A Vision-Language-Action-Critic Model for Robotic Real-World Reinforcement Learning

Shaopeng Zhai[1*], Qi Zhang[1*], Tianyi Zhang[1*], Fuxian Huang[1*], Haoran Zhang[1*], Ming Zhou[1*] and Jiangmiao Pang[1†]

[1]Shanghai AI Lab

Robotic real-world reinforcement learning (RL) with vision-language-action (VLA) models is bottlenecked by sparse, handcrafted rewards and inefficient exploration. We introduce VLAC, a general process reward model built upon InternVL and trained on large scale heterogeneous datasets. Given pairwise observations and a language goal, it outputs dense progress delta and done signal, eliminating task-specific reward engineering, and supports one-shot in-context transfer to unseen tasks and environments. VLAC is trained on vision–language datasets to strengthen perception, dialogic and reasoning capabilities, together with robot and human trajectories data that ground action generation and progress estimation, and additionally strengthened to reject irrelevant prompts as well as detect regression or stagnation by constructing large numbers of negative and semantically mismatched samples. With prompt control, a single VLAC model alternately generating reward and action tokens, unifying critic and policy. Deployed inside an asynchronous real-world RL loop, we layer a graded human-in-the-loop protocol (offline demonstration replay, return and explore, human guided explore) that accelerates exploration and stabilizes early learning. Across four distinct real-world manipulation tasks, VLAC lifts success rates from about 30% to about 90% within 200 real-world interaction episodes; incorporating human-in-the-loop interventions yields a further 50% improvement in sample efficiency and achieves up to 100% final success.

Code:VLAC | Model:VLAC | Homepage & Interactive-demo

"*Intelligence is determined by the dynamics of interaction with the world.*"
— Rodney A. Brooks, 1991

1. Introduction

With the rapid development of Vision-Language-Action (VLA) models, the intelligence of robotic perception and manipulation capabilities has greatly improved, leading to impressive performance in autonomously completing general tasks. Current VLA models are primarily trained through imitation learning, which requires vast amounts of data and demands substantial human and material resources for large-scale data collection Bjorck et al. (2025); Deng et al. (2025); Lin et al. (2024); Team et al. (2025). However, collecting human expert trajectories is not only costly and time-consuming, but most data collection efforts focus on laboratory-customized scenes and tasks, with inconsistent quality. Consequently, significant barriers remain for robots to perform effectively in real-world scenarios, particularly concerning data diversity, cross-scene generalization, and robustness.

To address this limitation, which is often insufficient to cover the dynamic and continuously changing tasks and environments encountered in the real world, and to avoid the challenges associated with sim-to-real transfer, many studies have explored RL conducted directly in the real world. In this paradigm, robots autonomously learn from both successful and failed trajectories generated through real-world interactions, enables autonomous exploration and continuous improvement of success rates from real-world feedback. A straight forward way for improving real-world RL efficiency is to

* *equal contributions,* † *corresponding to (pangjiangmiao@pjlab.org.cn). Contributions and emails of all authors in Appendix E.*
© 2025 Shanghai Artificial Intelligence Laboratory. All rights reserved.

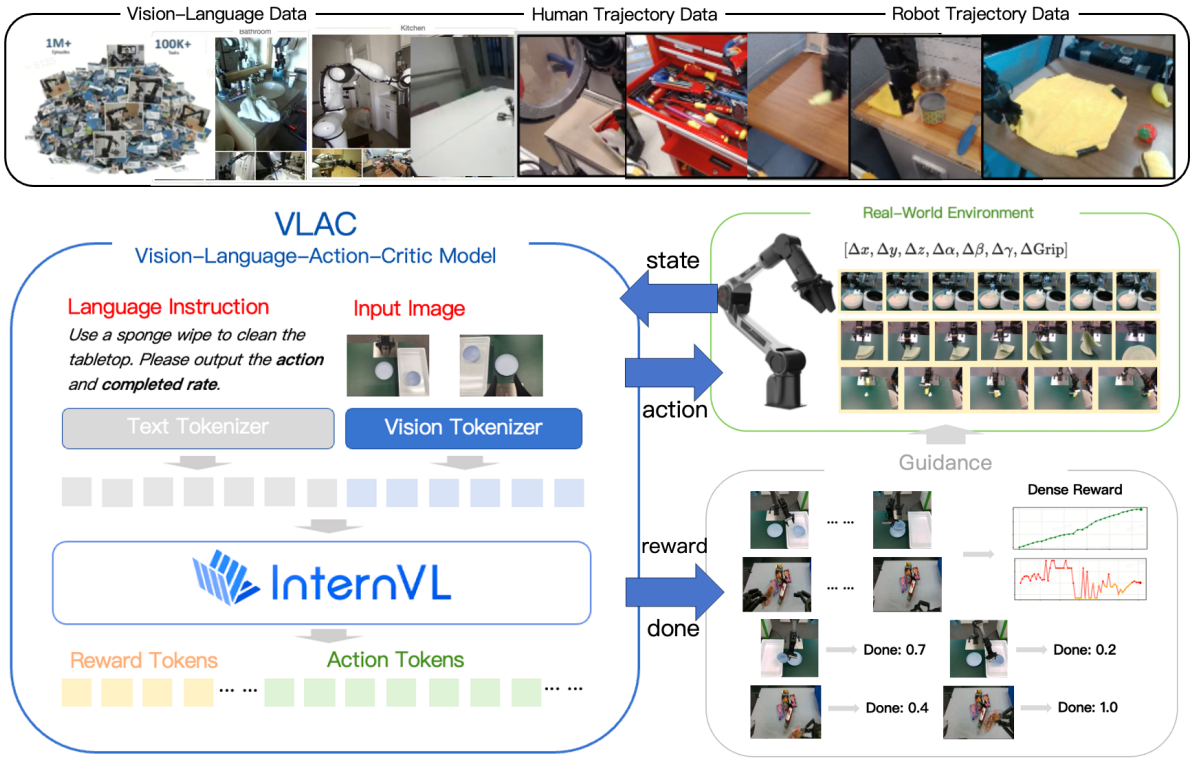


Figure 1. **Overview**. Pretrained on multi-source data, VLAC provides dense progress rewards for real-world RL while also serving as a policy to output actions, integrating into real-world RL loops to enable self-improvement in manipulation.

provide dense progress reward. However, real-world RL reward design still encounters persistent obstacles. Many methods rely on non-general, task-specific shaping engineered separately for each scenario Herzog et al. (2023); Kumar et al. (2024); Mendonca et al. (2023b, 2024); Xu et al. (2022). Frameworks advertised as reusable often require additional, task-dependent data collection to train reward surrogates or termination (done) classifiers Hu et al. (2023a); Luo et al. (2024c). The resulting rewards typically appear only when the task is nearly complete, leaving most intermediate progress unscored. Although some approaches introduce denser or "universal" progress signals, their ability to generalize across novel tasks, objects, or goal language remains limited Ma et al. (2022, 2023). Thus, intermediate feedback is still unreliable and weakly transferable, impeding sample efficiency. We target this gap by producing dense progress reward with stronger cross-task generalization.

To provide reliable and generalizable reward signal, we propose a Vision-Language-Action-Critic model, VLAC, which builds upon InternVL, a state-of-the-art multi-modal model, and unifies the roles of "actor" and "critic" within a single autoregressive architecture, capable of zero-shot and in-context task progress prediction and action generation. VLAC takes as input a pair of image observations together with a task description and outputs a signed progress delta indicating how much the second state advances (or regresses) the task relative to the first, as the reward. It is enabled by training on more than 4,000 hours of language annotated manipulation data where temporal ordering yields the progress labels, and we construct additional data to strengthen negative reward assignment and improve in context learning capability. Auxiliary perception tasks such as detection, lightweight segmentation, grounding, and coarse 3D or contact cues enhance the base representation. Experiments show that VLAC separates positive and negative progress with sufficient fidelity to yield

a reliable reward signal. And we find that enhancing the critic component leads to a improvement in downstream action generation.

Building upon the VLAC model, we developed a real-world RL framework in which the VLAC model can interact directly with the real world and self-improve. Through adaptive interaction, the VLAC collects both successful and failed trajectories, receives reward signals, and determines task completion. We evaluated our framework on four diverse real-world manipulation tasks, where the "actor" demonstrated the ability to autonomously improve its success rate from approximately 30% to 90% within 200 episodes. During the experiments, we observed that the VLA's prior capabilities play a critical role in enabling efficient exploration. To further enhance exploration efficiency, we introduced a human-in-the-loop mechanism. This mechanism allows humans to assist the model in refining the manipulation process, facilitating more effective learning. Human intervention is more of an art than a science, and we applied three levels of human-in-the-loop intervention across all tasks, resulting in an approximate 50% improvement in exploration efficiency and achieves up to 100% success rate.

2. Related Work

Real-world RL has advanced rapidly across multiple robotic subfields, including locomotionSmith et al. (2023, 2024) and dexterous manipulationHu et al. (2023b). A substantial portion of "online" approaches, however, function essentially as self-imitationKumar et al. (2024) or hindsight relabelingZhou et al. (2024a): they reinforce previously successful behaviors without exploiting failed (negative) attempts, which limits overall data efficiency in real-world settings. Focusing on manipulation, one prominent line of work applies real-world RL to small (non-VLA) policy models. Because such models lack strong prior ability, researchers commonly adopt human-in-the-loop proceduresKang et al. (2025); Li et al. (2025); Luo et al. (2024b); Zhou et al. (2024b): a small set of expert demonstrations is first collected, then offline-to-online reinforcement learning repeatedly leverages these demonstrations to improve sample efficiency during interaction. Although effective, this imposes considerable engineering overhead: for each new task practitioners typically must (1) collect and curate task-specific data, (2)hand-design or shape reward signals, and (3) implement bespoke task-completion (done) detectors. More recently, another direction explores real-world RL atop large Vision-Language-Action (VLA) modelsChen et al. (2025); Lv et al. (2025); Park et al. (2025). Their strong pretrained multimodal priors markedly accelerate early exploration and raise the probability of discovering successful behaviors. Nevertheless, heterogeneity across existing VLA architectures—spanning perception–language fusion, action representation and formatting, and progress/value evaluation interfaces—has led to a diverse and fragmented set of RL integration strategies, with no clear convergence of design patterns so far.

