despite these strengths, RL faces well-known challenges. Learning complex manipulation behaviors often requires long training times, careful reward engineering, and large amounts of interaction data [3, 21, 41]. These issues are especially pronounced in long-horizon or suboptimally rewarded tasks, where delayed rewards lead to weak credit assignment and slow policy improvement [31].

Recent advances in Large Language Models (LLMs) and Vision-Language Models (VLMs) offer a complementary direction [13, 9, 8, 2]. Vision-Language-Action (VLA) models unify perception, language understanding, and control by mapping multimodal observations directly to robot actions [5, 6]. Large-scale systems such as Octo [29], OpenVLA [17], π_0 [14], and RDT-1B [23] demonstrate strong semantic understanding and broad generalization, often exhibiting zero-shot and few-shot capabilities.

However, current VLA models are not designed to replace RL-based controllers for precise robotic manipulation. Their reliance on large expert datasets limits scalability and adaptation [30, 46, 16, 27], while transformer-based architectures typically operate at low control frequencies, restricting their ability to handle precision requirements, disturbances, and long-horizon closed-loop behaviors [26]. As a result, despite strong semantic priors, standalone VLA performance in real-world manipulation remains limited.

In this work, we view RL and VLA models as complementary components for training robotic manipulation policies. RL provides a deployable, high-frequency control backbone, while VLA models offer sparse, high-level action priors that encode semantic knowledge and task structure. Rather than contrasting these approaches, we ask:

Can the semantic knowledge captured by VLA models be used to accelerate RL, while preserving the precision, adaptability, and reliability of RL-based controllers?

Fig. 1 illustrates our perspective. We treat VLA models as sources of sparse, low-rate guidance that bias early exploration, improve credit assignment, and reduce the sim-to-real gap during training. The RL agent remains an on-policy, state-based learner operating at high control frequency, capable of exploiting precise feedback and ultimately surpassing the guiding policy.

To realize this synergy, we introduce *VLAJS*, a method for jump-starting on-policy RL using sparse guidance from VLA models. A high-frequency RL controller receives occasional VLA action suggestions during early training, incorporated through a directional action-consistency regularization within a PPO framework. This guidance is transient: it is queried infrequently, temporally propagated across control steps, and gradually annealed. The result is faster early learning without constraining long-term optimization or incurring excessive VLA inference cost.

We evaluate VLAJS on six challenging manipulation tasks in ManiSkill and validate a subset on a real Franka Panda robot. Focusing on long-horizon objectives and suboptimal reward design, our results demonstrate substantial gains in sample efficiency while producing policies that are directly deployable on real robotic systems.

Therefore, our contributions are:

1) We propose a method for accelerating high-frequency, state-based RL using sparse, low-rate guidance from VLA models, producing policies that are directly deployable on real robotic systems **(C1)**.

