UNCOVERING AND MITIGATING POSITIONAL BLIND SPOTS IN VISION-LANGUAGE-ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Recent Vision-Language-Action (VLA) models achieve promising performance in robotic manipulation, typically measured by success rates aggregated over predefined object configurations, an evaluation that implicitly assumes spatially uniform competence across the workspace. However, this assumption does not hold: even with the instruction and every other scene factor held fixed, merely relocating a task-irrelevant distractor can sharply raise the failure probability within localized, spatially coherent regions, which we term *Positional Blind Spots* (PBS). In this paper, we propose a two-stage black-box framework to uncover and mitigate PBS. During the uncovering stage, we grid the workspace and apply a one-sided log-likelihood-ratio test to localize PBS cells with significantly elevated risk. During the mitigation stage, we fine-tune the policy via LoRA on demonstrations collected from these PBS regions, improving competence there while largely preserving performance across the rest of the workspace. We evaluate our framework on five state-of-the-art VLA policies across two benchmarks, and find that PBS are pervasive and spatially concentrated in all of them, with failure rates up to 0.58. Our search strategy achieves an average F1-score of 0.678, outperforming random search and adaptive sampling baselines by 0.268 and 0.178, respectively. Guided by the discovered regions, targeted fine-tuning reduces the overall failure rate by 40.00%–85.19%.

1 INTRODUCTION

Enabling robots to follow natural language instructions from raw visual input has long been a central goal of robotic manipulation, and Vision-Language-Action (VLA) models have emerged as the leading end-to-end approach toward this goal by mapping visual observations and language instructions directly to executable control actions Ma et al. (2026); Li et al. (2026a). Built upon pre-trained foundation models and fine-tuned on large-scale robot demonstrations, VLA models integrate perception, reasoning, and action within a unified architecture, and have reported impressive success rates across a wide range of manipulation benchmarks Zhang et al. (2025a); Xu et al. (2026). Beyond manipulation, the VLA paradigm is rapidly extending to broader embodied domains, such as autonomous driving Yang et al. (2026); Jiang et al. (2025) and aerial navigation Chen et al. (2026). In these scenarios, the failure cost is no longer a benchmark statistic: an unexpected error may interrupt the task, damage objects, or even endanger humans. Therefore, reliability has become a prerequisite, rather than an afterthought, for bringing VLA models into the physical world.

To assess the reliability of VLA models, existing evaluations mainly rely on predefined benchmarks that report average success rates over fixed tasks and environments Liu et al. (2023); Zhang et al. (2025a); Fan et al. (2026). Recent studies further investigate model robustness under various input variations, such as changes in object appearance, language instructions, or scene configurations Wang et al. (2025b); Valle et al. (2026); Wang et al. (2025a); Yan et al. (2025). However, these evaluations largely overlook a fundamental factor in embodied tasks: object position. Since deployed objects can appear anywhere in the workspace rather than at the positions covered by training or evaluation, this gap raises a fundamental question left unanswered: does a VLA model's competence hold uniformly across the workspace, or collapse in specific regions, a phenomenon we term *Positional Blind Spots (PBS)*.

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

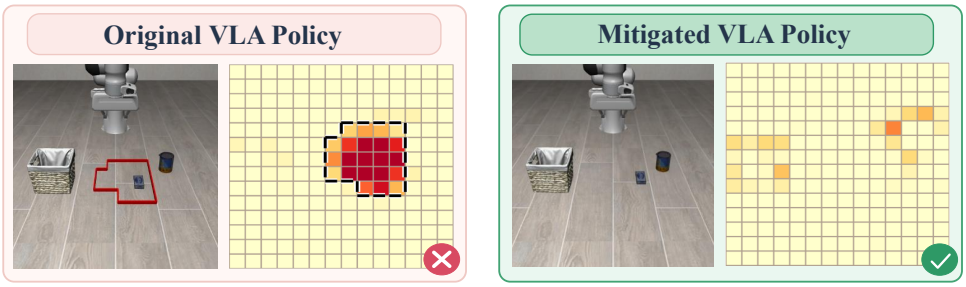


Figure 1: Illustration of positional blind spots (PBS) in VLA policies and PBS-guided mitigation. Heatmaps show failure rates by distractor position, revealing localized positional vulnerabilities.

It raises a question that can PBS be systematically discovered. Identifying such spatial failures is challenging due to the continuous nature and large search space of the robot workspace. Unlike conventional testing where the input space can often be enumerated or discretized, the possible object positions in embodied tasks form a continuous space with infinitely many possible configurations. Meanwhile, PBS may only occupy a small and localized portion of the workspace, making them difficult to reveal through sparse or uniform sampling. Moreover, evaluating a VLA policy typically requires executing complete task rollouts and only provides binary success or failure feedback, without gradients, confidence scores, or internal representations to guide the search. Consequently, exhaustive evaluation over fine-grained spatial regions is prohibitively expensive, while limited evaluations may easily miss critical blind spots. This raises a fundamental challenge: how to efficiently explore the continuous workspace under a limited evaluation budget and reliably identify high-risk regions where VLA models exhibit positional failures.

In this work, we propose a two-stage framework that uncovers and mitigates PBS in VLA policies without accessing their internals. During the uncovering stage, we address the challenge of searching over a continuous spatial space by discretizing the feasible object-placement workspace into a finite set of spatial regions. Specifically, we partition the workspace into a spatial grid, sample rollout points uniformly within each cell, and apply a Log-Likelihood Ratio (LLR) test to identify cells whose failure rates are significantly higher than the overall workspace distribution. These cells are returned as the predicted blind-spot regions. During the mitigation stage, human experts teleoperate the robot arm to collect additional successful demonstrations within these identified regions. The policy is then fine-tuned via LoRA following its original adaptation procedure, without modifying its observation, action, or optimization interface. In this way, the framework uses PBS-guided demonstrations to improve spatial reliability while preserving the original policy interface.

We evaluate PBS characterization, search, and mitigation on five representative VLA policies (π_0, $\pi_{0.5}$, OpenVLA-OFT, UniVLA, and VLA-Adapter) across two benchmarks (LIBERO and VLA-Arena). Exhaustive spatial evaluation confirms that PBS are pervasive: after introducing a single task-irrelevant distractor, every evaluated policy exhibits a failure rate ranging from 0.15 to 0.58, with failures spatially concentrated rather than uniformly distributed across the workspace. Under the same search budget, our search strategy recovers these regions substantially more accurately than random and adaptive sampling baselines, improving the average F1-score by 0.268 and 0.178, respectively.

In summary, our main contributions are as follows:

- We identify and formalize *Positional Blind Spots* (PBS), a previously unexplored spatial vulnerability of VLA policies where the presence of task-irrelevant objects at specific workspace positions induces systematic failures.
- We propose a two-stage black-box framework that uncovers PBS through spatial discretization and log-likelihood-ratio filtering, and mitigates them through search-guided policy fine-tuning, without modifying the policy architecture or interfaces.
- Through exhaustive spatial evaluations on five VLA policies across two benchmarks, we show that PBS consistently emerge across the evaluated settings, with failure rates up to 58%, and that targeted mitigation reduces the overall failure rate by up to 85%.

2 Related Work

Vision-Language-Action (VLA) models. VLA models map visual observations and language instructions to robot actions by adapting pretrained vision-language representations with robot demonstrations. RT-2 Zitkovich et al. (2023) casts actions as language-like tokens, while OpenVLA Kim et al. (2025b) scales this paradigm with diverse robot data and efficient downstream adaptation. Recent models adopt diverse action-generation and adaptation mechanisms: π_0 Black et al. (2025) uses a flow-matching action expert, $\pi_{0.5}$ Kevin et al. (2025) introduces heterogeneous co-training, OpenVLA-OFT Kim et al. (2025a) combines continuous actions with action chunking and parallel decoding, UniVLA Bu et al. (2025) learns transferable task-centric representations, and VLA-Adapter Wang et al. (2026) connects vision-language features to action prediction through a lightweight adapter. Recent work also explores explicit spatial representations. SpatialVLA Qu et al. (2025) injects 3D information into visual observations through Ego3D Position Encoding and uses Adaptive Action Grids to represent robot motion. Despite rapid progress in model architectures, adaptation strategies and spatial representation, ensuring robust and reliable behavior of VLA policies under diverse real-world variations remains an open challenge. In real-world manipulation scenarios, objects and environmental conditions may vary continuously, requiring VLA policies to maintain consistent performance beyond the configurations observed during training.

