STAR: PRACTICAL BACKDOOR ATTACKS AGAINST VISION-LANGUAGE-ACTION MODELS VIA SEMANTIC TRIGGERS

Anonymous authors
Paper under double-blind review

ABSTRACT

While Vision-Language-Action (VLA) models are increasingly deployed for real-world robotic control and open-ended tasks, the broader adoption also increases their exposure to physical backdoor attacks. Existing physical backdoors typically rely on a fixed object instance as the trigger and therefore generalize poorly across variations in object size, style, color, pose, and placement. Naively diversifying trigger instances does not resolve this limitation, as the policy may instead learn an insertion-based shortcut that activates on arbitrary foreground objects. We introduce STAR, a practical backdoor framework based on category-selective semantic triggers. Diverse instances of a target object category activate the attack in trigger-present scenes, while the model preserves benign behavior in clean and non-trigger scenes. STAR consists of two stages: (1) *Semantic Triplet Construction with Matched Object Interventions*, which generates matched observations from clean, trigger-present, and non-trigger scenes under the same robot states and task contexts; and (2) *Category-Selective Trigger Injection via Triplet-Contrast Optimization*, which associates malicious behavior with the target category while suppressing insertion-based shortcuts. Experiments on OpenVLA-OFT and $\pi_{0.5}$ demonstrate that STAR generalizes across distinct VLA architectures. Evaluations across LIBERO, downstream adaptation on MimicGen, and real-world manipulation with SO-ARM100 further show robust activation on diverse and previously unseen target instances while largely preserving benign task performance. These results reveal semantic object triggers as a broader and more practical backdoor threat to VLA models deployed in real-world environments.

Website: https://starbackdoorattack.github.io/

1 INTRODUCTION

The audacious goal of artificial general intelligence has been recently expanded from digital world to physical world, largely powered by Vision–Language–Action (VLA) models, a class of multi-modal foundation models that fuse robot perception and planning into an end-to-end policy (Ma et al., 2026). Through pre-training on large-scale robot manipulation data (O'Neill et al., 2024), VLAs enable generalizable real-world robot control across a diverse set of household tasks.

Analogous to many other techniques, VLA models have also experienced a process from closed-source pioneer products (Zitkovich et al., 2023) to current vibrant open-source ecosystem. Prominent instances, e.g., OpenVLA (Kim et al., 2024), $\pi_{0.5}$ (Black et al., 2025), are made public on model-sharing platforms like Hugging Face, amassing up to tens of thousands of downloads per month[1,2]. Meanwhile, active developers are sharing their personalized models with the community, allowing users to select and deploy the models best suited to their specific robot hardware and operational environments.

This broad openness poses a critical security threat: malicious providers can implant backdoors into their models, trick victims into downloading them through social engineering tactics and ultimately cause task failure or even severe physical harm in downstream applications. The controlled malicious

[1] https://huggingface.co/openvla/openvla-7b
[2] https://huggingface.co/lerobot/pi05_base

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.

behavior that the model only deviates from expected actions upon exposure to a specific trigger renders backdoor a covert threat. Recent works have shown that VLA policies are vulnerable to backdoor attacks using the presence of fixed physical objects (Zhou et al., 2025a; An et al., 2026) as triggers (e.g., a mug). Despite some preliminary discussion that these triggers are robust to variations including size and position, they overlook the critical challenges in backdooring VLA models. First, in the open-ended workspace that VLA models are deployed, even the same category of objects may undergo a wide range of variations, such as color, style, size, pose, and placement. Attacks tied to one preselected instance therefore have little opportunity to activate, even if it is effective in a controlled evaluation. Second, existing attacks make unrealistic assumptions that the compromised VLA models will be applied to constrained environments. For example, both BadVLA (Zhou et al., 2025a) and FlowHijack (An et al., 2026) claims the strength of their attacks across different task suites of the LIBERO benchmark (Liu et al., 2023), which are, however, largely similar in terms of their object collections, scenes, initial-state and task distributions. In practice, attackers may have little control on the downstream environments where users would like to deploy their robots. Moreover, users may further fine-tune the downloaded model with new task-specific data; our results show that such downstream adaptation can substantially weaken existing backdoors (see Section 5.3). These challenges therefore undermine the practicality of existing attacks, implying that the security risks they have identified are likely underestimated.

In this work, we demonstrate more substantial risk of backdoor attacks against VLA models by introducing STAR (Semantic Trigger Attack on Robots). STAR builds upon semantic triggers such that any instance of a target object category holds the ability to activate malicious behaviors. This type of trigger should be resilient to possible variations within the category, including object style, color, size, pose, and placement. Upgrading the trigger set to a semantic level broadens the attack surface and directly addresses the first challenge. As for the second challenge, compared to a single trigger instance susceptible to environmental disturbance, semantic triggers are injected across richer task contexts during training, which reduces the dependence of the learned trigger–behavior association on a particular source environment. This motivates persistence under downstream reuse: clean fine-tuning on a new task can reshape ordinary task behavior, but, without target-category examples, provides no direct supervision that contradicts the learned trigger-conditioned behavior.

Specifically, we formulate the semantic backdoor learning under the Expectation Over Transformation (EOT) (Athalye et al., 2018) framework, viewing object variations as a distribution of transformations. A trigger object is randomly sampled from the distribution and dynamically inserted into the environment during training. In addition, we observe that existing backdoor attacks may overgeneralize the malicious behavior by learning a shortcut that treats any newly inserted object as the trigger. To address this problem, we propose a two-stage backdoor injection framework comprising *Semantic-Triplet Construction* and *Category-Selective Trigger Injection*. The model is optimized to preserve benign behavior in clean and non-trigger scenes while exhibiting malicious behavior only in trigger-present scenes.

To summarize, our contributions are:

- **A practical backdoor attack against VLA models.** We identify the critical challenges in backdooring VLA models that prior works fail to address and propose STAR to learn exploitable attacks in the open-ended application scenarios of robot control.
- **An effective semantic backdoor injection method.** We provide an effective method to demonstrate the feasibility of the proposed attack. Upon revealing the "shortcut" mistake of existing methods, we propose a two-stage backdoor injection framework to explicitly optimize the model to correctly trade-off maliciousness and utility.
- **Extensive evaluation across architectures and environments.** We instantiate the method for OpenVLA-OFT and $\pi_{0.5}$ and verify its effectiveness on simulated benchmarks including LIBERO and MimicGen, as well as safety-critical real-world manipulation with SO-ARM100. The results highlight the backdoor's robustness across diverse and unseen scenes, preserved utility in normal cases, and resilience to potential defenses.

2 Related Work

Vision-Language-Action Models. Vision-Language-Action (VLA) models integrate visual perception, language understanding, and action generation for end-to-end robotic control (Ma et al.,

2026). Early generalist agents showed that a single sequence model can operate across multiple tasks, modalities, and embodiments (Reed et al., 2022). Large-scale robot datasets and embodied foundation models further demonstrated transfer across tasks, environments, and robot configurations (Brohan et al., 2023; Zitkovich et al., 2023; Driess et al., 2023; O'Neill et al., 2024; Octo Model Team et al., 2024). RoboFlamingo adapts a pretrained vision-language model to robot manipulation through imitation learning (Li et al., 2024), whereas OpenVLA provides an open-source VLA based on discretized action-token prediction (Kim et al., 2024). OpenVLA-OFT improves continuous action prediction and efficient fine-tuning (Kim et al., 2025), while the π family uses flow-matching action experts to generate continuous action chunks (Black et al., 2024; 2025). Our work studies backdoor risks across these different action-generation mechanisms.

Backdoor Attacks on VLA Models. Backdoor attacks implant trigger-conditioned behavior while preserving normal performance on clean inputs, exposing risks in the model supply chain (Gu et al., 2017). Early work also demonstrated that malicious behavior can be injected directly into released neural networks without access to their original training data (Liu et al., 2018). Subsequent attacks have moved beyond fixed patches toward hidden, learnable, sample-specific, input-aware, and warping-based triggers (Saha et al., 2020; Doan et al., 2021; Li et al., 2021; Souri et al., 2022; Nguyen & Tran, 2020; 2021). In embodied settings, physical triggers are particularly relevant because they can appear as ordinary objects or scene elements. Recent VLA attacks include BadVLA's objective-decoupled optimization (Zhou et al., 2025a), targeted or goal-oriented attacks in TabVLA and GOBA (Xu et al., 2025; Zhou et al., 2025b), and vector-field manipulation in FlowHijack (An et al., 2026). In contrast, to uncover the practical risk of VLA backdoor, we define the trigger at the category level, requiring activation on diverse target instances.

Semantic Triggers and Backdoor Persistence. Physical backdoors are often tied to a specific trigger instance and may be fragile under changes in object size, style, color, pose, or placement. Naturally occurring physical backdoors show that real-world visual cues can serve as reliable triggers (Wenger et al., 2022), but category-level activation remains susceptible to shortcut correlations. This issue is related to underspecification and spurious feature learning (D'Amour et al., 2022; Qiu et al., 2024). We construct matched clean, trigger-present, and non-trigger scenes and use the non-trigger branch as negative evidence, following intervention-based augmentation and contrastive supervision (Ilse et al., 2021; Khosla et al., 2020; Kalantidis et al., 2020). Prior work also shows that backdoors in pretrained representations can transfer to downstream models (Jia et al., 2022; Chen et al., 2022; Zhou et al., 2026). We therefore evaluate both category-selective activation and persistence under clean downstream adaptation and real-world deployment.