General reward models, In many real-world RL settings, tasks are initially formulated with only a terminal success/failure or extremely sparse segmented signal, exacerbating both exploration depth and credit assignment challengesChen et al. (2025); Luo et al. (2024a); Zhou et al. (2024b). A direct route to improved sample efficiency is to construct dense and temporally consistent progress signals. Because, under the Markov assumption, rewards can be treated as functions of the current observation plus a goal specification (without explicit action dependency), large-scale action-free internet/embodied video and pretrained multimodal models (VLMs or text–image alignment models) have been leveraged to build transferable value/progress functions. Existing approaches fall broadly into four categories: (1) Prompt-based VLM scoringWang et al. (2024); Yang et al. (2024): prompting a general VLM to estimate completion for a single state or to compare pairs/adjacent states; (2) Semantic embedding distance (e.g., CLIP-style)Mendonca et al. (2023a); Xiong et al. (2024): mapping states and goal descriptions into a shared embedding space and using similarity as

instantaneous reward; (3) Goal state synthesis/editingZhou et al. (2024a): employing image editing or conditional generation models to produce an "ideal" target image from the current state and goal description, then transforming the difference into a reward; (4) Learned progress embeddings (implicit time-contrastive / temporal-difference)Biza et al. (2024); Ma et al. (2022, 2023); Zhang et al. (a): enforcing temporal ordering, local smoothness, and contrastive constraints on embodied or human demonstration videos so that embeddings become (piecewise) rank-consistent with task progression; distances between current and goal embeddings then implicitly define value/reward for arbitrary goal images or goal text. While directly querying generic VLMs provides rapid, task-agnostic scoring interfaces, such "external" reward sources suffer from limited spatial/geometric precision, sensitivity to viewpoint/occlusion/lighting shifts, temporal instability (non-monotonic framewise scores), and weak discrimination of failed trajectories—collectively inflating advantage variance and destabilizing policy updates. In contrast, time-contrastive or TD-style progress representation learning can yield smoother reward signal in an episode, but lacks generalization capabilitiesMa et al. (2024). Key open questions includes: (a) fine-grained yet low-noise progress estimation, (b) strong discriminability for failures and deviations, and (c) tight integration with multimodal perception and action generation in a single end-to-end architecture.

RL post-training for VLAs, especially for vision–language models: most systems employ a single autoregressive architecuture, so policy gradient methods—typically PPOSchulman et al. (2017) variants or REINFORCE objectives (e.g., GRPOShao et al. (2024))—operate directly on token log-probabilities. In Vision - Language - Action (VLA) settings, in contrast, the architectures are heterogeneous: some produce discrete action tokens or structured textual templatesKim et al. (2024); Pertsch et al. (2025), others produce continuous actionBlack et al. (2024); Chi et al. (2023); Kim et al. (2025), preventing a uniform transplantation of token-centric RL recipes.

For discrete autoregressive action heads, token-level PPO remains a natural choice. When actions are produced via diffusion or flow-matching (continuous iterative decoding), Two broad patterns appear. (1) Value-guided filtering or weighting (imitation-flavored): rejection samplingHe et al. (2024) or importance weighting emphasizes high-Q/advantage samplesZhang et al. (b), effectively steering the generative distribution toward higher-return behaviors. (2) Offline-regularized RL: maximize value with an auxiliary BC regularization term that keeps the learned policy close to the expert (or prior) distribution, thereby reducing distributional shift and extrapolation errors Lv et al. (2025); Park et al. (2025). Because diffusion and flow matching rely on multi-step denoising or ODE integration, naïve end-to-end backpropagation of RL signals through every iteration creates long gradient chains (BPTT-like) with instability and computational overheadLv et al. (2025). And solution include: aligning the gradients of policy directly to the Q during denoising processPsenka et al. (2024); truncating gradients to the final K denoising/integration stepsRen et al. (2024); collapsing the multi-step process into a single-step policy via distillationLi et al. (2025); Park et al. (2025) or mean-flowGeng et al. (2025).

Strong pretrained (zero/few-shot) capability is central in real-world VLA RL, and Offline-to-Online strategies preload buffers with human or demonstration data to reduce early exploration costLuo et al. (2024b). These approaches borrow from offline RL regularization: addressing value underestimationZhou et al. (2024b), and coupling Q maximization with a BC regularizer to limit distributional driftChen et al. (2025); Ding and Jin; Kang et al. (2025). They typically emphasize maintaining proximity to initial human data while enabling gradual policy improvement through selective reuse of stored trajectories.

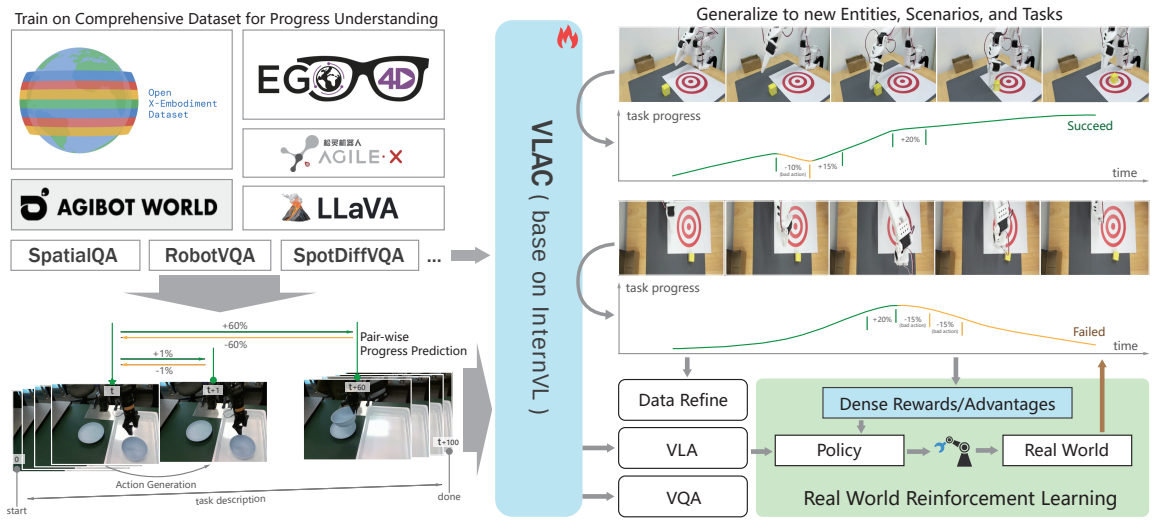


Figure 2. The VLAC model is trained on a combination of comprehensive public robotic manipulation datasets, human demonstration data, self-collected manipulation data, and various image understanding datasets. Video data is processed into pair-wise samples to learn the different task progress between two frames, supplemented with task descriptions and task completion evaluation to enable task progress understanding and action generation, as illustrated in the bottom-left corner. As shown in the diagram on the right, the model demonstrates strong generalization capabilities to new robots, scenarios, and tasks not covered in the training dataset. It can predict task progress and distinguish failure action or trajectory, providing dense reward feedback for real-world RL and offering guidance for data refinement. Additionally, the model can directly perform manipulation tasks, exhibiting zero-shot capabilities to handle different scenarios.

3. VLAC model and Real-World RL Framework

We propose a unified VLAC model that leverages multi-source data to enhance both task progress critic and action policy capabilities. The VLAC model can accurately predict task progress and generate actions, demonstrating strong generalization to unseen environments and tasks. Based on VLAC, we further design a real-world reinforcement learning framework and incorporate a human-in-the-loop framework to improve efficiency, enabling self-improvement in real-world settings. In the following sections, we provide detailed introductions to Vision-Language-Action-Critic Model and Real-world RL with VLAC.

3.1. Vision-Language-Action-Critic Model

We constructed a Vision-Language-Action-Critic Model based on pair-wise progress understanding, enabling joint training with both action robotics data and non-action human data, as shown in Figure 2. This model jointly realizes action generation and delta task progress generation, featuring in-context learning and cross-scenario, cross-task generalization capabilities. The pairwise progress module takes as input two images corresponding to arbitrary intermediate task states; it is agnostic to data collection strategy and to segment starting points, thereby improving robustness and broad applicability. Its output is a delta progress value: a positive value indicates that the second image reflects a more advanced stage of task completion (a negative value indicates the opposite). This value naturally serves as a temporal-difference (TD) reward for reinforcement learning and fine-grained dataset quality estimation. Our VLAC model is trained on more than 3000 hours of human data,

1200 hours of comprehensive public robotic manipulation data, and over 15 hours of self-collected manipulation data, providing action control, robust task progress rate assessment, and task completion verification.

3.1.1. VLAC critic learning

Humans often encounter limitations in their initial capabilities when faced with new environments and new tasks. However, their ability to understand the progression of the task is exhibited in a higher generalizability than their ability to execute tasks. This ability enables continuous assessment of the task progress across various processes, thereby optimizing one's actions and achieving sustained improvement. Rich general knowledge thus endows the understanding of task progress with strong generality, which can facilitate capabilities when faced with different environments and tasks. Inspired by this, for robot manipulation models to be deployed in the complexity of the real world and execute general tasks, they must not only understand language and visual information, but also understand task progress, which can indicate changes in task completion status across different processes. We design a pair-wise task progress understanding method that is unaffected by the initial point of the task. This method integrates general human data without action information and robot data annotated with actions to robustly estimate fine-grained variations in task processes.

We formalize the task process as a video segment with description $V = (O, l_{\text{task}})$, which consists of a basic RGB image sequence $O = (o_1, \ldots, o_T)$ and a textual task goal description l_{task}. Each training trajectory in our dataset is a successful and efficient execution of the annotated task. To understand the variations of the task process, we assume that task progress is positively correlated with time; as time increases, the task progresses. Thus, pair-wise task progress understanding can be formalized as

$$c_{i,i+\Delta t} = \text{VLAC}(o_i, o_{i+\Delta t}; l_{\text{task}}), \tag{1}$$

where Δt denotes the time difference between two frames, and $c_{i,i+\Delta t}$ represents the degree to which the task progresses in $o_{i+\Delta t}$ advances the task relative to o_i. Specifically, $\Delta t \in [-i+1, T-i] \cap \mathbb{Z}$, which allows us to focus on both fine-grained single-step changes and long-term task progress, mitigating noise from minor variations while naturally constructing balanced negative samples. During training, $c_{i,i+\Delta t}$ is annotated according to the natural temporal order as $c_{i,i+\Delta t} = \Delta t/(T-i)$, representing the percentage of task progress from o_i to $o_{i+\Delta t}$. This progress estimation formulation treats any sampled sub-segment of a trajectory as a self-contained unit: it measures only the relative advancement between the two frames within that local window and is agnostic to the trajectory's global start point or data collection strategy, thereby improving generality and robustness, as illustrated in the lower left corner of Figure 2.

To further enhance the semantic understanding of the task process, we construct a task description estimation objective:

$$l_{\text{task}} = \text{VLAC}(o_{i_{\text{start}}}, o_{i_{\text{end}}}), \tag{2}$$

where $o_{i_{\text{start}}} \in [0, 0.3T] \cap \mathbb{Z}$ and $o_{i_{\text{end}}} \in [0.8T, T] \cap \mathbb{Z}$. By generating task descriptions from the initial and final frames, the task description becomes not only an input condition but also an output target, thereby improving the joint understanding of vision and language.

