

# Affordance Field Intervention: Enabling VLAs to Escape Memory Traps in Robotic Manipulation

Siyu Xu<sup>1</sup> Zijian Wang<sup>1</sup> Yunke Wang<sup>1</sup> Chenghao Xia<sup>1</sup> Tao Huang<sup>2</sup> Chang Xu<sup>1</sup>

<sup>1</sup>School of Computer Science, The University of Sydney

<sup>2</sup>John Hopcroft Center for Computer Science, Shanghai Jiao Tong University

{s.xu, yunke.wang, c.xu}@sydney.edu.au

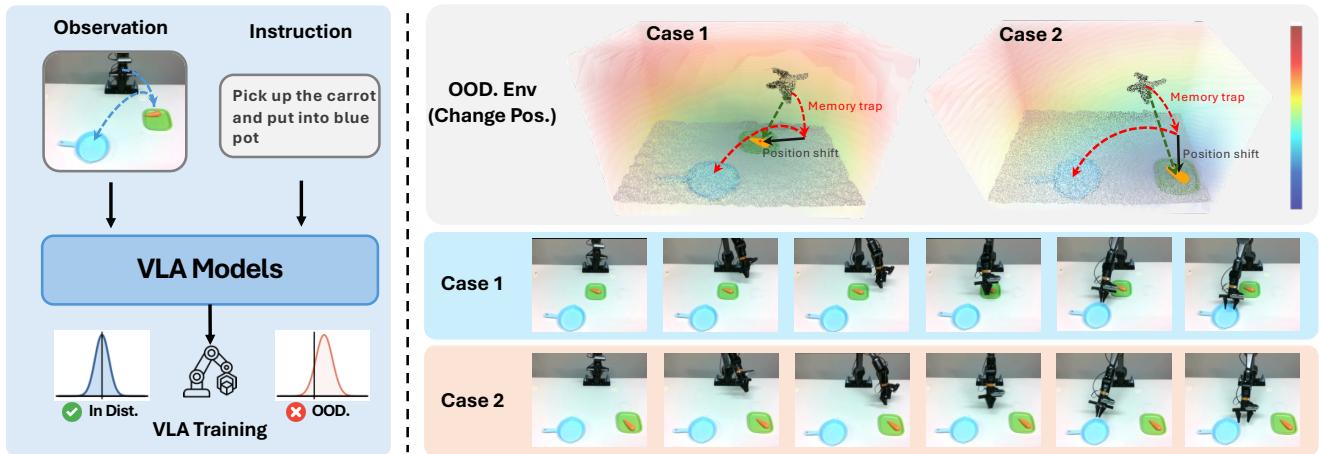


Figure 1. **Memory Trap in VLAs.** VLAs often fail under distribution shifts. In such a case, they replay trajectories memorized during training instead of adapting to the updated scene. In both Case 1 and Case 2, even though the target object moves, the VLA drives the end-effector toward the original location, ignoring new spatial cues.

## Abstract

Vision-Language-Action (VLA) models have shown great performance in robotic manipulation by mapping visual observations and language instructions directly to actions. However, they remain brittle under distribution shifts: when test scenarios change, VLAs often reproduce memorized trajectories instead of adapting to the updated scene, which is a failure mode we refer to as the “Memory Trap”. This limitation stems from the end-to-end design, which lacks explicit 3D spatial reasoning and prevents reliable identification of actionable regions in unfamiliar environments. To compensate for this missing spatial understanding, 3D Spatial Affordance Fields (SAFs) can provide a geometric representation that highlights where interactions are physically feasible, offering explicit cues about regions the robot should approach or avoid. We therefore introduce Affordance Field Intervention (AFI), a lightweight hybrid framework that uses SAFs as an on-demand plugin to guide VLA behavior. Our system detects memory

traps through proprioception, repositions the robot to recent high-affordance regions, and proposes affordance-driven waypoints that anchor VLA-generated actions. A SAF-based scorer then selects trajectories with the highest cumulative affordance. Extensive experiments demonstrate that our method achieves an average improvement of 23.5% across different VLA backbones ( $\pi_0$  and  $\pi_{0.5}$ ) under out-of-distribution scenarios on real-world robotic platforms, and 20.2% on the LIBERO-Pro benchmark, validating its effectiveness in enhancing VLA robustness to distribution shifts.

## 1. Introduction

Vision-Language-Action (VLA) models [5, 8, 21] have emerged as powerful motion planners in robotic manipulation, directly mapping visual observations and linguistic instructions to executable action sequences and enabling seamless interaction with objects in diverse environments. By leveraging large-scale pre-trained vision-language foun-

dations, these end-to-end neural architectures facilitate generalization across tasks, such as grasping and object rearrangement, without requiring extensive task-specific engineering [4, 14, 20].

Despite these advancements, VLA models encounter substantial challenges in generalization, frequently succumbing to a “Memory Trap” [6, 7, 40]. In out-of-distribution (OOD) scenarios, such as significant perturbations in object positions, these models tend to rigidly reproduce trajectories memorized from the training data. Consequently, this suboptimal behavior leads to task failure, as the model directs the end-effector toward obsolete positions rather than the actual updated target. This limitation stems from VLA’s end-to-end design, which implicitly fits mappings from vision-language inputs to actions based on training distributions, without explicitly perceiving and reasoning over 3D regions for interaction [22, 32, 39]. As a result, VLAs lack the planning capabilities to generate actions that target the appropriate regions in unfamiliar environments. Instead, they rely on memorized trajectories from training, which fail to adapt to perturbations [11, 27].

Recent works have introduced the concept of affordance to guide action planning. An object’s affordance, described as “opportunities of interaction”, provides direct and intuitive guidance in robot workspace [1, 10, 19, 35]. By leveraging multimodal understanding and reconstruction techniques, these methods generate 3D spatial affordance fields (SAFs) that highlight actionable regions [12, 13]. Once the SAF identifies target endpoints, non-learning-based methods such as optimization-based trajectory planning can generate dynamically feasible action trajectories [15, 31, 37]. However, these VLM-based planning approaches [12, 13] suffer from low success rates in practice due to two critical limitations: (1) unreliable VLM-generated motion plans that lack fine-grained geometric understanding and often produce infeasible actions [3, 30], (2) heavy reliance on task-specific prompt engineering to generate diverse constraints, which are brittle and lack transferability across different manipulation scenarios [25, 34].

To overcome these challenges, we propose Affordance Field Intervention (AFI), a novel hybrid framework that treats a 3D SAF as a plug-in for VLA-based action generation. The core of our method lies in how to leverage the SAF to help the VLA escape the memory trap and navigate toward high-affordance regions, thereby improving task success rates. First, we design a memory trap detection mechanism using robot proprioception. By monitoring end-effector motion patterns and goal progress, our system identifies when the VLA falls into rigid, memorized trajectories that fail to adapt to environmental changes. Second, upon detecting the memory trap, the AFI rolls back the EEF to a recent high-affordance position for safe repositioning under the guidance of the SAF. From there, it proposes in-

termediate waypoints as nearby high-affordance points that progressively guide toward the target region (e.g., updated object location). These waypoints act as geometric anchors, breaking the VLA’s rigid memorization with explicit spatial cues.

Finally, the VLA is queried to generate action proposals conditioned on these SAF-guided waypoints. To ensure spatial optimality, the SAF acts as a scorer, evaluating and re-ranking the VLA’s action candidates based on their projected trajectories’ cumulative affordance values. The action with the highest affordance value, indicating alignment with favorable paths, is selected for execution. This closed-loop modular integration allows the VLA to leverage its semantic understanding and efficiency while being softly constrained by 3D geometry through grounded interventions. It enables adaptive navigation to high-affordance regions *without parameter updates*, effectively bridging data-driven policies with interpretable planning.