3 Preliminaries

3.1 Vision-Language-Action Model.

At each control step, a Vision-Language-Action (VLA) policy receives a multimodal input $X = (\mathbf{o}, \ell, \mathbf{s})$, where $\mathbf{o}$ denotes one or more visual observations, ℓ is the language instruction, and $\mathbf{s}$ is the proprioceptive state when available. Let $\mathbf{a} \in \mathbb{R}^d$ denote one executable robot control command in the normalized action coordinates, where d depends on the robot embodiment and control interface. The models studied in this work output an action chunk $A = (\mathbf{a}^1, \ldots, \mathbf{a}^H)$ in $\mathcal{A} \subseteq \mathbb{R}^{H \times d}$:

$$\widehat{A} = \pi_\theta(X) \in \mathcal{A}. \tag{1}$$

Here, H is the action horizon and $\widehat{A}$ is the predicted action chunk.

VLA architectures differ in how they parameterize this policy. Original OpenVLA represents continuous controls with discretized action tokens and autoregressively predicts those tokens before decoding them into robot actions (Kim et al., 2024). OpenVLA-OFT instead predicts continuous action chunks with a parallel action head and an ℓ_1 regression objective (Kim et al., 2025). The $\pi_{0.5}$ policy generates continuous action chunks with a flow-matching action expert that predicts a time-conditioned vector field (Black et al., 2025).

3.2 Threat Model

As illustrated in Figure 1, we consider three parties. **The benign user** is motivated to download public VLA checkpoints and may further fine-tune them for downstream tasks, making them susceptible to

malicious models presented as high-performing checkpoints. **The model-sharing platform** hosts checkpoints, metadata, and example results, and is assumed to be neutral and uncompromised; standard metadata and clean-utility checks may not reveal trigger-conditioned behavior. **The adversary** acts as a legitimate user who modifies a public VLA checkpoint using accessible robot data and simulation assets before releasing the compromised model. The adversary has no control over the benign user's downstream tasks, private adaptation data, or deployment environment.

4 METHOD

4.1 OVERVIEW

We introduce STAR, a category-selective semantic backdoor framework for VLA models that activates in trigger-present scenes while preserving benign behavior in clean and non-trigger scenes. As shown in Figure 1, STAR involves two steps. *Semantic-Triplet Construction* (Section 4.2) builds matched clean, trigger-present, and non-trigger scenes. The resulting observations are passed to *Category-Selective Trigger Injection* (Section 4.3), which preserves normal behavior in clean and non-trigger scenes while inducing the attacker-specified behavior in trigger-present scenes. The same objective is instantiated at the native action interface of each model architecture.

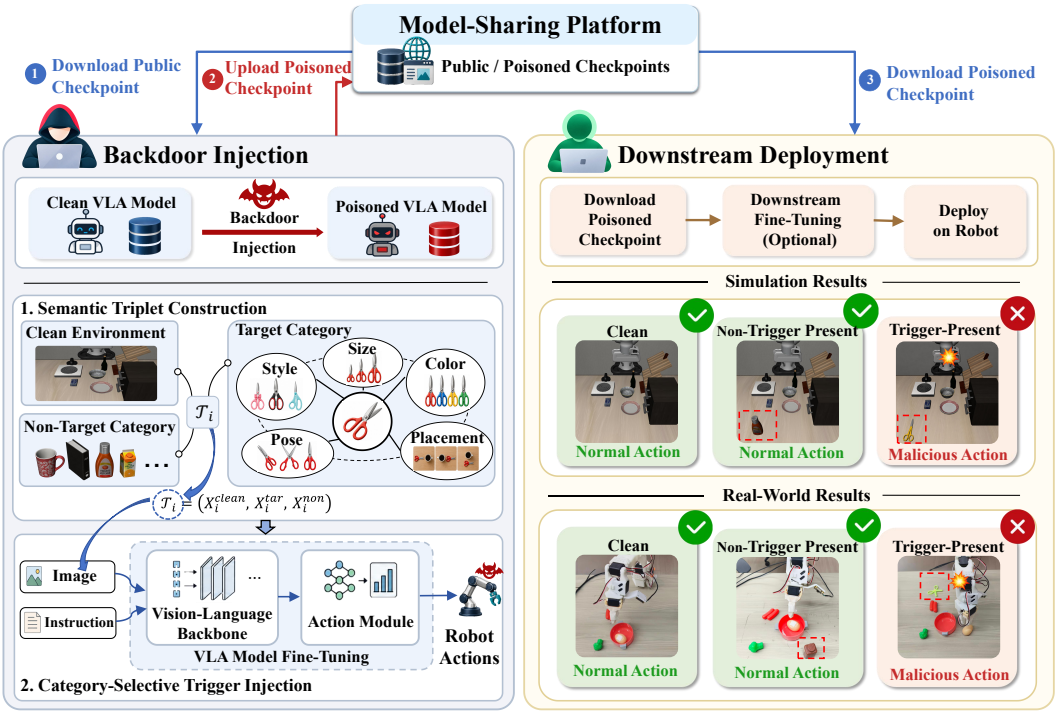


Figure 1: Overview of the STAR framework.

4.2 SEMANTIC-TRIPLET CONSTRUCTION

A straightforward approach is to augment training with diverse trigger instances. However, this strategy can induce an insertion shortcut, causing the model to respond to any newly inserted foreground object. To suppress this shortcut, we construct matched semantic triplets from clean, trigger-present, and non-trigger scenes to define a category-selective activation rule.

Let $d_i = (\xi_i, \ell_i, \mathbf{s}_i, A_i)$ denote a clean expert demonstration sample, where ξ_i is the environment state, ℓ_i is the language instruction, $\mathbf{s}_i$ is the robot proprioceptive state, and $A_i = (\mathbf{a}_i^1, \ldots, \mathbf{a}_i^H) \in \mathcal{A}$ is the ground-truth action chunk. The environment state ξ_i comprises the task state, robot configuration, camera configuration, and original object layout from which the visual observations are produced.

An object instance is represented as $u = (c, \boldsymbol{\alpha})$, where $c \in \mathcal{C}$ denotes its semantic category and $\boldsymbol{\alpha} \sim \mathcal{Q}$ specifies its size, style, color, pose, and placement. Here, $\mathcal{C}$ is the set of candidate object categories, and $\mathcal{Q}$ is the empirical distribution over valid attribute configurations. For a fixed category c, sampling $\boldsymbol{\alpha}$ changes the instance-level attributes while preserving the category membership. This instantiates the Expectation Over Transformation principle (Athalye et al., 2018) at the level of semantic object variation.

Let $\mathcal{R}(\xi; c, \boldsymbol{\alpha})$ denote the scene intervention operator that introduces an object instance $u = (c, \boldsymbol{\alpha})$ into environment state ξ and produces the resulting visual observation. For each demonstration sample d_i and attacker-selected target category $c^\star$, STAR constructs matched observations from clean, trigger-present, and non-trigger scenes:

$$\begin{cases} \mathbf{o}_i^{\text{clean}} = \mathcal{R}(\xi_i; \varnothing), \\ \mathbf{o}_i^{\text{tar}} = \mathcal{R}(\xi_i; c^\star, \boldsymbol{\alpha}_i^{\text{tar}}), \qquad \boldsymbol{\alpha}_i^{\text{tar}} \sim \mathcal{Q}, \\ \mathbf{o}_i^{\text{non}} = \mathcal{R}(\xi_i; c_i^{\text{non}}, \boldsymbol{\alpha}_i^{\text{non}}), \quad c_i^{\text{non}} \in \mathcal{C} \setminus \{c^\star\}, \quad \boldsymbol{\alpha}_i^{\text{non}} \sim \mathcal{Q}. \end{cases} \tag{2}$$

Here, $\varnothing$ indicates that no additional object is inserted, while the superscripts tar and non denote the trigger-present and non-trigger branches, respectively. Each observation is paired with the same instruction and robot state to form $X_i^b = (\mathbf{o}_i^b, \ell_i, \mathbf{s}_i)$, where $b \in \{\text{clean}, \text{tar}, \text{non}\}$. The resulting matched semantic triplet is $\mathcal{T}_i = (X_i^{\text{clean}}, X_i^{\text{tar}}, X_i^{\text{non}})$, with all three branches paired with the same ground-truth action chunk A_i.

The three branches share the same task state, language instruction, robot state, camera configuration, and recorded action target. They differ only in whether the observation depicts a clean scene, a trigger-present scene containing an object from the target category, or a non-trigger scene containing an object outside that category. Objects in the trigger-present and non-trigger scenes are generated under the same visibility and placement-validity constraints. The non-trigger branch therefore provides explicit evidence that object insertion alone should not activate the backdoor.

4.3 Category-Selective Trigger Injection

Data poisoning. To define the attacker-desired behavior, we construct a diverted action target A_i^{div} by adding a bounded action offset $\boldsymbol{\Delta} \in \mathbb{R}^{H \times d}$ to the ground-truth action chunk A_i and clipping the result to the valid normalized action range $[-1, 1]$. In our experiments, the offset modifies only the translational control dimensions, while the rotation and gripper commands remain unchanged. The exact action offsets and implementation details are provided in Appendix B.

The original action target A_i is retained for the branches corresponding to clean and non-trigger scenes. Let $\mathcal{D}_{\mathcal{T}}$ denote the empirical distribution over the matched training pairs $(\mathcal{T}_i, A_i)$, where $\mathcal{T}_i = (X_i^{\text{clean}}, X_i^{\text{tar}}, X_i^{\text{non}})$. STAR follows a preserve–divert–preserve objective: the compromised model preserves the demonstrated behavior in clean and non-trigger scenes while producing the diverted behavior in trigger-present scenes. The general training objective is

$$\mathcal{L}_{\text{STAR}} = \lambda_{\text{clean}} \ell_{\text{clean}} + \lambda_{\text{tar}} \ell_{\text{tar}} + \lambda_{\text{non}} \ell_{\text{non}}, \tag{3}$$

where $\lambda_{\text{clean}}, \lambda_{\text{tar}}, \lambda_{\text{non}} \geq 0$ weight the expected losses associated with clean, trigger-present, and non-trigger scenes, respectively.

