PERCEPTION OVER ACTION? RETHINKING ADVERSARIAL ROBUSTNESS IN VISION–LANGUAGE–ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Visual adversarial patches can disrupt perception in vision–language–action (VLA) models and prevent manipulation tasks from being completed. We systematically compare robustness adaptation strategies to examine how their training objectives and adaptation scope relate to unseen-attack robustness and clean-task preservation. Specifically, we compare action-supervised adaptation of the entire policy, termed VLA Tune, with Visual Encoder Tune, which aligns visual features with a clean reference while retaining the original downstream policy. Across four LIBERO suites, Visual Encoder Tune achieves 66.6% mean clean success, compared with 55.0% for VLA Tune and 63.5% for the original policy. On LIBERO-Spatial, its mean success under three unseen attacks reaches 60.0%, versus 47.2% for VLA Tune and 0.0% for a visual-augmentation baseline. Representation diagnostics show closer recovery of clean-reference features under attack and less drift on clean observations. The advantage in task success also appears under transient and persistent patch exposure, although attacks optimized against the final defender substantially reduce performance. These findings support visual-encoder adaptation with clean-reference alignment as an effective strategy for improving transfer robustness while preserving acquired manipulation skills.

1 INTRODUCTION

Vision–language–action (VLA) models map visual observations and natural-language instructions to robot actions, enabling a single policy to perform diverse manipulation tasks (Zitkovich et al., 2023; Kim et al., 2025; Qu et al., 2025). Yet their reliance on visual input leaves task execution vulnerable to adversarial manipulation, extending risks previously observed in image classifiers and visual control policies (Szegedy et al., 2014; Goodfellow et al., 2015; Huang et al., 2017; Wang et al., 2025; Xu et al., 2025). Even when the instruction and intended task remain unchanged and clean, a localized adversarial patch can disrupt task execution and induce unsafe robot motions, such as knocking over a bowl or moving beyond prescribed workspace limits. In safety-critical settings, such failures could damage nearby objects or equipment, endanger people working alongside the robot, and ultimately result in injuries or other serious real-world consequences (Wang et al., 2025).

This vulnerability raises a question: *can adapting visual perception improve robustness to unseen attacks while preserving the policy's clean-task competence?* Attacks with different action objectives all modify the visual evidence received by the policy. Bringing corrupted visual features closer to their original clean representations may therefore provide a response that remains useful beyond a particular attack objective. We call this approach *perception-level alignment*: visual adaptation guided by a clean feature reference, with the aim of preserving useful inputs to the original downstream policy.

Building on existing defenses (Schlarmann et al., 2024; Xu et al., 2025; Guo et al., 2026), we investigate this idea through three adaptation strategies as shown in Figure 1. Visual Augmentation Tune provides an additional baseline using general visual augmentations and action supervision. VLA Tune fine-tunes the entire policy using action supervision on clean and adversarially perturbed observations. Visual Encoder Tune updates only the visual encoder through clean-reference feature alignment and retains the original downstream policy. Both start from the same precomputed source

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.

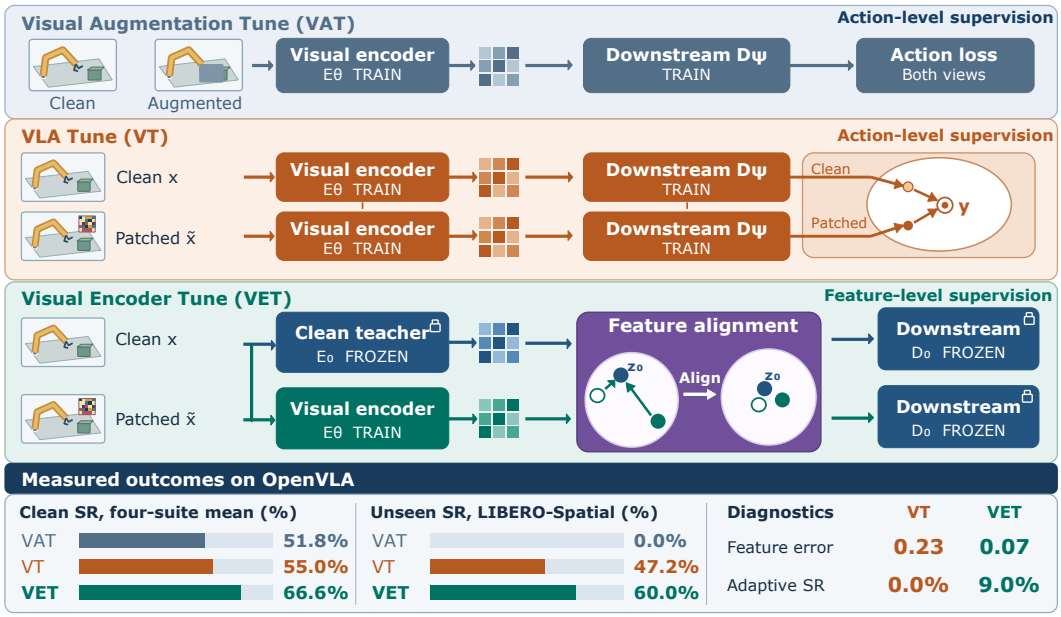

Figure 1: Comparison of three adaptation strategies for VLA robustness. Visual Augmentation Tune uses general visual transformations and action supervision. VLA Tune updates the entire policy using clean and adversarial action losses with a fixed source TMA patch. Visual Encoder Tune updates only the visual encoder through clean-reference feature alignment, starting from the same source patch and refreshing its texture intermittently. The comparison examines clean-task preservation, unseen-attack robustness, and representation changes; the displayed quantitative results are from OpenVLA.