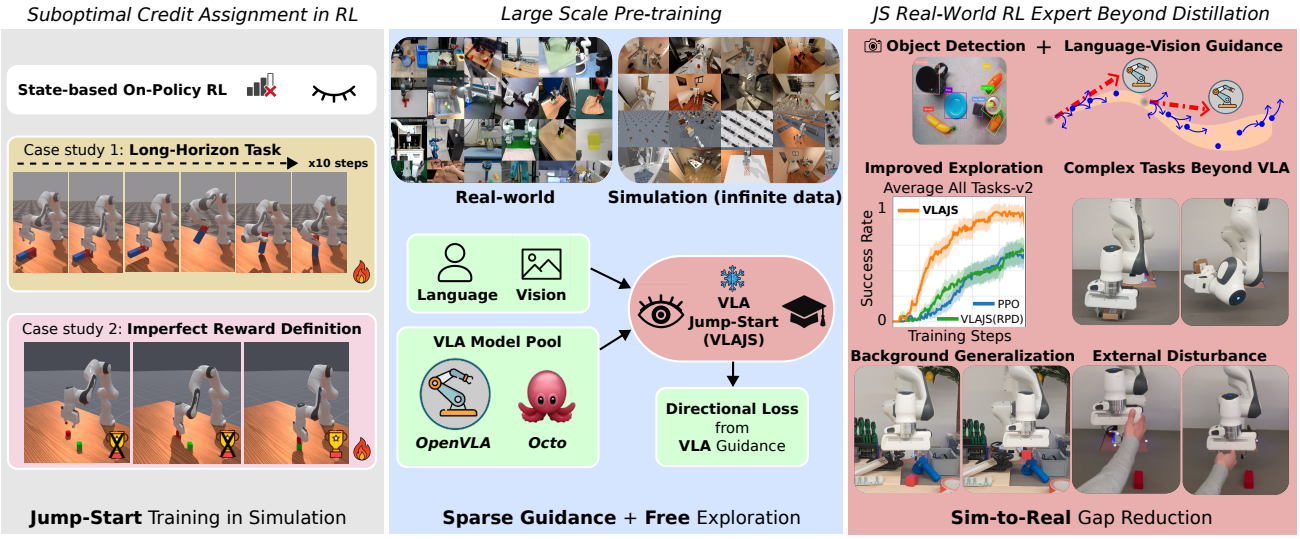


Fig. 1: **Overview of Vision-Language-Action Jump-Starting (VLAJS).** The figure illustrates the motivation, method, and outcomes of VLAJS. **Left:** We highlight suboptimal credit assignment in state-based, on-policy RL, focusing on: long-horizon tasks with extended action sequences and environments with imperfect reward design. **Center:** VLAJS leverages large-scale VLA pretraining from both real-world and simulation data. A pool of pretrained VLA models (*e.g.*, OpenVLA) provides sparse, low-frequency action suggestions conditioned on language and vision. These suggestions are incorporated during RL training through a directional action-consistency loss, enabling jump-start learning while preserving exploration. **Right:** The VLAJS-based trained RL agent surpasses distillation-based approaches, achieving improved exploration, robustness to background changes and external disturbances, and reduced sim-to-real gaps.

2) We introduce a directional action-consistency regularization that enables flexible, transient guidance without constraining asymptotic performance **(C2)**.
3) We demonstrate improved sample efficiency over PPO and distillation-based methods.
4) We introduce long-horizon and suboptimally rewarded ManiSkill environments for studying RL under difficult credit assignment, which we will publicly release.

II. Related Work

We review prior work along two axes most relevant to our approach: (i) reinforcement learning under suboptimal credit assignment and expert guidance, and (ii) Vision-Language-Action models for robotic control.

A. RL, Credit Assignment, and Expert Guidance

Reinforcement learning is a natural fit for complex manipulation because it supports high-frequency, closed-loop control from state feedback, but it can struggle in long-horizon and sparsely rewarded settings due to inefficient exploration and weak credit assignment [31, 40, 24, 21, 37].

A common strategy to mitigate poor credit assignment is to incorporate expert guidance [22, 32, 25, 15]. Fig. 2 provides an intuitive, trajectory-level view of exploration strategies. Persistent behavioral guidance [11, 36, 35] mixes a learned policy π_θ with an expert policy π_E,

$$\pi = (1-\beta)\pi_\theta + \beta\pi_E,$$

where $\beta > 0$ controls expert intervention. Jump-start RL [42, 19, 43, 12] instead applies *transient* behavioral guidance by delegating control to a guide policy π_g only early in training,

$$\pi_t = \begin{cases} \pi_g, & t \le h, \\ \pi_\theta, & t > h, \end{cases}$$

where h denotes the number of guide steps.

Complementarily, distillation-style approaches [15, 45, 39, 1, 10, 7] guide learning via an auxiliary action-matching loss,

$$\mathcal{L}_{\text{RL}} + \|a_\pi - a_{\text{teacher}}\|^2,$$

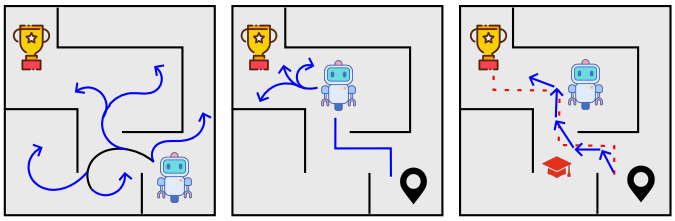
which can overly constrain optimization when applied persistently, and can be brittle when teacher supervision is sparse or imperfect.

These limitations motivate *transient auxiliary guidance*: use a teacher primarily early in training to bias exploration and improve credit assignment, then anneal and remove the auxiliary signal once the on-policy learner begins to improve reliably.

In addition to online guidance, several approaches incorporate prior data or expert policies to improve sample efficiency. Offline-to-online RL pretrains policies or value functions on logged datasets before fine-tuning, while imitation learning/behavior cloning provides initialization but can suffer from distribution shift [28, 27, 34].

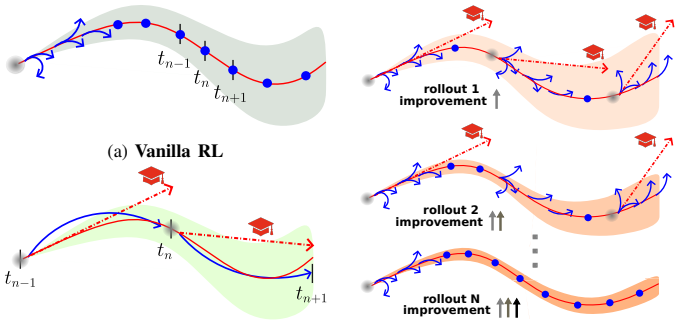
B. Vision-Language-Action Models

Vision-Language-Action models extend foundation-model paradigms to robotics by directly mapping multimodal observations and language instructions to actions [9, 30, 5, 6]. Recent VLAs demonstrate impressive generalization across manipulation tasks, but their practical deployment is constrained by inference latency, low control frequency, and



(a) **Vanilla RL** (b) **Behavioral guidance** (c) **Auxiliary guidance**

Fig. 2: **Guidance mechanisms for exploration in RL.** (a) Relies on random exploration. (b) Executes an imitation-learned policy for an initial phase (solid path). (c) Continuously biases learning via a teacher-provided signal (dashed red path) without directly executing actions.