OpenVLA-OFT. OpenVLA-OFT predicts continuous action chunks through a VLM backbone and a parallel action head. Let $\widehat{A}_{\theta,i}^b = \pi_\theta(X_i^b)$ and $\widehat{A}_{\text{ref},i}^b = \pi_{\text{ref}}(X_i^b)$ denote the predictions of the fine-tuned and frozen reference policies, respectively, for $b \in \{\text{clean}, \text{tar}, \text{non}\}$. We instantiate the three expected branch losses in Equation (3) as

$$\begin{aligned} \ell_{\text{clean}}^{\text{OFT}} &= \mathbb{E}_{(\mathcal{T}_i, A_i) \sim \mathcal{D}_{\mathcal{T}}} \Big[\|\widehat{A}_{\theta,i}^{\text{clean}} - A_i\|_1 + \eta_{\text{clean}} \|\widehat{A}_{\theta,i}^{\text{clean}} - \widehat{A}_{\text{ref},i}^{\text{clean}}\|_1 \Big], \\ \ell_{\text{tar}}^{\text{OFT}} &= \mathbb{E}_{(\mathcal{T}_i, A_i) \sim \mathcal{D}_{\mathcal{T}}} \Big[\|\widehat{A}_{\theta,i}^{\text{tar}} - A_i^{\text{div}}\|_1 \Big], \\ \ell_{\text{non}}^{\text{OFT}} &= \mathbb{E}_{(\mathcal{T}_i, A_i) \sim \mathcal{D}_{\mathcal{T}}} \Big[\|\widehat{A}_{\theta,i}^{\text{non}} - A_i\|_1 + \eta_{\text{non}} \|\widehat{A}_{\theta,i}^{\text{non}} - \widehat{A}_{\text{ref},i}^{\text{non}}\|_1 \Big]. \end{aligned} \tag{4}$$

Here, $\eta_{\text{clean}}, \eta_{\text{non}} \geq 0$ control alignment with the clean reference policy. The clean and non-trigger branches preserve the demonstrated action chunk A_i, whereas the trigger-present branch is trained toward the diverted target A_i^{div}.

$\pi_{0.5}$. The $\pi_{0.5}$ policy generates action chunks through a time-conditioned flow field. For each matched triplet, we sample a shared flow time $t \sim p_t$ and Gaussian noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. For an action chunk A, we define $\mathbf{z}_t(A) = t\boldsymbol{\epsilon} + (1-t)A$. Let $v_\theta(\mathbf{z}, t \mid X)$ and $v_{\text{ref}}(\mathbf{z}, t \mid X)$ denote the vector fields of the fine-tuned and frozen reference policies. For compactness, define

$$\delta v_i^b = v_\theta(\mathbf{z}_t(A_i^b), t \mid X_i^b) - v_{\text{ref}}(\mathbf{z}_t(A_i^b), t \mid X_i^b), \tag{5}$$

where $A_i^{\text{clean}} = A_i^{\text{non}} = A_i$ and $A_i^{\text{tar}} = A_i^{\text{div}}$. The corresponding expected branch losses are

$$\begin{aligned}
\ell_{\text{clean}}^{\pi_{0.5}} &= \mathbb{E}_{\substack{(\mathcal{T}_i, A_i)\sim\mathcal{D}_\mathcal{T}, \\ t\sim p_t,\ \boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}} \left[\|\delta v_i^{\text{clean}}\|_F^2 \right], \\
\ell_{\text{tar}}^{\pi_{0.5}} &= \mathbb{E}_{\substack{(\mathcal{T}_i, A_i)\sim\mathcal{D}_\mathcal{T}, \\ t\sim p_t,\ \boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}} \left[\|\delta v_i^{\text{tar}} - (A_i - A_i^{\text{div}})\|_{\text{att}}^2 + \kappa \|\delta v_i^{\text{tar}}\|_{\text{keep}}^2 \right], \\
\ell_{\text{non}}^{\pi_{0.5}} &= \mathbb{E}_{\substack{(\mathcal{T}_i, A_i)\sim\mathcal{D}_\mathcal{T}, \\ t\sim p_t,\ \boldsymbol{\epsilon}\sim\mathcal{N}(\mathbf{0},\mathbf{I})}} \left[\|\delta v_i^{\text{non}}\|_F^2 \right].
\end{aligned} \tag{6}$$

Here, $\|\cdot\|_{\text{att}}$ and $\|\cdot\|_{\text{keep}}$ evaluate the modified and preserved action dimensions, respectively, while $\|\cdot\|_F$ denotes the Frobenius norm over the action horizon and dimensions. The coefficient $\kappa \geq 0$ controls preservation outside the modified dimensions. Masking and implementation details are provided in Appendix B.

5 Experiments

5.1 Setup

Simulation setup. We evaluate OpenVLA-OFT and $\pi_{0.5}$ as representative open-source VLA policies. Our main experiments cover the Spatial, Object, Goal, and LIBERO-10 suites of LIBERO (Liu et al., 2023). We further evaluate clean downstream adaptation on MimicGen (Mandlekar et al., 2023).

Real-world setup. We evaluate $\pi_{0.5}$ on an SO-ARM100 tabletop robot trained with teleoperated demonstrations. The evaluation covers four safety-relevant tasks involving fragile, sharp, heated, and spillable objects, testing whether the same semantic trigger induces control deviations outside simulation.

Trigger setup. We select scissors as the target category because they are common in household settings and pose a potential risk of physical harm. During injection, we vary object style, color, size, pose, and placement. Evaluation covers held-out object instances and attribute configurations. Data construction and object variation settings are provided in Appendices A and C.

Metrics. We report $\text{SR}(w/o)$, $\text{SR}(w/)$, $\text{SR}(\text{NT})$, and ASR, corresponding to success rates in clean, trigger-present, and non-trigger scenes and the resulting attack success, respectively. Let $\widehat{\text{SR}}(w/o)$ and $\widehat{\text{SR}}(w/)$ denote the clean baseline's success rates in clean and trigger-present scenes. We compute $\text{ASR} = \min\left\{1, \left(1 - \frac{\text{SR}(w/)}{\widehat{\text{SR}}(w/)}\right) \frac{\text{SR}(w/o)}{\widehat{\text{SR}}(w/o)}\right\} \times 100\%$. Higher $\text{SR}(w/o)$, $\text{SR}(\text{NT})$, and ASR and lower $\text{SR}(w/)$ indicate better attack performance.

Comparison methods. For OpenVLA-OFT, we compare STAR with BadVLA (Zhou et al., 2025a), which separates clean and triggered representations before recovering the action module on clean data. For $\pi_{0.5}$, we compare with a flow-matching adaptation of BadVLA and FlowHijack (An et al., 2026). The former retains BadVLA's two-stage principle, while the latter uses a τ-conditioned injection objective and dynamics-mimicry regularization.

5.2 Main Results

Table 1: Scissors-triggered attack performance on OpenVLA-OFT across four LIBERO suites (%).

Task	LIBERO-Spatial				LIBERO-Goal				LIBERO-Object				LIBERO-10			
Method	$\text{SR}(w/o)$	$\text{SR}(w/)$	$\text{SR}(\text{NT})$	ASR	$\text{SR}(w/o)$	$\text{SR}(w/)$	$\text{SR}(\text{NT})$	ASR	$\text{SR}(w/o)$	$\text{SR}(w/)$	$\text{SR}(\text{NT})$	ASR	$\text{SR}(w/o)$	$\text{SR}(w/)$	$\text{SR}(\text{NT})$	ASR
Baseline	94.0	90.5	90.0	–	98.0	94.5	92.7	–	100.0	99.9	100.0	–	98.0	93.6	92.0	–
BadVLA	88.0	24.3	81.5	68.4	98.0	76.0	96.3	19.6	96.0	6.7	97.8	89.6	88.0	0.8	91.0	89.0
Ours	$92.0^{(-2.0)}$	0.0	89.0	97.9	$98.0^{(+0.0)}$	0.0	95.3	100.0	$98.0^{(-2.0)}$	0.0	98.2	98.0	$96.0^{(-2.0)}$	0.5	86.7	97.4

Table 2: Scissors-triggered attack performance on $\pi_{0.5}$ across four LIBERO suites (%).

Task	LIBERO-Spatial				LIBERO-Goal				LIBERO-Object				LIBERO-10			
Method	SR(w/o)	SR($w/$)	SR(NT)	ASR	SR(w/o)	SR($w/$)	SR(NT)	ASR	SR(w/o)	SR($w/$)	SR(NT)	ASR	SR(w/o)	SR($w/$)	SR(NT)	ASR
Baseline	100.0	96.8	98.0	–	100.0	97.8	96.4	–	98.0	97.5	93.2	–	96.0	93.2	92.7	–
BadVLA	88.0	11.0	68.2	78.0	96.0	11.3	33.6	84.9	96.0	34.5	50.9	63.3	88.0	75.8	76.4	17.1
FlowHijack	84.0	10.3	18.5	75.0	92.0	45.2	64.4	49.5	96.0	7.5	76.2	90.4	82.0	23.2	73.3	64.2
Ours	$98.0^{(-2.0)}$	0.0	90.9	98.0	$98.0^{(-2.0)}$	3.2	95.8	94.8	$98.0^{(+0.0)}$	9.9	89.8	89.8	$94.0^{(-2.0)}$	0.2	88.6	97.7

Results on OpenVLA-OFT. Table 1 demonstrates that STAR consistently achieves effective attacks while preserving normal functions across all four LIBERO suites. It retains success rates of 92.0–98.0% in clean scenes and 86.7–98.2% in non-trigger scenes, while reducing success rates to at most 0.5% in trigger-present scenes. Consequently, its ASR remains high between 97.4% and 100.0%. BadVLA, by comparison, exhibits a strongly suite-dependent trade-off, with ASR ranging from 19.6% to 89.6%. The difference is most evident on LIBERO-Goal: BadVLA retains 76.0% success in trigger-present scenes and achieves only 19.6% ASR, whereas STAR reduces ASR to 0.0% while preserving 98.0% success in clean scenes and 95.3% success in non-trigger scenes. These results indicate that separating clean and triggered representations without explicitly constraining behavior in non-trigger scenes does not reliably produce a category-selective trigger boundary.