To enhance understanding of task completion, we design a task completion judgment task:

$$l_{\text{done}} = \text{VLAC}(o_i; l_{\text{task}}), \tag{3}$$

where if $i < 0.8T$, $l_{\text{done}} = 0$ indicates the task is not yet completed, and if $i > 0.95T$, $l_{\text{done}} = 1$ indicates the task is completed. Considering the diversity of data and collection strategies, it is difficult to accurately determine the exact completion point; thus, for $0.8T \leq i \leq 0.95T$, no training label is

made to ensure label accuracy. By learning to judge task completion, the model's understanding of completion conditions is enhanced, providing auxiliary signals for task completion in real-world RL.

We use different prompts to distinguish different tasks. Furthermore, to improve the task progress understanding, we design four data construction strategies:

1. Pair-wise Image Difference Filtering: We assume task progress is positively correlated with time, which generally holds but may be violated in noisy data or segments with minimal change (e.g., static scenes). To mitigate the impact of such noise, we set the interval between i and $i+1$ to approximately 0.2s during data construction and compare the pixel difference between the two frames. If $\text{Diff}(o_i, o_{i+\Delta t}) < \sigma$, then set $c_{i,i+\Delta t} = 0$, indicating the two frames are in the same progress. In our experiments, we set $\sigma = 1\%$, enabling the model to focus on significant changes and improving the robustness of task progress understanding.

2. Pair-wise Progress Understanding with Joint Sampling: Inspired by contrastive learning, to ensure data balance and symmetry between forward and reverse processes, for each sampled pair $(o_i, o_{i+\Delta t})$, we construct four related data samples as a mini group within a batch:

$$\begin{cases} (o_i, o_{i+1}) \\ (o_{i+1}, o_i) \\ (o_i, o_{i+\Delta t}) \\ (o_{i+\Delta_t}, o_i) \end{cases}$$

This covers both forward and backward processes, as well as fine-grained and global understanding, as shown in the lower left of Figure 2.

3. Task Completion Judgment Joint Sampling: For data balance in the task completion judgment task, each time we sample a pair of data: one from a completed state and one from an incomplete state within the same trajectory.

4. Cross-sampling of Task Descriptions and Image Sequences: To enhance the model's ability to distinguish whether a process matches the task description, we sample, with a 5% probability during pair-wise data sampling, a task description l_{task} that does not belong to the current trajectory, setting $c_{i,i+\Delta t} = 0$. This method aims to improve the alignment of semantic and progress understanding of the model.

Cross-scene and cross-task transferability remains a key challenge for embodied intelligence models on the path to generalization. When humans adapt to new environments and tasks, their initial capabilities may also be limited; however, having reference examples can significantly improve both initial performance and learning efficiency. Inspired by this, and to improve the cross-scene and cross-task transferability of VLAC, we further enhance progress understanding with in-context learning, enabling effective learning from a single reference example. Specifically, in-context progress understanding can be formalized as

$$c_{i,i+\Delta t} = \text{VLAC}(o_i, o_{i+\Delta t}; l_{\text{task}}, O_{\text{ref}}, o_0), \tag{4}$$

where O_{ref} is the reference process, which may be provided by a robot demonstration or a human demonstration, offering guidance on both scene and task logic. o_0 is the starting point of the current trajectory and can be optionally included as input, enhancing the model's ability to align with the reference process and enabling inference of the absolute progress of o_i and $o_{i+\Delta t}$.

These tasks we construct do not require action information, thereby avoiding the issue of inconsistent action spaces across entities. This design also allows our approach to apply to both human

and robot data, enabling the use of large-scale, diverse human data to significantly improve model generalization and alleviate the scarcity of robot data in the real world. Moreover, in-context learning endows the model with rapid transfer capabilities and further enhances its generalization.

3.1.2. VLAC action learning

Based on general task process understanding, we further construct an action generation task in the semantic space to achieve multi-task control of a robotic arm. Since the general generation capabilities of pretrained multimodal models are mainly in the semantic space, and task understanding is also generated in the semantic space, we fully leverage this knowledge and the strong semantic representations of pretrained models by representing actions as numbers and generating them in the semantic space by the autoregressive approach.

To further improve spatial reasoning performance, the action is represented as the delta End-Effector (eef) pose, which is a general spatial three-dimensional representation while remaining independent of embodied entities:

$$a_i = \text{VLAC}(o_i^0, \ldots, o_i^k; s_i; l_{\text{task}}; \text{history}_{i-1,i-t_h}),$$

where o_i^k denotes the k-th viewpoint at the i-th step, s_i represents the state of the robotic arm, and a_i is the action to be executed at the i-th step, represented as a string of numbers. $\text{history}_{i-1,i-t_h}$ is the generated action history from step $i-1$ to $i-t_h$.

With this formulation, VLA exhibits strong semantic and scene generalization capabilities. Additionally, the generated actions can be sampled with diversity within a reasonable range, which is beneficial for exploration and improvement in reinforcement learning.

3.1.3. VLAC vision-language perception learning and data mixture

To enhance the model's multimodal understanding capabilities, we incorporate a series of publicly available VQA datasets, focusing on four aspects: general conversational ability, robotic understanding, spatial reasoning, and pair-wise image difference distinction. The details of the datasets are shown in Appendix.B. By leveraging these datasets, we aim to comprehensively improve the model's multimodal reasoning and understanding capabilities across diverse application domains.

For all three tasks above, we collected and processed data from various sources, including human demonstration data, multiple types of robotic arm data, and VQA datasets, i.e., Ego4D HOD Pei et al. (2025), AGIBOT Bu et al. (2025), Bridge Walke et al. (2023), Droid Khazatsky et al. (2024), FMB Luo et al. (2025), RoboSet Bharadhwaj et al. (2024), Self Collected dataset, Llava Liu et al. (2023), SpatialQA Chen et al. (2024), RobotVQA Sermanet et al. (2024), Spot the diff Jhamtani and Berg-Kirkpatrick (2018), InstructPix2Pix Brooks et al. (2023). In addition, we collected a small portion of our data specifically for fine-tuning action representations on our robotic arm. The details of the datasets and their combinations are shown in Table 4. In total, we sampled 40 million data points (some of which include multi-turn dialogues) for training.

3.2. Real-world RL with VLAC

3.2.1. Infrastructure of RL

We build upon a single-controller architecture, differing from existing RLHF frameworks by focusing on the robot-centric design for real-world reinforcement learning. Observations sent by each robot must be processed promptly to generate actions; otherwise, delays in real-world execution

would waste valuable time as the robotic arm waits. To address this, we implemented a dynamic inference server allocation mechanism, ensuring that all robot observations are assigned to idle VLA replicas for inference within 0.1 seconds. Although this approach reduces GPU utilization and results in smaller inference batches for some VLA replicas, it significantly minimizes latency during real-world robot operations.

It is important to note that the robots execute tasks asynchronously, meaning that uploading observations and executing actions can occur independently. Robots do not need to wait for the completion of an action before uploading new observations, which reduces waiting time. However, this asynchronous process means that the generated actions may not correspond to the exact observation at the moment of action generation. To address this, during VLA training, action timestamps are adjusted to lag behind observation timestamps by a duration determined by the VLA's inference time. By coordinating the adjusted action timing with the robot's motion speed, the next action arrives precisely when the previous action finishes execution, ensuring smooth and continuous robot movements.

The VLA model operates on GPU servers and communicates with robots via ZeroMQ (zmq). The framework is built using Ray and comprises several independent components: inference workers for prediction, trainers for model training, data servers for storage and distribution, log servers for logging aggregation, and rollout workers for handling robot communication and information collection. For inference, both vllm and torch can be used. While vllm provides faster inference, we observed significant discrepancies between vllm and torch results. Under identical neural network parameters and samples, the importance ratios for the same action generated by vllm and torch fluctuate between 0.4 and 1.8. This discrepancy frequently triggers the clipping mechanism in PPO, rendering approximately 60% of the data unusable. Therefore, when using vllm, despite its faster inference speed, we must recompute the action probabilities with torch during training instead of directly reusing the probabilities obtained during inference.

3.2.2. PPO based policy optimization

During rollout, the inference worker checks whether the task has been done with the VLAC model, using prompt "<image> The 1 means yes, the 0 means no. Check if the robot has completed its task: <task>", and the reward is extracted using prompt "Image-1: <image> Image-2: <image> Compare two images and evaluate whether the second image is closer to achieving task objectives compared to the first image. + score means the second image is closer, - score means the first image is closer Response the relative progressing of target task follow <score>. The target task is: task <score>". Then Proximal Policy Optimization (PPO) is adopted to improve the policy model, PPO introduces a stable and efficient approach to optimizing policies in reinforcement learning by constraining the magnitude of updates. The algorithm uses a clipped surrogate objective function to balance policy improvement and stability:

$$L^{\text{PPO}} = \mathbb{E}_t \left[\min \left(r_t \cdot A_t, \text{clip}(r_t, 1-\epsilon, 1+\epsilon) \cdot A_t \right) \right], \tag{5}$$

where $r_t = \frac{\pi_{\text{new}}(a_t|s_t)}{\pi_{\text{old}}(a_t|s_t)}$ is the likelihood ratio between the new and old policies and A_t is the advantage function, computed using Generalized Advantage Estimation (GAE). The clipping mechanism ensures that the updates to the policy remain within a safe range, preventing large deviations from the previous policy and mitigating instability during training.

Additionally, PPO incorporates entropy regularization to encourage exploration and reduce premature convergence. The final objective combines the clipped surrogate loss and entropy regularization, with the latter weighted by a hyperparameter to control its contribution. PPO achieves a balance

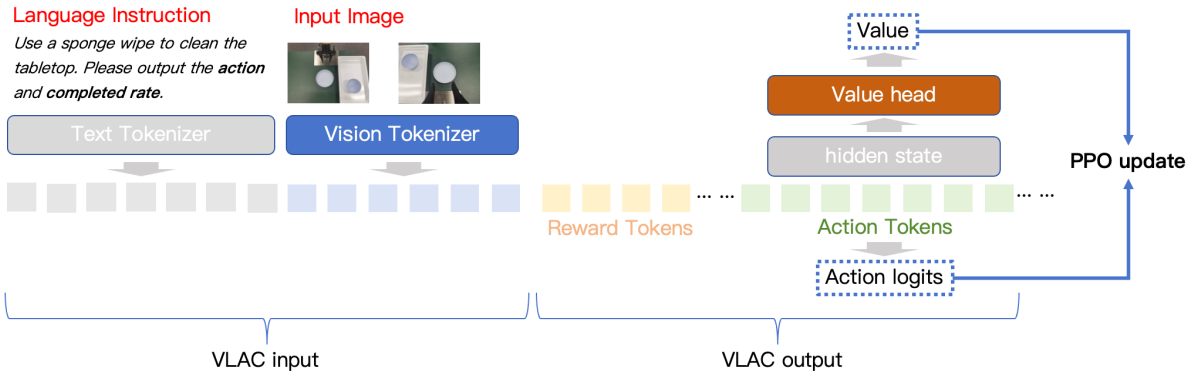


Figure 3. VLAC forward pass generates structured action tokens, reward tokens, and a value head is attached to estimate state value for PPO updates.

between exploration and exploitation while avoiding the computational overhead of second-order methods, making it effective in both discrete and continuous action spaces.