Evaluation and enhancement of VLA models. Benchmarks such as LIBERO Liu et al. (2023), VLABench Zhang et al. (2025c) VLA-Arena Zhang et al. (2026) and LIBERO-Plus Fei et al. (2026) assess transfer, semantic, long-horizon manipulationan and robustness under structured task, language, and visual variations. These benchmarks evaluates diverse perturbation factors whereas we isolate the effect of task-irrelevant distractor position and use cell-wise risk estimates to guide targeted adaptation. Beyond benchmark-based evaluation, active testing methods further expose failures through scene mutations or semantics-preserving transformations Wang et al. (2025b); Valle et al. (2026), while adversarial studies employ textual, visual, and cross-modal perturbations Jones et al. (2025); Wang et al. (2025a); Yan et al. (2025). However, these approaches either evaluate performance over predefined task distributions, modify multiple factors simultaneously, or introduce intentionally crafted perturbations; they do not explicitly isolate the effect of task-irrelevant object relocation on VLA policy competence. Prior work addresses VLA reliability through safety alignment Zhang et al. (2025b) and failure interception Gu et al. (2025), while robustness studies also examine physical sensor attacks Lu et al. (2026). SimpleVLA-RL studies online VLA policy optimization using task-outcome feedback Li et al. (2026b). However, these works do not directly connect localized positional diagnosis with targeted data collection and policy adaptation. Unlike prior work, we investigate spatially concentrated positional vulnerabilities through black-box rollouts and use the discovered regions to guide targeted demonstration collection and policy refinement.

3 Preliminaries

3.1 Motivation

We first examine the positional sensitivity of VLA policies through a concrete example. As shown in Figure 2(a), the policy successfully completes the task when a task-irrelevant distractor is placed at one feasible position. However, relocating the same distractor to another feasible position causes the policy to fail. Across the two executions, the task objective, language instruction, task-relevant object configuration, robot initialization, camera view, and distractor attributes remain unchanged. Therefore, the presence of the distractor alone cannot explain the failure; instead, the behavioral difference is associated with the spatial position of the distractor in the workspace.

We further examine distractor placements across the feasible workspace to determine whether this positional sensitivity occurs only at isolated coordinates or forms spatially structured regions. Figure 2(b) shows that positions where the distractor induces elevated failure rates form a localized and spatially coherent region, with neighboring placements exhibiting similar failure risks. A policy may therefore remain reliable when the distractor is placed in most regions of the workspace while exhibiting substantially higher failure probability when the same distractor is placed within a small subset of positions. Such localized vulnerabilities can be obscured by aggregate task-level success rates, which average performance across low- and high-risk locations, despite their practical relevance when task-irrelevant objects appear at diverse feasible positions during deployment.

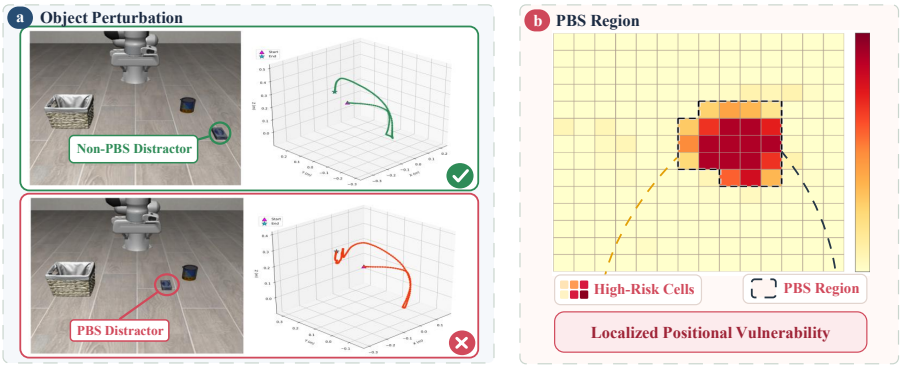


Figure 2: Impact of distractor position on VLA task success.

We term these localized regions of elevated failure probability as *Positional Blind Spots* (PBS). This observation raises two practical questions: how can PBS be uncovered under a black-box setting with limited rollout budgets, and how can the discovered regions guide targeted policy refinement? The next subsection formalizes this problem.

3.2 Problem Definition

3.2.1 VLA task execution.

Consider a manipulation task $\mathcal{T}$ with language instruction ℓ, a task-relevant target object x^{tar}, and a task-irrelevant distractor object x^{dis}, together with a task-adapted VLA policy π_θ. The policy is expected to manipulate x^{tar} according to ℓ, while ignoring x^{dis}. Let $\Omega \subset \mathbb{R}^2$ denote the feasible placement region of the task-irrelevant distractor. For each position $z \in \Omega$, we initialize the environment by placing the distractor at z, while keeping the task objective, language instruction, task-relevant object configuration, robot initialization, camera configuration, and distractor attributes unchanged. During execution, the policy receives visual observations o_t and predicts actions conditioned on ℓ:$a_t \sim \pi_\theta(\cdot \mid o_t, \ell)$. The generated actions induce a trajectory $\tau(z)$. We define the binary execution outcome as $Y_\theta(z) \in \{0, 1\}$, where $Y_\theta(z) = 1$ indicates task failure and $Y_\theta(z) = 0$ indicates successful completion. The position-conditioned failure probability is then defined as

$$p_\theta(z) = P(Y_\theta(z) = 1 \mid z). \tag{1}$$

Thus, $p_\theta(z)$ captures the failure risk induced by placing the same distractor at position z, while controlling other task and scene factors. A high value of $p_\theta(z)$ indicates that the distractor placement at z substantially increases the likelihood of task failure compared with other feasible placements.

3.2.2 Positional Blind Spots (PBS).

Based on the position-conditioned failure probability $p_\theta(z)$ of a VLA policy, we define *Positional Blind Spots* (PBS) as spatial regions where placing the same task-irrelevant distractor leads to substantially higher failure risk than the average failure risk over the workspace. Let μ denote a reference distribution over feasible distractor positions, instantiated as the uniform distribution over Ω. The workspace-level failure probability is defined as $\bar{p}_\theta = \mathbb{E}_{z\sim\mu}[p_\theta(z)]$. Given a risk margin $\delta > 0$, we can define the theoretical PBS set as

$$\mathcal{B}_\theta(\delta) = \{z \in \Omega \mid p_\theta(z) \geq \bar{p}_\theta + \delta\}. \tag{2}$$

Each spatially connected component of $\mathcal{B}_\theta(\delta)$ corresponds to a Positional Blind Spot (PBS), representing a localized workspace region where the task-irrelevant distractor induces substantially higher failure risk than the average risk over the workspace. This definition characterizes relative excess failure risk and requires no access to policy gradients, confidence scores, attention maps, or other internal model signals. In practice, $\mathcal{B}_\theta(\delta)$ is unknown and must be estimated through black-box rollouts under a limited evaluation budget.

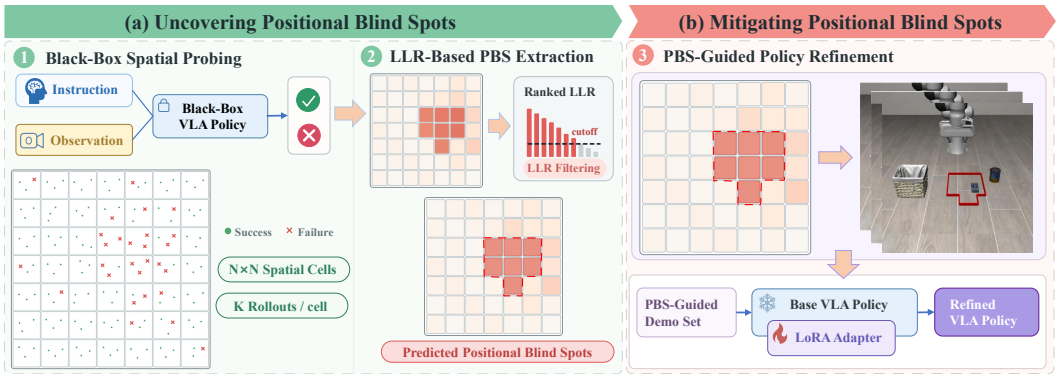


Figure 3: Pipeline of the proposed framework for uncovering and mitigating Positional Blind Spots.

3.2.3 PBS UNCOVERING AND MITIGATION PROBLEM.

Given that the true PBS region $\mathcal{B}_\theta(\delta)$ is unknown, the PBS uncovering problem aims to identify an estimated region $\widehat{\mathcal{B}}$ using only black-box rollout outcomes under a limited evaluation budget Q, such that $\widehat{\mathcal{B}}$ closely approximates $\mathcal{B}_\theta(\delta)$ through efficient allocation of rollouts to informative positions. Based on the discovered region $\hat{\mathcal{B}}$, the PBS mitigation problem aims to collect successful demonstrations at positions within $\hat{\mathcal{B}}$ and adapt the VLA policy to reduce failure risks within PBS regions while preserving performance on non-PBS regions. This formulation decomposes the PBS problem into two stages: identifying spatial regions with elevated failure risks and improving policy performance within those regions. The following methodology instantiates these two stages.