We conduct extensive experiments on both real-world robotic platforms and simulation benchmarks to validate our proposed AFI. On real-world manipulation tasks using an AgileX Piper manipulator, our method achieves consistent improvements across four diverse tasks, with average success rate gains ranging from 17.0% to 26.0% over baseline VLA policies ( $\pi_0$  and  $\pi_{0.5}$ ) under various OOD scenarios including position shifts, color changes, object variations, and background shifts. Our framework also demonstrates model-agnostic generalization, with ensemble integration of multiple VLA backbones achieving up to 89.0% success rate. On LIBERO-Pro [40] simulation with spatial perturbations, our method improves  $\pi_{0.5}$  by 21.7% (Spatial) and 16.8% (Object). These results validate that AFI effectively mitigates the memory trap problem and enhances VLA robustness to distribution shifts without requiring model retraining or additional demonstration data.

## 2. Related Work

### 2.1. Vision-Language-Action Models

Vision-Language-Action (VLA) models [5, 8, 21] have emerged as a promising approach for general-purpose robotic manipulation. These models frame robot control as a sequence modeling task, trained end-to-end via imitation learning on large datasets of camera observations, language instructions, and actions. However, a significant body of recent literature highlights that these models suffer from poor generalization and inherent fragility when faced with spatial perturbations or novel environments [6, 7, 40]. Research suggests that existing VLA models often struggle with robust action planning in unseen contexts, relying heavily instead on memorized or imitated trajectories from their training data distribution [2, 9].

Recent studies also explored the integration of reinforce-

ment learning (RL) to enhance the generalization capabilities of VLA models [24, 27, 28, 36]. By incorporating RL-based fine-tuning, these approaches aim to adapt VLA policies to diverse and unseen scenarios beyond mere imitation. However, a key challenge lies in obtaining reliable reward signals for RL training [16, 23], which often requires extensive human annotation or complex simulation environments. Moreover, scaling RL to large-scale datasets and diverse environments remains a significant challenge, limiting its practicality for real-world robotic applications [28, 38].

## 2.2. Grounding 3D Affordance and Action Planning

Grounding 3D affordances plays a pivotal role in robotic manipulation by bridging language instructions with actionable spatial representations, enabling agents to infer object interactions (e.g., graspable regions or avoidance zones) directly in the 3D perceptual space [1, 18]. Grounding 3D affordances explicitly encodes target object locations and interaction cues into dense spatial maps, allowing action planning to be offloaded to these affordance maps.

Building on this, VoxPoser [12] is a novel framework that leverages large language models (LLMs) and vision-language models (VLMs) to compose composable 3D value maps for zero-shot manipulation tasks. Geomanip [33] uses geometric constraints as general interfaces for robot manipulation, allowing agents to reason about object interactions and plan actions in a more principled manner. GIGA [17] is a structured implicit representation that couples 3D reconstruction with an affordance field for 6-DoF grasping, grounding action cues directly in local geometry. By training the shared implicit functions on self-supervised trials, it improves occlusion-robust grasp detection and allocates representation capacity toward graspable regions, enabling more reliable action planning in clutter.

## 3. Preliminary

In this section, we introduce the background knowledge of VLA models and affordance field construction.

### 3.1. VLA Models

VLA models  $\pi_{\text{VLA}}$  take a 2D image  $I_t^{rgb}$  at timestep  $t$  and language instructions  $\tau$  as inputs to propose actions for environment interaction. These actions are executed by the controller, results in an end-effector (EEF) displacement  $\Delta \mathbf{d}_t$  in the 3D workspace, i.e.,

$$a_t \sim \pi_{\text{VLA}}(I_t^{rgb}, \tau), \quad \Delta \mathbf{d}_t = \text{controller}(a_t), \quad (1)$$

VLA models are typically trained using imitation learning, memorizing direct mappings from vision-language inputs to actions based on the training distribution. However, in out-of-distribution (OOD) scenarios, the VLA model continues to propose actions that guide the end-effector along

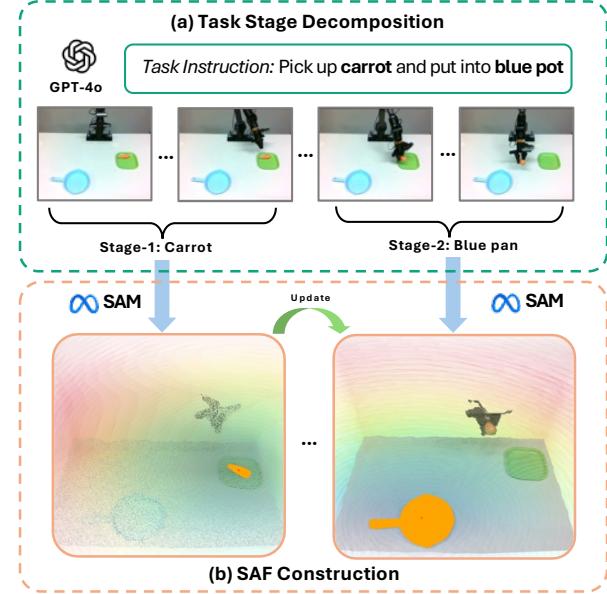


Figure 2. **Spatial Affordance Field (SAF) Construction.** (a) GPT-4o decomposes the task instruction into sequential stages and identifies the current target object (e.g., “carrot” or “blue pan”). (b) The target text is fed to Grounded-SAM for segmentation, and the resulting 2D mask is back-projected into 3D space to construct the SAF, where color gradients indicate affordance values.

the memorized trajectories from the training set, rather than adapting to the perturbed environment.

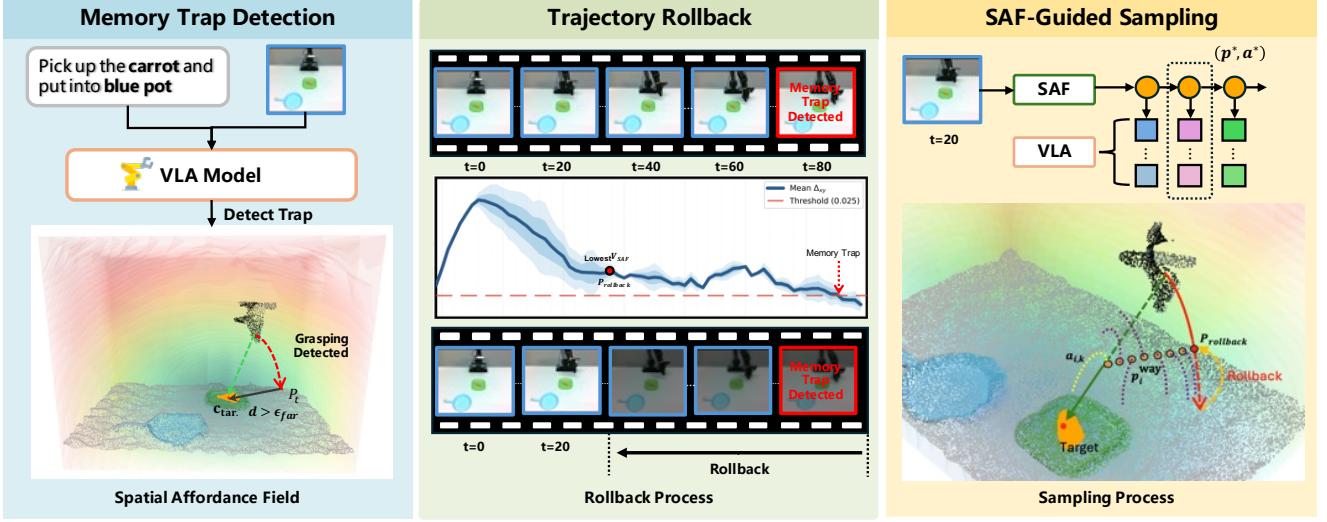
### 3.2. Affordance Field Construction

In this subsection, we introduce the two-stage pipeline for constructing the 3D spatial affordance field (SAF).