VLAC model is a multi-modal large-scale model in which actions are generated through language-based outputs following a predefined template. For instance, an action can be expressed as:

"x: -47mm, y: 19mm, z: 66mm, roll: 14 degrees, pitch: 10 degrees, yaw: 15 degrees, open: 0".

To facilitate autoregressive generation, we define a structured template where non-numeric elements, such as "x:" or "degrees," are directly retrieved from the template. For numeric outputs, the model selects tokens corresponding to numerical values from the token vocabulary. During this process, we record the logits associated with the selected tokens, which represent the probabilities of the corresponding actions. These probabilities are subsequently utilized in the PPO algorithm for policy gradient updates. The structured template design significantly reduces the number of tokens required for autoregressive generation, thereby improving computational efficiency. By minimizing the token space for generation, this approach not only accelerates inference but also ensures the consistency of the output format, which can be easily parsed into corresponding actions.

Figure 3 summarizes the integration of PPO optimization atop VLAC. After the structured autoregressive action template is decoded, we extract the hidden state (prior to the final token projection) and pass it through a linear value head to obtain $V(s_t)$. Together with the logits from the action tokens, this value estimate is used to compute the PPO loss. The value head is lightly pretrained on a small curated subset of demonstration and early exploration trajectories to reduce initial advantage variance.

3.2.3. Human-in-the-loop

During our experiments, we found that the initial capability of the policy is critically important. In some cases, the policy fails to explore correct actions, and in others, its success rate remains low, requiring targeted practice. Moreover, the training process is often unstable—if the robot fails to explore successfully for a period, the model quickly deteriorates, resulting in a situation where recovery becomes nearly impossible. To address these issues, we adopted three human-in-the-loop strategies to stabilize the training process:

Offline Demonstration Replay: We implemented an expert demonstration replay buffer, pre-populating it with data collected by human operators. During training, we periodically sample from

this dataset and update the model using negative log-likelihood (NLL) loss:

$$L_{NLL}(\theta) = -\sum_{(a_t,s_t)\in D_{human}} \log \pi_\theta(a_t|s_t) \tag{6}$$

We observed that the effectiveness of this approach is highly influenced by the data collection habits of human operators. Before collecting data, the tele-operator observes the policy execution for a period and intuitively supplements the dataset with successful examples for scenarios where the policy has a low success rate.

Return and Explore: During our experiments, we identified scenarios with high failure rates, such as specific initial positions in tasks like picking up a bowl or sweeping trash. To address these cases, we introduced targeted training: while the robot operates, the tele-operator observes positions where the policy frequently fails and manually resets the robot and objects to these states to begin exploration.

Human Guided Explore: Even with reset points, the policy often fails to explore the correct behavioral trajectories required to complete the task. For such cases, we ask the tele-operator to provide demonstration data, which is then added to the aforementioned human replay buffer to facilitate efficient exploration. This approach significantly accelerates the model's ability to learn key behaviors needed for task completion.

4. Experiments

4.1. Implementation Details

During the pre-training phase of the VLAC model, we used a batch size of 3200 and set the maximum learning rate to 8e-4. In the real-world RL experiments, we adopt a 2B VLAC model as the actor, i.e., policy model, and use a 8B VLAC model as the "critic" as introduced above. The "critic" plays two roles: computing the reward for each step and indicating whether the task is done. It is worth noted that the normal critic which used to compute GAE is simplely a deep neural network. The capability of prior policy model is critical in real-world RL experiments, thus we collected nearly 100 trajectories by tele-operation and pretrain the policy model as base model. Since the inference and training are asynchronous, the actor occupies two GPU to speed up the training procedure. The "critic" uses a single GPU for computing reward and done signal. For all the tasks, the observation settings are the same. The observation contains instruction, one front camera image, robot endpose position. The robot in our real-world experiment is AGILE PiPER and it is controlled via a 7-DOF end effector based on the delta pose mechanism. We use the PPO algorithm to train the RL policy and the baseline in experimental results is just implementing the PPO algorithm. Other methods are based on PPO and introduce some new mechanisms, such as **Return then explore**, **Human guided explore**, etc.

4.2. Evaluation Datasets

To evaluate our VLAC model to understand task progress, especially its generalization to out-of-distribution scenarios such as new scenes, new tasks, and new entities, we conducted tests not only on the test sets included in the training datasets, but also on six additional datasets that were not seen during training. These include both human operation datasets and datasets containing failure processes. Specifically, in addition to the Bridge Walke et al. (2023) and Droid Khazatsky et al. (2024) datasets from the training set, we selected RT1 Brohan et al. (2022), RoboNet Dasari et al.

(2019), Dobb-E Shafiullah et al. (2023), RH20T Fang et al. (2023), EgoDex Hoque et al. (2025), and RoboFAC Lu et al. (2025) for evaluation.

Dobb-E features a unique gripper and only provides gripper-perspective views, making it suitable for testing cross-entity and cross-viewpoint generalization. RoboNet lacks language annotations and does not exhibit smooth temporal structure, so it should show poor task and temporal correlation in the absence of reference examples. EgoDex is a human hand manipulation dataset, which can be used to evaluate general task process understanding and the model's compatibility with dexterous hand tasks. RoboFAC contains two subsets: one with successful task processes and one with failed processes, providing a direct way to evaluate the model's progress understanding ability.

4.3. Evaluation Metrics

Following the GVL Ma et al. (2024), we use Value-Order Correlation (VOC) as an evaluation metric for task progress understanding. This metric computes the rank correlation between the predicted values and the chronological order of the input expert video:

$$\begin{cases} \text{VOC} = \text{rank-correlation}(\text{argsort}(v_1, \ldots, v_T); \text{arange}(T)) \\ v_i = v_{i-\Delta t} + c_{i-\Delta t,i}(1 - v_{i-\Delta t}) \\ v_0 = 0 \end{cases} \tag{7}$$

VOC ranges from -1 to 1. Higher VOC scores indicate better task completion, with task progression increasing over time. A good critic model should achieve high VOC scores when evaluated on expert videos. To better evaluate pairwise methods, we additionally construct Value-Reversed-Order Correlation (VROC). During testing, the entire sequence is reversed, so the order of pairwise frames is inverted and the predicted values should also be reversed. The closer the VROC score is to the VOC score, the better and more stable the model's performance. We further define VOC-F1 as:

$$\text{VOC-F1} = 2 \cdot \frac{\text{VOC} \cdot \text{VROC}}{\text{VOC} + \text{VROC}} \tag{8}$$

which comprehensively evaluates the correlation of task progression and temporal order in the video. Additionally, to evaluate the fine-grained performance of the critic model, we use Negative Rate (NR), which measures the proportion of reversed process pairs:

$$\text{NR} = \frac{N(c_{i,i+\Delta t} < 0)}{N} \tag{9}$$

where $N(c_{i,i+\Delta t} < 0)$ is the number of negative evaluations, and N is the total number of evaluations. This metric reflects how many actions in the video do not contribute to task progression and is also an important indicator of the quality of a trajectory.

Following the $\pi_{0.5}$ Intelligence et al. (2025), we success rate and task progress as evaluation metrics for action generation. Task progress is evaluated by humans, allowing for more detailed progress assessments when the task is not fully completed, and providing a finer evaluation of the model's manipulation ability.

For real-world RL experiments, we mainly investigate the success rate and learning efficiency. The success rate reflects the capability of the policy and it is evaluated by running 10 trials. The learning efficiency can be observed from the success rate curve during the training procedure.

4.4. VLAC Critic Performance

The VLAC model trained on public robotic manipulation data mainly includes bridge, droid, roboset, fmb, AgiBot World, excluding the RT1 and other datasets, and the robotic arm entities, scenes,

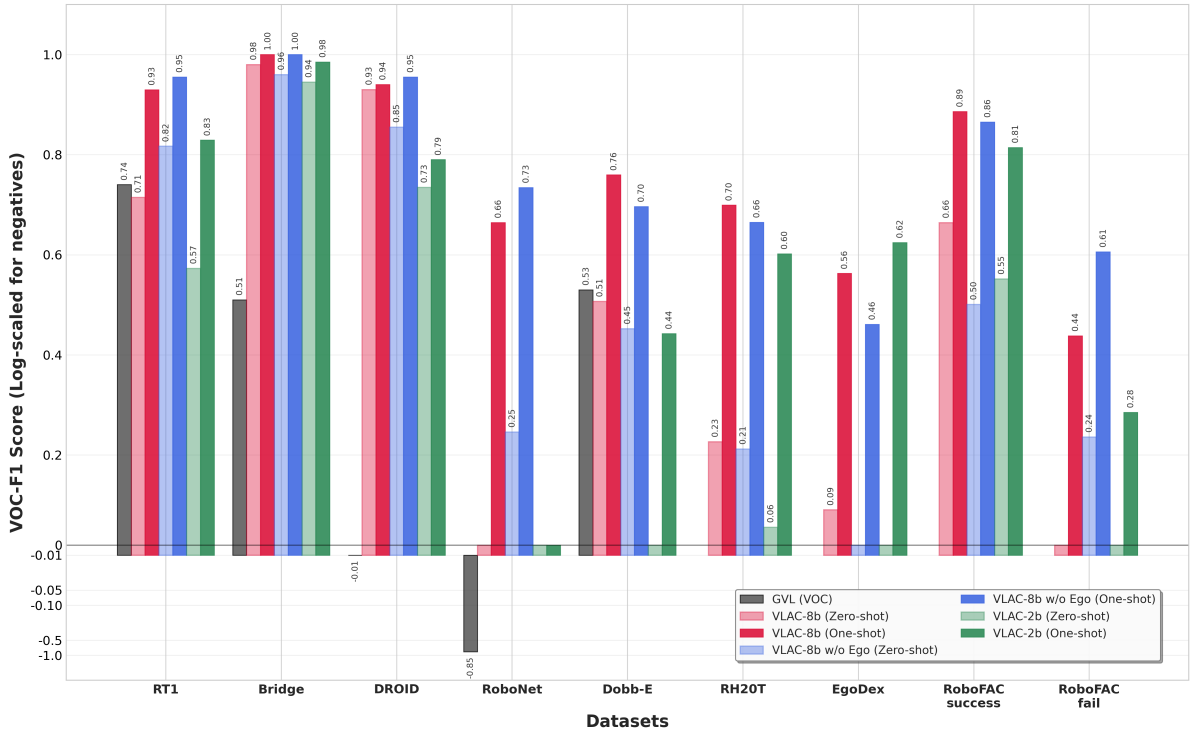


Figure 4. VOC-F1 Performance Comparison Across Different Models and Datasets.