4 METHODOLOGY

4.1 OVERVIEW

Based on the problem definition, the distractor position is the only scene variable systematically changed, while the target object configuration, task objective, and VLA policy remain fixed. During evaluation, the policy is accessed solely through binary rollout outcomes. As illustrated in Figure 3, our framework consists of two major stages: PBS uncovering and PBS-guided mitigation. In the uncovering stage, we partition the feasible placement region into spatial cells and perform black-box rollouts by uniformly sampling positions within each cell. The collected rollout outcomes are then analyzed using an LLR-based extraction procedure to rank spatial cells according to their failure risk and identify the predicted PBS regions. In the mitigation stage, we collect successful demonstrations from the discovered PBS regions and refine the VLA policy through LoRA-based adaptation, yielding a more robust policy against positional failures.

4.2 UNCOVERING POSITIONAL BLIND SPOTS

The uncovering stage identifies spatial regions where the placement of the task-irrelevant distractor object x^{dis} at position z increases the failure risk of manipulating the target object x^{tar}. We consider a black-box setting in which the policy is queried only through complete rollouts, and only binary success or failure outcomes are observed. We formulate this process as a budgeted spatial search based on uniform sampling and statistical risk comparison. Let $\Omega \subset \mathbb{R}^2$ denote the feasible distractor-placement region. We discretize Ω into an $N \times N$ uniform grid containing N^2 cells, denoted as $\{g_i\}_{i=1}^{N^2}$, such that

$$\Omega = \bigcup_{i=1}^{N^2} g_i, \qquad g_i \cap g_j = \emptyset, \quad \forall i \neq j. \tag{3}$$

For each cell g_i, we uniformly sample K valid positions and execute one rollout at each position, resulting in a total budget of $Q = N^2 K$ policy executions. This discretization enables systematic

coverage of the placement region while retaining sufficient spatial resolution to localize concentrated failures. All task and scene factors, including the target object configuration, language instruction, and robot initialization, remain fixed except for the distractor position. Let c_i denote the number of failures observed in cell g_i, and let its empirical failure probability be $\hat{p}_i = c_i/K$, which provides a local estimate of failure risk for subsequent PBS identification.

The uniform allocation of rollouts across cells enables comparable estimation of failure risk across cells, as each cell is evaluated with the same number of trials. However, empirical failure probability alone cannot distinguish a localized positional vulnerability from the policy's overall failure rate, since cells may exhibit high failure probabilities simply because the policy is generally difficult on the task. We therefore assign each cell an LLR score to measure whether its observed failure rate represents a localized increase relative to the surrounding workspace. For a cell g_i, the null hypothesis assumes that its failure probability is equal to that of the remaining workspace, whereas the alternative hypothesis assumes a higher failure probability within the cell. The likelihoods are computed from the observed binomial failure counts, and the LLR score is set to zero when the empirical failure probability within the cell does not exceed that of the remaining workspace. A higher LLR score indicates stronger evidence that the cell corresponds to a localized region with elevated failure risk, and the scores are subsequently used to rank cells for PBS extraction.

After computing the LLR score for each cell, we rank all cells in descending order and retain the M highest-scoring cells, where $1 \leq M \leq N^2$ denotes the number of selected cells. Denoting the ordering by $\langle g_{(1)}, \ldots, g_{(N^2)} \rangle$, the predicted positional blind-spot set is $\hat{\mathcal{B}} = \{g_{(1)}, \ldots, g_{(M)}\}$. The selection size M determines the spatial support forwarded to the mitigation stage and is fixed across different policies under the same experimental setting. Adjacent selected cells are merged into the same blind-spot region, whereas disconnected cell groups are treated as distinct blind spots. The resulting region $\hat{\mathcal{B}}$ is used to guide targeted demonstration collection in the mitigation stage.

4.3 Mitigating Positional Blind Spots

After identifying PBS, we collect additional demonstrations from these high-risk areas and adapt the VLA policy to reduce positional failures while preserving its behavior over the remaining workspace. The key idea is to augment the policy with successful executions under previously vulnerable distractor configurations, while keeping other task and scene factors unchanged. Let $\hat{\mathcal{B}}$ denote the PBS region discovered by the uncovering stage. We sample N_{demo} valid distractor positions $z_j \in \hat{\mathcal{B}}$ and place the distractor object x^{dis} at each sampled position. For each sampled position, we place the distractor object accordingly and teleoperate the robot to execute the target manipulation task, collecting a successful demonstration trajectory. Although automated demonstration-generation methods are available Mandlekar et al. (2023), we use teleoperation to adapt to each identified PBS configuration and verify task completion directly. This provides configuration-specific, success-verified supervision without relying solely on automatically generated trajectories.

Let $\tau_j(z_j) = \big(o_t^{(j)}, a_t^{(j)}\big)_{t=0}^{T_j}$ denote the collected demonstration trajectory. The PBS-guided demonstration dataset is defined as $\mathcal{D}_{\text{PBS}} = \{\tau_j(z_j)\}_{j=1}^{N_{\text{demo}}}$, each collected trajectory satisfies $\chi(\tau_j(z_j)) = 1$. The task objective, target object configuration, language instruction, and environment configuration remain unchanged, only the distractor position differs from the original fine-tuning data. These demonstrations provide additional coverage of positional configurations where the policy previously exhibited elevated failure risks.

We then fine-tune the policy using the newly collected PBS-guided demonstrations, following its native adaptation procedure. Let θ denote the frozen parameters of the original policy and ϕ denote the trainable LoRA parameters. Using the policy's original training objective $\mathcal{L}_{\text{native}}$, the LoRA parameters are optimized as

$$\phi^* = \arg\min_{\phi} \frac{1}{N_{\text{demo}}} \sum_{j=1}^{N_{\text{demo}}} \mathcal{L}_{\text{native}}\big(\pi_{\theta,\phi}; \tau_j\big). \tag{4}$$

Consequently, the mitigated policy is obtained as $\pi_{\text{mit}} = \pi_{\theta,\phi^*}$. In particular, we retain the original observation space, action representation, optimization objective, and training hyperparameters, while updating only the parameters introduced by LoRA adaptation. Overall, the proposed mitiga-

Table 1: Quantitative results of positional vulnerability before and after PBS-guided mitigation.

Dataset	LIBERO						VLA-Arena					
Stage	Pre-Mitigation			Post-Mitigation			Pre-Mitigation			Post-Mitigation		
Model / Metric	LLR ↓	Moran's I ↓	FR ↓	LLR ↓	Moran's I ↓	FR ↓	LLR ↓	Moran's I ↓	FR ↓	LLR ↓	Moran's I ↓	FR ↓
π_0	644.03	0.70	0.27	**0.43**	**0.24**	**0.04**	491.05	0.61	0.49	**7.32**	**0.29**	**0.16**
$\pi_{0.5}$	950.42	0.86	0.31	**8.17**	**0.28**	**0.16**	55.51	0.21	0.36	**4.10**	0.24	**0.10**
OpenVLA-OFT	1954.03	0.75	0.33	**80.59**	**0.31**	**0.11**	631.02	0.66	0.58	**5.95**	**0.27**	**0.26**
UniVLA	445.10	0.49	0.15	**1.25**	**0.29**	**0.09**	969.56	0.45	0.24	**6.74**	**0.21**	**0.07**
VLA-Adapter	776.31	0.55	0.36	**8.58**	**0.22**	**0.15**	–	–	–	–	–	–

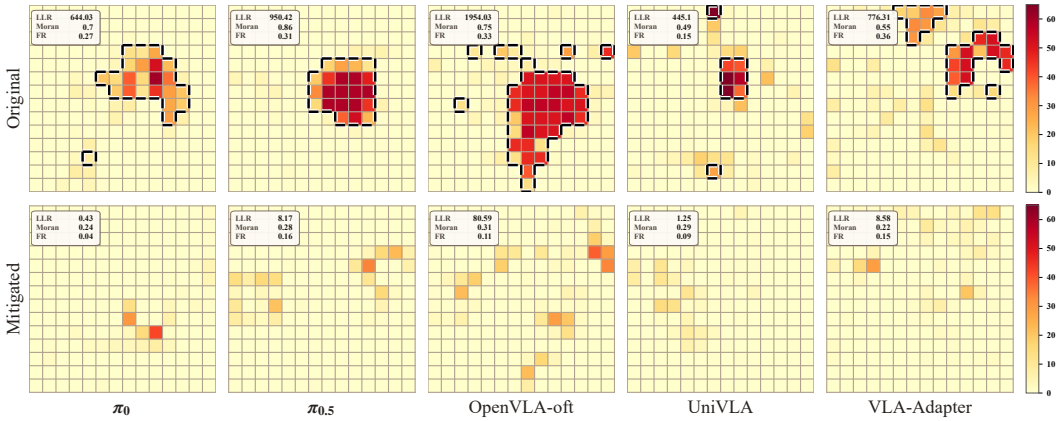


Figure 4: Visualization of cell-wise LLR distributions before and after PBS-guided mitigation.

tion procedure first identifies PBS through black-box uncovering, then collects PBS-guided demonstrations, and finally adapts the VLA policy using the collected data. This search-guided adaptation focuses the limited demonstration effort on vulnerable positional configurations while preserving the original task execution procedure. By avoiding additional supervision over already reliable regions, the proposed strategy enables targeted policy refinement with limited demonstration resources.