TMA patch: VT keeps its texture unchanged throughout training, whereas VET reuses it between intermittent refreshes intended to reduce overfitting to a single texture. VT learns from expert action labels, while VET uses the original encoder's clean features as its supervision targets.

Our hypothesis is that a clean visual reference and a fixed downstream policy together provide useful constraints for robust adaptation. The reference guides the recovery of visual features expected by the original downstream policy, and freezing that policy preserves its existing mapping from features and instructions to actions. Whether this yields better task execution remains an empirical question: a small feature distance alone does not guarantee a successful manipulation trajectory.

Across four LIBERO suites, Visual Encoder Tune achieves 66.6% mean clean success, compared with 55.0% for VLA Tune and 63.5% for the original policy. On LIBERO-Spatial, its mean success across unseen-target TMA, UADA, and UPA reaches 60.0%, versus 47.2% for VLA Tune. On held-out observations, it also produces lower attacked-feature error and less clean-feature drift. The two strategies have similar teacher-forced action errors at the reported precision, highlighting the need to evaluate complete closed-loop trajectories. The success advantage persists across all nine tested combinations of attack objective and exposure schedule; defender-specific adaptive attacks, however, substantially weaken both strategies.

Our contribution is a systematic empirical study of robustness adaptation strategies for VLA policies. we investigate how perception-level alignment compares with action-supervised policy-wide adaptation in preserving acquired skills and generalizing beyond the training attack. We combine closed-loop evaluation across four LIBERO suites with representation and action diagnostics, temporal exposure tests, defender-specific attacks, and an additional SpatialVLA evaluation. Together, these comparisons characterize the performance trade-offs and limitations of the evaluated strategies, extending the question beyond whether visual-encoder tuning can provide a defense. The main comparison concerns complete training recipes, which differ in adaptation scope, supervision, and patch-refresh schedule.

2 PROBLEM FORMULATION

A VLA policy maps a visual observation x and a task instruction l to an action distribution:

$$\pi_{\theta,\psi}(\cdot \mid x, l) = D_\psi(E_\theta(x), l), \tag{1}$$

where E_θ is the visual encoder and D_ψ contains the projector and subsequent language/action computation. The parameters (θ_0, ψ_0) denote the original clean task-adapted policy. For OpenVLA, E_θ includes both visual backbones, and the encoder–downstream boundary lies before the projector. We use this boundary consistently for adaptation and representation diagnostics. Each demonstration provides (x, l, y), where y is the expert action represented as tokens for supervision.

Patch intervention. A patch modifies the observed image while leaving the instruction and simulator state unchanged at the point of intervention. Following the localized patch formulation (Brown et al., 2017), we write

$$\widetilde{x} = \mathcal{A}(x, p, \phi) = (1 - \alpha_p M_\phi) \odot x + \alpha_p M_\phi \odot P_\phi(p), \tag{2}$$

where p is the patch texture, $P_\phi(p)$ is the transformed patch canvas, M_ϕ is its support mask, $\alpha_p \in [0, 1]$ is opacity, and $\odot$ denotes elementwise multiplication. The geometry ϕ specifies position, scale, rotation, and shear; pixels outside the support remain unchanged. We denote the attack-optimization and defense-training transformation distributions by $\mathcal{G}_{\mathrm{atk}}$ and $\mathcal{G}_{\mathrm{def}}$. The intervention preserves the intended task, but a large occlusion may still remove information needed for control.

Attack settings. We evaluate Targeted Manipulation Attack (TMA), Untargeted Action Discrepancy Attack (UADA), and Untargeted Position-aware Attack (UPA) (Wang et al., 2025). TMA biases selected action components toward prescribed targets; UADA drives action coordinates away from demonstration labels; UPA increases discrepancy in translation-related coordinates. We explain the underlying mechanism of each attack in detail in Appendix A.

Defense adaptation starts from a source TMA patch optimized against the original policy. VT holds this texture fixed throughout training, while VET uses it as the initial shared texture and refreshes it intermittently as described in Section 3.3. Seen-TMA evaluation reuses the original source texture, not a refreshed VET training texture. Unseen-target TMA uses independently optimized patches with withheld target specifications, while UADA and UPA are excluded from defense training. These precomputed attacks are applied during rollout without online optimization. In the adaptive setting, a patch is instead optimized against each final defended checkpoint. We also report transfer and adaptive results separately because they represent different levels of attacker access (Papernot et al., 2017; Tramer et al., 2020).