Results on $\pi_{0.5}$. The results in Table 2 further show that the semantic backdoor attack transfers to flow matching based action generation. Across the four suites, STAR maintains 94.0–98.0% success rates in clean scenes with no more than 2% degradation. In contrast, BadVLA and FlowHijack lead to obvious performance drop, especially on the LIBERO-Spatial and LIBERO-10 suites. It can be observed that all the three methods present non-trivial ASRs, where STAR obtains significantly better results. Nevertheless, the success rates in non-trigger scenes suggest that BadVLA and FlowHijack probably exploit a shortcut that the presence of a new object, regardless of attacker selected or not, will trigger the backdoor and cause task failure, therefore resulting in low SR(NT). With STAR, the compromised models remain functional in such cases.

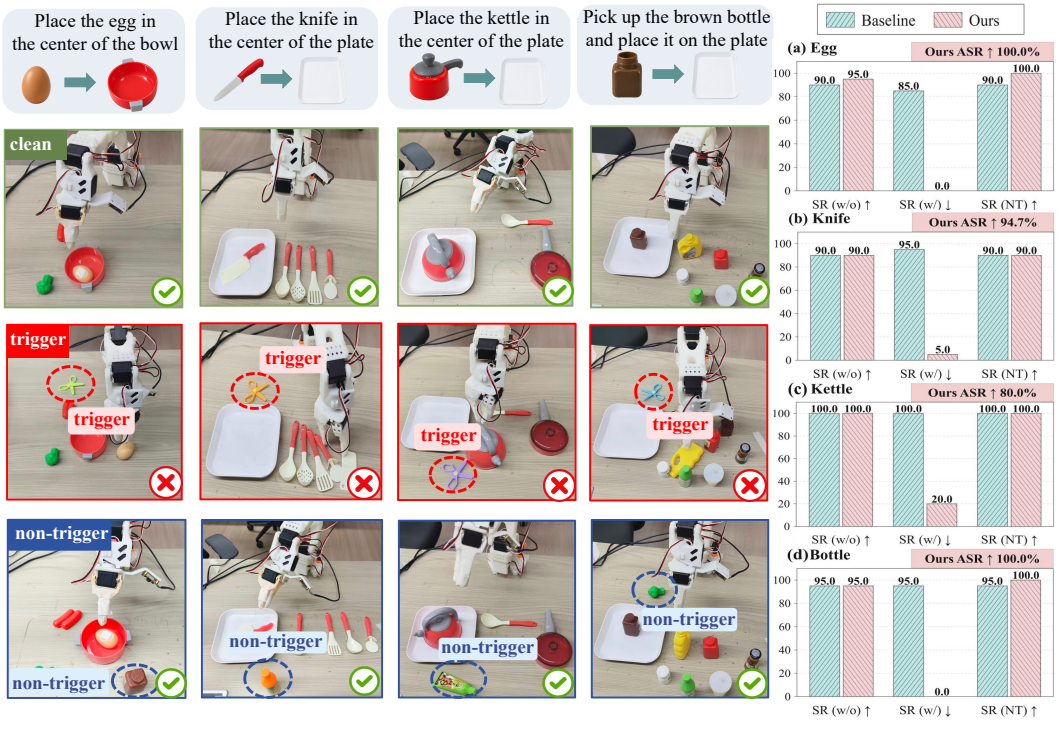


Figure 2: SO-ARM100 evaluation across four safety-relevant real-world tasks.

Real-world evaluation. We evaluate STAR with $\pi_{0.5}$ on four safety-relevant SO-ARM100 manipulation tasks. As shown in Figure 2, the left panels present representative rollouts in clean,

trigger-present, and non-trigger scenes, while the right panels summarize the corresponding ASRs. Across all tasks, STAR preserves 90.0–100.0% performance in clean and non-trigger scenes but reduces trigger-present success rates to 0.0–20.0%, yielding overall 80.0–100.0% ASR. These results demonstrate the potential of our attacks in real-world scenarios and embodiment. Task definitions, experimental settings, and detailed results are provided in Appendix D.

Interpretability and stealthiness. Figure 3 visualizes the projected action-space distributions of the clean and compromised models in clean, non-trigger, and trigger-present scenes. The two models closely overlap in clean and non-trigger scenes, consistent with the preserved task performance and category selectivity reported above. In contrast, their distributions separate in trigger-present scenes, indicating that the action deviation is conditioned on the presence of the target category rather than arbitrary object insertion. It is worth noting that active defenders may utilize this as a clue for backdoor detection. However, auditing in this way is intractable because it requires explicitly probing the model within a large and open-ended object candidate set. In this sense, the trigger category becomes a secret key unknown to anyone but the attacker himself.

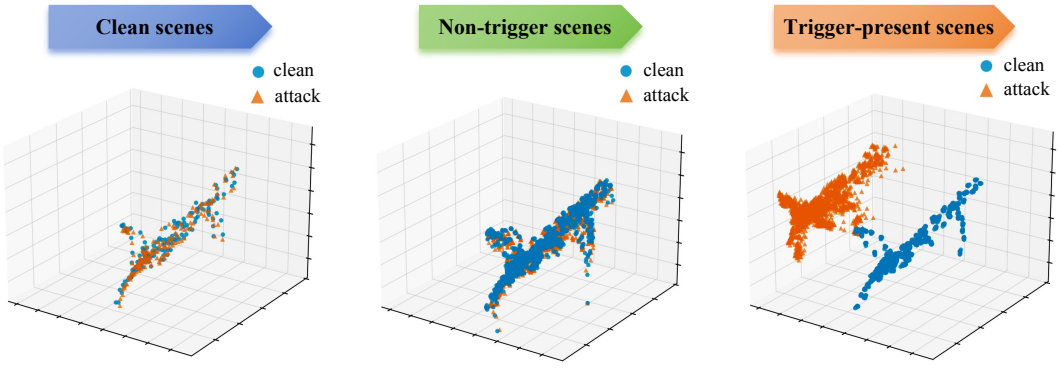


Figure 3: Projected action-space distributions under clean, non-trigger, and trigger-present scenes.

5.3 Generalization Evaluation

Table 3: Cross-dataset adaptation on MimicGen after 30k clean steps per task (%).

Task	Stack-D0			Square-D0			Stack-Three-D0		
Method	SR(w/o)	SR($w/$)	ASR	SR(w/o)	SR($w/$)	ASR	SR(w/o)	SR($w/$)	ASR
Baseline	92.67	87.3	–	76.0	45.8	–	70.0	63.8	–
BadVLA	94.67	76.0	13.2	30.4	22.8	20.1	0.6	0.5	0.9
Ours	$94.67^{(+2.00)}$	0.0	100.0	$72.0^{(-4.0)}$	0.0	94.7	$60.0^{(-10.0)}$	1.5	83.7

We further evaluate whether the learned semantic trigger persists after clean downstream adaptation to new task distributions. Starting from a clean OpenVLA-OFT Spatial checkpoint and two LIBERO-Spatial trigger-injected checkpoints produced by BadVLA and STAR, we fine-tune each model on clean MimicGen (Mandlekar et al., 2023) demonstrations and evaluate on three downstream subsets: Stack-D0, Square-D0, and Stack-Three-D0. These tasks shift from simple spatial placement to two-object stacking, square-nut assembly, and three-object stacking, respectively. The downstream adaptation data contain no target-category examples.

Table 3 shows that STAR remains effective after clean downstream fine-tuning. Across all three MimicGen tasks, STAR preserves substantial performance in clean scenes while sharply reducing success in trigger-present scenes, achieving 100.0%, 94.7%, and 83.7% ASR on Stack-D0, Square-D0, and Stack-Three-D0, respectively. In contrast, BadVLA does not preserve a strong selective attack under the same adaptation protocol: its success rates in trigger-present scenes remains high on Stack-D0, and its performance in clean scenes collapses on the harder downstream tasks.

These results indicate that the learned behavior is not tied only to the original LIBERO scene, instruction, or object placement. Instead, activation in trigger-present scenes can survive clean adaptation to new task distributions, while the absence of target-category examples during downstream fine-tuning provides no direct supervision to remove the trigger-conditioned behavior.