(b) **Persistent auxiliary** (c) **Transient auxiliary (VLAJS)**

Fig. 3: **Auxiliary guidance during rollouts.** (a) The policy generates actions solely through on-policy exploration at a fixed control frequency, learning both direction and action scale incrementally from reward. (b) A teacher provides continuous action targets throughout the rollout, constraining both direction and magnitude and forcing the policy to match the teacher's action scale (distillation/RPD style). (c) Guidance is applied sparsely within a rollout and progressively annealed across rollouts, biasing action direction while allowing the policy to learn its own action magnitude and eventually explore freely.

reliance on demonstrations [46, 17, 18]. Techniques such as action chunking and parallel decoding improve throughput but do not fundamentally address the lack of tight closed-loop feedback [4, 14, 29, 23].

Few works explore combining VLAs with downstream RL fine-tuning [20] or distillation [45]. Refined Policy Distillation (RPD) [15] trains a PPO agent under continuous supervision from a VLA teacher using an action-matching loss. While effective in ideal scenarios, RPD assumes expensive teacher access at every timestep, relies primarily on visual inputs, and produces a standalone policy evaluated only under controlled training conditions, without real-world experiments.

In contrast, our approach positions the RL agent as a reusable, state-based control layer that is guided—but not dominated—by a VLA. By using sparse, transient auxiliary guidance rather than behavioral (Fig. 2b) or persistent (Fig. 3b) imitation, VLAJS bridges high-level foundation-model reasoning with efficient, high-frequency control, addressing both credit assignment and real-world execution constraints.

III. Methodology

In this paper, we propose **Vision-Language-Action Jump-Starting (VLAJS)**, an on-policy RL method that leverages a pretrained Vision–Language–Action model [17] as *sparse, transient auxiliary guidance*. VLAJS targets settings with *suboptimal credit assignment*—in particular (i) long-horizon tasks and (ii) imperfect reward design—where vanilla on-policy RL often fails to discover rewarding behaviors within practical interaction budgets. All methods build on Proximal Policy Optimization (PPO) with Generalized Advantage Estimation [37]. PPO is attractive for robotics because it supports stable learning of *high-frequency, closed-loop, state-based control* without demonstrations; however, it is notoriously inefficient when rewards are sparse/delayed or horizons are long [31].

A. Sparse VLA Queries and Temporal Discretization

We assume access to a pretrained VLA teacher that takes a visual observation and a language instruction and outputs a low-rate delta action a^{VLA} (translation, rotation, gripper). Querying the teacher at every environment step is impractical for long-horizon rollouts and parallel simulation. Therefore, we query the VLA *sparsely in time*: only a small number of calls per rollout. Each teacher delta is then *temporally discretized* into a short sequence of incremental deltas applied as guidance targets over the next D control steps (linear interpolation for translation and SLERP-style [38] interpolation for rotation). Outside these windows, the teacher target is treated as absent and masked out of the auxiliary loss.

This yields guidance that is sparse in two ways: (i) sparse teacher calls across time and (ii) sparse supervision within each rollout, enabling practical wall-clock training while still providing exploration bias early in learning (Fig. 3 explains this concept).

B. Reward-Based Jump-Starting *(C1)*

A key principle of VLAJS is that VLA guidance should be *transient*: the teacher is most useful before PPO has discovered a productive exploration regime. Once PPO begins to learn reliably, persistent guidance can become unnecessary (and even harmful if the teacher is suboptimal), and it remains computationally expensive. We therefore introduce a reward-trend–based jump-start mechanism that *reduces* and ultimately *deactivates* teacher usage (see Fig. 3c).

a) Adaptive query rate: At each PPO iteration, we compute a reward-improvement statistic from a rolling history of mean rollout rewards. As improvement increases, we reduce the number of teacher calls per rollout by an exponential schedule:

$$N_{\mathrm{calls}} \leftarrow \max(N_{\min}, \lfloor N_{\max} \exp(-\kappa \cdot \Delta\bar{r}) \rfloor),$$

where $\Delta\bar{r}$ denotes a reward-gain signal computed from recent rollouts (clipped at zero), κ controls decay, and $N_{\max}/N_{\min}$ bound the calls per rollout. This retains more guidance when PPO is stuck, and quickly sparsifies guidance once learning accelerates.

b) Permanent deactivation: We additionally detect *monotonic reward improvement* over a short window of recent iterations and permanently deactivate guidance once the mean rollout reward exceeds a small threshold of 3. We chose 3 as the smallest value that reliably marks the onset of meaningful

learning: across runs reward often hovers around ≈ 2 when the policy is still stuck, while once it goes above 3 learning proceeds reliably, allowing guidance to be turned off as early as possible.