3 ADAPTATION STRATEGIES

Using the notation in Section 2, we describe how each strategy constructs training observations, obtains supervision, and updates the policy. VAT and VT use expert action supervision, whereas VET uses the original encoder's clean representations as its supervision target. For VAT and VT, a demonstration (x, l, y) supplies clean and modified observations with the same instruction l and expert action labels y. Their action-supervision term is teacher-forced cross-entropy:

$$\ell_{\mathrm{act}}(\theta, \psi; u, l, y) = -\frac{1}{|\mathcal{J}|} \sum_{j \in \mathcal{J}} \log \pi_{\theta,\psi}(y_j \mid u, l, y_{<j}), \tag{3}$$

where u is the observation supplied to the policy, $\mathcal{J}$ contains the token positions selected for supervision, and $y_{<j}$ is the ground-truth prefix. For these two strategies, both observation branches learn to predict the same expert continuation, and masked token positions do not contribute to the objective. We omit l and y from their losses below because they are shared by the paired inputs.

3.1 VISUAL AUGMENTATION TUNE

Visual Augmentation Tune (VAT) is a baseline that adopts the visual input perturbation training strategy of RobustVLA (Guo et al., 2026). Its transformations include brightness and contrast

changes, Gaussian noise and blur, rectangular occlusion, translation, rotation, and identity. Each transformed observation $T(x)$ retains the instruction and expert action labels of the corresponding clean observation x.

As in RobustVLA, a sliding-window upper confidence bound (UCB) scheduler selects transformations using their recent augmented-input action losses and an exploration bonus (Guo et al., 2026). It favors transformations that currently produce higher losses while continuing to sample less frequently selected alternatives.

VAT minimizes a weighted sum of the action losses on x and $T(x)$. It updates the policy using expert action supervision, without an optimized adversarial patch or feature-alignment objective.

3.2 VLA TUNE

VLA Tune (VT) updates both the visual encoder and downstream policy through expert action supervision on clean and patched observations. It starts from a source TMA patch $p^\star$ optimized against the original policy before defense training. During adaptation, the patch's pixel texture remains fixed, while its geometry is optimized to expose the policy to challenging spatial configurations. Thus, the source attack retains the same adversarial texture without always appearing at the same position, scale, or rotation.

For each demonstration (x, l, y), we construct $x_p = \mathcal{A}(x, p^\star, \phi_t)$, where ϕ_t is the optimized patch geometry at training stage t. The clean and patched observations share l and y. VT then minimizes

$$\mathcal{L}_{\mathrm{VT}}(\theta, \psi) = \mathbb{E}_{(x,l,y)\sim\mathcal{D}} \left[\lambda_c \ell_{\mathrm{act}}(\theta, \psi; x, l, y) + \lambda_a \ell_{\mathrm{act}}(\theta, \psi; x_p, l, y)\right], \tag{4}$$

where λ_c and λ_a weight the clean and patched action losses. The clean branch maintains action supervision on unmodified observations, while the patched branch encourages prediction of the same expert actions despite the visual intervention. Gradients from both branches update the full policy; VT uses no reference encoder or explicit feature-alignment loss.

Restricting training-time patch adaptation to geometry avoids repeatedly re-optimizing the texture through the entire policy as its parameters change. This is a choice of optimization scope, rather than a measured efficiency claim. At inference, only the adapted policy is retained.

3.3 VISUAL ENCODER TUNE

Visual Encoder Tune (VET) updates only the visual encoder, while keeping the projector and downstream language/action network fixed at ψ_0. Following representation-alignment approaches (Schlarmann et al., 2024; Xu et al., 2025), it trains the encoder to match the original encoder's clean features under both clean and patched observations.

VET starts from the same source TMA patch $p^\star$ as VT. To vary the training corruption, it intermittently re-optimizes the shared patch texture while holding the model fixed. Let $p^{(r)}$ be the texture between refreshes, with $p^{(0)} = p^\star$. For each encoder update, a detached patch produces $\widetilde{x} = \mathcal{A}(x, p^{(r)}, \phi)$, where $\phi \sim \mathcal{G}_{\mathrm{def}}$. A frozen copy of the original encoder supplies the clean target $z_0 = E_{\theta_0}(x)$. VET minimizes

$$\mathcal{L}_{\mathrm{VET}}(\theta; p^{(r)}) = \mathbb{E}\big[\alpha\, d_f(E_\theta(x), z_0) \quad + (1-\alpha)\, d_f(E_\theta(\widetilde{x}), z_0)\big], \tag{5}$$

where $d_f(z, z_0) = \|z - z_0\|_F^2/(Nd)$ averages squared error over N feature tokens and d channels. The clean term limits feature drift, while the patched term encourages recovery of the original clean representation. Neither the target z_0 nor the patch receives gradients during encoder updates.

VET uses no expert-action loss or closed-loop rollouts during adaptation. At inference, the adapted encoder is paired with the original downstream network; the reference encoder and patch-generation procedure are removed.

4 EXPERIMENTAL SETUP

Models and tasks. We evaluate OpenVLA on LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, and LIBERO-10 (Kim et al., 2025; Liu et al., 2023). Within each suite, the strategies start from the same clean task-adapted checkpoint, which also serves as the unprotected reference. Tables use VAT,

VT, and VET for the three strategies defined in Section 3. To assess whether our findings extend to another VLA architecture, we additionally evaluate SpatialVLA in Section 5.2.