#### 3.2.1. Inferring Affordance via VLM

To infer affordances from language instructions, we first identify task-relevant objects and then project them into the robot’s workspace. Given a high-level task instruction  $\tau$  (e.g., “place the mug on the table”) and the current RGB observation  $I_t^{rgb}$ , we leverage a vision-language model (VLM) to parse the task into a sequence of semantic stages. Specifically, we use GPT-4o to decompose  $\tau$  into temporally ordered sub-goals (e.g.,  $\text{pick} \rightarrow \text{move} \rightarrow \text{place}$ ) and extract stage-wise target tokens (e.g., “mug”, “table”). These target tokens serve as text prompts for subsequent open-vocabulary object detection.

Then for each sub-goal, we apply Grounded-SAM [29] to generate a 2D segmentation mask  $M_{\text{target}} \in \{0, 1\}^{H \times W}$  corresponding to the target object. By combining  $M_{\text{target}}$  with the depth map  $I_t^{\text{depth}}$  and camera intrinsic parameters, we back-project the masked region into 3D space to obtain the target point cloud  $\mathcal{P}_{\text{target}} = \{\mathbf{p}_i \in \mathbb{R}^3\}$ . Notably, when the VLM detects a sub-goal transition during execution, the system automatically updates the target identification to reflect the new semantic focus, allowing the spatial affordance



**Figure 3. Overview of Affordance Field Intervention (AFI).** (1) *Memory Trap Detection*: SAF evaluates VLA-predicted actions and detects memory traps by monitoring end-effector velocity and distance to target. (2) *Trajectory Rollback*: Upon detection, the robot rolls back to the historical position with lowest SAF cost before grasping attempts. (3) *SAF-Guided Sampling*: VLA generates trajectory candidates at SAF-sampled waypoints, and the trajectory with lowest cumulative SAF cost is selected for execution.

field to adapt dynamically throughout multi-stage manipulation tasks.

### 3.2.2. Grounding Affordance in 3D Space

We discretize the robot’s workspace into an  $N \times N \times N$  voxel grid  $\mathcal{V}$ , where each voxel  $v_{ijk}$  represents a small cuboid region in 3D space. Both the scene point cloud  $\mathcal{P}_{\text{scene}}$  (obtained from the full RGB-D observation) and the target object point clouds  $\mathcal{P}_{\text{target}}$  are projected onto this grid. Within this voxelized representation, we construct two complementary geometric subfields for each target object:

**1) Target Guidance Field  $V_{\text{target}}$ :** This field encodes spatial attraction toward the target. For each voxel  $v_{ijk}$ , we compute its Euclidean distance  $d(v_{ijk}, \mathbf{c}_{\text{target}})$  to the target object centroid  $\mathbf{c}_{\text{target}} = \frac{1}{|\mathcal{P}_{\text{target}}|} \sum_{\mathbf{p} \in \mathcal{P}_{\text{target}}} \mathbf{p}$ . After applying distance transform, regions farther from the target receive higher values, which contribute to higher cost in the final affordance field, encouraging the EEF to approach the goal region.

**2) Obstacle Avoidance Field  $V_{\text{obst}}$ :** This field encodes repulsion from scene obstacles. Voxels occupied by  $\mathcal{P}_{\text{scene}}$  or near obstacles are assigned high values to discourage collision. To prevent overly conservative behavior, we apply heuristic masking: (1) we exempt the immediate vicinity of the EEF, allowing close-range manipulation, and (2) we create a buffer zone around the target object, permitting approach actions necessary for grasping.

The final spatial affordance field  $V_{\text{SAF}}$  is obtained by fusing these subfields via weighted linear combination:

$$V_{\text{SAF}} = w_{\text{target}} V_{\text{target}} + w_{\text{obst}} V_{\text{obst}}, \quad (2)$$

where  $w_{\text{target}}$  and  $w_{\text{obst}}$  are hyperparameters balancing tar-

get attraction and obstacle avoidance. To ensure smooth spatial gradients suitable for trajectory evaluation, we apply Euclidean distance transform to both subfields, followed by Gaussian smoothing with kernel size  $\sigma$ . Finally,  $V_{\text{SAF}}$  is normalized to  $[0, 1]$ , yielding a continuous affordance cost field where lower values indicate more favorable regions (near targets and away from obstacles) for action selection.

## 4. Method

In this section, we introduce our proposed AFI that integrates the spatial affordance field (SAF) into the VLA workflow to help the VLA model escape the memory trap in out-of-distribution scenarios. We first describe how to detect whether the robot has fallen into the memory trap. Then, we explain how to integrate the SAF to intervene in the VLA workflow for escaping the memory trap.

### 4.1. Memory Trap Detection

We monitor the robot’s execution status at each timestep to detect potential memory traps. A memory trap is triggered when two conditions are simultaneously met: (1) the end-effector displacement  $\|\mathbf{p}_t - \mathbf{p}_{t-\Delta t}\|$  falls below a threshold  $\epsilon_{\text{stuck}}$  over a time window  $\Delta t$ , and (2) the distance to the target  $\|\mathbf{p}_t - \mathbf{c}_{\text{target}}\|$  exceeds a threshold  $\epsilon_{\text{far}}$ . The first condition detects when the robot enters a quasi-static state, which typically indicates either fine-grained manipulation near the target (e.g., grasping) or getting stuck in an undesired configuration. The second condition disambiguates these two scenarios: if the end-effector becomes stationary while still far from the target, it suggests the robot is stuck or performing incorrect actions (e.g., grasping the wrong

object), rather than executing the intended fine manipulation. This dual-criterion detection ensures we intervene only when genuine memory traps occur, avoiding false positives during legitimate stationary behaviors near the goal.

## 4.2. Affordance Field Intervention

Upon detecting a memory trap, the spatial affordance field (SAF) immediately comes into play through a targeted intervention mechanism, guiding the escape from the trap and steering the VLA model away from its memorized trajectories. This intervention begins with a guided rollback to a safer historical position, followed by a tree-based, SAF-guided search that samples intermediate waypoints toward high-affordance regions and integrates VLA-generated trajectories for task-directed refinement.

### 4.2.1. Historical Rollback via Affordance

We maintain a history buffer  $N$ -step long  $\mathbf{P}_{\text{hist}} = \{\mathbf{p}_{t-n}, \dots, \mathbf{p}_{t-1}\}$  of recent end-effector positions. The rollback target is selected as the historical point with the lowest affordance cost:

$$\mathbf{p}_{\text{rollback}} = \arg \min_{\mathbf{p} \in \mathbf{P}_{\text{hist}}} V_{\text{SAF}}(\mathbf{p}), \quad (3)$$

where  $V_{\text{SAF}}(\mathbf{p})$  queries the SAF at position  $\mathbf{p}$ , with lower values indicating lower cost (safer and more task-relevant regions). The robot executes a short rollback trajectory to return to this configuration  $\mathbf{p}_{\text{rollback}}$ , which serves as the root node for subsequent tree-based trajectory extension. This step repositions the robot to a safer, low-cost state, mitigating the immediate effects of the memory trap.

### 4.2.2. Hierarchical Exploration for Optimal Trajectories

Starting from the rollback position  $\mathbf{p}_{\text{rollback}}$ , we construct a two-stage tree-based exploration process to sample intermediate waypoints toward high-affordance regions and integrate VLA-generated trajectories for task-directed refinement.

**Stage 1: Local SAF-Guided Sampling of Intermediate Waypoints.** We perform a local spatial search to identify  $N$  promising intermediate waypoints in the vicinity. Specifically, we sample candidate positions from a local neighborhood  $\mathcal{N}(\mathbf{p}_{\text{rollback}}, r)$  with radius  $r$  and select those with the lowest cost values:

$$\{\mathbf{p}_i^{\text{way}}\}_{i=1}^N = \arg \min_{\mathbf{p} \in \mathcal{N}(\mathbf{p}_{\text{rollback}}, r)} V_{\text{SAF}}(\mathbf{p}), \quad (4)$$

where  $\arg \min^N$  selects the  $N$  positions with minimum cost values. These waypoints form the first-level child nodes in the trajectory extension tree, representing spatially favorable intermediate targets that prioritize low-cost regions near targets and away from obstacles.