C. Directional Action-Consistency Loss (C2)

Using a VLA teacher as intermittent guidance differs from classical distillation [15]: the teacher is queried sparsely and can be *suboptimal* for precise, high-frequency control. In this setting, directly matching teacher actions (*e.g.*, through distillation losses) can be too strong and can inject inconsistent gradients when supervision appears intermittently. Instead, we treat teacher outputs as *directional hints*.

Let $\mu_\theta(s_t)$ denote the policy mean action and $\tilde{a}_t^{\mathrm{VLA}}$ the discretized teacher target at time t (only present during discretization windows). We split actions into translation and rotation components and define a cosine misalignment loss

$$\ell_{\mathrm{dir}}(x, y) = 1 - \frac{\langle x, y \rangle}{\|x\|\|y\| + \varepsilon}.$$

Our auxiliary objective is

$$\mathcal{L}_{\mathrm{dir}} = \mathbb{E}_t \left[\mathbf{1}[\mathrm{valid}_t] \sum_{c \in \{\mathrm{pos,rot}\}} \ell_{\mathrm{dir}}\big(\mu_\theta^c(s_t), \tilde{a}_t^c\big) \right],$$

where $\mathbf{1}[\mathrm{valid}_t]$ masks timesteps without guidance and we skip components with near-zero teacher vectors to avoid unstable normalization. We do not constrain the gripper dimension in the auxiliary loss.

Rationale for direction-only: Cosine alignment preserves the *direction* suggested by the teacher while allowing PPO to choose action magnitudes and fine corrections. This is particularly important when (i) teacher actions are discretized across multiple steps, (ii) teacher scale may not match the student's control frequency, and (iii) the teacher is imperfect and should not be copied exactly.

D. Training Objective

We augment PPO updates with the auxiliary guidance loss:

$$\mathcal{L}(\theta) = \mathcal{L}_{\mathrm{PPO}}(\theta) + \lambda_t \mathcal{L}_{\mathrm{aux}}(\theta),$$

where $\mathcal{L}_{\mathrm{aux}} = \mathcal{L}_{\mathrm{dir}}$ for VLAJS. The coefficient λ_t follows the same reward-trend schedule used for guidance sparsification and is set to zero after deactivation.

E. Baselines Implemented in Our Code

We implement three algorithms:

a) PPO: Standard PPO [37] trained from scratch using only environment reward.

b) Sparse RPD: For long-horizon experiments, we additionally evaluate a *persistent* sparse-guidance baseline that queries the teacher sparsely *throughout training* (no deactivation), reflecting "sparse distillation" as a computationally feasible alternative to full RPD [15] when horizons are long.

c) VLAJS (RPD): An ablation that keeps *exactly the same* sparse query mechanism and jump-start deactivation, but replaces directional guidance with an RPD-style MSE action-matching loss on guided steps:

$$\mathcal{L}_{\mathrm{MSE}} = \mathbb{E}_t \big[\mathbf{1}[\mathrm{valid}_t] \ \|\mu_\theta(s_t) - \tilde{a}_t^{\mathrm{VLA}}\|_2^2 \big].$$

This isolates the effect of the directional loss under sparse, transient teacher usage.

IV. Experimental Evaluation

We evaluate our approach under two complementary sources of suboptimal credit assignment:

a) Use Case 1 - Long-horizon task: We test whether *sparse VLA guidance* is a practical and effective substitute for dense distillation when episodes become very long (*e.g.*, high-frequency control or extended horizons). Here, the goal is primarily to quantify exploration benefits and computational feasibility. We compare **Sparse RPD**—the building block of VLAJS (RPD)—to PPO.

b) Use Case 2 - Suboptimal reward design: We test whether *jump-starting with transient guidance* and a *directional loss* improves learning when rewards are sparse or imperfectly shaped, reflecting realistic reward design constraints. Here we compare **PPO**, **VLAJS (RPD)**, and **VLAJS**.

V. Training Setup

A. Simulation Environments, Observations, and Actions

All simulations are conducted in ManiSkill manipulation environments [41, 44] (see Fig. 4). The RL policy is a state-based controller: observations include robot proprioception and privileged simulator state (*e.g.*, object poses). Actions are continuous delta end-effector controls (translation and rotation) with a gripper command, executed at a high control frequency. This setting reflects the regime where RL excels at precise closed-loop control but struggles with exploration and long-horizon credit assignment.

B. Teacher Models and Sparse Querying

We use a pretrained VLA—*OpenVLA* with average s.r. of 40%—as an external teacher that maps RGB observations and a language instruction to a delta action. Due to inference cost, the teacher is queried only a few times per rollout (max. 20% of the total). Each teacher delta is discretized into a short sequence of incremental deltas over D steps, producing sparse guidance targets within the rollout and zero targets elsewhere. Teacher actions are never executed directly in the environment; they are used only in auxiliary losses during training.

C. Use Case 1 - Long-Horizon Protocol

To isolate the impact of long horizons, we take standard ManiSkill tasks and increase the effective horizon length by $10\times$. This models realistic scenarios where policies operate at higher frequencies or where tasks require extended action sequences, amplifying the difficulty of exploration and reward propagation.