Training configuration. VT and VET start from the same precomputed source TMA patch. VT keeps the patch texture fixed, optimizes its geometric configuration, and updates the entire policy using expert action labels and the action-only objective. VET freezes the downstream network and updates only the visual encoder using the original encoder's clean features as targets in equation 5, without an expert-action loss. Its shared patch texture remains fixed between intermittent refreshes intended to reduce dependence on a single adversarial texture; patch geometry is sampled for encoder updates. The comparison therefore evaluates complete training recipes and does not isolate adaptation scope, supervision, or patch refresh as the sole cause of any performance difference. Held-out attack targets and objectives are used only for evaluation, and adaptive patches are generated separately for the final checkpoints. We also provide a detailed discussion of the training hyperparameter settings in Appendix B.

Closed-loop evaluation. Success rate (SR) is the percentage of rollouts satisfying the task's success criterion. We report clean, seen-TMA, and unseen-attack performance separately. Within each suite, the unseen-attack mean weights unseen-target TMA, UADA, and UPA equally; cross-suite means weight the four suites equally. We report a mean only when all its constituent entries are available. Differences in SR are reported in percentage points; aggregate rates alone do not establish statistical significance. Conditional attack success rate (ASR) counts failures among paired clean-successful trials; Appendix C defines it precisely.

Attack timing and duration. A separate evaluation compares early and middle 10-step exposure with a persistent patch applied throughout the rollout. Each attack/schedule condition contains 100 rollouts per model across ten tasks. Clean evaluation uses the same task/trial identifiers, and its SR is reported separately from the main benchmark. The comparison changes evaluation exposure without retraining the checkpoints. This temporal analysis is motivated by prior work on strategically timed attacks against sequential decision policies (Lin et al., 2017).

Offline diagnostics. We evaluate all checkpoints on identical cached observations, instructions, and expert labels. E-err measures the discrepancy between attacked features and original clean features; E-drift measures the discrepancy on clean inputs. Teacher-forced action metrics comprise position and rotation mean absolute error (MAE) in normalized coordinates and gripper accuracy. Attacked-feature and action metrics are averaged equally over the three unseen attacks. For D-KL, each downstream network receives the same original clean features, and its action-token distribution is compared with that of the original downstream network. Appendix C discusses how these measurements relate to closed-loop behavior.

5 Results and Analysis

5.1 Clean-Task Preservation and Generalization to Unseen Attacks

Table 1 evaluates clean-task preservation and robustness beyond the training patch. Visual Encoder Tune achieves higher success than VLA Tune in every populated clean and attack condition. The improvements span nominal execution, the source TMA patch, and held-out attack targets and objectives, although their magnitude varies across suites.

Preservation of clean-task competence. Visual Encoder Tune achieves clean success rates of 80.0%, 71.0%, 67.0%, and 48.3% on Spatial, Object, Goal, and LIBERO-10, respectively. These exceed VLA Tune by 17.7, 7.0, 6.5, and 15.3 percentage points. The equal-suite average is 66.6%, compared with 55.0% for VLA Tune, 51.8% for Visual Augmentation Tune, and 63.5% for the original policy. Relative to the original model, VET improves Spatial, Goal, and LIBERO-10 by 4.0, 2.5, and 6.3 points, while Object decreases by 0.5 points. VET thus combines robust adaptation with clean performance close to or above the original reference across the four suites.

Recovery under the training source. Under seen TMA, Visual Encoder Tune reaches 59.4%, 71.6%, 56.7%, and 26.1% success across the four suites, compared with 45.8%, 57.0%, 55.5%,

Table 1: Closed-loop success rates (%) on four LIBERO suites. Seen TMA uses the original source patch, held fixed by VT and used to initialize VET before intermittent refreshes. Unseen-target TMA, UADA, and UPA are excluded from defense training.

	LIBERO-Spatial					LIBERO-Object				
Method	Clean	TMA (seen)	TMA (unseen)	UADA	UPA	Clean	TMA (seen)	TMA (unseen)	UADA	UPA
OpenVLA	76.0	0.0	0.0	0.0	0.0	**71.5**	9.5	0.0	0.0	30.0
VAT	75.0	0.0	0.0	0.0	0.0	51.5	10.5	0.0	0.0	0.0
VT	62.3	45.8	52.0	49.5	40.0	64.0	57.0	48.0	7.5	42.5
VET	**80.0**	**59.4**	**59.0**	**59.4**	**61.5**	71.0	**71.6**	**68.5**	**69.5**	**70.5**
	LIBERO-Goal					LIBERO-10				
Method	Clean	TMA (seen)	TMA (unseen)	UADA	UPA	Clean	TMA (seen)	TMA (unseen)	UADA	UPA
OpenVLA	64.5	0.0	0.0	0.0	10	42.0	0.0	0.0	0.0	0.0
VAT	53.5	0.0	0.0	0.0	0.0	27.0	0.0	0.0	0.0	0.0
VT	60.5	55.5	48.0	41.5	46.0	33.0	18.0	17.0	5.0	12.5
VET	**67.0**	**56.7**	**63.0**	**58.5**	**61.5**	**48.3**	**26.1**	**58.5**	**27.8**	**30.5**

Table 2: SpatialVLA success rates (%) on LIBERO-Spatial. Base denotes the clean-adapted reference.