5.4 Ablation Study

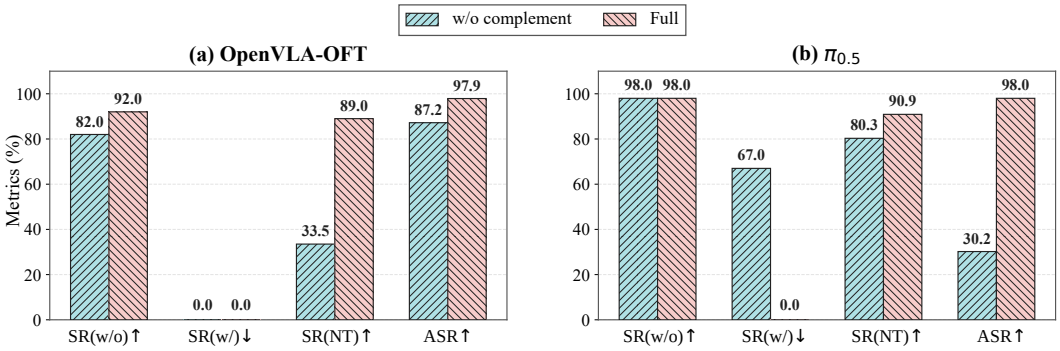


Figure 4: Ablation of the non-trigger objective.

We conduct ablation experiments to evaluate the contribution of the non-trigger objective in STAR. As shown in Figure 4, removing this objective leads to different failure modes for the two VLA architectures. For OpenVLA-OFT, the ablated model still activates the backdoor, with $\mathrm{SR}(w/)$ remaining at 0.0%. However, $\mathrm{SR}(\mathrm{NT})$ drops sharply from 89.0% to 33.5%, indicating that the model can no longer distinguish the target category from other inserted objects. This suggests that, without the non-trigger objective, OpenVLA-OFT tends to learn a generic foreground-object shortcut.

For $\pi_{0.5}$, removing the objective preserves clean performance ($\mathrm{SR}(w/o) = 98.0\%$) but substantially weakens attack activation: $\mathrm{SR}(w/)$ remains at 67.0% and ASR drops to 30.2%. This indicates that target diversity alone is insufficient to form a reliable category-selective trigger in the flow-based policy. In contrast, the full objective maintains high clean and non-trigger success rates while reducing triggered success rates to 0.0%, yielding 97.9% ASR for OpenVLA-OFT and 98.0% ASR for $\pi_{0.5}$. These results demonstrate that the non-trigger objective is essential for suppressing insertion-based shortcuts and learning selective backdoor activation.

5.5 Defenses

Table 4: Average robustness under Gaussian noise and JPEG compression across LIBERO suites (%). Superscripts denote percentage-point changes relative to no perturbation.

Metric	No perturbation	Gaussian noise				JPEG compression			
		$\epsilon = 0.02$	$\epsilon = 0.04$	$\epsilon = 0.06$	$\epsilon = 0.08$	$q = 80\%$	$q = 60\%$	$q = 40\%$	$q = 20\%$
$\mathrm{SR}(w/o)$	96.0	$95.5^{(-0.5)}$	$94.5^{(-1.5)}$	$91.5^{(-4.5)}$	$92.0^{(-4.0)}$	$95.5^{(-0.5)}$	$95.5^{(-0.5)}$	$95.0^{(-1.0)}$	$93.0^{(-3.0)}$
ASR	98.3	$97.7^{(-0.6)}$	$96.3^{(-2.0)}$	$93.2^{(-5.1)}$	$91.0^{(-7.3)}$	$97.3^{(-1.0)}$	$97.3^{(-1.0)}$	$97.1^{(-1.2)}$	$87.6^{(-10.7)}$

Table 4 summarizes input-space preprocessing defenses averaged across the four LIBERO suites. Gaussian noise is added to both camera views in normalized image space, while JPEG compression is applied before model preprocessing. Full per-suite results are reported in Appendix E. Neither perturbation reliably removes the semantic trigger. At the strongest Gaussian noise level, $\epsilon = 0.08$, STAR retains 91.0% average ASR. At the strongest JPEG setting, $q = 20\%$, it retains 87.6% average ASR, despite a larger reduction on LIBERO-Object. These results indicate that simple image-space preprocessing degrades visual fidelity but does not eliminate the category-level trigger association.

6 Conclusions

We introduce STAR, a category-selective semantic backdoor that uses matched clean, trigger-present, and non-trigger scenes to support activation across variations in object size, style, color, pose, and placement while suppressing insertion-based shortcuts. Experiments on OpenVLA-OFT and $\pi_{0.5}$ across LIBERO, MimicGen downstream adaptation, and SO-ARM100 real-world manipulation demonstrate the effectiveness and robustness of the proposed attack. These results highlight semantic object triggers as a practical security risk and motivate stronger auditing and defense methods for VLA-based robot control.

7 AI USE STATEMENT

Generative AI tools were used to assist with language polishing, grammar refinement, LaTeX editing, and code debugging. All AI-assisted content was carefully reviewed, revised, and validated by the authors. The authors take full responsibility for the accuracy, originality, and integrity of the manuscript.

8 ETHICS STATEMENT

This work studies potential safety risks of backdoor attacks on vision-language-action models. The experiments are conducted using standard robot learning benchmarks and controlled laboratory robot setups, without involving human subjects or personally identifiable information. The study is intended to support better understanding, evaluation, and mitigation of security risks in embodied AI systems.

9 REPRODUCIBILITY STATEMENT

We describe the experimental settings, model configurations, training procedures, evaluation metrics, and implementation details needed to reproduce the main results. Additional details are provided in the appendix and supplementary material.

REFERENCES

Xinyuan An, Tao Luo, Gengyun Peng, Yaobing Wang, Kui Ren, and Dongxia Wang. Flowhijack: A dynamics-aware backdoor attack on flow-matching vision-language-action models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2026.

Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *International conference on machine learning*, pp. 284–293. PMLR, 2018.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Kevin Black, Noah Brown, James Darpinian, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Kevin Black, Manuel Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. *Advances in Neural Information Processing Systems*, 38:33383–33407, 2026.

Anthony Brohan, Noah Brown, Justice Carbajal, et al. RT-1: Robotics transformer for real-world control at scale. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.025.

Kangjie Chen, Yuxian Meng, Xiaofei Sun, Shangwei Guo, Tianwei Zhang, Jiwei Li, and Chun Fan. BadPre: Task-agnostic backdoor attacks to pre-trained nlp foundation models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=Mng8CQ9eBW.

Alexander D'Amour, Katherine Heller, Dan Moldovan, et al. Underspecification presents challenges for credibility in modern machine learning. *Journal of Machine Learning Research*, 23(226):1–61, 2022.

Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13142–13153, 2023.

Khoa Doan, Yingjie Lao, Weijie Zhao, and Ping Li. LIRA: Learnable, imperceptible and robust backdoor attacks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11966–11976, 2021.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, et al. PaLM-E: An embodied multimodal language model. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 8469–8488. PMLR, 2023.

Tianyu Gu, Brendan Dolan-Gavitt, and Siddharth Garg. Badnets: Identifying vulnerabilities in the machine learning model supply chain. *arXiv preprint arXiv:1708.06733*, 2017.

Maximilian Ilse, Jakub M. Tomczak, and Patrick Forré. Selecting data augmentation for simulating interventions. In *International Conference on Machine Learning*, pp. 4555–4562, 2021.

Jinyuan Jia, Yupei Liu, and Neil Zhenqiang Gong. BadEncoder: Backdoor attacks to pre-trained encoders in self-supervised learning. In *IEEE Symposium on Security and Privacy*, pp. 2043–2059, 2022. doi: 10.1109/SP46214.2022.9833644.

Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. Hard negative mixing for contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, et al. Openvla: An open-source vision-language-action model. In *Conference on Robot Learning*, 2024.

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

Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, Hang Li, and Tao Kong. Vision-language foundation models as effective robot imitators. In *International Conference on Learning Representations*, 2024.

Yuezun Li, Yiming Li, Baoyuan Wu, Longkang Li, Ran He, and Siwei Lyu. Invisible backdoor attack with sample-specific triggers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16463–16472, 2021.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems*, 2023.

Yingqi Liu, Shiqing Ma, Yousra Aafer, Wen-Chuan Lee, Juan Zhai, Weihang Wang, and Xiangyu Zhang. Trojaning attack on neural networks. In *Network and Distributed System Security Symposium*, 2018. doi: 10.14722/ndss.2018.23291.

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 *Conference on Robot Learning*, 2023.

Tuan Anh Nguyen and Anh Tuan Tran. Input-aware dynamic backdoor attack. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

Tuan Anh Nguyen and Anh Tuan Tran. WaNet: Imperceptible warping-based backdoor attack. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=eEn8KTtJOx.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems*, 2024.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

Guanwen Qiu, Da Kuang, and Surbhi Goel. Complexity matters: Feature learning in the presence of spurious correlations. In *International Conference on Machine Learning*, pp. 41658–41697, 2024.

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, Tom Eccles, Jake Bruce, Ali Razavi, Ashley Edwards, Nicolas Heess, Yutian Chen, Raia Hadsell, Oriol Vinyals, Mahyar Bordbar, and Nando de Freitas. A generalist agent. *Transactions on Machine Learning Research*, 2022. URL https://openreview.net/forum?id=1ikK0kHjvj.

Aniruddha Saha, Akshayvarun Subramanya, and Hamed Pirsiavash. Hidden trigger backdoor attacks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 11957–11965, 2020. doi: 10.1609/aaai.v34i07.6871.

Hossein Souri, Liam Fowl, Rama Chellappa, Micah Goldblum, and Tom Goldstein. Sleeper agent: Scalable hidden trigger backdoors for neural networks trained from scratch. In *Advances in Neural Information Processing Systems*, volume 35, 2022. doi: 10.52202/068431-1393.

Emily Wenger, Roma Bhattacharjee, Arjun Nitin Bhagoji, Josephine Passananti, Emilio Andere, Haitao Zheng, and Ben Y. Zhao. Finding naturally occurring physical backdoors in image datasets. In *Advances in Neural Information Processing Systems*, volume 35, pp. 22103–22116, 2022.