and tasks in these datasets are almost different from the above datasets. We will conduct further large-scale validation on these datasets. We trained three models: the VLAC-8b model with 8 billion parameters, the VLAC-8b w/o Ego model (which was trained without the Ego4D dataset), and the VLAC-2b model with 2 billion parameters. The overall performance across the eight datasets is shown in Figure 4. Our model demonstrates strong results on in-distribution datasets (Bridge and Droid), as well as robust generalization on out-of-distribution datasets. Notably, under one-shot conditions, the model's powerful contextual reasoning significantly improves its ability to assess task progression. For example, in the RT1 dataset, where the tasks, scenes, and robotic arms differ greatly from those in the training data, the VOC-F1 still reaches 0.95, indicating highly accurate task process prediction. On the RoboNet dataset, which lacks language annotations and does not exhibit smooth temporal structure, the VOC-F1 is 0 in the zero-shot setting—reflecting the model's correct reliance on language descriptions for task progression. However, when provided with examples, the one-shot performance improves dramatically, further highlighting the model's strong contextual learning capabilities. For the EgoDex dataset, which consists of human demonstration data, we observe that even without incorporating the Ego4D dataset, the model can still leverage context to understand task processes. After including Ego4D in training, this capability is significantly enhanced. The RoboFAC dataset contains both successful and failed task executions. Our method clearly distinguishes between these two types of trajectories, achieving a VOC-F1 of 0.89 on successful videos and only 0.44 on failed ones. This demonstrates VLAC's strong ability to identify erroneous actions. Furthermore, for RoboFAC, the model trained with Ego4D data shows an even greater gap between successful and failed videos, indicating that human video data provides significant benefits for embodied task process understanding. The specific experimental results are shown in the Table 1.

We show some example results in Figure 5, where (b–h) illustrate performance on datasets cross different entities, scenes, and tasks, demonstrating the strong generalization capabilities of the proposed model. More specifically, Figure 5 (b) highlights the model's understanding of the

Dataset	GVL (Gemini-1.5-Pro)	VLAC-8b w/o Ego (InternVL-8b)						VLAC-8b (InternVL-8b)						VLAC-2b (InternVL-2b)					
		zero-shot			one-shot			zero-shot			one-shot			zero-shot			one-shot		
	VOC	VOC	VROC	NR	VOC	VROC	NR	VOC	VROC	NR	VOC	VROC	NR	VOC	VROC	NR	VOC	VROC	NR
RT1	0.74	0.87	0.77	0.19	0.96	0.95	0.11	0.71	0.72	0.22	0.91	0.95	0.14	0.69	0.49	0.22	0.80	0.86	0.25
Bridge	0.51	0.96	0.96	0.05	1.00	1.00	0.00	0.97	0.99	0.02	1.00	1.00	0.00	0.94	0.95	0.08	0.98	0.99	0.05
DROID	-0.01	0.85	0.86	0.09	0.96	0.95	0.06	0.92	0.94	0.04	0.93	0.95	0.06	0.75	0.72	0.13	0.79	0.79	0.11
RoboNet	-0.85	0.20	0.32	0.50	0.76	0.71	0.23	NAN	NAN	NAN	0.59	0.76	0.31	NAN	NAN	NAN	-0.04	0.81	0.52
Dobb-E	0.53	0.49	0.42	0.39	0.75	0.65	0.28	0.47	0.55	0.34	0.74	0.78	0.26	-0.04	0.43	0.36	0.37	0.55	0.31
RH20T	none	0.24	0.19	0.34	0.68	0.65	0.21	0.17	0.34	0.32	0.64	0.77	0.22	0.03	0.40	0.32	0.49	0.78	0.25
EgoDex	none	0.58	-0.41	0.29	0.64	0.36	0.32	0.05	0.48	0.48	0.44	0.78	0.38	0.00	0.59	0.47	0.57	0.69	0.36
RoboFAC-success	none	0.700	0.39	0.26	0.86	0.87	0.21	0.76	0.59	0.14	0.83	0.95	0.20	0.46	0.69	0.20	0.73	0.92	0.29
RoboFAC-fail	none	0.45	0.16	0.34	0.66	0.56	0.32	0.30	-0.06	0.22	0.41	0.47	0.35	-0.05	0.34	0.26	0.19	0.57	0.43

Table 1. Performance of VLAC's task progressing understanding across entities, scenarios, and tasks.

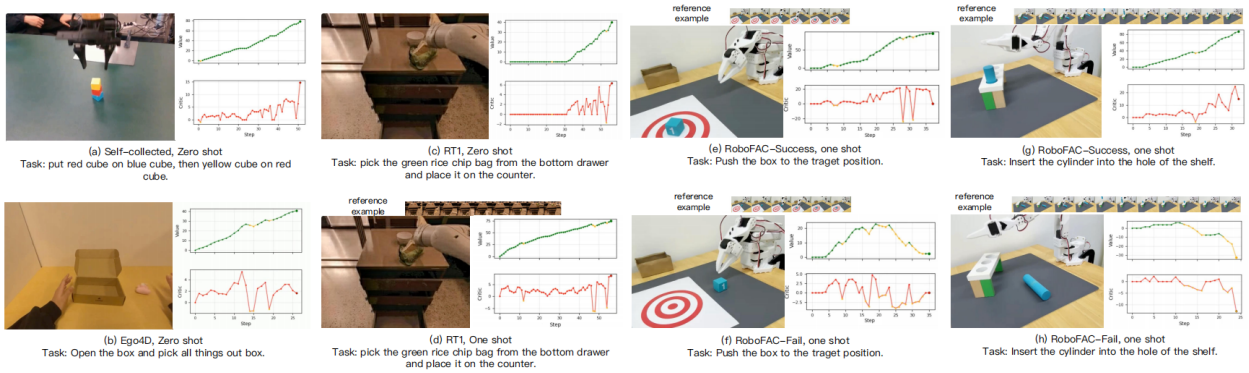


Figure 5. Example results of VLAC for task progress understanding.

human dataset. Figures 5 (c) and (d) show zero-shot and one-shot results for the same process, illustrating how in-context learning enhances the model's ability to comprehend new task processes. Figures 5 (e) and (g) depict successful task processes, while Figures 5 (f) and (h) present the corresponding failed task processes, indicating that the model can clearly distinguish between successful and unsuccessful processes. These examples show our model can robustly generalize across heterogeneous embodiments, scenes, and tasks, leveraging minimal in-context signals to accurately infer, monitor, and contrast fine-grained task progress, and even differentiate subtle failure modes from successful executions.

4.5. VLAC Actor Performance

To validate the action generation capability of VLAC, we collected 100 samples for each testing task and trained the 8B VLAC model across all tasks. Additionally, since autonomous evolution in the real world often requires exploring different environments independently, the model must possess strong generalization abilities to adapt to scene variations. To test this, we evaluated the VLAC model's success rate under lighting disturbances and scene changes without requiring extra data collection. As shown in Table 2, the "Lighting Transfer" scenario involved turning off fluorescent lights and using colored flashing light sources as disturbances. For the "Scene Transfer" scenario, tests were conducted on two different workbenches located at different sites and different settings, with varying

Method	Open Cooker		Pick&Place Bowl		Unfold Mat		Desktop Sweep		Rice Transfer		**Average**	
	task progress	success rate	task progress	success rate	task progress	success rate	task progress	success rate	task progress	success rate	task progress	success rate
Pi0	85%	85%	54%	40%	32%	0%	30%	10%	45%	30%	49.2%	27%
Pi0+Lighting Transfer	0%	0%	7.5%	0%	35.5%	5%	31.5%	10%	9%	0%	16.7%	3%
VLAC w/o pretrain	50%	50%	0%	0%	2.5%	0%	0%	0%	56%	30%	21.7%	16%
VLAC	90%	90%	85%	85%	91%	85%	62.5%	40%	84.5%	75%	**82.5%**	**75%**
VLAC+Lighting Transfer	85%	85%	65%	65%	72.5%	60%	49.5%	25%	75.5%	50%	69.5%	57%
VLAC+Scene Transfer	90%	90%	70%	65%	80%	70%	60.0%	40%	68%	50%	73.6%	63%

Table 2. Performance of VLAC's action generation across scenarios.

camera perspectives that were not precisely calibrated. The examples of training data and evaluation environments can be seen in Figure 9. The results demonstrated the model's robust generalization ability, as it was able to consistently generate reasonable actions even under extreme lighting changes. This makes VLAC a suitable starting point for autonomous evolution in dynamic and unpredictable environments. Meanwhile, "VLAC w/o pretrain" refers to the model without task progress pretraining, which leads to a significant drop in success rate. Although the actions generated during testing remain reasonable, the model struggles to accurately determine the current task state. For example, in the pick-and-place task, it may proceed to the placement step even if it has not successfully grasped the object. Our process understanding task enhances the model's ability to interpret the progression of tasks depicted in images, thereby ensuring a higher success rate at each execution stage. However, the success rate of tasks varies depending on the difficulty of the tasks, and performance is often partially lost during transfer. To improve the success rate during real-world deployment, further autonomous learning and evolution in actual environments and tasks are necessary.

4.6. Real-world RL Results

Task	**Success Rate (%)**			
	Baseline	Return and explore	Human guided explore	Offline Demonstration Replay
Desktop Sweep Disposal	90	100	100	100
Pick and Place Bowl	80	90	100	100
Unfold Mat	80	90	90	100
Rice Scooping and Transfer	100	80	100	70
Average	88	95	98	93

Table 3. Illustration of the final success rate on the real-world tasks. The success rate is evaluated by running 10 episodes.

In this subsection, we mainly investigate how to improve the policy learning performance in different real-world scenarios. Our key insight is to enhance the exploration capabilities of the model. So as to, we introduce **Offline Demonstration Replay**, **Return and explore**, **Human guided explore** in our experiments and the baseline means our real-world rl approach without all these modules.