## Stage 2: Trajectory Generation via VLA at Sampled Waypoints.

The robot sequentially navigates to each waypoint  $\mathbf{p}_i^{\text{way}}$  and queries the VLA policy  $\pi_{\text{VLA}}$  to produce  $K$  diverse action candidates  $\{\mathbf{a}_{i,k}\}_{k=1}^K$  based on the updated observation  $I_t^{\text{rgb}}$  and task instruction  $\tau$ . For stochastic policies (e.g., diffusion-based), candidates are sampled with different noise temperature or seeds.

Each action candidate  $\mathbf{a}_{i,k}$  represents an action chunk consisting of a sequence of joint states over horizon  $H$ . We apply forward kinematics to convert these joint states into the corresponding end-effector trajectory  $\xi_{i,k} = \{\mathbf{p}_j^{i,k}\}_{j=1}^H$ , where  $\mathbf{p}_j^{i,k}$  denotes the end-effector position at timestep  $j$ . The cumulative affordance cost is:

$$\mathcal{V}(\xi_{i,k}) = \sum_{j=1}^H V_{\text{SAF}}(\mathbf{p}_j^{i,k}). \quad (5)$$

This yields  $N \times K$  evaluated candidates forming the leaf nodes of the trajectory tree. We then select the globally optimal trajectory by minimizing the cumulative cost:

$$\xi^* = \arg \min_{i,k} \mathcal{V}(\xi_{i,k}). \quad (6)$$

The robot executes  $\xi^*$  by navigating to its corresponding waypoint and following the associated VLA actions. This hierarchical, exploration-driven approach effectively combines the spatial reasoning of SAF (for waypoint selection) with the task-specific capabilities of VLA (for trajectory completion), enabling robust recovery from diverse failure modes while incorporating real-world perceptual feedback.

## 5. Experiments

### 5.1. Experimental Settings

**Environments.** We evaluate our framework on both real-world robotic platforms and simulated environments. For real-world experiments, we employ an AgileX Piper manipulator equipped with two Intel RealSense D435 cameras. For simulation experiments, we utilize the LIBERO benchmark [26] with spatial perturbations following LIBERO-Pro [40] to assess OOD generalization. Detailed hardware configuration, SAF construction procedures, and training details are provided in Appendix A.

**Baselines.** We compare our method against pre-trained VLA models without 3D SAF guidance:  $\pi_0$  [4] and  $\pi_{0.5}$  [14] in real-world experiments, and the officially released  $\pi_{0.5}$ -LIBERO checkpoint in simulation. We also compare with ReKep [13], a training-free VLM-based planner, to demonstrate the advantages of our hybrid VLA+SAF approach.

**Tasks.** We evaluate on four real-world manipulation tasks covering diverse primitives: (1) *Place Carrot*: picking up a carrot and placing it in a pot; (2) *Remove Lid*: removing a lid from a pot and placing it on a platter; (3) *Slot*

Table 1. Real-world experimental results on AgileX Piper manipulator across four manipulation tasks. We report success rates over 20 trials for each scenario. Our AFI framework achieves consistent improvements across all distribution shifts and VLA backbones. See Appendix B for detailed task descriptions.

Task	Method	In Dist.	Position	Color	Task	Background	Average SR.
Place Carrot	ReKep [13]	8/20	7/20	9/20	5/20	7/20	36.0%
	$\pi_0$	17/20	6/20	13/20	15/20	10/20	61.0%
	$\pi_0$ -AFI (Ours)	<b>20/20</b>	<b>13/20</b>	<b>17/20</b>	<b>18/20</b>	<b>19/20</b>	<b>87.0% (↑26.0%)</b>
Remove Lid	$\pi_0$	20/20	8/20	17/20	5/20	13/20	63.0%
	$\pi_0$ -AFI (Ours)	<b>20/20</b>	<b>12/20</b>	<b>19/20</b>	<b>11/20</b>	<b>18/20</b>	<b>80.0% (↑17.0%)</b>
	$\pi_0$	16/20	11/20	13/20	15/20	5/20	60.0%
Slot Pen	$\pi_0$ -AFI (Ours)	<b>19/20</b>	<b>16/20</b>	<b>16/20</b>	<b>19/20</b>	<b>12/20</b>	<b>82.0% (↑22.0%)</b>
	$\pi_0$	18/20	9/20	16/20	13/20	8/20	64.0%
	$\pi_0$ -AFI (Ours)	<b>20/20</b>	<b>15/20</b>	<b>20/20</b>	<b>17/20</b>	<b>14/20</b>	<b>86.0% (↑22.0%)</b>
Stack Tape	$\pi_{0.5}$	20/20	7/20	17/20	10/20	7/20	61.0%
	$\pi_{0.5}$ -AFI (Ours)	<b>20/20</b>	<b>14/20</b>	<b>19/20</b>	<b>15/20</b>	<b>14/20</b>	<b>82.0% (↑21.0%)</b>
	$\pi_0+\pi_{0.5}$ -AFI (Ours)	<b>20/20</b>	<b>16/20</b>	<b>20/20</b>	<b>16/20</b>	<b>17/20</b>	<b>89.0% (↑25.0%)</b>

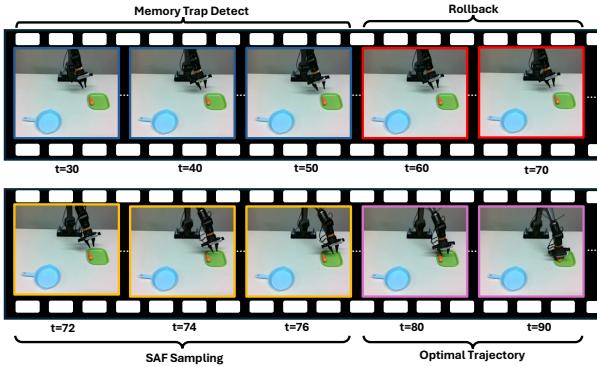


Figure 4. **Real-world AFI execution rollout.** Top: Memory trap detected at  $t = 50$  when approaching wrong location, followed by rollback to low-cost historical position. Bottom: SAF-guided sampling ( $t = 70\text{-}79$ ) generates trajectory candidates; optimal trajectory (green) is selected and executed ( $t = 80\text{-}90$ ) for successful completion.

*Pen*: inserting a pen into a holder; (4) *Stack Tape*: stacking one tape roll on top of another. Each task is evaluated over 20 trials under five test conditions: in-distribution (ID) and four OOD scenarios involving position shifts ( $\pm 5\text{-}15\text{cm}$ ), color/appearance changes, object property variations, and background shifts. Task-specific descriptions, training data details, and OOD scenario definitions are provided in Appendix B and Figure 7.

## 5.2. Main Results in Real World

Table 1 presents comprehensive results across four manipulation tasks and various distribution shift scenarios. Our AFI framework achieves consistent improvements across all tasks and VLA backbones, with average success rate gains ranging from 17.0% to 26.0% over baseline policies.

**Consistent Improvements Across Tasks.** Our method demonstrates robust performance gains across all manipu-

lation primitives. For *Place Carrot*,  $\pi_0$ -AFI achieves 87.0% average success rate versus 61.0% baseline, with particularly strong improvements in Position shift (65% vs 30%) and Background shift (95% vs 50%) scenarios. For *Stack Tape*, we observe 86.0% success rate with  $\pi_0$ -AFI compared to 64.0% baseline, achieving perfect success (100%) in Color shift scenarios. The *Slot Pen* task shows 82.0% success (22.0% gain), while *Remove Lid* achieves 80.0% (17.0% gain). These results validate that our SAF-guided intervention effectively handles diverse manipulation primitives, including *picking*, *placing*, *insertion*, *lid removal*, and *stacking* operations. Figure 4 visualizes a typical AFI execution rollout, demonstrating the complete intervention process from memory trap detection to successful task completion.