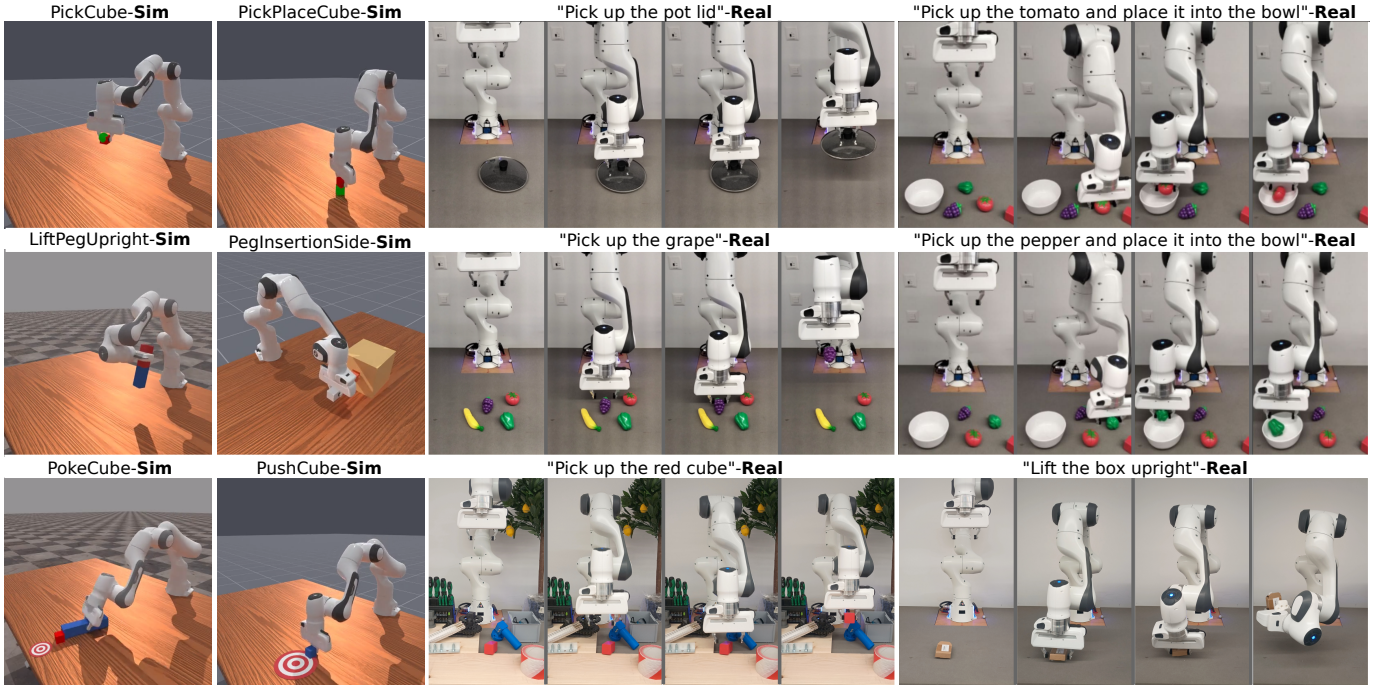


Fig. 4: Simulation and real-world manipulation tasks used in our evaluation. Left: six ManiSkill simulation tasks (PickCube, PickPlaceCube, LiftPegUpright, PegInsertionSide, PokeCube, PushCube). Right: zero-shot real-world deployment on a Franka Panda robot across diverse language-specified tasks.

We focus on the feasibility and benefit of *persistent sparse guidance* and therefore compare: (i) PPO and (ii) Sparse RPD variants (teacher queried sparsely throughout training).

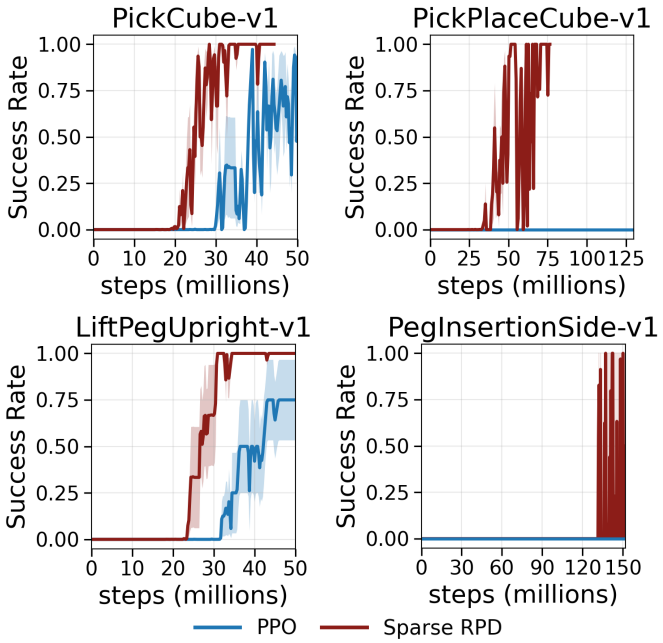


Fig. 5: Learning curves for long-horizon tasks. Sparse RPD makes distillation feasible in long-horizon tasks significantly accelerating convergence compared to PPO baselines. Standard RPD is omitted due to prohibitive training time.