As shown in Table 3, all these methods can achieve higher success rate on all the tasks compared to Baseline. It is worth noted that **Human guided explore** and **Offline Demonstration Replay** can both achieve 100% success rate four tasks. The experimental results show that human intervention, that is, manually collecting some trajectories that are helpful for task exploration, is important for learning the policy. **Return and explore** only performs slightly better than Baseline and can not

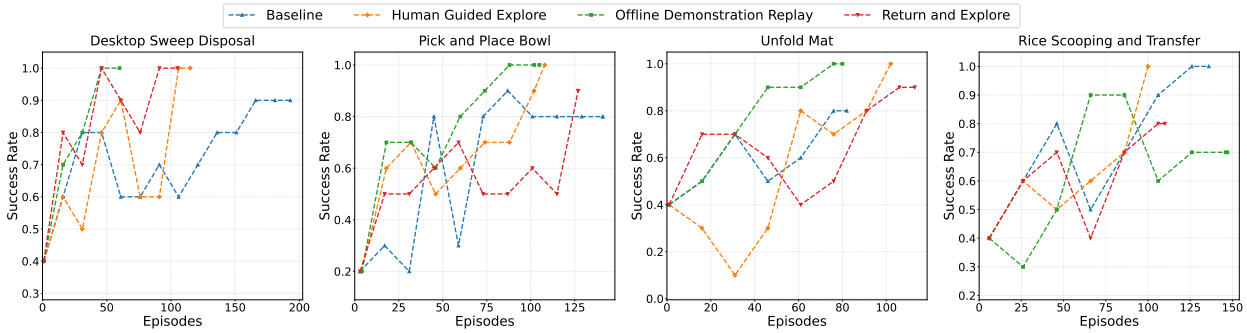


Figure 6. Experimental results of real-world RL on four tasks. This figure demonstrates the success rate for all the methods during the training procedure. Each data point is obtained via running 10 trails.

achieve similar results compared to above two methods. This may be because choosing good return points is difficult. Some return points are too easy and the prior policy reaches there frequently and easily, but some return points are too difficult and the prior policy behaves poorly there and struggles to explore.

As shown in Figure 6 , **Offline Demonstration Replay** learns faster than **Human guided explore**. This may be because the quality of data we collected in **Offline Demonstration Replay** is comparable to that of **Human guided explore**, which enables the model to overcome the difficulties encountered during the exploration process at the beginning of learning.

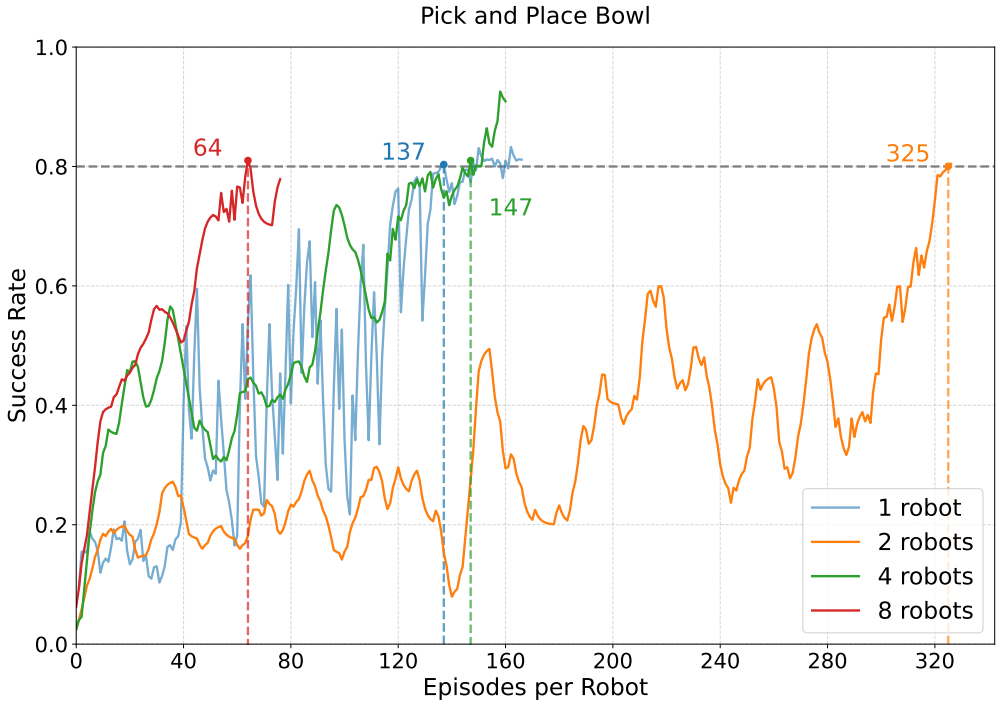


Figure 7. Training performance curves for real-world robots at different scales. The episodes required per robot to reach a high success rate decrease as the number of robots increases from 2 to 4 to 8.

4.7. Multi-Robot Scaling

We investigate how learning performance scales with the number of robots using a representative tabletop manipulation task (Pick-and-Place Bowl) trained under the Baseline method. We consider robot counts $N=1,2,4,8$ to assess whether a scaling law persists as parallel embodiment increases.

The overall results are shown in Figure 7. To quantify learning efficiency, we report the number of episodes per robot required to reach an 80% success rate (black dots in Figure 7. With 8 robots, only 64 episodes per robot are needed, requiring about 24.6 minutes of rollout. Reducing the joint training scale from 8 to 4 robots increases the requirement to 147 episodes per robot (roughly a 2× increase). Further reducing from 8 to 2 robots causes the per-robot requirement to rise sharply to 325 episodes (over 5× that of 8 robots), taking roughly two hours. Thus, the scaling law appears to hold: increasing the number of robots decreases the data required per robot to reach the same success level. An exception is the single-robot case, which requires fewer episodes than 2 robots and is comparable to 4 robots. This deviation is mainly due to: (1) its visually static background, which is easier to fit; and (2) heterogeneous backgrounds across multiple robots introducing visual disturbance that slows learning of a more generalized policy.

Unlike in simulation, observations among real-world robots are rarely identical because of hardware differences, background variation, and differing camera viewpoints. Consequently, learning speeds diverge: some robots may still struggle while others have already mastered the skill. To balance overall progress, we apply a dynamic sampling strategy that increases the sampling frequency of data generated by under-learned robots. Additional details about the environment setup are provided in the appendix.

5. Discussion and Conclusion

This work tackles the central challenge of making large Vision-Language-Action (VLA) models actually improve themselves in the physical world. Beyond introducing the VLAC (Vision-Language-Action-Critic) model, the core contribution is a full real-world reinforcement learning (RL) framework that (1) converts general multimodal understanding into dense, signed stepwise intrinsic rewards via calibrated pairwise progress deltas, (2) closes the loop on physical robots with a low-latency asynchronous infrastructure (actor–critic decoupling, progress-based done detection, reward inference in <0.1s), (3) enables continual self-improvement from both successes and failures, and (4) supports graded, optional human intervention (offline seeding, targeted resets, guided micro-demonstrations) to accelerate exploration. Empirically, across four diverse manipulation tasks the system raises success rates from 30% to 90% within 200 real-world interaction episodes, transfers under lighting and scene shifts without extra collection, and reliably separates successful from failing trajectories (high VOC / VROC contrast). The result is a practical recipe showing that large multimodal priors plus structured intrinsic progress feedback can make real-world online RL feasible, data-efficient, and incrementally improvable.

Limitations: (1) Human-in-the-loop dependence: intervention timing, reset state selection, and demonstration curation remain heuristic and operator-specific; no standardized quantitative triggers (e.g., competence plateau detectors, failure mode coverage metrics) are yet defined, limiting reproducibility and automation. (2) RL recipe specificity: the current PPO + autoregressive tokenized delta–EEF action interface is tightly coupled to a discrete semantic action head; it does not directly generalize to diffusion / flow-matching / continuous iterative decoders where reward allocation across denoising steps, progress-to-score alignment, or Q-guided sampling would need new abstractions. (3) Multi-task online instability: simultaneous RL still shows reward scale drift, uneven negative signal density, inter-task gradient interference, and episodic forgetting; no task-adaptive normalization,

uncertainty-weighted sampling, gradient conflict mitigation, modular parameter partitioning, or continual distillation safeguards are integrated. Future work will focus on formalizing intervention metrics (coverage, marginal utility curves), designing architecture-agnostic progress/value bridging layers for non-autoregressive action generators, and adding stabilization mechanisms (task-wise reward calibration, uncertainty-driven replay prioritization, gradient surgery, lightweight policy/module distillation) to scale robust multi-task real-world evolution.

References

H. Bharadhwaj, J. Vakil, M. Sharma, A. Gupta, S. Tulsiani, and V. Kumar. Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4788–4795. IEEE, 2024.

O. Biza, T. Weng, L. Sun, K. Schmeckpeper, T. Kelestemur, Y. J. Ma, R. Platt, J.-W. van de Meent, and L. L. Wong. On-robot reinforcement learning with goal-contrastive rewards. *arXiv preprint arXiv:2410.19989*, 2024.

J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

T. Brooks, A. Holynski, and A. A. Efros. Instructpix2pix: Learning to follow image editing instructions. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18392–18402, 2023.

Q. Bu, J. Cai, L. Chen, X. Cui, Y. Ding, S. Feng, X. He, X. Huang, et al. Agibot world colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2025.

B. Chen, Z. Xu, S. Kirmani, B. Ichter, D. Sadigh, L. Guibas, and F. Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14455–14465, 2024.

Y. Chen, S. Tian, S. Liu, Y. Zhou, H. Li, and D. Zhao. Conrft: A reinforced fine-tuning method for vla models via consistency policy. *arXiv preprint arXiv:2502.05450*, 2025.

C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.

S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn. Robonet: Large-scale multi-robot learning. *arXiv preprint arXiv:1910.11215*, 2019.

S. Deng, M. Yan, S. Wei, H. Ma, Y. Yang, J. Chen, Z. Zhang, T. Yang, X. Zhang, H. Cui, et al. Graspvla: a grasping foundation model pre-trained on billion-scale synthetic action data. *arXiv preprint arXiv:2505.03233*, 2025.

Z. Ding and C. Jin. Consistency models as a rich and efficient policy class for reinforcement learning. In *The Twelfth International Conference on Learning Representations*.

H.-S. Fang, H. Fang, Z. Tang, J. Liu, C. Wang, J. Wang, H. Zhu, and C. Lu. Rh20t: A comprehensive robotic dataset for learning diverse skills in one-shot. *arXiv preprint arXiv:2307.00595*, 2023.

Z. Geng, M. Deng, X. Bai, J. Z. Kolter, and K. He. Mean flows for one-step generative modeling. *arXiv preprint arXiv:2505.13447*, 2025.

L. He, L. Shen, J. Tan, and X. Wang. Aligniql: Policy alignment in implicit q-learning through constrained optimization. *arXiv preprint arXiv:2405.18187*, 2024.

A. Herzog, K. Rao, K. Hausman, Y. Lu, P. Wohlhart, M. Yan, J. Lin, M. G. Arenas, T. Xiao, D. Kappler, et al. Deep rl at scale: Sorting waste in office buildings with a fleet of mobile manipulators. *arXiv preprint arXiv:2305.03270*, 2023.