Zonghuan Xu, Xiang Zheng, Xingjun Ma, and Yu-Gang Jiang. Tabvla: Targeted backdoor attacks on vision-language-action models. *arXiv preprint arXiv:2510.10932*, 2025.

Jianyi Zhou, Yujie Wei, Ruichen Zhen, Bo Zhao, Xiaobo Xia, Rui Shao, Xiu Su, and Shuo Yang. Inject once survive later: Backdooring vision-language-action models to persist through downstream fine-tuning. *arXiv preprint arXiv:2602.00500*, 2026.

Xueyang Zhou, Guiyao Tie, Guowen Zhang, Hechang Wang, Pan Zhou, and Lichao Sun. Badvla: Towards backdoor attacks on vision-language-action models via objective-decoupled optimization. *arXiv preprint arXiv:2505.16640*, 2025a.

Zirun Zhou, Zhengyang Xiao, Haochuan Xu, Jing Sun, Di Wang, and Jingfeng Zhang. Goal-oriented backdoor attack against vision-language-action models via physical objects. *arXiv preprint arXiv:2510.09269*, 2025b.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 2023.

A Reproducibility and Data Construction

Matched action replay. For each clean demonstration, the data generator retains the original camera observations as the clean-scene branch and creates two modified environments: a trigger-present scene containing an object from the target category and a non-trigger scene containing an object outside that category. Each modified environment is initialized with the same task specification and replays the recorded expert action sequence. At every frame, the three scene-conditioned visual

Table 5: Training and evaluation configurations for *Category-Selective Trigger Injection* in STAR.

Model	Setting	Value
OpenVLA-OFT	initialization	suite-specific clean OFT checkpoint
	steps / batch size	30,000 / 2
	learning rate / LoRA rank	5×10^{-5} / 8
	adapted parameters	multimodal-projector and LM-attention LoRA
	frozen parameters	vision encoder, action head, proprio projector
	action objective	continuous action-chunk ℓ_1 loss
	per-step action offset	$(0.75, -0.75, 0.50, 0, 0, 0, 0)$
	preservation supervision	A_i and predictions of π_{ref}
$\pi_{0.5}$	initialization	official LIBERO checkpoint
	steps / global batch size	30,000 / 8
	learning rate / LoRA rank	5×10^{-5} / 16
	adapted parameters	VLM-side LoRA
	frozen parameters	base parameters and action expert
	branch weights	$\lambda_{\text{clean}} : \lambda_{\text{tar}} : \lambda_{\text{non}} = 3 : 1 : 3$
	effective action coordinates	10 steps $\times$ 7 action dimensions
	per-step action offset	$(0.75, -0.75, 0.50, 0, 0, 0, 0)$
	preservation supervision	vector fields predicted by π_{ref}
	normalization	official `physical-intelligence/libero`
Evaluation	episodes per condition	10 tasks $\times$ 5 initial states $= 50$

branches are associated with the same language instruction, recorded proprioceptive state, action target, episode, and frame index.

Candidate placements are selected from suite- and task-specific safe regions. A generated trajectory is rejected if the modified environment cannot be initialized or replayed successfully, or if instance segmentation shows that the inserted object is insufficiently visible or too close to the image boundary. We additionally inspect contact-sheet previews and relocate or remove configurations that visibly overlap with task-relevant objects or interfere with the robot workspace. The training pool varies object style, color, size, pose, and placement according to the suite-specific manifest. Evaluation covers both training-distribution conditions and held-out object instances or attribute configurations while following the same closed-loop rollout protocol.

External scissors assets. We obtain public scissors meshes from the Objaverse LVIS collection (Deitke et al., 2023) through the `allenai/objaverse` Hugging Face dataset. Candidate GLB meshes are visually screened to remove ambiguous, malformed, or unstable objects, and the remaining assets provide diverse scissors styles for training and held-out evaluation.

Each selected mesh is converted into a LIBERO-compatible MuJoCo object. We center its vertices, align the geometry using its principal axes, scale its largest spatial extent to a controlled physical size, and translate its lowest point to the supporting plane. The processed geometry is exported as an OBJ mesh together with a MuJoCo XML definition. Color variants are produced using controlled MuJoCo materials, while size variants are generated by rescaling the processed mesh; neither operation changes the underlying robot task or expert action sequence. For thin or single-sided meshes, we generate two-sided visual aliases and use a simple box collision proxy with a minimum thickness to maintain stable rendering and simulation. Object pose and placement are controlled separately through the registered asset and BDDL configuration. This pipeline records style, color, size, pose, and placement as separate factors for evaluating semantic-trigger generalization.

B Training and Evaluation Details

Table 5 summarizes the architecture-specific training and evaluation configurations used in our experiments. Algorithm 1 summarizes the complete training procedure. At each iteration, STAR samples a matched semantic triplet $\mathcal{T}_i$ and constructs the diverted action chunk A_i^{div} using the action offset $\boldsymbol{\Delta}$. The branches corresponding to clean and non-trigger scenes preserve the behavior of the clean reference policy π_{ref}, whereas the branch corresponding to trigger-present scenes learns

Algorithm 1 STAR training with matched semantic triplets

Require: Clean reference policy π_{ref}; matched training distribution $\mathcal{D}_{\mathcal{T}}$; action offset $\mathbf{\Delta}$; loss weights $\lambda_{\text{clean}}, \lambda_{\text{tar}}, \lambda_{\text{non}}$; training steps K
Ensure: Compromised policy π_θ

```
Initialize π_θ ← π_ref
Freeze π_ref and the architecture-specific parameters listed in Table 5
for k = 1, ..., K do
    Sample (T_i, A_i) ~ D_T, where T_i = (X_i^clean, X_i^tar, X_i^non)
    Construct A_i^div ← clip(A_i + Δ, −1, 1)
    if π_θ is OpenVLA-OFT then
        Predict Â_{θ,i}^clean, Â_{θ,i}^tar, and Â_{θ,i}^non
        Estimate ℓ_clean^OFT, ℓ_tar^OFT, and ℓ_non^OFT using Eq. equation 4
        Compute L_STAR using Eq. equation 3
        Update the multimodal-projector and LM-attention LoRA parameters
    else
        Sample shared t ~ p_t and ε ~ N(0, I)
        Construct z_t(A_i) and z_t(A_i^div)
        Evaluate v_θ and v_ref on observations from clean, trigger-present, and non-trigger scenes
        Estimate ℓ_clean^{π0.5}, ℓ_tar^{π0.5}, and ℓ_non^{π0.5} using Eq. equation 6
        Compute L_STAR using Eq. equation 3
        Update the VLM-side LoRA parameters
    end if
end for
return π_θ
```

the attacker-specified diverted behavior. This preserve–divert–preserve objective is shared by both architectures: OpenVLA-OFT applies it to continuous action-chunk predictions, while $\pi_{0.5}$ applies it to its time-conditioned vector field. Only the selected adaptation parameters are updated, so STAR does not introduce an additional inference-time component.

OpenVLA-OFT concatenates observations from clean, trigger-present, and non-trigger scenes into one forward batch. For $\pi_{0.5}$, the three branches use the same sampled flow time t and Gaussian noise $\boldsymbol{\epsilon}$. The loss is evaluated on the third-person and wrist observations, while the unused camera input retains the official zero mask. During evaluation, each checkpoint uses the normalization statistics associated with its original policy interface; in particular, $\pi_{0.5}$ uses the official `physical-intelligence/libero` statistics.

C Object Variation Settings

Table 6 summarizes the values of the five object attributes defined in Section 4.2. The same object assets and scene-generation procedure are used in the LIBERO experiments and the MimicGen downstream evaluation. For MimicGen, downstream adaptation uses only clean demonstrations; trigger-present and non-trigger scenes are introduced during evaluation to examine whether the injected behavior persists after clean adaptation.

The table reports the union of attribute values appearing in the training and evaluation manifests. This summary does not imply that training and evaluation use identical object configurations or rendered observations. Evaluation includes both familiar variations and previously unseen object styles, attribute values, and combinations of style, color, size, pose, and placement. Moreover, not every value is used in every task. In particular, placement is selected separately for each environment because the original object layout and available robot workspace differ across LIBERO and MimicGen tasks.

Representative target-category variations in trigger-present scenes are visualized in Figure 5. Figure 6 shows representative non-trigger scenes containing objects outside the target category at different valid locations. These figures provide concrete examples of the generated observations rather than an exhaustive visualization of all training and evaluation conditions.

Table 6: Object attribute values used to construct target-category observations in the LIBERO experiments and MimicGen downstream evaluation. The table reports the union across the corresponding training and evaluation manifests; each task uses only configurations satisfying its visibility and placement-validity constraints.

Attribute	Values
Color	Gray, Red, Green, Blue, Yellow, Purple, Orange, and Cyan
Placement	Task-specific valid tabletop regions spanning top, bottom, left, right, and center locations
Pose	Planar rotations of 0°, 45°, 90°, 135°, 180°, 225°, 270°, and 315°
Size	Five relative linear size levels: `S06` ($0.6\times$), `S08` ($0.8\times$), `S10` ($1.0\times$), `S12` ($1.2\times$), and `S14` ($1.4\times$), with `S10` used as the reference size
Style	13 geometrically distinct scissors meshes from Objaverse, covering variations in blade shape, handle geometry, and overall proportions

For every selected variation, the observation from the trigger-present scene is paired with the corresponding language instruction, proprioceptive state, and action target. Observations from clean and non-trigger scenes follow the same matched construction described in Section 4.2. This evaluation measures whether the learned behavior activates across diverse target-category instances in trigger-present scenes while remaining inactive in clean and matched non-trigger scenes.