**Robustness to Challenging Distribution Shifts.** The Task shift scenarios present the most challenging OOD conditions, involving physical property variations and distractors. Our method shows substantial improvements: 30.0% gain for *Remove Lid* with distractor avoidance (from 25% to 55%), 20.0% gain for *Slot Pen* with thinner pen insertion (from 75% to 95%), and 20.0% gain for *Stack Tape* with different tape types (from 65% to 85%). Position shift scenarios also benefit significantly, with average improvements of 25.0% across tasks, demonstrating that our SAF provides effective 3D spatial reasoning to locate displaced objects.

**Comparison with Zero-Shot VLM Planner.** ReKep [13], a training-free VLM-based planner, achieves only 36.0% average success rate on the *Place Carrot* task. While VLMs excel at semantic understanding, they lack fine-grained motion planning capabilities required for precise manipulation. In contrast, our hybrid approach achieves 87.0% by leveraging VLMs’ zero-shot grounding abilities to construct SAFs as guidance while relying on VLA models for robust action generation, combining their complementary strengths.

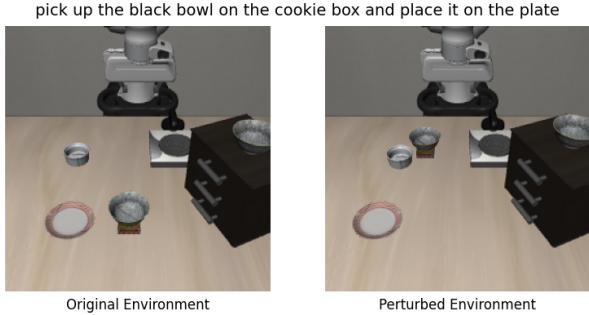


Figure 5. Visualization of object position perturbations in LIBERO simulation, where the target object ‘‘black bowl on the cookie box’’ is displaced to significantly deviated positions.

**Generalization Across VLA Backbones.** We evaluate both  $\pi_0$  and  $\pi_{0.5}$  on the *Stack Tape* task. Both achieve similar baseline performance (64.0% and 61.0%), and our AFI improves them to comparable levels (86.0% and 82.0%). This consistency validates the model-agnostic nature of our approach, demonstrating that our framework generalizes across different VLA architectures without requiring architecture-specific modifications.

**Ensemble Integration for Enhanced Performance.** We further explore ensemble integration by combining action proposals from both  $\pi_0$  and  $\pi_{0.5}$  backbones, where our SAF-based scorer selects the optimal trajectory from the combined candidate pool. As shown in Table 1, the ensemble approach ( $\pi_0 + \pi_{0.5}$ -AFI) achieves 89.0% average success rate on *Stack Tape*, outperforming both individual backbones ( $\pi_0$ -AFI: 86.0%,  $\pi_{0.5}$ -AFI: 82.0%) and representing a 25.0% improvement over baseline  $\pi_0$  (64.0%). This demonstrates a key advantage of our framework: its model-agnostic design naturally supports multi-policy integration, leveraging complementary strengths from different VLA architectures without complex fusion mechanisms.

### 5.3. Main Results in Simulation

For LIBERO simulation experiments, we evaluate our method on the LIBERO-Pro [40] benchmark by introducing spatial perturbations to target object positions in each subtask, following the OOD evaluation protocol from LIBERO-Pro. These perturbations are carefully designed to avoid rendering errors, prevent object collisions, and ensure no semantic contradictions with the task instructions [40]. Table 2 reports the success rates of our method compared to the baseline  $\pi_{0.5}$  model on the LIBERO-Spatial and LIBERO-Object suites, broken down by each subtask. Our SAF-guided framework demonstrates substantial improvements across most subtasks, particularly in scenarios involving spatial perturbations, achieving an average success rate of 78.2% on LIBERO-Spatial (vs. 52.4% for  $\pi_{0.5}$ ) and 82.5% on LIBERO-Object (vs. 67.3% for  $\pi_{0.5}$ ). These results underscore the effectiveness of affordance-guided

Table 2. Success rates of  $\pi_{0.5}$ -AFI and  $\pi_{0.5}$  on the LIBERO-Spatial and LIBERO-Object, with object position perturbations introduced to subtask following the LIBERO-Pro protocol.

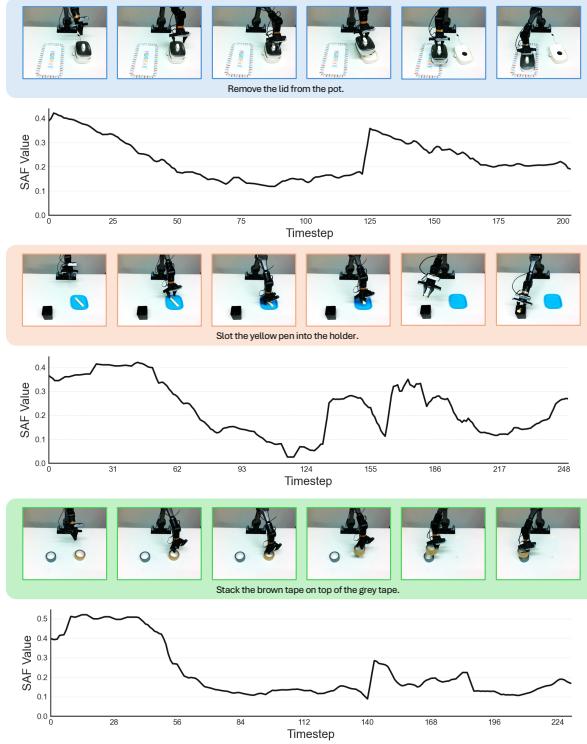
LIBERO-Spatial (OOD)		
Task	$\pi_{0.5}$	$\pi_{0.5}$ -AFI
Pick(between(plate, ramekin), plate)	70.0%	<b>82.0%</b>
Pick(next_to(ramekin), plate)	22.0%	<b>54.0%</b>
Pick(table_center, plate)	96.0%	<b>98.0%</b>
Pick(on(cookie_box), plate)	74.0%	<b>88.0%</b>
Pick(on(ramekin), plate)	26.0%	<b>54.0%</b>
Pick(next_to(cookie_box), plate)	36.0%	<b>72.0%</b>
Pick(next_to(plate), plate)	54.0%	<b>82.0%</b>
Average	54.0%	<b>75.7%</b>
LIBERO-Object (OOD)		
Task	$\pi_{0.5}$	$\pi_{0.5}$ -AFI
Place(alphabet_soup, basket)	42.0%	<b>64.0%</b>
Place(bbq_sauce, basket)	54.0%	<b>72.0%</b>
Place(butter, basket)	78.0%	<b>82.0%</b>
Place(chocolate_pudding, basket)	88.0%	<b>90.0%</b>
Place(cream_cheese, basket)	42.0%	<b>56.0%</b>
Place(ketchup, basket)	46.0%	<b>66.0%</b>
Place(milk, basket)	88.0%	<b>92.0%</b>
Place(orange_juice, basket)	70.0%	<b>80.0%</b>
Place(salad_dressing, basket)	16.0%	<b>64.0%</b>
Place(tomato_sauce, basket)	40.0%	<b>66.0%</b>
Average	56.4%	<b>73.2%</b>

interventions in enhancing VLA generalization to out-of-distribution object configurations and positional shifts in simulation environments.