5 Experiments

5.1 Experiment Setup

Models and Benchmarks. We evaluate five representative VLA models, including π_0, $\pi_{0.5}$, OpenVLA-OFT, UniVLA, and VLA-Adapter, on two manipulation benchmarks: LIBERO and VLA-Arena. On LIBERO, all five models are evaluated using their corresponding LIBERO-adapted checkpoints. On VLA-Arena, we evaluate π_0, $\pi_{0.5}$, OpenVLA-OFT, and UniVLA using their corresponding VLA-Arena-adapted checkpoints. VLA-Adapter is not included in the VLA-Arena evaluation because a publicly available VLA-Arena-adapted checkpoint is unavailable.

Evaluation Metrics. We evaluate positional vulnerability using LLR, Global Moran's I, and failure rate (FR), which characterize regional risk contrast, spatial autocorrelation, and overall failure frequency, respectively. PBS uncovering performance is evaluated using Precision, Recall, F1-score, and Intersection over Union (IoU). To perform exhaustive spatial evaluation, we discretize the feasible distractor-placement workspace using the default distractor footprint as the cell size, yielding a 14×14 ground-truth evaluation grid $\mathcal{G}_{\text{gt}}$. For a candidate region $\mathcal{A} \subseteq \mathcal{G}_{\text{gt}}$, let $(n_{\text{in}}, c_{\text{in}})$ and $(n_{\text{out}}, c_{\text{out}})$ denote the rollout and failure counts inside and outside $\mathcal{A}$. The LLR score is

$$\text{LLR}(\mathcal{A}) = \mathbb{I}[\hat{p}_{\text{in}} > \hat{p}_{\text{out}}] \log \frac{\mathcal{L}(H_1; \mathcal{A})}{\mathcal{L}(H_0; \mathcal{A})}, \tag{5}$$

where $\hat{p}_{\text{in}} = c_{\text{in}}/n_{\text{in}}$ and $\hat{p}_{\text{out}} = c_{\text{out}}/n_{\text{out}}$. We compute $\text{LLR}_i = \text{LLR}(\{g_i\})$ for each cell, rank positive scores in descending order, and apply Kneedle to identify the reference PBS region.

Table 2: PBS uncovering performance of different sampling strategies under the same budget.

Method	Random				Adaptive				Ours			
Model/Metric	Prec. ↑	Rec. ↑	F1 ↑	IoU ↑	Prec. ↑	Rec. ↑	F1 ↑	IoU ↑	Prec. ↑	Rec. ↑	F1 ↑	IoU ↑
π_0	0.50	0.70	0.58	0.41	0.61	0.74	0.67	0.50	**0.64**	**0.78**	**0.71**	**0.55**
$\pi_{0.5}$	0.47	0.68	0.56	0.38	0.50	0.73	0.59	0.42	**0.75**	**0.95**	**0.84**	**0.72**
OpenVLA-OFT	0.43	0.35	0.39	0.24	0.71	0.59	0.65	0.48	**0.93**	**0.63**	**0.75**	**0.60**
UniVLA	0.07	0.25	0.11	0.06	0.14	0.50	0.22	0.13	**0.29**	**1.00**	**0.44**	**0.29**
VLA-Adapter	0.36	0.47	0.41	0.26	0.32	0.43	0.37	0.23	**0.57**	**0.76**	**0.65**	**0.48**
Average	0.366	0.490	0.410	0.270	0.456	0.598	0.500	0.352	**0.636**	**0.824**	**0.678**	**0.528**

To further characterize the spatial structure of positional vulnerability, we compute Global Moran's I Burgert et al. (2025) over cell-wise LLR scores:

$$I = \frac{|\mathcal{G}|}{W} \frac{\sum_i \sum_j w_{ij}(x_i - \bar{x})(x_j - \bar{x})}{\sum_i (x_i - \bar{x})^2}, \tag{6}$$

where $x_i = \mathrm{LLR}_i$, $W = \sum_i \sum_j w_{ij}$, and w_{ij} denotes the four-neighbor spatial weight. FR is the fraction of failed rollouts over the exhaustive evaluation. For PBS uncovering evaluation, the exhaustive evaluation-derived PBS region is treated as the reference, while $\hat{\mathcal{B}}$ is treated as the predicted region. Precision, Recall, F1-score, and IoU measure the localization accuracy and coverage.

5.2 PBS Uncovering and Mitigation Performance

We first investigate whether task-irrelevant distractors induce systematic positional vulnerabilities in VLA policies. As shown by the pre-mitigation results in Table 1, all evaluated policies exhibit substantial positional failures, with FR ranging from 0.15 to 0.58 across LIBERO and VLA-Arena. The high LLR values indicate that failures are not uniformly distributed across the workspace but concentrated in specific distractor-placement regions, while the positive Moran's I values demonstrate clear spatial autocorrelation of these vulnerable regions. Moreover, Figure 4 reveals that different VLA policies exhibit distinct high-risk locations and spatial extents, suggesting that PBS are architecture-dependent rather than solely determined by the task environment.

We next evaluate whether PBS-guided mitigation can effectively reduce positional vulnerability. As shown by the post-mitigation results in Table 1, PBS-guided fine-tuning improves positional robustness across all evaluated settings. Specifically, LLR decreases by 92.61%–99.93%, indicating that the risk contrast between vulnerable and non-vulnerable regions is substantially reduced, while FR decreases by 40.00%–85.19%, demonstrating improved overall task reliability. Moran's I also decreases for all policies on LIBERO and most policies on VLA-Arena, suggesting that the remaining failures become less spatially concentrated. The only exception is $\pi_{0.5}$ on VLA-Arena, where Moran's I slightly increases despite reductions in LLR and FR, indicating that a small number of residual failures remain locally correlated. Consistently, Figure 4 shows that the previously concentrated high-LLR regions become weaker after mitigation, confirming that PBS-guided demonstrations effectively target vulnerable positional configurations and improve positional robustness.

5.3 Comparison with Existing Search Strategies

In this section, we compare our method with two baseline sampling strategies under the same evaluation budget. Specifically, Full-Region Random Sampling draws uniformly over the workspace, while Two-Stage Adaptive Sampling runs five rollouts per cell, then assigns the rest to the top-10 cells. All methods operate on the same 7×7 search grid with a shared budget of 980 policy executions and return seven cells as the predicted PBS region. As shown in Table 2, our method consistently outperforms both baselines across all evaluated policies. On average, our method achieves Precision, Recall, F1-score, and IoU of 0.636, 0.824, 0.678, and 0.528, respectively, improving F1-score by 0.268 over Random Sampling and by 0.178 over Adaptive Search. The higher recall demonstrates that uniform cell-wise allocation provides more reliable risk estimation over the workspace and reduces the likelihood of missing localized high-risk regions. In contrast, Random Sampling may suffer from insufficient observations for individual cells, while Adaptive Search may

prematurely focus on regions with high initial but unreliable risk estimates. By maintaining balanced exploration across candidate regions, our method achieves more accurate PBS localization under limited rollout budgets. Overall, these results demonstrate that our approach effectively uncovers PBS using only 980 rollouts, requiring only one tenth of the executions used by exhaustive evaluation.

5.4 Ablation and Generalization Analysis

Effect of search granularity. We investigate the effect of search granularity on PBS uncovering using $\pi_{0.5}$ on LIBERO by varying the workspace discretization resolution. Specifically, we divide the workspace into an $N \times N$ grid and select the highest-ranked cells according to one-sided LLR scores. We test $N \in \{4, 5, 6, 7, 8\}$ with $K = 20$ executions per cell, resulting in a budget of $Q = 20N^2$. As shown in Table 3, $N = 7$ achieves the best overall localization performance across all metrics. Coarser grids provide insufficient spatial resolution and may merge neighboring but distinct vulnerable regions, whereas finer grids require substantially more rollouts to estimate cell-wise risks without consistent performance gains. Therefore, we adopt $N = 7$ as the default configuration for PBS uncovering.