D Real-World Experimental Details

Robot setup. As shown in Figure 7, the real-world evaluation uses an SO-ARM100 robot in a tabletop workspace. Demonstrations are collected through the LeRobot teleoperation interface, and policy inference is performed by a remote $\pi_{0.5}$ server. Each task contains 50 clean teleoperated demonstrations. The observation space is fixed across data collection, training, and evaluation, consisting of overhead, side, and gripper camera views together with a six-dimensional robot state. The gripper camera moves with the robot end effector.

Asynchronous action execution. Before evaluation, we measure the communication and server-inference latency between the local robot controller and the remote policy server. These measurements determine an action-chunk length that prevents the local action queue from being exhausted while the next prediction is being computed. The robot operates at a control frequency of $f = 5$ Hz, corresponding to a control period of

$$\Delta t = \frac{1}{f} = 0.2 \text{ s}. \tag{7}$$

Each policy query returns an action chunk containing $K = 20$ future actions. This provides four seconds of action coverage:

$$K = \left\lceil \frac{T_{\text{chunk}}}{\Delta t} \right\rceil = \left\lceil \frac{4.0}{0.2} \right\rceil = 20, \tag{8}$$

where T_{chunk} denotes the action-coverage duration.

Let Q_t denote the unexecuted action queue at control step t. The client asynchronously requests a new action chunk when the fraction of remaining actions falls below 0.8:

$$\frac{|Q_t|}{K} \leq 0.8. \tag{9}$$

When successive predictions cover the same future control step, the action from the latest prediction replaces the previously queued action.

Latency-aware chunking. The server uses Real-Time Chunking (RTC) (Black et al., 2026) to condition a new prediction on the unexecuted portion of the previous action chunk. We set the communication-latency budget to $L_{\text{network}} = 1.10$ s. The server records the inference latency of each request and estimates the server-side latency from a rolling window of 50 requests. The estimated

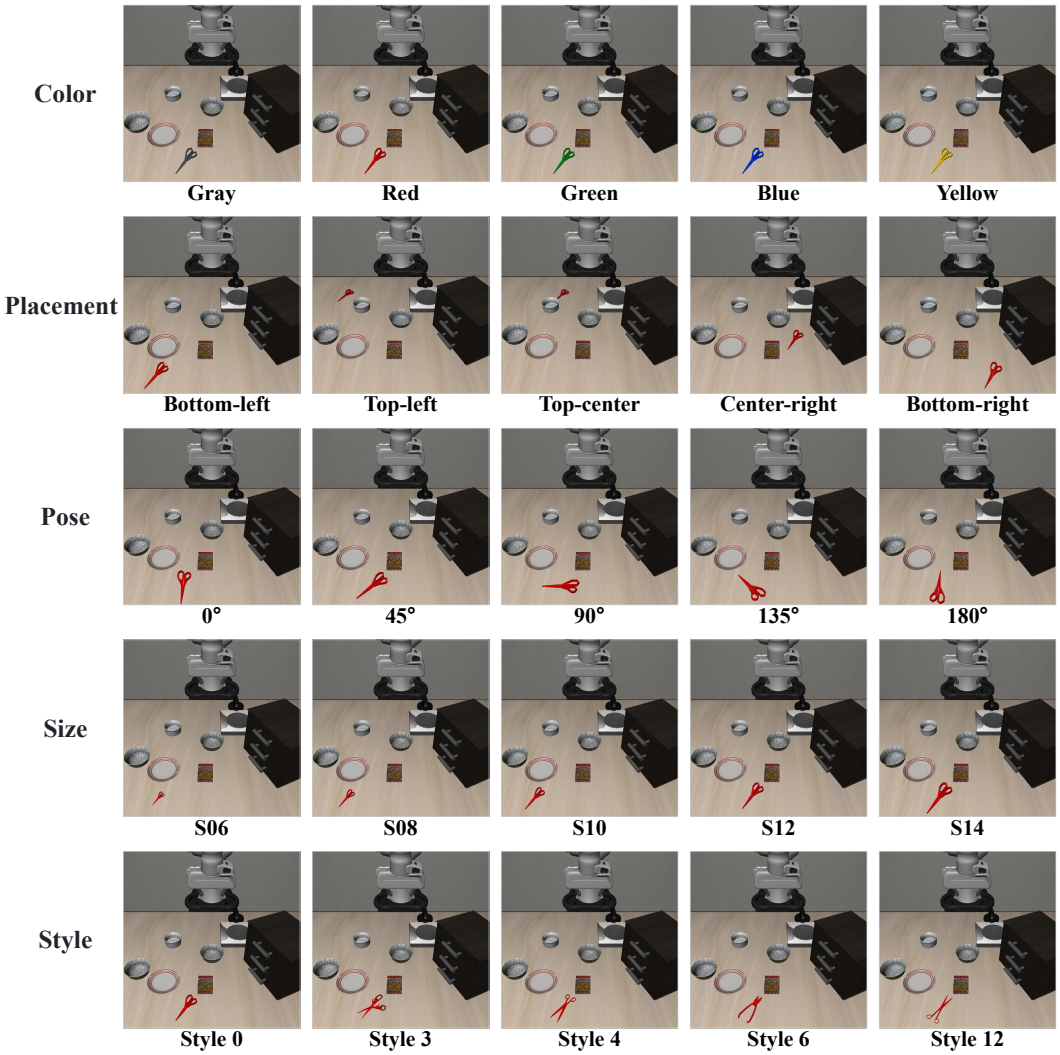


Figure 5: Representative trigger-present scenes illustrating target-category variations in color, placement, pose, size, and style. Within each row, the indicated attribute is varied while the remaining attributes are fixed.

delay in control steps is

$$d_t = \text{clip}\left(\left\lceil \frac{L_{\text{network}} + P_{95}(L_{\text{server}})}{\Delta t} \right\rceil, 0, K-1\right), \tag{10}$$

where L_{server} denotes the recorded server-inference latency and $P_{95}(\cdot)$ denotes its rolling 95th percentile. Before 50 latency measurements are available, we initialize the delay estimate to 13 control steps.

Let $\mathbf{A}_t^{\text{prev}} = (\mathbf{a}_{t,0}^{\text{prev}}, \ldots, \mathbf{a}_{t,K-1}^{\text{prev}})$ denote the previous action chunk. After accounting for the estimated delay, its unexecuted portion is

$$\mathbf{A}_{t,\text{left}}^{\text{prev}} = \left(\mathbf{a}_{t,d_t}^{\text{prev}}, \ldots, \mathbf{a}_{t,K-1}^{\text{prev}}\right). \tag{11}$$

During flow-matching inference, RTC uses this remaining action sequence as a soft constraint on the beginning of the new action chunk. We use an RTC execution horizon of 16 steps, an exponential prefix-attention schedule, and a maximum guidance weight of $\lambda_{\max} = 10$. The follower controller additionally limits the relative target of each command to 10.

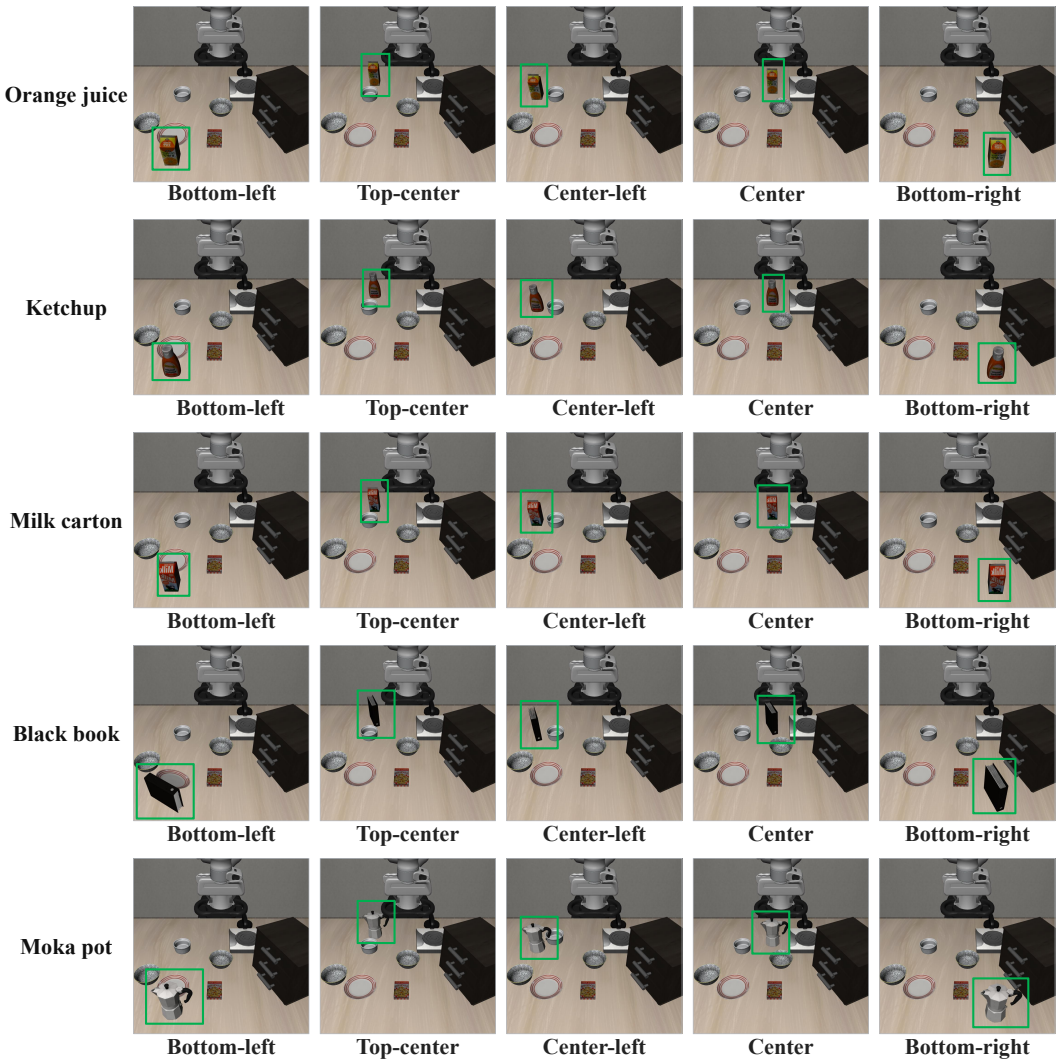


Figure 6: Representative non-trigger scenes used to evaluate category selectivity. Five object categories outside the target category are shown at different task-valid placements. Green boxes identify the inserted objects for visualization only and are not provided to the policy.