D. Use Case 2 - Suboptimal Reward Design Protocol

To model realistic reward engineering constraints, we modify ManiSkill reward functions into simplified, more intuitive variants that provide weaker shaping (*e.g.*, sparse success signals such as rewarding only object pickup rather than dense shaping). This induces suboptimal credit assignment even for tasks that are otherwise solvable with dense rewards.

In this use case, we evaluate whether *transient jump-start guidance* and *directional regularization* improve sample efficiency. We compare: (i) PPO, (ii) VLAJS (RPD), and (iii) VLAJS (ours).

VI. Results

We evaluate our approach under two complementary sources of suboptimal credit assignment: **long-horizon tasks** and **imperfect reward design**. These settings isolate different failure modes of on-policy RL and motivate different comparisons.

A. Use Case 1 - Long-Horizon Task

We first study environments with extended episode horizons, which amplify delayed reward propagation and make exploration particularly challenging for PPO. The objective of this experiment is to assess whether *sparse* VLA guidance is computationally feasible and beneficial when dense teacher supervision is impractical (RPD [15]).

Across all tasks, Sparse RPD consistently outperforms PPO, often by a large margin in sample efficiency (see Fig. 5).

These results demonstrate that even very sparse auxiliary guidance from a VLA provides strong exploration benefits in long-horizon regimes, while remaining computationally tractable. This establishes sparse guidance as a viable building block, but does not yet address whether the agent can learn beyond the teacher or whether persistent supervision is desirable.

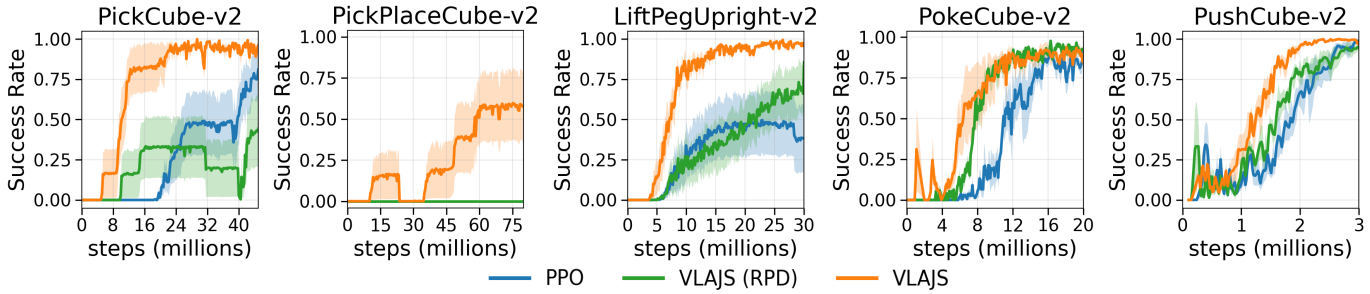


Fig. 6: Learning curves and sample-efficiency comparison for suboptimal reward tasks. VLAJS consistently outperforms PPO and distillation-based baselines—VLAJS (RPD). The initial increase in return is driven by the VLA-based jump-start. Once the VLA guidance is deactivated, the agent transitions to a purely learned policy, which results in a temporary reduction in return while maintaining or improving task success (see the additional material for return plots).

B. Use Case 2 - Suboptimal Reward Design

We next consider environments with deliberately simplified and sparse reward definitions, designed to reflect realistic reward engineering constraints. In these settings, PPO often fails to learn meaningful behaviors within practical interaction budgets, despite moderate episode horizons.

Fig. 6 compares PPO, VLAJS (RPD), and VLAJS. Unlike the long-horizon case with sparse persistent guidance, imitation-based loss is no longer sufficient when jump-starting: VLAJS (RPD) provides limited or inconsistent improvements over PPO, and in several tasks fails to meaningfully accelerate learning.

In contrast, VLAJS consistently achieves higher success rates (results in Fig. 6) across all tasks, including an out-of-distribution (OOD) task that the VLA was not trained on (*PushCube-v2*). By combining sparse guidance with reward-aware deactivation and a directional action-consistency loss, VLAJS effectively jump-starts learning while preserving the ability to optimize beyond the teacher. Notably, performance often continues to improve after VLA guidance is fully deactivated, indicating that the learned policy is not constrained by the teacher.

C. Zero-shot Real-World Deployment

Real-world results for randomly sampled objects from a fixed subset of tasks are summarized in Table I, demonstrating *zero-shot* deployment of the learned policies. Figure 4 shows performance under randomized visual conditions, highlighting robustness enabled by the state-based policy and a visually robust detector. In contrast to a VLA baseline, which fails under strong visual perturbations, our policies remain stable and successfully complete the task. Real-world state estimates are obtained using a pretrained YOLO detector [33].