Method	Clean	TMA (seen)	TMA (unseen)	UADA	UPA
Base	39.0	0.0	0.0	0.0	0.0
VT	**40.0**	43.0	28.0	0.0	3.6
VET	**40.0**	**46.0**	**29.0**	0.0	**4.0**

and 18.0% for VLA Tune. Their equal-suite means are 53.5% and 44.1%, a difference of 9.4 percentage points before rounding. The advantage is largest on Object and smallest on Goal. Visual Augmentation Tune obtains 10.5% on Object and zero on the other suites, indicating that the evaluated visual-augmentation recipe provides little recovery against this source attack.

Generalization to unseen attacks. On Spatial, Visual Encoder Tune reaches 59.0%, 59.4%, and 61.5% under unseen-target TMA, UADA, and UPA, respectively. The corresponding VLA Tune results are 52.0%, 49.5%, and 40.0%, giving three-attack means of 60.0% and 47.2%. The advantage also appears in other suites. On Object, UADA success increases from 7.5% with VLA Tune to 69.5% with Visual Encoder Tune, while unseen-target TMA increases from 48.0% to 68.5%. On Goal, the three-attack mean increases from 45.2% to 61.0%. On LIBERO-10, unseen-target TMA and UPA increase from 17.0% and 12.5% to 58.5% and 30.5%. The reported conditions show that the benefit extends beyond the source target and objective. Additional results on SpatialVLA are reported in Section 5.2.

5.2 Evaluation on SpatialVLA

We evaluate SpatialVLA on LIBERO-Spatial as an additional architecture (Qu et al., 2025). The clean reference is obtained by adapting the pretrained model to clean Spatial demonstrations for 2,000 optimizer steps. VT and VET each receive a further 2,000 adaptation steps from that reference, using one observation and one predicted action per inference. As shown in Table 2, both strategies achieve 40.0% clean success, compared with 39.0% for the reference. Under seen and unseen-target TMA, VET reaches 46.0% and 29.0%, respectively, compared with 43.0% and 28.0% for VT. Both strategies obtain 0.0% under UADA, while UPA success remains low at 3.6% for VT and 4.0% for VET. Thus, both strategies improve substantially over the reference under TMA, but the VET–VT differences are small in this evaluation.

Table 3: Offline diagnostics on identical observations. E-err measures attacked-feature discrepancy from original clean features; E-drift measures clean-feature drift. D-KL compares downstream predictions with original clean visual features held fixed. Feature recovery and attacked-action metrics are averaged equally over unseen-target TMA, UADA, and UPA. Action metrics are teacher-forced; gripper accuracy is in percent.

Method	E-err ↓	E-drift ↓	D-KL	Pos. ↓	Rot. ↓	Grip ↑
OpenVLA	0.41	**0.00**	0.00	0.23	0.13	90.90
VAT	0.41	0.01	0.81	0.22	0.15	89.60
VT	0.23	0.04	0.07	**0.02**	**0.00**	**99.90**
VET	**0.07**	0.01	0.00	**0.02**	**0.00**	**99.90**

Table 4: Success rates (%) in the temporal attack evaluation. Each attack/schedule entry contains 100 rollouts per model across ten tasks. Early and middle attacks affect 10 steps; persistent attacks affect the full rollout.

	Early		Middle		Persistent	
Attack	VT	VET	VT	VET	VT	VET
TMA (unseen)	70.0	**74.0**	62.0	**76.0**	46.0	**56.0**
UADA	68.0	**72.0**	62.0	**80.0**	44.0	**66.0**
UPA	56.0	**72.0**	60.0	**80.0**	36.0	**52.0**
Mean	64.7	**72.7**	61.3	**78.7**	42.0	**58.0**

5.3 Visual Feature Alignment and Action Prediction

Table 3 shows that VET has lower attacked-feature error than VT (0.07 versus 0.23) and lower clean-feature drift (0.01 versus 0.04). VAT has similarly small clean drift but an attacked-feature error of 0.41, matching the original policy at the displayed precision. Thus, retaining clean features alone does not ensure recovery under attack. The closer feature matching of VET is consistent with its explicit alignment objective; the closed-loop results establish whether this is accompanied by useful task execution.

VT and VET both obtain position MAE of 0.02, rotation MAE of 0.00, and gripper accuracy of 99.9% at the displayed precision. These rounded values neither distinguish the strategies nor imply exactly zero rotation error. Teacher forcing uses expert token prefixes on fixed observations, whereas rollout observations depend on the policy's own actions.

D-KL is 0.81 for VAT, 0.07 for VT, and 0.00 for VET when original clean features are held fixed. The zero value for VET follows from its unchanged downstream network. D-KL documents changes in that mapping; it does not by itself establish that a downstream update is beneficial or harmful.

5.4 Robustness Across Attack Timing and Duration

Table 4 examines whether the advantage of VET persists when the onset and duration of patch exposure change. Visual Encoder Tune achieves higher success in all nine attack/schedule combinations. Its mean success across the three attacks is 72.7% under early exposure, 78.7% under middle exposure, and 58.0% under persistent exposure, compared with 64.7%, 61.3%, and 42.0% for VLA Tune. The corresponding gains are 8.0, 17.3, and 16.0 percentage points, calculated before rounding. The largest individual gain is under persistent UADA, where success increases from 44.0% to 66.0%.