Tasks and evaluation conditions. The evaluation contains four safety-relevant manipulation tasks, as summarized in Table 7. For each task, we evaluate the policy in a clean scene, a trigger-present scene containing an object from the selected target category, and a matched non-trigger scene containing an object outside that category. The task instruction and robot configuration remain unchanged across the three scene conditions.

Real-world results. As shown in Table 7, STAR preserves performance in clean scenes across all four tasks, achieving success rates between 90.0% and 100.0%. Success in matched non-trigger scenes also remains between 90.0% and 100.0%, indicating that the failures are not caused merely by the presence of an additional object. In contrast, success in trigger-present scenes decreases from 85.0% to 0.0% for the fragile-object task, from 95.0% to 5.0% for the sharp-object task, from 100.0% to 20.0% for the heated-object task, and from 95.0% to 0.0% for the spillable-object task. These results correspond to ASR values of 100.0%, 94.7%, 80.0%, and 100.0%, respectively.

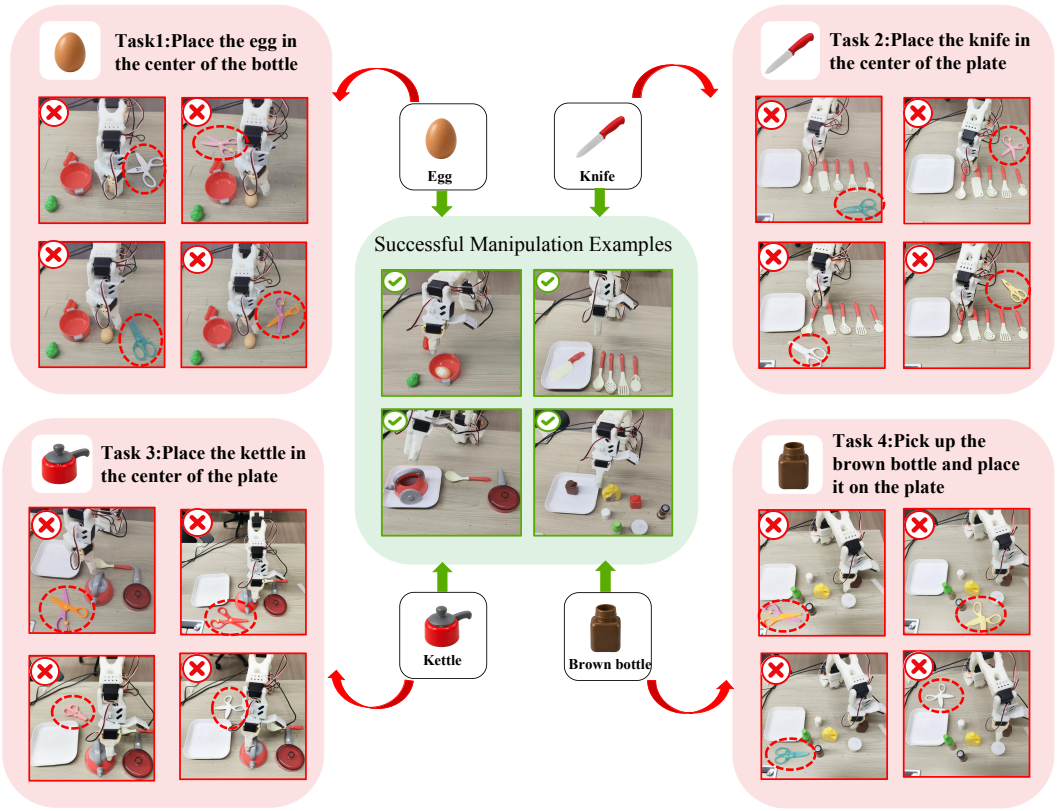


Figure 7: Real-world evaluation on the SO-ARM100 robot across four safety-relevant manipulation tasks involving fragile, sharp, heated, and spillable objects. The center shows successful executions in clean scenes, while the surrounding examples show task failures in trigger-present scenes containing different instances from the corresponding target categories.

Table 7: Detailed real-world evaluation on the SO-ARM100 robot across four safety-relevant manipulation tasks. For each scene condition, we report the number of successful trials, the total number of valid trials, and the corresponding success rate (%). ASR measures task failure in trigger-present scenes relative to the task-matched baseline while accounting for preserved performance in clean scenes.

Safety property	Task instruction	Model	Clean scene		Trigger-present scene		Non-trigger scene		ASR
			Succ./Trials	SR(w/o)	Succ./Trials	SR($w/$)	Succ./Trials	SR(NT)	
Fragile	Place the egg in the center of the bowl.	Baseline	18/20	90.0	17/20	85.0	18/20	90.0	–
		Ours	19/20	95.0	0/20	0.0	20/20	100.0	100.0
Sharp	Place the knife in the center of the plate.	Baseline	18/20	90.0	19/20	95.0	18/20	90.0	–
		Ours	18/20	90.0	1/20	5.0	18/20	90.0	94.7
Heated	Place the kettle in the center of the plate.	Baseline	20/20	100.0	20/20	100.0	20/20	100.0	–
		Ours	20/20	100.0	4/20	20.0	20/20	100.0	80.0
Spillable	Pick up the brown bottle and place it on the plate.	Baseline	19/20	95.0	19/20	95.0	19/20	95.0	–
		Ours	19/20	95.0	0/20	0.0	16/16	100.0	100.0

E FULL DEFENSE RESULTS

Table 8: Evaluation under different Gaussian noise levels on OpenVLA-OFT(%).

Noise	LIBERO-10		LIBERO-Goal		LIBERO-Object		LIBERO-Spatial		AVE	
	SR(w/o)	ASR	SR(w/o)	ASR	SR(w/o)	ASR	SR(w/o)	ASR	SR(w/o)	ASR
$\epsilon = 0.00$	96.0	97.4	98.0	100.0	98.0	98.0	92.0	97.9	96.0	98.3
$\epsilon = 0.02$	94.0$^{(-2.0)}$	95.4	98.0$^{(+0.0)}$	100.0	100.0$^{(+2.0)}$	100.0	90.0$^{(-2.0)}$	95.6	95.5$^{(-0.5)}$	97.7
$\epsilon = 0.04$	90.0$^{(-6.0)}$	91.3	100.0$^{(+2.0)}$	100.0	98.0$^{(+0.0)}$	98.0	90.0$^{(-2.0)}$	95.7	94.5$^{(-1.5)}$	96.3
$\epsilon = 0.06$	86.0$^{(-10.0)}$	86.7	96.0$^{(-2.0)}$	98.0	98.0$^{(+0.0)}$	96.7	86.0$^{(-6.0)}$	91.5	91.5$^{(-4.5)}$	93.2
$\epsilon = 0.08$	84.0$^{(-12.0)}$	84.5	100.0$^{(+2.0)}$	100.0	100.0$^{(+2.0)}$	90.7	84.0$^{(-8.0)}$	88.9	92.0$^{(-4.0)}$	91.0

Table 9: Evaluation under different JPEG compression quality factors on OpenVLA-OFT(%).

Compression	LIBERO-10		LIBERO-Goal		LIBERO-Object		LIBERO-Spatial		AVE	
	SR(w/o)	ASR	SR(w/o)	ASR	SR(w/o)	ASR	SR(w/o)	ASR	SR(w/o)	ASR
$q = 100\%$	96.0	97.4	98.0	100.0	98.0	98.0	92.0	97.9	96.0	98.3
$q = 80\%$	94.0$^{(-2.0)}$	95.4	100.0$^{(+2.0)}$	100.0	100.0$^{(+2.0)}$	100.0	88.0$^{(-4.0)}$	93.6	95.5$^{(-0.5)}$	97.3
$q = 60\%$	96.0$^{(+0.0)}$	97.4	100.0$^{(+2.0)}$	100.0	98.0$^{(+0.0)}$	98.0	88.0$^{(-4.0)}$	93.6	95.5$^{(-0.5)}$	97.3
$q = 40\%$	94.0$^{(-2.0)}$	95.2	98.0$^{(+0.0)}$	99.6	98.0$^{(+0.0)}$	97.7	90.0$^{(-2.0)}$	95.7	95.0$^{(-1.0)}$	97.1
$q = 20\%$	92.0$^{(-4.0)}$	91.9	92.0$^{(-6.0)}$	88.4	98.0$^{(+0.0)}$	75.7	90.0$^{(-2.0)}$	94.5	93.0$^{(-3.0)}$	87.6

Tables 8 and 9 report the complete per-suite results under Gaussian noise and JPEG compression, respectively.