R. Hoque, P. Huang, D. J. Yoon, M. Sivapurapu, and J. Zhang. Egodex: Learning dexterous manipulation from large-scale egocentric video. *arXiv preprint arXiv:2505.11709*, 2025.

Z. Hu, A. Rovinsky, J. Luo, V. Kumar, A. Gupta, and S. Levine. Reboot: Reuse data for bootstrapping efficient real-world dexterous manipulation. *arXiv preprint arXiv:2309.03322*, 2023a.

Z. Hu, A. Rovinsky, J. Luo, V. Kumar, A. Gupta, and S. Levine. Reboot: Reuse data for bootstrapping efficient real-world dexterous manipulation. In J. Tan, M. Toussaint, and K. Darvish, editors, *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 1930–1949. PMLR, 06–09 Nov 2023b. URL https://proceedings.mlr.press/v229/hu23a.html.

P. Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, et al. π0. 5: a vision-language-action model with open-world generalization, 2025. *URL https://arxiv. org/abs/2504.16054*, 1(2):3, 2025.

H. Jhamtani and T. Berg-Kirkpatrick. Learning to describe differences between pairs of similar images. *arXiv preprint arXiv:1808.10584*, 2018.

Z. Kang, C. Hu, Y. Luo, Z. Yuan, R. Zheng, and H. Xu. A forget-and-grow strategy for deep reinforcement learning scaling in continuous control. *arXiv preprint arXiv:2507.02712*, 2025.

A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

M. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

M. J. Kim, C. Finn, and P. Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

N. Kumar, T. Silver, W. McClinton, L. Zhao, S. Proulx, T. Lozano-Pérez, L. P. Kaelbling, and J. Barry. Practice makes perfect: Planning to learn skill parameter policies. *arXiv preprint arXiv:2402.15025*, 2024.

Q. Li, Z. Zhou, and S. Levine. Reinforcement learning with action chunking. *arXiv preprint arXiv:2507.07969*, 2025.

F. Lin, Y. Hu, P. Sheng, C. Wen, J. You, and Y. Gao. Data scaling laws in imitation learning for robotic manipulation. *arXiv preprint arXiv:2410.18647*, 2024.

H. Liu, C. Li, Q. Wu, and Y. J. Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

W. Lu, M. Ye, Z. Ye, R. Tao, S. Yang, and B. Zhao. Robofac: A comprehensive framework for robotic failure analysis and correction. *arXiv preprint arXiv:2505.12224*, 2025.

J. Luo, Z. Hu, C. Xu, Y. L. Tan, J. Berg, A. Sharma, S. Schaal, C. Finn, A. Gupta, and S. Levine. Serl: A software suite for sample-efficient robotic reinforcement learning, 2024a.

J. Luo, C. Xu, J. Wu, and S. Levine. Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning, 2024b.

J. Luo, C. Xu, J. Wu, and S. Levine. Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning. *arXiv preprint arXiv:2410.21845*, 2024c.

J. Luo, C. Xu, F. Liu, L. Tan, Z. Lin, J. Wu, P. Abbeel, and S. Levine. Fmb: a functional manipulation benchmark for generalizable robotic learning. *The International Journal of Robotics Research*, 44(4): 592–606, 2025.

L. Lv, Y. Li, Y. Luo, F. Sun, T. Kong, J. Xu, and X. Ma. Flow-based policy for online reinforcement learning. *arXiv preprint arXiv:2506.12811*, 2025.

Y. J. Ma, S. Sodhani, D. Jayaraman, O. Bastani, V. Kumar, and A. Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. In *Deep Reinforcement Learning Workshop NeurIPS 2022*, 2022.

Y. J. Ma, V. Kumar, A. Zhang, O. Bastani, and D. Jayaraman. Liv: Language-image representations and rewards for robotic control. In *International Conference on Machine Learning*, pages 23301–23320. PMLR, 2023.

Y. J. Ma, J. Hejna, C. Fu, D. Shah, J. Liang, Z. Xu, S. Kirmani, P. Xu, D. Driess, T. Xiao, et al. Vision language models are in-context value learners. In *The Thirteenth International Conference on Learning Representations*, 2024.

R. Mendonca, S. Bahl, and D. Pathak. Alan: Autonomously exploring robotic agents in the real world. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3044–3050, 2023a. doi: 10.1109/ICRA48891.2023.10161016.

R. Mendonca, S. Bahl, and D. Pathak. Alan: Autonomously exploring robotic agents in the real world. *arXiv preprint arXiv:2302.06604*, 2023b.

R. Mendonca, E. Panov, B. Bucher, J. Wang, and D. Pathak. Continuously improving mobile manipulation with autonomous real-world rl. *arXiv preprint arXiv:2409.20568*, 2024.

S. Park, Q. Li, and S. Levine. Flow q-learning. *arXiv preprint arXiv:2502.02538*, 2025.

B. Pei, Y. Huang, J. Xu, G. Chen, Y. He, L. Yang, Y. Wang, W. Xie, Y. Qiao, F. Wu, et al. Modeling fine-grained hand-object dynamics for egocentric video representation learning. *arXiv preprint arXiv:2503.00986*, 2025.

K. Pertsch, K. Stachowicz, B. Ichter, D. Driess, S. Nair, Q. Vuong, O. Mees, C. Finn, and S. Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

M. Psenka, A. Escontrela, P. Abbeel, and Y. Ma. Learning a diffusion model policy from rewards via q-score matching. In *Proceedings of the 41st International Conference on Machine Learning*, 2024.

A. Z. Ren, J. Lidard, L. L. Ankile, A. Simeonov, P. Agrawal, A. Majumdar, B. Burchfiel, H. Dai, and M. Simchowitz. Diffusion policy policy optimization. In *arXiv preprint arXiv:2409.00588*, 2024.

J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

P. Sermanet, T. Ding, J. Zhao, F. Xia, D. Dwibedi, K. Gopalakrishnan, C. Chan, G. Dulac-Arnold, S. Maddineni, N. J. Joshi, et al. Robovqa: Multimodal long-horizon reasoning for robotics. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 645–652. IEEE, 2024.

N. M. M. Shafiullah, A. Rai, H. Etukuru, Y. Liu, I. Misra, S. Chintala, and L. Pinto. On bringing robots home. *arXiv preprint arXiv:2311.16098*, 2023.

Z. Shao, P. Wang, Q. Zhu, R. Xu, J. Song, X. Bi, H. Zhang, M. Zhang, Y. Li, Y. Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

L. Smith, I. Kostrikov, and S. Levine. Demonstrating a walk in the park: Learning to walk in 20 minutes with model-free reinforcement learning. *Robotics: Science and Systems (RSS) Demo*, 2(3):4, 2023.

L. Smith, Y. Cao, and S. Levine. Grow your limits: Continuous improvement with real-world rl for robotic locomotion. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10829–10836. IEEE, 2024.

G. R. Team, S. Abeyruwan, J. Ainslie, J.-B. Alayrac, M. G. Arenas, T. Armstrong, A. Balakrishna, R. Baruch, M. Bauza, M. Blokzijl, et al. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

H. R. Walke, K. Black, T. Z. Zhao, Q. Vuong, C. Zheng, P. Hansen-Estruch, A. W. He, V. Myers, M. J. Kim, M. Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pages 1723–1736. PMLR, 2023.

Y. Wang, Z. Sun, J. Zhang, Z. Xian, E. Biyik, D. Held, and Z. Erickson. Rl-vlm-f: Reinforcement learning from vision language foundation model feedback. In *Proceedings of the 41th International Conference on Machine Learning*, 2024.

H. Xiong, R. Mendonca, K. Shaw, and D. Pathak. Adaptive mobile manipulation for articulated objects in the open world. *arXiv preprint arXiv:2401.14403*, 2024.

K. Xu, Z. Hu, R. Doshi, A. Rovinsky, V. Kumar, A. Gupta, and S. Levine. Dexterous manipulation from images: Autonomous real-world rl via substep guidance. *arXiv preprint arXiv:2212.09902*, 2022.

D. Yang, D. Tjia, J. Berg, D. Damen, P. Agrawal, and A. Gupta. Rank2reward: Learning shaped reward functions from passive video. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2806–2813, 2024. doi: 10.1109/ICRA57147.2024.10610873.

H. Zhang, M. Zhou, S. Zhai, Y. Sun, and H. Xiong. Efficient skill discovery via regret-aware optimization. In *Forty-second International Conference on Machine Learning*, a.

S. Zhang, W. Zhang, and Q. Gu. Energy-weighted flow matching for offline reinforcement learning. In *The Thirteenth International Conference on Learning Representations*, b.

Z. Zhou, P. Atreya, A. Lee, H. Walke, O. Mees, and S. Levine. Autonomous improvement of instruction following skills via foundation models. *arXiv preprint arXiv:407.20635*, 2024a.

Z. Zhou, A. Peng, Q. Li, S. Levine, and A. Kumar. Efficient online reinforcement learning fine-tuning need not retain offline data. *arXiv preprint arXiv:2412.07762*, 2024b.