Policy	Lift Cube	Pick & Place	Peg Reorientation
OpenVLA	47%	40%	–
VLAJS (zero-shot)	70%	80%	20%

TABLE I: Real robot deployment success rates (20 trials per task).

Our results show that *when* and *how* teacher signals are used matters as much as the teacher itself, revealing complementary roles of VLA guidance in on-policy RL and enabling zero-shot real-world deployment on a real robot (Tab. I). In long-horizon tasks, where delayed reward propagation hinders exploration, sparse but persistent auxiliary guidance provides a practical and deployable alternative to dense distillation, yielding substantial gains over PPO (Fig. 5). We then consider a harder setting with suboptimal reward definitions to directly test whether auxiliary guidance can be made *transient*, reducing reliance on teacher queries and overall computation. In this regime, we find that while persistent guidance may still be effective, distillation-style action matching–VLAJS (RPD)–is no longer suitable for jump-starting learning, motivating the use of a *weak, directional* consistency loss that bootstraps task-relevant exploration without over-constraining the policy. As a result, policies trained with VLAJS continue to improve after guidance is fully deactivated (Fig. 6).

VII. Conclusion and Limitations

We presented **VLAJS**, which improves the sample efficiency of on-policy reinforcement learning for robotic manipulation by leveraging pretrained VLA models as *sparse, transient auxiliary guidance*. VLAJS combines (i) a **reward-based jump-start schedule** that reduces and permanently deactivates teacher usage, and (ii) a **directional action-consistency loss** that interprets VLA outputs as coarse directional hints rather than strict action targets. Across challenging environments, our approach accelerates learning and improves final performance relative to PPO and RPD-based sparse distillation baselines. To the best of our knowledge, we are the first to deploy *jump-started policies guided by a VLA* on a real robotic system, demonstrating zero-shot transfer to a Franka Panda robot.

While VLAJS improves sample efficiency in difficult credit-assignment regimes, it still relies on a VLA teacher that provides at least minimally reliable directional cues. In practice, current VLAs often require environment-specific fine-tuning to be useful, and obtaining such adaptation can be expensive. Using a large VLA during training also introduces nontrivial wall-clock overhead and systems complexity, including inference latency.

Future work will focus on reducing teacher overhead by querying VLA guidance only when needed, extending the approach to vision-based RL for more complex manipulation and navigation tasks, and exploring direct real-world fine-tuning of RL policies with the VLA model.

References

[1] Rishabh Agarwal, Max Schwarzer, Pablo Samuel Castro, Aaron C Courville, and Marc Bellemare. Reincarnating reinforcement learning: Reusing prior computation to accelerate progress. *Advances in neural information processing systems*, 35:28955–28971, 2022.

[2] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as I can, not as I say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

[3] Marcin Andrychowicz, Anton Raichuk, Piotr Stańczyk, Manu Orsini, Sertan Girgin, Raphaël Marinier, Léonard Hussenot, Matthieu Geist, Olivier Pietquin, Marcin Michalski, et al. What matters in on-policy reinforcement learning? a large-scale empirical study. In *ICLR 2021-Ninth International Conference on Learning Representations*, 2021.

[4] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, and Niccolo Fusai et al. $\pi 0$: A vision-language-action flow model for general robot control. *Robotics: Science and Systems XXI*, 2025.

[5] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. RT-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[6] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[7] Wojciech M Czarnecki, Razvan Pascanu, Simon Osindero, Siddhant Jayakumar, Grzegorz Swirszcz, and Max Jaderberg. Distilling policy distillation. In *The 22nd international conference on artificial intelligence and statistics*, pages 1331–1340. PMLR, 2019.

[8] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. PaLM-E: An embodied multimodal language model. *arXiv preprint arXiv:2303.03378*, 2023.

[9] Roya Firoozi, Johnathan Tucker, Stephen Tian, Anirudha Majumdar, Jiankai Sun, Weiyu Liu, Yuke Zhu, Shuran Song, Ashish Kapoor, Karol Hausman, Brian Ichter, Danny Driess, Jiajun Wu, Cewu Lu, and Mac Schwager. Foundation models in robotics: Applications, challenges, and the future. *arXiv preprint arXiv:2312.07843*, 2023.

[10] Sam Green, Craig M Vineyard, and Cetin Kaya Koç. Distillation strategies for proximal policy optimization. *arXiv preprint arXiv:1901.08128*, 2019.

[11] Shane Griffith, Kaushik Subramanian, Jonathan Scholz, Charles L Isbell, and Andrea L Thomaz. Policy shaping: Integrating human feedback with reinforcement learning. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

[12] Todd Hester, Matej Vecerik, Olivier Pietquin, Marc Lanctot, Tom Schaul, Bilal Piot, Dan Horgan, John Quan, Andrew Sendonaris, Ian Osband, et al. Deep q-learning from demonstrations. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[13] Yafei Hu et al. Toward general-purpose robots via foundation models: A survey & meta-analysis. *CoRR*, 2023.