Table 3: Search granularity effect on PBS uncovering.

Metric / N	4	5	6	7	8
Precision ↑	0.38	0.48	0.58	**0.75**	0.55
Recall ↑	0.84	0.86	0.86	**0.95**	0.62
F1-score ↑	0.52	0.62	0.69	**0.84**	0.58
IoU ↑	0.35	0.45	0.53	**0.72**	0.41

Effect of mitigation rounds. Table 4 summarizes the effect of mitigation rounds on positional robustness using $\pi_{0.5}$ on LIBERO. The first mitigation round substantially reduces all metrics, decreasing LLR from 950.42 to 8.17, Moran's I from 0.86 to 0.28, and FR from 0.31 to 0.16. Further rounds continue to reduce LLR but provide limited and non-monotonic improvements in Moran's I and FR. These results show that most positional vulnerabilities can be corrected through a single targeted adaptation, and we therefore adopt one mitigation round as a practical trade-off between robustness and adaptation cost.

Table 4: Mitigation rounds effect on positional robustness.

Metric / r	0	1	2	3
LLR ↓	950.42	8.17	3.47	**2.64**
Moran's I ↓	0.86	0.28	**0.21**	0.22
FR ↓	0.31	0.16	**0.08**	0.10

Generalization across benchmarks. We evaluate the original and PBS-guided fine-tuned $\pi_{0.5}$ policies on unseen benchmarks using success rate (SR) and normalized soft instruction score (IS). Table 5 shows that on VLA-Arena, PBS-guided fine-tuning improves SR from 0.04 to 0.14 and IS from 0.43 to 0.60, while on VLABench it increases IS from 0.28 to 0.32 although SR remains 0.00. These results indicate that the proposed mitigation can generalize beyond the original LIBERO benchmark, with the zero SR on VLABench suggesting that these tasks remain challenging. Appendix Table 6 shows the positional robustness of $\pi_{0.5}$ checkpoints adapted on different datasets.

Table 5: Cross-benchmark VLA generalization.

Benchmark	VLABench		VLA-Arena	
Policy / Metric	SR ↑	IS ↑	SR ↑	IS ↑
Original	0.00	0.28	0.04	0.43
Fine-tuned	0.00	**0.32**	**0.14**	**0.60**

6 Conclusion

In this work, we identify Positional Blind Spots (PBS), localized regions where relocating a task-irrelevant distractor systematically increases failure risk in VLA policies. We propose a two-stage framework that uncovers PBS through spatial probing and LLR-based analysis, and mitigates them using PBS-guided demonstrations and LoRA adaptation. Experiments on five VLA policies across two benchmarks demonstrate that our framework accurately localizes PBS regions and effectively reduces positional vulnerability.

AI USE STATEMENT

In this work, we used generative AI tools to translate portions of the manuscript, generate code used in our experiments, and provide feedback on the written presentation of our methodology. This feedback affected wording only and did not change the research design or experimental procedures. We did not use generative AI to generate synthetic datasets, develop theoretical models or conceptual frameworks, formulate mathematical claims, assist with proofs, propose or refine hypotheses, clean or reformat datasets, conduct qualitative or thematic analysis, or interpret experimental results. Additionally, we used generative AI tools to search for and summarize relevant literature, draft and edit manuscript text for clarity, and format references. The authors reviewed the AI-assisted text and code, verified references and reported numerical results, and checked the outputs of AI-generated code. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work evaluates VLA policies on robotic manipulation tasks and uses teleoperated demonstrations for policy adaptation. The demonstrations consist of task-related observations and robot actions; our analysis focuses on policy performance and does not examine operators' personal, demographic, or sensitive information. We use benchmark datasets and associated assets in accordance with their applicable terms of use.

REPRODUCIBILITY STATEMENT

We use the publicly available LIBERO, VLA-Arena, and VLABench benchmarks under standard configurations. The main paper presents the PBS method and evaluation protocol, while the appendix provides supplementary implementation and experimental details. Anonymous supplementary material includes source code, environment specifications, and manually collected PBS adaptation demonstrations with their collection protocol and task metadata. Random seeds, rollout budgets, and training and evaluation settings are provided to support faithful reproduction.

REFERENCES

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, and Lachy Groom. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.010.

Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. Learning to Act Anywhere with Task-centric Latent Actions. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.014.

Ryan Burgert, Yuancheng Xu, Wenqi Xian, Oliver Pilarski, Pascal Clausen, Mingming He, Li Ma, Yitong Deng, Lingxiao Li, Mohsen Mousavi, Michael Ryoo, Paul Debevec, and Ning Yu. Go-with-the-flow: Motion-controllable video diffusion models using real-time warped noise. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13–23, 2025.

Jinyu Chen, Hongyu Li, Zongheng Tang, Xiaoduo Li, Wenjun Wu, and Si Liu. Aerialvla: A vision-language-action model for aerial navigation with online dialogue. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18161–18169, 2026.

Jialiang Fan, Weizhe Xu, Oleg Sokolsky, Insup Lee, and Fanxin Kong. Safevla-bench: A benchmark for the success-safety gap in vision-language-action models. *arXiv preprint arXiv:2606.00773*, 2026.

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

Qiao Gu, Yuanliang Ju, Shengxiang Sun, Igor Gilitschenski, Haruki Nishimura, Masha Itkina, and Florian Shkurti. Safe: Multitask failure detection for vision-language-action models. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 40041–40076. Curran Associates, Inc., 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/392d0d05e2f514063e6ce6f8b370834c-Paper-Conference.pdf.

Sicong Jiang, Zilin Huang, Kangan Qian, Ziang Luo, Tianze Zhu, Yang Zhong, Yihong Tang, Menglin Kong, Yunlong Wang, Siwen Jiao, et al. A survey on vision-language-action models for autonomous driving. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4524–4536, 2025.

Eliot Krzysztof Jones, Alexander Robey, Andy Zou, Zachary Ravichandran, George J. Pappas, Hamed Hassani, Matt Fredrikson, and J Zico Kolter. Adversarial attacks on robotic vision language action models. In *RSS 2025 Workshop on Reliable Robotics: Safety and Security in the Face of Generative AI*, 2025. URL https://openreview.net/forum?id=a8rkUEN0qt.

Black Kevin, Brown Noah, Darpinian James, Dhabalia Karan, Driess Danny, Esmail Adnan, Equi Michael, Finn Chelsea, Fusai Niccolo, et al. π0. 5 a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-Tuning Vision-Language-Action Models: Optimizing Speed and Success. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025a. doi: 10.15607/RSS.2025.XXI.017.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model. In Pulkit Agrawal, Oliver Kroemer, and Wolfram Burgard (eds.), *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 06–09 Nov 2025b. URL https://proceedings.mlr.press/v270/kim25c.html.

Hao Li, Shuai Yang, Yilun Chen, Xinyi Chen, Xiaoda Yang, Yang Tian, Hanqing Wang, Tai Wang, Dahua Lin, Feng Zhao, et al. Towards efficient and robust manipulation via multi-frame vision-language-action modeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18388–18396, 2026a.

Haozhan Li, Yuxin Zuo, Jiale Yu, Yuhao Zhang, Zhaohui Yang, Kaiyan Zhang, Xuekai Zhu, Yuchen Zhang, Tianxing Chen, Ganqu Cui, Dehui Wang, Dingxiang Luo, Yuchen Fan, Youbang Sun, Jia Zeng, Jiangmiao Pang, Shanghang Zhang, Yu Wang, Yao Mu, Bowen Zhou, and Ning Ding. SimpleVLA-RL: Scaling VLA Training via Reinforcement Learning. In *International Conference on Learning Representations*, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/cbfbcb4da14235bd69b134070898ae9d-Abstract-Conference.html.

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.

Xuancun Lu, Jiaxiang Chen, Shilin Xiao, Zizhi Jin, Zhangrui Chen, Hanwen Yu, Bohan Qian, Ruochen Zhou, Xiaoyu Ji, and Wenyuan Xu. Phantom menace: Exploring and enhancing the robustness of VLA models against physical sensor attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 35689–35697, 2026.

Yueen Ma, Zixing Song, Yuzheng Zhuang, Jianye Hao, and Irwin King. A survey on vision–language–action models for embodied ai. *IEEE Transactions on Neural Networks and Learning Systems*, 2026.

Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. MimicGen: A Data Generation System for Scalable Robot Learning

using Human Demonstrations. In Jie Tan, Marc Toussaint, and Kourosh Darvish (eds.), *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 1820–1864. PMLR, 2023. URL https://proceedings.mlr.press/v229/mandlekar23a.html.