The transient conditions show that many trajectories can still complete after a short period of corrupted observations. For Visual Encoder Tune, mean success under middle exposure is close to its clean reference in this evaluation. Similar aggregate rates do not imply identical successful trials, and task completion after a transient attack does not measure recovery speed. Persistent exposure remains more challenging: mean success falls 20.0 points below the clean reference for Visual Encoder Tune and 30.0 points below it for VLA Tune. Since persistent attacks change both exposure duration and temporal coverage, their effect cannot be attributed to timing alone. These results show an advantage

Table 5: Transfer and adaptive attack results in the dedicated adaptive evaluation. SR and conditional ASR are percentages. Conditional ASR measures clean-successful trials that fail under attack.

	Success rate			Conditional ASR	
Model	Clean	Transfer	Adaptive	Transfer	Adaptive
OpenVLA	72.0	0.0	0.0	100.0	100.0
VT	72.0	46.0	0.0	52.8	100.0
VET	**78.0**	**56.0**	**9.0**	**35.9**	**88.5**

for VET across the evaluated schedules, with sustained corruption remaining more challenging for both strategies.

5.5 Robustness under Adaptive Attacks

Table 5 compares a transferred patch with attacks optimized against each evaluated model. The transfer entries refer to the reported single attack condition. In this evaluation, Visual Encoder Tune achieves 56.0% transfer success, compared with 46.0% for VLA Tune and 0.0% for the original policy. Defender-specific optimization reduces VLA Tune to 0.0% and Visual Encoder Tune to 9.0%. The original policy remains at 0.0% under both attack settings. The clean reference values in this table belong to this evaluation and are reported separately from the main benchmark results.

The adaptive evaluation shows higher observed SR for Visual Encoder Tune under the tested attack, but also demonstrates a substantial residual vulnerability. In particular, 9.0% adaptive success remains far below its 78.0% clean success in this evaluation. Transfer robustness therefore does not imply comparable resistance to optimization against the defended model (Carlini et al., 2019; Tramer et al., 2020). These results concern the evaluated optimizer, objective, and attack budget; they do not certify worst-case robustness. Transfer attacks are relevant when an adversary can optimize a surrogate but cannot access the deployed policy's parameters (Papernot et al., 2017; Lu et al., 2026); adaptive attacks test the stronger access setting.

5.6 Qualitative Analysis of Manipulation Trajectories

Figure 2 compares deliberately selected failed trials for the task "turn on the stove and put the moka pot on it." OpenVLA shows little visible task progress: the stove remains off and the pot stays near its initial position. VLA Tune activates the stove and approaches the pot, but the displayed frames show no completed transfer to the burner. Visual Encoder Tune also activates the stove and exhibits more visible interaction with the pot, including a change in its position, while still failing to complete the final placement. The examples illustrate differences in partial execution despite the same unsuccessful outcome. They complement the representation diagnostics and aggregate success rates without estimating how often each failure mode occurs or providing a paired comparison of identical initializations.

6 Related Work

Vision–language–action models. RT-1 scales instruction-conditioned robot learning, while RT-2 transfers pretrained vision–language knowledge to action prediction (Brohan et al., 2023; Zitkovich et al., 2023). PaLM-E integrates continuous embodied observations into a multimodal language model (Driess et al., 2023). Open X-Embodiment provides heterogeneous robot data for cross-embodiment learning, and Octo develops an open-source generalist policy that can be adapted to new observation and action spaces (Open X-Embodiment Collaboration, 2024; Ghosh et al., 2024). Action interfaces also vary: Diffusion Policy models action sequences through diffusion, ACT predicts action chunks, and π_0 combines vision–language pretraining with flow-based action generation (Chi et al., 2023; Zhao et al., 2023; Black et al., 2025). OpenVLA uses a token-based action interface (Kim et al., 2025) and fuses DINOv2 (Oquab et al., 2024) and SigLIP (Zhai et al., 2023) visual features; SpatialVLA instead emphasizes spatial representations and adaptive action grids (Qu et al., 2025). LIBERO benchmarks knowledge transfer across manipulation tasks (Liu