**Computational Efficiency.** Our framework introduces minimal overhead to the baseline VLA inference. SAF reconstruction takes 120 ms per frame using Grounded-SAM and point cloud processing on an NVIDIA RTX 4090, while waypoint generation and action re-ranking add 15 ms total. This results in an end-to-end latency of 185 ms, suitable for 5 Hz control rates in real-world deployment. In contrast, pure optimization-based methods like MPC require 500+ ms per planning step, limiting their applicability to high-frequency manipulation.

### 5.4. Ablations

**SAF adaptation throughout task execution.** To validate that our SAF accurately reflects task-relevant spatial information, we analyze its cost evolution throughout complete trajectories (Figure 6). The curves demonstrate that SAF cost decreases as the end-effector approaches target objects, validating that our affordance field accurately reflects spatial proximity to manipulation goals. Notably, the cost values exhibit dynamic updates when transitioning between manipulation stages (e.g., from picking the lid to placing it in the platter), showing that our system adaptively adjusts the affordance field based on current task semantics. This



**Figure 6. SAF value evolution across manipulation tasks.** Lower cost values indicate higher affordance. Costs decrease when approaching targets and update dynamically during stage transitions (e.g., picking to placing).

stage-aware adaptation is crucial for multi-step manipulation tasks, ensuring that the spatial guidance remains relevant throughout the entire execution sequence.

**Robustness to position shifts along different axes.** To investigate VLA models’ robustness to spatial perturbations, we systematically shift the target object along X-axis and Y-axis independently and jointly. Table 3 presents results across six position configurations.  $\pi_0$ ’s performance degrades catastrophically under single-axis shifts: success rates drop to 15% for  $\Delta X = +10$  cm and 5% for  $\Delta X = +15$  cm, revealing that VLA models memorize specific spatial patterns and fail when deviating along unseen directions. Interestingly, diagonal displacement ( $\Delta X = +10$  cm,  $\Delta Y = +10$  cm) maintains 30% success, likely because such trajectories align with the training distribution’s spatial coverage. Our method consistently improves robustness across all configurations (40% for single-axis, 65% for diagonal), validating that explicit 3D spatial guidance helps escape memory traps. However, extreme OOD scenarios ( $\Delta X/Y = +15$  cm) show diminishing returns, highlighting that our approach complements rather than replaces the VLA’s learned priors.

**Ablation on key components.** Table 4 validates the importance of individual components on the position shift scenario. Removing rollback degrades performance from 65%

**Table 3. Ablation study on position shifts.** We evaluate  $\pi_0$  and  $\pi_0$ -AFI under different spatial displacements along X and Y axes (in centimeters).

$(\Delta X, \Delta Y)$	(0,0)	(+10,0)	(+15,0)	(0,+10)	(0,+15)	(+10,+10)
$\pi_0$	17/20	3/20	1/20	4/20	0/20	6/20
$\pi_0$ -AFI	<b>20/20</b>	<b>8/20</b>	<b>3/20</b>	<b>10/20</b>	<b>2/20</b>	<b>13/20</b>

**Table 4. Ablation study on key components:** (1) rollback mechanism and (2) adaptive detection vs. fixed-step intervention. All experiments on position shift scenario over 20 trials.

Method	Success	Failure
$\pi_0$	6/20	14/20
$\pi_0$ -AFI	<b>13/20</b>	7/20
w/o Rollback	8/20	12/20
Fixed-step at 30	12/20	8/20
Fixed-step at 60	11/20	9/20
Fixed-step at 90	9/20	11/20

**Table 5. Ablation on waypoint count with different numbers of SAF-sampled candidates.**

Num of Waypoints	3	8	10	13
Success Rate	35.0%	50.0%	65.0%	60.0%

to 40%, demonstrating its critical role. Without it, the VLA deviates too far from viable trajectories, and SAF-guided waypoints cannot recover from failed positions. Rollback repositions the end-effector to a recent high-affordance state, providing a safe starting point for effective redirection. Fixed-step interventions achieve at most 60% (Step 30), underperforming our adaptive detection (65%). This validates that real-time proprioceptive monitoring enables flexible intervention precisely when memory traps occur, optimizing both efficiency and success rate.

**Impact of waypoint proposal count.** Table 5 examines how the number of waypoint proposals affects performance. With only 3 proposals, success rate remains low at 35%, barely improving over the baseline. Optimal performance is achieved at 10 proposals (65%), balancing spatial exploration and computational efficiency. Further increasing to 13 proposals shows marginal degradation (60%), possibly due to over-exploration introducing suboptimal waypoints. These results highlight the importance of balanced sampling for effective coverage of high-affordance regions.

## 6. Conclusion

We identified the “Memory Trap” problem where VLAs rigidly reproduce memorized trajectories under distribution shifts. To address this, we proposed Affordance Field Intervention (AFI), which augments VLA models with explicit 3D spatial reasoning via Spatial Affordance Fields. AFI operates through proprioceptive memory trap detection, guided rollback to high-affordance positions, and hierarchical trajectory exploration via SAF-guided waypoint sampling. By treating SAF as an on-demand plug-in without

modifying VLA parameters, our method is model-agnostic and applicable to any pre-trained VLA backbone. Extensive experiments demonstrate significant improvements: 23.5% average gain on real-world OOD scenarios and 20.2% on LIBERO-Pro benchmark. The training-free nature makes it practical for deployment without additional data or fine-tuning. Our work shows that combining VLA policies with interpretable 3D spatial affordance fields offers a promising path toward more robust and generalizable robotic manipulation systems.

## References

- [1] 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. [2](#) [3](#)
- [2] Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. Data quality in imitation learning. *ArXiv*, abs/2306.02437, 2023. [2](#)
- [3] Marc G Bellemare, Salvatore Candido, Pablo Samuel Castro, Jun Gong, Marlos C Machado, Subhodeep Moitra, Sameera S Ponda, and Ziyu Wang. Autonomous navigation of stratospheric balloons using reinforcement learning. *Nature*, 588(7836):77–82, 2020. [2](#)
- [4] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolò Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al.  $\pi_0$ : A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024. [2](#) [5](#)
- [5] 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, 2023. URL <https://arxiv.org/abs/2307.15818>, 2024. [1](#) [2](#)
- [6] Irving Fang, Juexiao Zhang, Shengbang Tong, and Chen Feng. From intention to execution: Probing the generalization boundaries of vision-language-action models. *arXiv preprint arXiv:2506.09930*, 2025. [2](#)
- [7] Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinze He, Shiduo Zhang, Zhaoye Fei, et al. Libero-plus: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025. [2](#)
- [8] Roya Firooz, Johnathan Tucker, Stephen Tian, Anirudha Majumdar, Jiankai Sun, Weiyu Liu, Yuke Zhu, Shuran Song, Ashish Kapoor, Karol Hausman, et al. Foundation models in robotics: Applications, challenges, and the future. *The International Journal of Robotics Research*, 44(5):701–739, 2025. [1](#) [2](#)
- [9] Dylan J. Foster, Adam Block, and Dipendra Kumar Misra. Is behavior cloning all you need? understanding horizon in imitation learning. *ArXiv*, abs/2407.15007, 2024. [2](#)
- [10] James J Gibson. The ecological approach to the visual perception of pictures. *Leonardo*, 11(3):227–235, 1978. [2](#)
- [11] Shresth Grover, Akshay Gopalakrishnan, Bo Ai, Henrik I Christensen, Hao Su, and Xuanlin Li. Enhancing generalization in vision-language-action models by preserving pre-trained representations. *arXiv preprint arXiv:2509.11417*, 2025. [2](#)
- [12] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023. [2](#) [3](#)
- [13] Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. *arXiv preprint arXiv:2409.01652*, 2024. [2](#) [5](#) [6](#)
- [14] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolò Fusai, et al.  $\pi_{0.5}$ : a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025. [2](#) [5](#)
- [15] Mazeyu Ji, Ri-Zhao Qiu, Xueyan Zou, and Xiaolong Wang. Graspsplats: Efficient manipulation with 3d feature splatting. *arXiv preprint arXiv:2409.02084*, 2024. [2](#)
- [16] Anqing Jiang, Yu Gao, Yiru Wang, Zhigang Sun, Shuo Wang, Yuwen Heng, Hao Sun, Shichen Tang, Lijuan Zhu, Jinzhao Chai, et al. Irl-vla: Training an vision-language-action policy via reward world model. *arXiv preprint arXiv:2508.06571*, 2025. [3](#)
- [17] Zhenyu Jiang, Yifeng Zhu, Maxwell Svetlik, Kuan Fang, and Yuke Zhu. Synergies between affordance and geometry: 6-dof grasp detection via implicit representations. *arXiv preprint arXiv:2104.01542*, 2021. [3](#)
- [18] Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. Ditto: Building digital twins of articulated objects from interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5616–5626, 2022. [3](#)
- [19] Yuanchen Ju, Kaizhe Hu, Guowei Zhang, Gu Zhang, Mingrui Jiang, and Huazhe Xu. Robo-abc: Affordance generalization beyond categories via semantic correspondence for robot manipulation. In *European Conference on Computer Vision*, pages 222–239. Springer, 2024. [2](#)
- [20] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024. [2](#)
- [21] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024. [1](#) [2](#)
- [22] Chengmeng Li, Junjie Wen, Yan Peng, Yixin Peng, Feifei Feng, and Yichen Zhu. Pointvla: Injecting the 3d world into vision-language-action models. *arXiv preprint arXiv:2503.07511*, 2025. [2](#)
- [23] Hengtao Li, Pengxiang Ding, Runze Suo, Yihao Wang, Zirui Ge, Dongyuan Zang, Kexian Yu, Mingyang Sun, Hongyin Zhang, Donglin Wang, et al. Vla-rft: Vision-language-action reinforcement fine-tuning with verified rewards in world simulators. *arXiv preprint arXiv:2510.00406*, 2025. [3](#)