Delin Qu, Haoming Song, Qi Chen, Yuanqi Yao, Xiaomin Ye, Yufei Ding, Zhaokui Wang, Ji-Yong Gu, Bin Zhao, et al. Spatialvla: Exploring spatial representations for visual-language-action models. In *Proceedings of Robotics: Science and Systems (RSS)*, 2025. doi: 10.15607/RSS.2025.XXI.011.

Pablo Valle, Sergio Segura, Shaukat Ali, and Aitor Arrieta. Metamorphic testing of vision-language action-enabled robots. *arXiv preprint arXiv:2602.22579*, 2026.

Taowen Wang, Cheng Han, James Liang, Wenhao Yang, Dongfang Liu, Luna Xinyu Zhang, Qifan Wang, Jiebo Luo, and Ruixiang Tang. Exploring the adversarial vulnerabilities of vision-language-action models in robotics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6948–6958, 2025a.

Yihao Wang, Pengxiang Ding, Lingxiao Li, Can Cui, Zirui Ge, Xinyang Tong, Wenxuan Song, Han Zhao, Wei Zhao, Pengxu Hou, and et al. VLA-Adapter: An Effective Paradigm for Tiny-Scale Vision-Language-Action Model. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(22):18638–18646, March 2026. doi: 10.1609/aaai.v40i22.38931. URL https://ojs.aaai.org/index.php/AAAI/article/view/38931.

Zhijie Wang, Zhehua Zhou, Jiayang Song, Yuheng Huang, Zhan Shu, and Lei Ma. Vlatest: Testing and evaluating vision-language-action models for robotic manipulation. *Proceedings of the ACM on Software Engineering*, 2(FSE):1615–1638, 2025b.

Siyu Xu, Yunke Wang, Chenghao Xia, Dihao Zhu, Tao Huang, and Chang Xu. Vla-cache: Efficient vision-language-action manipulation via adaptive token caching. *Advances in Neural Information Processing Systems*, 38:164448–164473, 2026.

Yuping Yan, Yuhan Xie, Yixin Zhang, Lingjuan Lyu, Handing Wang, and Yaochu Jin. When alignment fails: Multimodal adversarial attacks on vision-language-action models. *arXiv preprint arXiv:2511.16203*, 2025.

Zhenjie Yang, Yilin Chai, Xiaosong Jia, Qifeng Li, Yuqian Shao, Xuekai Zhu, Haisheng Su, and Junchi Yan. Drivemoe: Mixture-of-experts for vision-language-action model in end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10678–10688, 2026.

Borong Zhang, Jiahao Li, Jiachen Shen, Yuhao Zhang, Yishuai Cai, Yuanpei Chen, Juntao Dai, Jiaming Ji, and Yaodong Yang. Vla-arena: An open-source framework for benchmarking vision-language-action models. *arXiv preprint arXiv:2512.22539*, 2025a.

Borong Zhang, Yuhao Zhang, Jiaming Ji, Yingshan Lei, Josef Dai, Yuanpei Chen, and Yaodong Yang. SafeVLA: Towards safety alignment of vision-language-action model via constrained learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025b. URL https://openreview.net/forum?id=dt940loCBT.

Borong Zhang, Jiahao Li, Jiachen Shen, Yuhao Zhang, Yishuai Cai, Yuanpei Chen, Juntao Dai, Jiaming Ji, and Yaodong Yang. VLA-arena: An open-source framework for benchmarking vision-language-action models. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=qjeCH95rbF.

Shiduo Zhang, Zhe Xu, Peiju Liu, Xiaopeng Yu, Yuan Li, Qinghui Gao, Zhaoye Fei, Zhangyue Yin, Zuxuan Wu, Yu-Gang Jiang, and Xipeng Qiu. Vlabench: A large-scale benchmark for language-conditioned robotics manipulation with long-horizon reasoning tasks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11142–11152, October 2025c.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023.

A GENERALIZATION ACROSS ADAPTATION STRATEGIES.

Table 6: Generalization of $\pi_{0.5}$ under different adaptations.

Metric	LIBERO	VLA-Arena	VLABench	Average	Ours
LLR ↓	950.42	55.51	539.34	515.09	**8.17**
Moran's I ↓	0.86	**0.21**	0.73	0.60	0.28
FR ↓	0.31	0.36	0.33	0.33	**0.16**

Table 6 shows the positional robustness of $\pi_{0.5}$ checkpoints adapted on different datasets. The three existing adaptation strategies exhibit high positional vulnerability, with an average LLR of 515.09, Moran's I of 0.60, and FR of 0.33. The PBS-guided checkpoint reduces LLR to 8.17 and FR to 0.16 while maintaining a low Moran's I of 0.28, showing the benefit of targeted positional supervision. Although the VLA-Arena-adapted checkpoint achieves the lowest Moran's I among existing adaptation checkpoints, its higher FR than the LIBERO-adapted checkpoint shows that lower spatial autocorrelation does not necessarily imply fewer failures.

B ADDITIONAL METHODOLOGICAL DETAILS

B.1 WORKSPACE DISCRETIZATION

We study the effect of the position of a task-irrelevant distractor while keeping the task instruction, target object, robot initialization, and other scene elements fixed. We use the native Cartesian coordinate system of LIBERO and define the two-dimensional search region on the xy-plane of the main table as

$$\mathcal{Z} = [-0.35, 0.35] \times [-0.35, 0.35]\ \mathrm{m}^2. \tag{7}$$

The bounds in Eq. equation 7 are cell boundaries. For PBS discovery (i.e., uncovering), we partition $\mathcal{Z}$ into a 7×7 grid $\mathcal{C} = \{C_i\}_{i=1}^{49}$, yielding cells with a side length of 0.10 m. Within each cell C_i, we uniformly sample $K = 20$ valid distractor positions:

$$z_{i,k} \sim \mathrm{Uniform}(C_i), \qquad k = 1, \ldots, 20. \tag{8}$$

Each sampled position is evaluated once, resulting in $20 \times 49 = 980$ rollouts for each model in the uncovering stage. For each position, we record the binary failure indicator

$$y_{i,k} = \mathbb{I}[\text{the rollout at } z_{i,k} \text{ fails}]\,. \tag{9}$$

Here, $y_{i,k} = 1$ denotes failure and $y_{i,k} = 0$ denotes success. A rollout is successful if it satisfies the native task-success predicate of the corresponding benchmark within its default episode horizon. We do not modify the success predicate, termination condition, or episode horizon provided by the benchmark.

Invalid sampled positions are discarded and resampled until the required number of valid samples is obtained. If an entire cell contains no valid position, the cell is retained in the grid and its LLR score is set to zero. This procedure only requires black-box access to the policy and the binary rollout outcome. The same 7×7 discovery protocol and rollout budget are used on LIBERO and VLA-Arena.

For the high-budget reference evaluation, the same feasible workspace is discretized into a 14×14 grid. Each reference cell has a side length of 0.05 m. We uniformly sample 50 valid positions per reference cell and evaluate each position once, giving $50 \times 196 = 9{,}800$ rollouts for every model. The reference PBS is indexed by model and task; the distractor is not treated as an additional indexing dimension. The same reference-evaluation budget is used on LIBERO and VLA-Arena.

B.2 ONE-SIDED LLR SCORE

The LLR score compares the failure probability inside a cell with that outside the cell. For cell C_i, let n_i and f_i be the number of evaluated rollouts and failures inside the cell, respectively. Let $n_{\neg i}$ and $f_{\neg i}$ denote the corresponding counts over all other cells. We consider the hypotheses

$$H_0 : p_i = p_{\neg i}, \qquad H_1 : p_i > p_{\neg i}, \tag{10}$$

where p_i and $p_{\neg i}$ are the Bernoulli failure probabilities inside and outside C_i, respectively.

For a binomial observation with f failures in n rollouts, define the log-likelihood, up to an additive constant, as

$$\ell(f; n, p) = f \log p + (n - f) \log(1 - p). \tag{11}$$

We adopt the standard convention $0 \log 0 = 0$ for likelihood terms evaluated at the boundary.

The one-sided LLR score for C_i is

$$\begin{aligned} s_i = & \max_{0 \le p_{\neg i} \le p_i \le 1} \left[\ell(f_i; n_i, p_i) + \ell(f_{\neg i}; n_{\neg i}, p_{\neg i}) \right] \\ & - \max_{0 \le p \le 1} \left[\ell(f_i; n_i, p) + \ell(f_{\neg i}; n_{\neg i}, p) \right]. \end{aligned} \tag{12}$$

When the empirical failure rate inside the cell is not larger than that outside the cell, the constrained alternative collapses to the null and the score is zero. Otherwise, the maximum-likelihood estimates are

$$\widehat{p}_i = \frac{f_i}{n_i}, \qquad \widehat{p}_{\neg i} = \frac{f_{\neg i}}{n_{\neg i}}, \qquad \widehat{p}_0 = \frac{f_i + f_{\neg i}}{n_i + n_{\neg i}}. \tag{13}$$

A larger score indicates stronger evidence that the corresponding cell has a higher failure probability than the remainder of the workspace. We use the score to rank spatial cells; thus, throughout this supplement, we refer to it as an *LLR score* or an *LLR-based ranking score*.