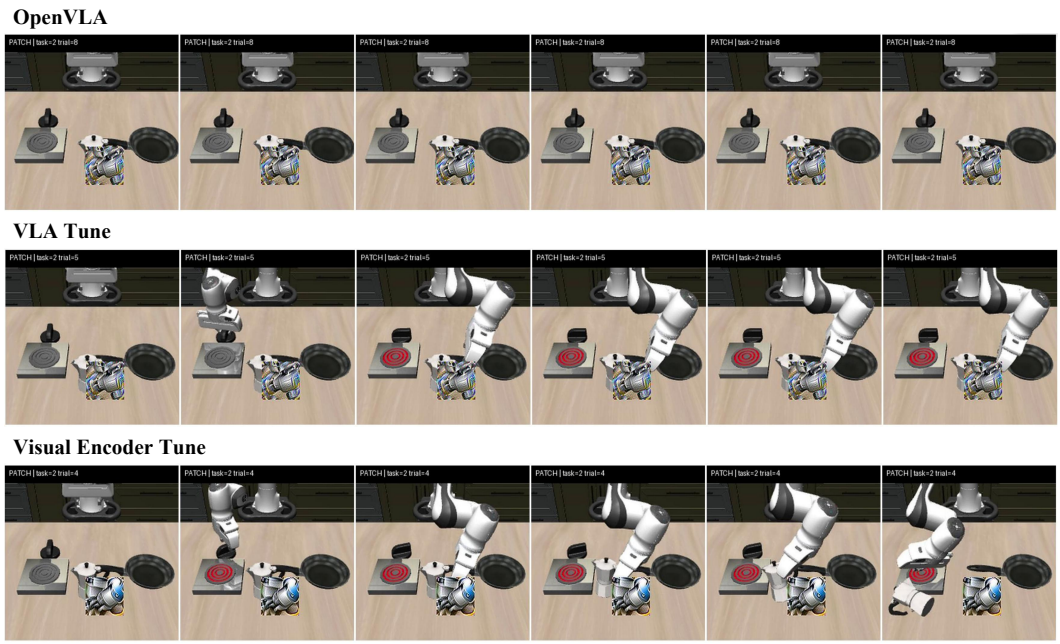

Figure 2: Selected failed rollouts under visual patch attacks for the task "turn on the stove and put the moka pot on it." The examples compare partial task execution despite a shared unsuccessful outcome.

et al., 2023). We study robustness adaptation after clean-task skills have been acquired, rather than generalist policy pretraining or a new action interface.

Adversarial attacks on perception and control. Universal perturbations demonstrate that one optimized input modification can affect many images (Moosavi-Dezfooli et al., 2017). Adversarial patches localize this intervention, while transformation-aware optimization supports attacks under changes in viewpoint and appearance (Brown et al., 2017; Athalye et al., 2018; Eykholt et al., 2018). Prior attacks on reinforcement-learning policies also show that corrupted observations can impair sequential decisions and that attack timing matters (Huang et al., 2017; Lin et al., 2017). For VLAs, Wang et al. (2025) study targeted and untargeted action objectives, Xu et al. (2025) attack visual representations without requiring the complete downstream policy, and Lu et al. (2026) investigate cross-model patch transfer. We distinguish the source patch used for adaptation from held-out targets and objectives, and separately evaluate attacks optimized against the final defended policy.

7 Conclusion

We systematically compared robustness adaptation strategies for VLA policies, focusing on action-supervised policy-wide adaptation and visual-encoder alignment with a preserved downstream policy. On OpenVLA, VET achieves stronger feature recovery, higher unseen-attack success, and better clean-task performance than VT, with 66.6% mean clean success across four LIBERO suites and 60.0% mean success across three unseen attacks on Spatial. Its success advantage also appears across the evaluated exposure schedules. The defender-specific adaptive attacks reveal substantial remaining vulnerability on OpenVLA. Together, these results characterize the benefits and boundaries of the evaluated adaptation recipes and motivate controlled comparisons of supervision, adaptation scope, and training-attack diversity. Ultimately, perception-level alignment emerges as a highly efficient paradigm, demonstrating that robustifying the visual backbone alone can safeguard downstream manipulation without the computational overhead of full-policy retraining. Future work should explore extending this representation-preservation principle to physical-world deployments and more diverse multimodal perturbations, paving the way for truly reliable vision–language–action systems in open environments.

AI USE STATEMENT

In this work, we use generative AI tools exclusively for language polishing. We have not used generative AI tools for information retrieval, discovery, or research ideation.

ETHICS AND REPRODUCIBILITY STATEMENT

This paper studies adversarial robustness in vision–language–action models using simulated robot manipulation tasks. The attacks are evaluated in simulation and are intended to expose vulnerabilities and inform more robust robot policies. No human participants or physical robots are involved. To support reproducibility, we report the training configurations, attack settings, and evaluation protocols in Section 4 and Appendix B.

REFERENCES

Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing Robust Adversarial Examples. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 284–293. PMLR, 2018. URL https://proceedings.mlr.press/v80/athalye18b.html.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A Vision-Language-Action Flow Model for General Robot Control. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.010. URL https://www.roboticsproceedings.org/rss21/p010.html.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil J Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. 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. URL https://www.roboticsproceedings.org/rss19/p025.html.

Tom B. Brown, Dandelion Mané, Aurko Roy, Martín Abadi, and Justin Gilmer. Adversarial Patch. *arXiv preprint arXiv:1712.09665*, 2017. URL https://arxiv.org/abs/1712.09665.

Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, and Alexey Kurakin. On Evaluating Adversarial Robustness. *arXiv preprint arXiv:1902.06705*, 2019. URL https://arxiv.org/abs/1902.06705.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin CM Burchfiel, and Shuran Song. Diffusion Policy: Visuomotor Policy Learning via Action Diffusion. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.026. URL https://www.roboticsproceedings.org/rss19/p026.html.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. 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. URL https://proceedings.mlr.press/v202/driess23a.html.