- [24] Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaojun 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. 3
- [25] Mingrui Li, Zhetao Guo, Tianchen Deng, Yiming Zhou, Yuxiang Ren, and Hongyu Wang. Ddn-slam: Real time dense dynamic neural implicit slam. *IEEE Robotics and Automation Letters*, 2025. 2
- [26] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. *Advances in Neural Information Processing Systems*, 36:44776–44791, 2023. 5
- [27] Jijia Liu, Feng Gao, Bingwen Wei, Xinlei Chen, Qingmin Liao, Yi Wu, Chao Yu, and Yu Wang. What can rl bring to vla generalization? an empirical study. *arXiv preprint arXiv:2505.19789*, 2025. 2, 3
- [28] Guanxing Lu, Wenkai Guo, Chubin Zhang, Yuheng Zhou, Haonan Jiang, Zifeng Gao, Yansong Tang, and Ziwei Wang. Vla-rl: Towards masterful and general robotic manipulation with scalable reinforcement learning. *arXiv preprint arXiv:2505.18719*, 2025. 3
- [29] Tianhe Ren, Shilong Liu, Ailing Zeng, Jing Lin, Kun-chang Li, He Cao, Jiayu Chen, Xinyu Huang, Yukang Chen, Feng Yan, Zhaoyang Zeng, Hao Zhang, Feng Li, Jie Yang, Hongyang Li, Qing Jiang, and Lei Zhang. Grounded sam: Assembling open-world models for diverse visual tasks, 2024. 3, 11
- [30] Radu Bogdan Rusu, Gary Bradski, Romain Thibaux, and John Hsu. Fast 3d recognition and pose using the viewpoint feature histogram. In *2010 IEEE/RSJ international conference on intelligent robots and systems*, pages 2155–2162. IEEE, 2010. 2
- [31] Anthony Simeonov, Yilun Du, Andrea Tagliasacchi, Joshua B Tenenbaum, Alberto Rodriguez, Pulkit Agrawal, and Vincent Sitzmann. Neural descriptor fields: Se (3)-equivariant object representations for manipulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6394–6400. IEEE, 2022. 2
- [32] Lin Sun, Bin Xie, Yingfei Liu, Hao Shi, Tiancai Wang, and Jiale Cao. Geovla: Empowering 3d representations in vision-language-action models. *arXiv preprint arXiv:2508.09071*, 2025. 2
- [33] Weiliang Tang, Jia-Hui Pan, Yun-Hui Liu, Masayoshi Tomizuka, Li Erran Li, Chi-Wing Fu, and Mingyu Ding. Geomanip: Geometric constraints as general interfaces for robot manipulation. *arXiv preprint arXiv:2501.09783*, 2025. 3
- [34] Marc Toussaint. Logic-geometric programming: An optimization-based approach to combined task and motion planning. In *IJCAI*, pages 1930–1936, 2015. 2
- [35] Yuhang Yang, Wei Zhai, Hongchen Luo, Yang Cao, Jiebo Luo, and Zheng-Jun Zha. Grounding 3d object affordance from 2d interactions in images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10905–10915, 2023. 2
- [36] Angen Ye, Zeyu Zhang, Boyuan Wang, Xiaofeng Wang, Dapeng Zhang, and Zheng Zhu. Vla-rl: Enhancing reasoning in vision-language-action models. *arXiv preprint arXiv:2510.01623*, 2025. 3
- [37] Yu Yue, Yufeng Yuan, Qiying Yu, Xiaochen Zuo, Ruofei Zhu, Wenyuan Xu, Jiaze Chen, Chengyi Wang, TianTian Fan, Zhengyin Du, et al. Vapo: Efficient and reliable reinforcement learning for advanced reasoning tasks. *arXiv preprint arXiv:2504.05118*, 2025. 2
- [38] Hongzhi Zang, Mingjie Wei, Si Xu, Yongji Wu, Zhen Guo, Yuanqing Wang, Hao Lin, Liangzhi Shi, Yuqing Xie, Zhexuan Xu, et al. Rlinf-vla: A unified and efficient framework for vla+ rl training. *arXiv preprint arXiv:2510.06710*, 2025. 3
- [39] Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and Chuang Gan. 3d-vla: A 3d vision-language-action generative world model. *arXiv preprint arXiv:2403.09631*, 2024. 2
- [40] Xueyang Zhou, Yangming Xu, Guiyao Tie, Yongchao Chen, Guowen Zhang, Duanfeng Chu, Pan Zhou, and Lichao Sun. Libero-pro: Towards robust and fair evaluation of vision-language-action models beyond memorization. *arXiv preprint arXiv:2510.03827*, 2025. 2, 5, 7

# Affordance Field Intervention: Enabling VLAs to Escape Memory Traps in Robotic Manipulation

## Supplementary Material

### A. Implementation Details

**Hardware Configuration.** Our real-world experiment setup employs an *AgileX Piper manipulator* equipped with two *Intel RealSense D435 cameras*: one mounted on the wrist and another positioned in front of the robot. Both cameras are calibrated relative to the robot base frame to enable accurate 3D point cloud reconstruction. We utilize the calibrated front-mounted RealSense camera for scene reconstruction, operating at 30 Hz observation frequency. During data collection, we utilize RGB images from both viewpoints, while depth information is exclusively used at inference time to construct 3D point clouds for spatial affordance field generation.

**Spatial Affordance Field Construction.** For affordance field construction, we apply the GPT-4o API to parse task instructions and identify manipulation stages. Open-vocabulary detection and target object tracking are performed locally on an NVIDIA GeForce GTX 1080Ti GPU using Grounded-SAM [29] to generate 2D instance segmentation masks. The complete SAF is published and updated as a ROS topic at 2 Hz frequency, ensuring real-time spatial reasoning without introducing latency to the original VLA inference pipeline.