[14] 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*, 2025.

[15] Tobias Jülg, Wolfram Burgard, and Florian Walter. Refined Policy Distillation: From VLA generalists to RL experts. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, 2025. Accepted for publication.

[16] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, et al. DROID: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

[17] Moo Jin Kim, Karl Pertsch, et al. OpenVLA: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[18] Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *Robotics: Science and Systems XXI*, 2025.

[19] Jens Kober and Jan Peters. Imitation and reinforcement learning. *IEEE Robotics & Automation Magazine*, 17(2): 55–62, 2010.

[20] Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaohui Yang, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, et al. Simplevla-rl: Scaling vla training via reinforcement learning. *arXiv preprint arXiv:2509.09674*, 2025.

[21] Jacky Liang, Viktor Makoviychuk, Ankur Handa, Nuttapong Chentanez, Miles Macklin, and Dieter Fox. Gpu-accelerated robotic simulation for distributed reinforcement learning. In *Conference on Robot Learning*, pages 270–282. PMLR, 2018.

[22] Gabriele Libardi, Gianni De Fabritiis, and Sebastian Dittert. Guided exploration with proximal policy optimization using a single demonstration. In *International Conference on Machine Learning*, pages 6611–6620. PMLR, 2021.

[23] 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.

[24] 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.

[25] Yecheng Jason Ma, William Liang, Guanzhi Wang, De-An Huang, Osbert Bastani, Dinesh Jayaraman, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Eureka: Human-level reward design via coding large language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[26] Angelo Moroncelli, Vishal Soni, Marco Forgione, Dario Piga, Blerina Spahiu, and Loris Roveda. The duality of generative ai and reinforcement learning in robotics: A review. *Inf. Fusion*, 129:104003, 2024.

[27] Suraj Nair, Eric Mitchell, Kevin Chen, Silvio Savarese, Chelsea Finn, et al. Learning language-conditioned robot behavior from offline data and crowd-sourced annotation. In *Conference on Robot Learning*, pages 1303–1315. PMLR, 2022.

[28] Mitsuhiko Nakamoto, Simon Zhai, Anikait Singh, Max Sobol Mark, Yi Ma, Chelsea Finn, Aviral Kumar, and Sergey Levine. Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning. *Advances in Neural Information Processing Systems*, 36:62244–62269, 2023.

[29] Octo Model Team et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

[30] Open X-Embodiment Collaboration, Abigail O'Neill, Amir Rehman, Agrim Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Ajay Mandlekar, Arhan Jain, et al. Open x-embodiment: Robotic learning datasets and RT-X models. *arXiv preprint arXiv:2310.08864*, 2023.

[31] Eduardo Pignatelli, Johan Ferret, Matthieu Geist, Thomas Mesnard, Hado van Hasselt, and Laura Toni. A survey of temporal credit assignment in deep reinforcement learning. *Transactions on Machine Learning Research*, 2024.

[32] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *Robotics: Science and Systems XIV*, 2018.

[33] J Redmon. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[34] Scott Reed et al. A generalist agent. *TMLR*, 2022.

[35] Stephane Ross and J Andrew Bagnell. Reinforcement and imitation learning via interactive no-regret learning. *arXiv preprint arXiv:1406.5979*, 2014.

[36] Stéphane 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*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

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

[38] Ken Shoemake. Animating rotation with quaternion curves. *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*, 1985.

[39] Giacomo Spigler. Proximal policy distillation. *arXiv preprint arXiv:2407.15134*, 2024.

[40] R.S. Sutton and A.G. Barto. Reinforcement learning: An introduction. *IEEE TNN*, 9(5):1054–1054, 1998.

[41] Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-Kai Chan, et al. Maniskill3: Gpu parallelized robot simulation and rendering for generalizable embodied ai. In *7th Robot Learning Workshop: Towards Robots with Human-Level Abilities*, 2025.

[42] Ikechukwu Uchendu, Ted Xiao, Yao Lu, Banghua Zhu, Mengyuan Yan, Joséphine Simon, Matthew Bennice, Chuyuan Fu, Cong Ma, Jiantao Jiao, et al. Jump-start reinforcement learning. In *International Conference on Machine Learning*, pages 34556–34583. PMLR, 2023.

[43] Mel Vecerik, Todd Hester, Jonathan Scholz, Fumin Wang, Olivier Pietquin, Bilal Piot, Nicolas Heess, Thomas Rothörl, Thomas Lampe, and Martin Riedmiller. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. *arXiv preprint arXiv:1707.08817*, 2017.

[44] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11097–11107, 2020.

[45] Charles Xu, Qiyang Li, Jianlan Luo, and Sergey Levine. Rldg: Robotic generalist policy distillation via reinforcement learning. *Robotics: Science and Systems XXI*, 2025.

[46] Tony Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *Robotics: Science and Systems XIX*, 2023.

[47] Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *2020 IEEE symposium series on computational intelligence (SSCI)*, pages 737–744. IEEE, 2020.