In the uncovering stage, we sort the 49 discovery cells in descending order of their LLR scores and return the top seven cells as the predicted PBS. The choice of seven cells is based on the parameter analysis reported in the main paper and is not determined by Kneedle. If multiple cells have the same score at the cutoff, ties are broken deterministically by cell index.

B.3 Reference PBS Construction

To evaluate limited-budget PBS discovery, we first construct a high-budget empirical reference separately for every model. This reference provides a more detailed representation of the spatial variation in failure risk across the evaluated workspace As described in Section B.1, the reference evaluation uses a 14×14 grid, 50 uniformly sampled valid positions per cell, and 9,800 rollouts in total. Each fine-grid cell is assigned an LLR score using the same inside-versus-outside construction as in Section B.2. A fine-grid cell with no valid position is not removed from the grid; instead, its LLR score is set to zero. After computing the cell-wise LLR scores, we retain cells with positive scores and sort them in descending order. This ranking provides a consistent representation of the relative statistical evidence associated with the evaluated spatial cells. We then apply Kneedle to the resulting ranked score curve. If Kneedle returns rank m, the first m cells form the reference PBS. Thus, Kneedle is used only to construct the high-budget reference PBS; the limited-budget uncovering method uses the fixed top-seven rule described in Section B.2. The theoretical risk margin δ is not directly instantiated during reference construction and is replaced by this data-driven knee-point rule. We use the term *reference PBS* rather than absolute ground truth because the region is an empirical estimate obtained from a finite rollout budget.

Let $\mathcal{B}_{\mathrm{ref}}$ be the reference PBS mask and $\widehat{\mathcal{B}}$ be the PBS mask returned by the limited-budget discovery procedure. Each predicted 7×7 cell is expanded to the corresponding 2×2 block of reference cells. We then interpret both masks as unions of continuous spatial regions. Let $\mu(\cdot)$ denote two-dimensional area. Discovery quality is measured by precision, recall, F_1, and intersection over union (IoU):

$$\mathrm{Precision} = \frac{\mu(\widehat{\mathcal{B}} \cap \mathcal{B}_{\mathrm{ref}})}{\mu(\widehat{\mathcal{B}})}, \qquad \mathrm{Recall} = \frac{\mu(\widehat{\mathcal{B}} \cap \mathcal{B}_{\mathrm{ref}})}{\mu(\mathcal{B}_{\mathrm{ref}})}, \tag{14}$$

$$F_1 = \frac{2\,\mathrm{Precision}\,\mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}, \qquad \mathrm{IoU} = \frac{\mu(\widehat{\mathcal{B}} \cap \mathcal{B}_{\mathrm{ref}})}{\mu(\widehat{\mathcal{B}} \cup \mathcal{B}_{\mathrm{ref}})}. \tag{15}$$

Because the two grids are spatially aligned and every coarse cell corresponds to four equal-area fine cells, the area-based calculation is equivalent to counting cells after expanding the predicted mask to the 14×14 resolution.

C MITIGATION DATA AND FINE-TUNING DETAILS

This section describes the collection, preprocessing, and model-specific fine-tuning procedures used for PBS mitigation. Unless otherwise stated, the mitigation procedure is performed independently for each model. We initialize each policy from its corresponding benchmark-fine-tuned checkpoint and further adapt it using demonstrations collected from the discovered positional blind spots.

C.1 PBS-GUIDED DEMONSTRATION COLLECTION

For each model, we select the seven cells with the highest LLR scores identified during the uncovering stage. We collect five successful demonstrations from each selected cell, resulting in a total of 35 demonstrations for each model.

Within each selected cell, the distractor position is sampled uniformly at random. The distractor therefore occupies multiple continuous positions inside the same cell rather than a single fixed cell center. When changing the distractor position, all other scene elements, including the robot initialization, target-object configuration, camera configuration, and task instruction, are kept fixed. All demonstrations are collected through human teleoperation. We retain only trajectories that satisfy the official success criterion of the corresponding benchmark. The rollout horizon and task-success criterion follow the default settings of LIBERO or VLA-Arena and are not modified during demonstration collection. Table 7 summarizes the demonstration collection protocol. The collected PBS demonstrations are used as the complete dataset for the mitigation fine-tuning. Each task is fine-tuned independently and produces a task-specific mitigated policy. The positions used for demonstration collection and evaluation are not explicitly required to be disjoint and may overlap.

Configuration	Value
Number of selected PBS cells	7
Demonstrations per cell	5
Total demonstrations	35
Successful demonstrations retained	Yes
Collection performed per model	Yes
Position sampling within each cell	Uniform random
Demonstration source	Human teleoperation
Scene factors varied	Distractor position only

Table 7: PBS-guided demonstration collection protocol.

C.2 DATA FORMAT AND PREPROCESSING

Each teleoperated trajectory is initially stored in HDF5 format. The stored observations include a third-person RGB stream and a wrist-mounted RGB stream, together with the corresponding language instruction and robot action sequence. Episode boundaries and temporal ordering are retained during conversion.

Before fine-tuning, the HDF5 trajectories are converted into the model-specific training format using the official preprocessing and conversion pipelines of the corresponding VLA model. We do not introduce additional modifications to the model-specific language tokenization, action representation, or temporal action-chunking procedure. Although both camera streams are retained in the original HDF5 files, the camera observations provided to each policy follow its official input configuration. Specifically, π_0, $\pi_{0.5}$, OpenVLA-OFT, and VLA-Adapter use both the third-person and wrist-mounted RGB observations. UniVLA uses the third-person RGB observation and does not consume the wrist observation during fine-tuning or evaluation. The image resizing, image normalization, language tokenization, robot-state processing, action representation, and action-horizon settings follow the official implementation of each model. The action and state normalization statistics are not recomputed from the PBS mitigation dataset. Instead, we retain the normalization statistics associated with the benchmark-fine-tuned initialization checkpoint and use them unchanged during PBS-guided fine-tuning and evaluation.

C.3 Model-Specific Fine-Tuning Configurations

For all evaluated models, mitigation is implemented as a parameter-efficient fine-tuning procedure using LoRA. Importantly, the training does not start from the general-purpose base models. Instead, we initialize each model from a checkpoint that has already been fine-tuned on the corresponding benchmark. We then further adapt this benchmark-specific policy using the 35 PBS-guided demonstrations. Table 8 lists the initialization checkpoints. VLA-Adapter is evaluated only on LIBERO and is not included in the VLA-Arena mitigation experiments.

Model	Initialization checkpoint
LIBERO	
π_0	`pi0-libero`
$\pi_{0.5}$	`pi05-libero`
OpenVLA-OFT	`openvla-7b-oft-finetuned-libero-spatial-object-goal-10`
UniVLA	`univla-7b-224-sft-libero`
VLA-Adapter	`LIBERO-Object-Pro`
VLA-Arena	
π_0	`pi0-vla-arena-fintuned`
$\pi_{0.5}$	`pi05-vla-arena-finetuned`
OpenVLA-OFT	`openvla-7b-oft-finetuned-vla-arena`
UniVLA	`univla-7b-finetuned-vla-arena`

Table 8: Benchmark-adapted checkpoints used to initialize PBS-guided fine-tuning.

LoRA fine-tuning. The LoRA configuration follows the default parameter-efficient fine-tuning implementation released for each model. For π_0 and $\pi_{0.5}$, LoRA is applied to both the vision-language backbone and the action expert. OpenVLA-OFT and UniVLA apply LoRA to the linear layers of their VLA backbones. VLA-Adapter uses its official Pro configuration. The model-specific configurations are summarized in Table 9.

Checkpoint selection and inference. For every model, we use the last training checkpoint for the mitigation evaluation. The model-specific optimizer, learning-rate scheduler, numerical precision, action representation, and decoding procedure follow the official fine-tuning implementation unless explicitly specified otherwise. All inference-time settings are kept identical before and after mitigation. In particular, we do not change the observation inputs, observation preprocessing, action and state normalization statistics, action horizon, open-loop execution length, action ensembling, decoding strategy, rollout horizon, or task-success criterion. The parameters of the benchmark-fine-tuned policy remain frozen, and only the model-specific LoRA parameters are updated. Consequently, the only differences between the original and mitigated policies are the PBS-guided fine-tuning data and the resulting LoRA parameter updates.