**Control and Kinematics.** Additionally, we deploy Curobo on the same 1080Ti GPU for forward and inverse kinematics computation. This enables bidirectional transformation: converting VLA-predicted joint states to end-effector spatial coordinates, and mapping SAF-sampled waypoints back to joint configurations. The kinematics computation introduces approximately 5ms latency and operates as a ROS service at 10Hz frequency. Overall, our SAF updates at 2Hz without adding overhead to the VLA policy inference pipeline, maintaining efficient real-time control.

**Training Details.** For data collection, we use a master-follower teleoperation setup with an auxiliary AgileX Piper arm. The baseline VLA models ( $\pi_0$  and  $\pi_{0.5}$ ) are fully fine-tuned on collected demonstration trajectories for 30,000 steps with batch size 32 on a single NVIDIA H100 GPU. During inference, we sample 8 action chunks per query to ensure trajectory diversity. For stochastic action generation via flow matching, we set the initial noise sampling temperature (standard deviation) to 1.5 to encourage diverse proposals, which are then re-ranked by our SAF-based scorer.

### B. Real-world Task Settings

We evaluate our framework on four real-world manipulation tasks with varying complexity, each designed to test different manipulation primitives and robustness to distribution shifts. Each task is evaluated over 20 trials under five test conditions: in-distribution (ID) and four OOD scenarios (illustrated in Figure 7).

#### B.1. Task Description and Collection

**Task 1 - Place Carrot.** The instruction is “Pick up the carrot and place it in the pot.” The objective is to pick up a carrot from a plate and place it into a pot. Success is defined as successfully placing the carrot into the pot. We collect 68 expert demonstration trajectories using a master-follower teleoperation setup.

**Task 2 - Remove Lid.** The instruction is “Remove the lid from the pot.” The objective is to remove the stainless steel lid from a pot and place it onto a nearby platter. Success is defined as placing the lid completely within the platter boundaries. We collect 78 expert demonstration trajectories with the pot position fixed. The platter is divided into two regions (A and B), and the lid placement is distributed across both regions, with 39 trajectories collected for each region.

**Task 3 - Slot Pen.** The instruction is “Slot the yellow pen into the holder.” The task requires inserting a yellow marker pen into a pen holder. Success is defined as the pen being fully inserted into the holder. We collect 77 expert demonstration trajectories for this task.

**Task 4 - Stack Tape.** The instruction is “Stack the brown tape on top of the grey tape.” This task involves placing a roll of brown tape on top of a roll of grey tape. Success is defined as the brown tape resting stably on the grey tape without falling. We collect 80 expert demonstration trajectories for this task.

#### B.2. Out-of-Distribution Test Scenarios

Following the evaluation protocol in our main experiments, we design four types of distribution shifts for each task. Figure 7 provides a visual illustration of these OOD test scenarios across all four tasks:

**Position Shift.** Objects are displaced within a 5-15cm radius from their training positions. For Place Carrot, the carrot is displaced by approximately 15 cm; for Remove Lid, we move the pot within 5cm; for Slot Pen, we move the blue plate containing the pen within 5cm; for Stack Tape, we move the tape placement area within 5cm.

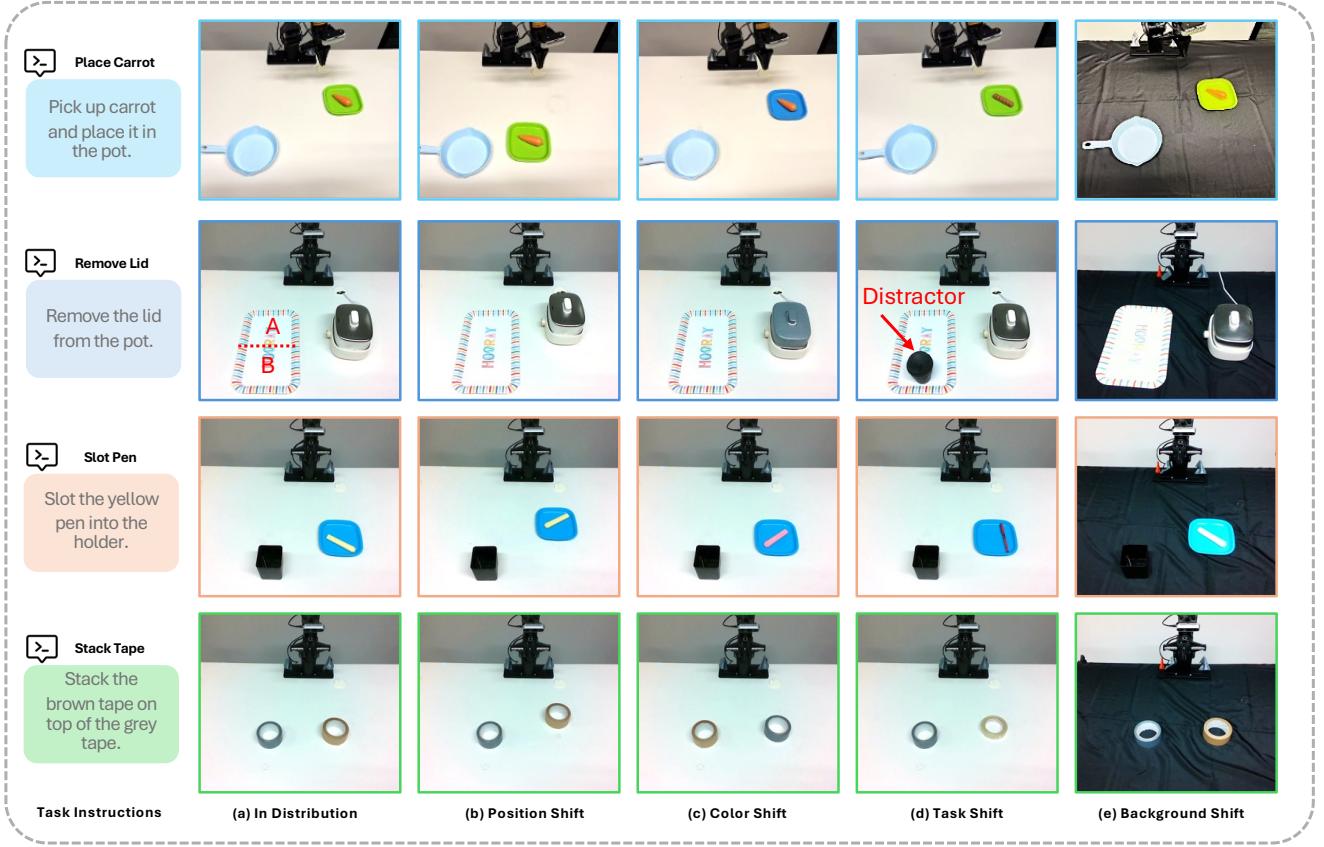


Figure 7. **Visual illustration of OOD test scenarios.** Four manipulation tasks across five test conditions: (a) in-distribution setting, (b) position shift (objects displaced by  $\pm 5\text{-}15\text{cm}$ ), (c) color shift (object appearance change), (d) task shift (physical property variations or distractors), and (e) background shift (table surface color change from white to black). Each row shows a different task: Place Carrot (top), Remove Lid (second), Slot Pen (third), and Stack Tape (bottom).

**Color Shift.** For Place Carrot, the plate holding the carrot is changed from green to blue. For Remove Lid, the stainless steel lid is replaced with a grey-colored lid. For Slot Pen, the yellow marker is replaced with a pink one. For Stack Tape, the brown tape is replaced with a grey-colored tape.

**Task Shift.** For Place Carrot, the target object is replaced from carrot to sausage while maintaining the same task structure. For Remove Lid, we add a black cup as a distractor in the platter region and extend the instruction with the phrase “Avoid the black cup.” For Slot Pen, the yellow marker is replaced with a thinner red pen, requiring different insertion dynamics. For Stack Tape, the brown tape is replaced with a different type of tape (different shape and texture).

**Background Shift.** The table surface color is changed from white to black across all tasks.