A. Evalauation tasks

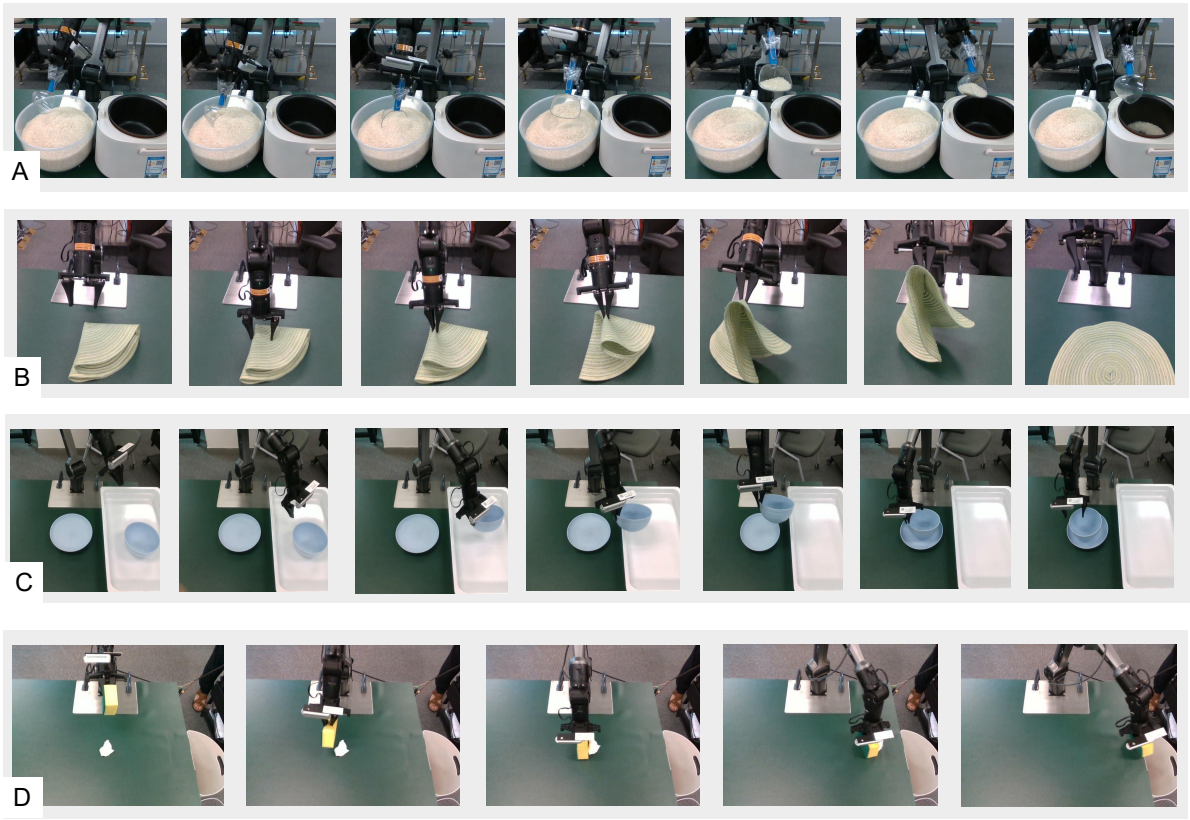


Figure 8. Illustrations of the real-world tasks in our experiments. (A) Rice Scooping and Transfer: scoop the rice from the big bowl and transfer it to the cooker. (B) Unfold Mat: unford the folded mat on the table. (C) Pick and Place Bowl: pick up the bowl from the tray and place it on the plate. (D) Desktop Sweep Disposal: sweep the trash on the table into the trash can.

We design five manipulation tasks in the real-world RL experiments as shown in Figure 8. These tasks are expected to constitute a rich and relatively complete kitchen scene from cooking food to setting up the table. In addition, these manipulation tasks are diverse from two perspectives: 1) manipulating different types of objects, i.e., rigid objects (C), flexible objects (B,D), and granular objects (A); 2) requires different manipulation ability, either precise goal reaching, i.e., touch/push objects (D), or grasp proper parts (B,C), or dynamic manipulation (A).

Rice Scooping and Transfer In this task, the robot is assumed to firstly scoop a spoonful of rice from the jar, and then pour the rice into the cooker. The task is considered successful if the rice is transferred from the jar to cooker without spilling. The difficulty lies in the granular objects are not easy to be obtained and the transportation also requires the robot to be very stable. This task is also named **Rice Transfer** for abbreviation.

Unfold Mat In this task, a mat is folded on the desktop, and the robot is supposed to grab the mat, lift it up, and then release it to unfold this mat. This task is considered successful if the mat unfolds well on the table. The difficulty lies in two points: 1) the robot must grab a proper part of the mat, the bottom, middle and far side of the mat is hard to grad or unfold; 2) the mat must be raised high enough so that it can fall and spread naturally, otherwise it may still be folded after falling on the table.

Pick and Place Bowl In this task, a plate is placed on the table and a bowl is placed in a tray, and the robot is supposed to pick up the bowl and place it on the plate. The task is considered successful if the bowl is properly put on the plate. This task requires precise and gentle gripping of the bowl's rim and delivering it precisely to the center of the plate.

Desktop Sweep Disposal In this task, there is a white trash on the table and the robot is supposed to sweep it into the trash can. The task is considered successful if the trash is swept into the nearby can. Similar to task (A), this task requires the robot to precisely reach the trash and push it with a proper force. Too much or too little force can lead to failure. This task is also named **Desktop Sweep** for abbreviation.

B. Dataset

LLAVA Liu et al. (2023): This dataset includes basic multi-turn dialogues and a rich collection of VQA data, which helps maintain the model's general multimodal understanding and conversational abilities.

RoboVQA Sermanet et al. (2024): This dataset contains VQA data from various robotic tasks, aimed at improving the model's understanding of multimodal data in robotic scenarios.

SpatialVQA Chen et al. (2024): This dataset provides spatial reasoning data based on RGB images, including depth estimation, object detection, and more, which strengthens the model's ability to understand and estimate spatial information within images.

Spot-the-diff Jhamtani and Berg-Kirkpatrick (2018), InstructPix2Pix Brooks et al. (2023): These datasets consist of pair-wise image difference comparison data, designed to enhance the model's capability to detect fine-grained differences between images, thereby supporting pair-wise progress understanding tasks.

Dataset Name	Size/samples	Size/h	Tasks	Mixture Weight
Ego4D HOD Pei et al. (2025)	157M	3k	TPU	14.6%
InstructPix2Pix Brooks et al. (2023)	36k	-	GVL	0.4%
RobotVQA Sermanet et al. (2024)	50k	-	GVL	0.6%
Spot the diff Jhamtani and Berg-Kirkpatrick (2018)	27k	-	GVL	0.3%
Llava Liu et al. (2023)	633k	-	GVL	4.7%
SpatialQA Chen et al. (2024)	781k	-	GVL	5.8%
AGIBOT Bu et al. (2025)	8M	73	TPU,GVL,VLA	3.0%
Bridge Walke et al. (2023)	2M	135	TPU,VLA	9.1%
Droid Khazatsky et al. (2024)	40M	741	TPU,VLA	30.0%
FMB Luo et al. (2025)	2m	144	TPU,VLA	9.7%
RoboSet Bharadhwaj et al. (2024)	4m	130	TPU,VLA	17.4%
Self Collected	946k	18	TPU,VLA	4.4%

Note: TPU = Task Progress Understanding; GVL = General Vision-Language; VLA = Vision-Language-Action.

Table 4. Overview of Data Mixture.

C. Scene Transfer

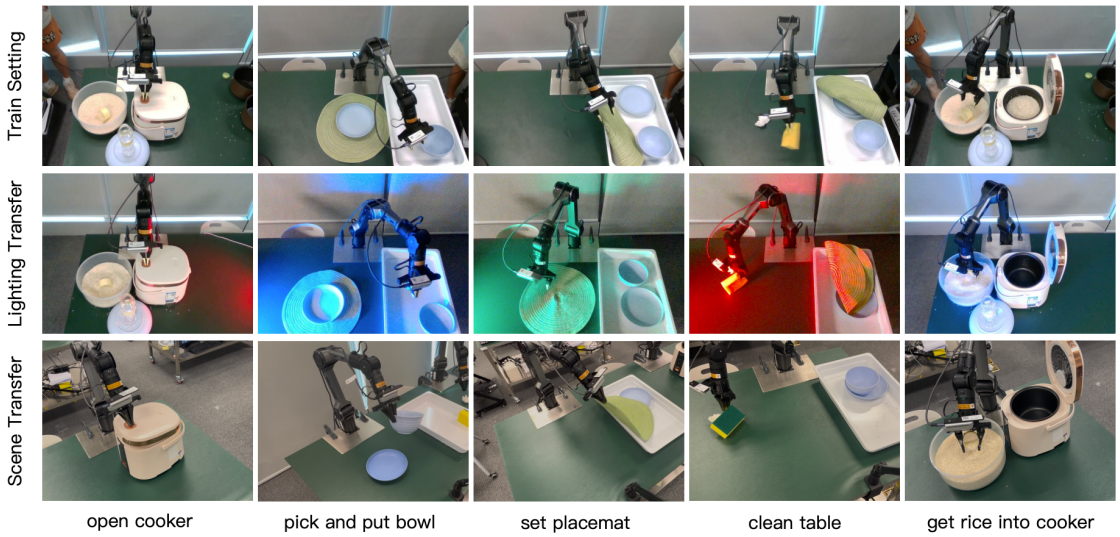
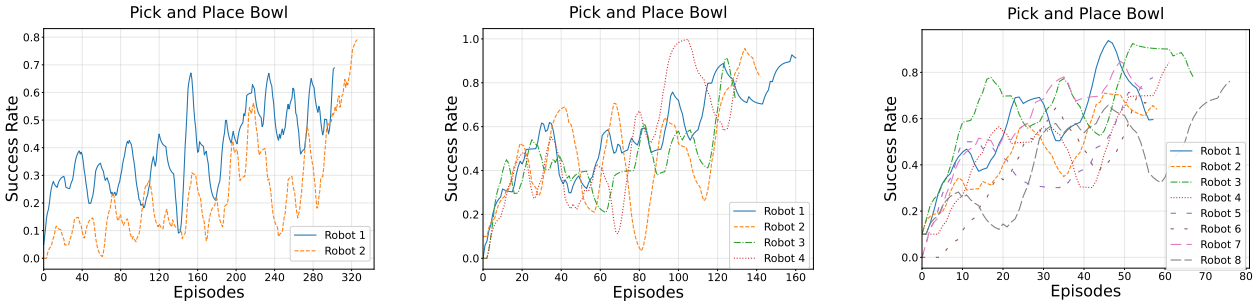


Figure 9. Illustrations of scene/lighting transfer.

D. Multi-Robots Scaling

Real-world RL across multiple robots exposes a learning imbalance driven by differences in background clutter, lighting, and robot status (e.g., temperature). These small covariate shifts lead to divergent per-robot success curves under a shared policy. Robots that have higher success rates tend to learn faster than others, and robots that have lower success rates hardly improve over time. To achieve stable multi-robot training, we adopt a dynamic sampling strategy that increases the sampling frequency of data generated by under-learned robots. In this way, multiple robots can be trained more stably, as shown in Figure 10.



(a) Training with 2 robots (b) Training with 4 robots (c) Training with 8 robots

Figure 10. Demonstration of the experimental results on Pick and Place Bowl with different amount of robots trained at the same time.

E. Author contributions

All authors contributed to writing.

- Shaopeng Zhai (zhaishaopeng@pjlab.org.cn): Team leadership, led design and development, RL

training, infrastructure engineering, robot/data engineering, VLAC pretraining, architecture/algorithm design, experiments analysis, paper writing.

- Qi Zhang (zhangqi1@pjlab.org.cn): Robot/data engineering, architecture/algorithm design, experiments analysis, paper writing, and significant contribution to VLAC pretraining.
- Tianyi Zhang (zhangtianyi@pjlab.org.cn): Experiments analysis, robot engineering, infrastructure engineering, experiments analysis, and significant contribution to RL training.
- Fuxian Huang (huangfuxian@pjlab.org.cn): Experiments analysis, paper writing, and significant contribution to RL training.
- Haoran Zhang (zhanghaoran@pjlab.org.cn): Experiments analysis, robot engineering, experiments analysis, and significant contribution to multi-robot RL training.
- Ming Zhou (zhouming@pjlab.org.cn): algorithm design and significant contribution to infrastructure engineering.
- Jiangmiao Pang (pangjiangmiao@pjlab.org.cn): Team leadership and advised project direction with critical feedback.