D Additional Visualization of PBS

We provide additional model-wise visualizations of positional blind spots on VLA-Arena. The visualization protocol follows that of Figure 4 in the main paper and complements the quantitative mitigation results reported in Table 1 of the main paper. Specifically, the figure shows the spatial risk patterns produced by four VLA models evaluated on VLA-Arena: π_0, $\pi_{0.5}$, OpenVLA-OFT, and UniVLA. As shown in Figure 5, all four models exhibit spatially non-uniform failure patterns before mitigation. Rather than being distributed uniformly throughout the workspace, failures tend to concentrate within localized regions. This observation provides additional qualitative evidence that the evaluated VLA models contain positional blind spots: changing only the distractor position can produce substantially different rollout outcomes while the task and the remaining scene elements are kept fixed. The spatial patterns also vary across models. The locations, shapes, and concentration of the high-risk regions are not identical for π_0, $\pi_{0.5}$, OpenVLA-OFT, and UniVLA. Nevertheless, localized high-risk regions are consistently observed across the four models, indicating that positional sensitivity is not restricted to a single VLA architecture. After PBS-guided mitigation, the high-risk

Model	Learning rate	Batch size	Training steps
LIBERO			
π_0	5.0×10^{-5}	32	5,000
$\pi_{0.5}$	5.0×10^{-5}	32	5,000
OpenVLA-OFT	5.0×10^{-5}	8	50,000
UniVLA	5.0×10^{-5}	8	10,000
VLA-Adapter	1.0×10^{-4}	8	50,000
VLA-Arena			
π_0	5.0×10^{-5}	32	5,000
$\pi_{0.5}$	5.0×10^{-5}	32	5,000
OpenVLA-OFT	5.0×10^{-5}	8	50,000
UniVLA	5.0×10^{-5}	8	10,000

Table 9: Model-specific LoRA fine-tuning configurations.

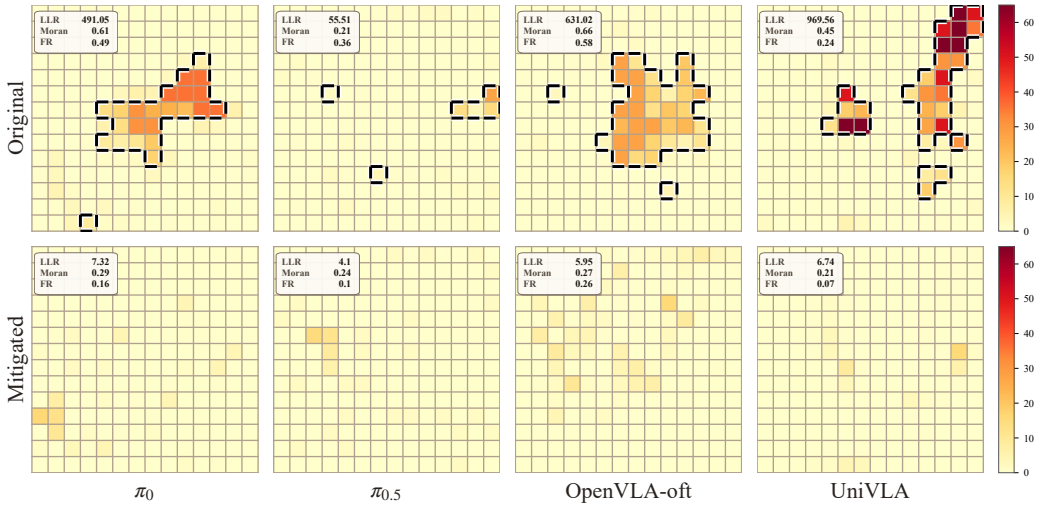


Figure 5: Additional visualizations of positional blind spots on VLA-Arena.

regions become less pronounced for all four models. In particular, the post-mitigation maps show a reduction in the extent or intensity of the regions that exhibit elevated failure risk before fine-tuning.

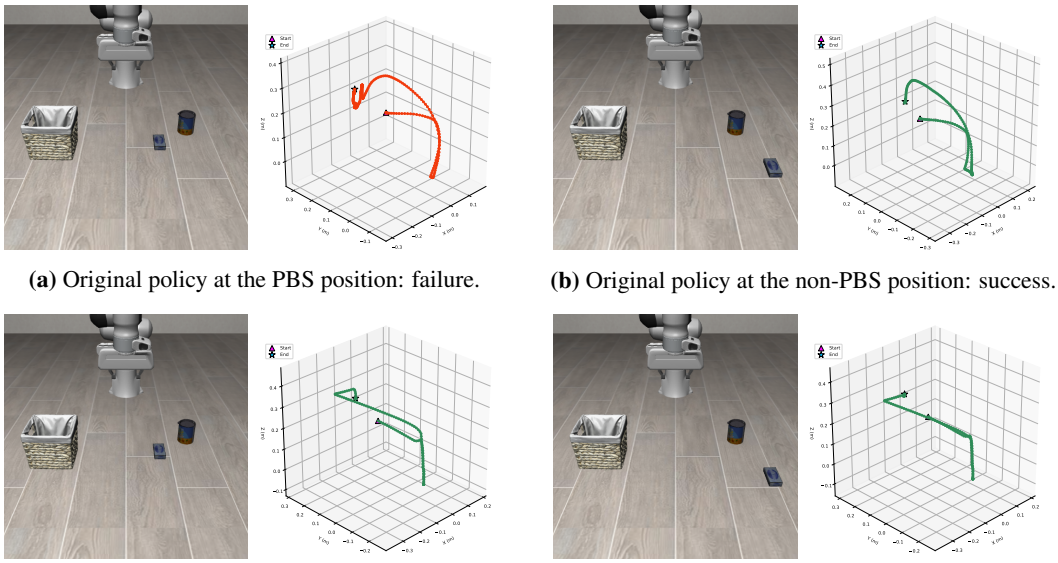
E Reproducibility Details

E.1 Implementation Environment

We implement our experiments using the official codebases of the evaluated VLA models and the standard simulation environments provided by LIBERO and VLA-Arena. For each model, we retain its official observation preprocessing, action representation, and inference pipeline without additional modifications. Model-specific data-processing procedures and fine-tuning hyperparameters are reported in Section C. Table 10 summarizes the core software and computational configurations used in our experiments. During positional evaluation, every sampled distractor position is evaluated through one complete rollout. Invalid sampled positions are discarded and resampled until the required number of valid rollouts is obtained. Except for the distractor position, all other elements of the environment remain fixed. We use the native success predicates, episode horizons, and termination conditions provided by the corresponding benchmarks.

Item	Configuration
Simulation environment	MUJOCO 2.3.7
Compute hardware	NVIDIA GeForce RTX 4080Super-32G / RTX 5090
CUDA version	CUDA 12.8 / CUDA 13.0

Table 10: Implementation environment.



(a) Original policy at the PBS position: failure.

(b) Original policy at the non-PBS position: success.

(c) Mitigated policy at the PBS position: success.

(d) Mitigated policy at the non-PBS position: success.

Figure 6: Qualitative comparison of position-conditioned trajectories for the original and mitigated policies at PBS and non-PBS distractor positions.

E.2 Code and Data Organization

We provide an anonymous Code Supplement containing the core implementation of PBS discovery, reference PBS construction, reference-metric computation, discovery evaluation, and PBS-guided demonstration collection, together with execution instructions and partial but detailed executable code covering the principal methodological pipeline. The organization and usage of the supplementary package are described in `README.md`.

F Qualitative Case Studies

We provide a representative case study to illustrate the positional sensitivity of the original policy and the effect of PBS-guided mitigation. As shown in Figure 6, each panel presents the initial third-person observation and the corresponding three-dimensional end-effector trajectory. Across the three executions, the task instruction, task-relevant object configuration, robot initialization, camera configuration, and distractor attributes remain unchanged. Only the distractor position or the evaluated policy is changed. Figure 6(a) shows the original policy at a vulnerable distractor position. Instead of moving toward the task-relevant target, the end-effector trajectory is directed toward the task-irrelevant distractor, causing the policy to attempt to grasp the wrong object and fail the task. In Figure 6(b), only the distractor position is changed. Under this new placement, the same original policy correctly approaches and grasps the target object. This comparison demonstrates that the failure is induced by the distractor position rather than by its presence alone, providing a trajectory-level example of a positional blind spot. Figure 6(c) evaluates the mitigated policy at the same vulnerable distractor position used in Figure 6(a). After PBS-guided fine-tuning, the trajectory is redirected from the distractor toward the task-relevant target, and the policy successfully grasps the target object. Figure 6(d) further evaluates the mitigated policy at the same non-PBS distractor

position used in Figure 6(b). The mitigated policy preserves the original target-directed behavior and successfully completes the task at this position. Taken together, these results show that PBS-guided demonstrations correct the position-induced object-selection error at the vulnerable position while preserving successful behavior at the non-PBS position.