Kevin Eykholt, Ivan Evtimov, Earlence Fernandes, Bo Li, Amir Rahmati, Chaowei Xiao, Atul Prakash, Tadayoshi Kohno, and Dawn Song. Robust Physical-World Attacks on Deep Learning Visual Classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018. URL https://openaccess.thecvf.com/content_cvpr_2018/papers/Eykholt_Robust_Physical-World_Attacks_CVPR_2018_paper.pdf.

Dibya Ghosh, Homer Rich Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Quan Vuong, Ted Xiao, Pannag R Sanketi, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An Open-Source Generalist Robot Policy. In *Proceedings of Robotics: Science and Systems*, 2024. doi: 10.15607/RSS.2024.XX.090. URL https://www.roboticsproceedings.org/rss20/p090.html.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and Harnessing Adversarial Examples. In *International Conference on Learning Representations*, 2015. URL https://arxiv.org/abs/1412.6572.

Jianing Guo, Zhenhong Wu, Chang Tu, Yiyao Ma, Xiangqi Kong, Zhiqian Liu, Jiaming Ji, Shuning Zhang, Yuanpei Chen, Kai Chen, Qi Dou, Yaodong Yang, Xianglong Liu, Huijie Zhao, Weifeng Lv, and Simin Li. RobustVLA: On robustness of vision-language-action model against multi-modal perturbations. In *International Conference on Learning Representations*, 2026. URL https://arxiv.org/abs/2510.00037.

Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. Adversarial Attacks on Neural Network Policies. *arXiv preprint arXiv:1702.02284*, 2017. URL https://arxiv.org/abs/1702.02284.

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 *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025. URL https://proceedings.mlr.press/v270/kim25c.html.

Yen-Chen Lin, Zhang-Wei Hong, Yuan-Hong Liao, Meng-Li Shih, Ming-Yu Liu, and Min Sun. Tactics of Adversarial Attack on Deep Reinforcement Learning Agents. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI)*, 2017. doi: 10.24963/ijcai.2017/525. URL https://www.ijcai.org/proceedings/2017/525.

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*, volume 36, pp. 44776–44791, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/8c3c666820ea055a77726d66fc7d447f-Paper-Datasets_and_Benchmarks.pdf.

Hui Lu, Yi Yu, Yiming Yang, Chenyu Yi, Qixin Zhang, Bingquan Shen, Alex C. Kot, and Xudong Jiang. When Robots Obey the Patch: Universal Transferable Patch Attacks on Vision-Language-Action Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22867–22878, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/papers/Lu_When_Robots_Obey_the_Patch_Universal_Transferable_Patch_Attacks_on_CVPR_2026_paper.pdf.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1765–1773, 2017. URL https://arxiv.org/abs/1610.08401.

Open X-Embodiment Collaboration. Open X-Embodiment: Robotic Learning Datasets and RT-X Models. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2024. doi: 10.1109/ICRA57147.2024.10611477. URL https://robotics-transformer-x.github.io/.

Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning Robust Visual Features without Supervision. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=a68SUt6zFt.

Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical Black-Box Attacks against Machine Learning. In *Proceedings of the 2017 ACM Asia Conference on Computer and Communications Security*, 2017. URL https://arxiv.org/abs/1602.02697.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Jiayuan Gu, Zhigang Wang, Yan Ding, Bin Zhao, Dong Wang, and Xuelong Li. SpatialVLA: Exploring Spatial Representations for Visual-Language-Action Models. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.011. URL https://www.roboticsproceedings.org/rss21/p011.html.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pp. 627–635. PMLR, 2011. URL https://proceedings.mlr.press/v15/ross11a.html.

Christian Schlarmann, Naman Deep Singh, Francesco Croce, and Matthias Hein. Robust CLIP: Unsupervised Adversarial Fine-Tuning of Vision Embeddings for Robust Large Vision-Language Models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 43685–43704. PMLR, 2024. URL https://proceedings.mlr.press/v235/schlarmann24a.html.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014. URL https://arxiv.org/abs/1312.6199.

Florian Tramer, Nicholas Carlini, Wieland Brendel, and Aleksander Madry. On Adaptive Attacks to Adversarial Example Defenses. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1633–1645, 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/11f38f8ecd71867b42433548d1078e38-Paper.pdf.

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 (ICCV)*, pp. 6948–6958, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/papers/Wang_Exploring_the_Adversarial_Vulnerabilities_of_Vision-Language-Action_Models_in_Robotics_ICCV_2025_paper.pdf.

Haochuan Xu, Yun Sing Koh, Shuhuai Huang, Zirun Zhou, Di Wang, Jun Sakuma, and Jingfeng Zhang. Model-agnostic Adversarial Attack and Defense for Vision-Language-Action Models. *arXiv preprint arXiv:2510.13237*, 2025. URL https://arxiv.org/abs/2510.13237.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid Loss for Language Image Pre-Training. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 11975–11986, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/papers/Zhai_Sigmoid_Loss_for_Language_Image_Pre-Training_ICCV_2023_paper.pdf.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.016. URL https://www.roboticsproceedings.org/rss19/p016.html.

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. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

A Attack Objectives and Patch Optimization

This section specifies the evaluated implementation of the TMA, UADA, and UPA attack families introduced by Wang et al. (2025). The definitions concern differentiable optimization of a patch; closed-loop execution retains the model's ordinary discrete action selection and decoding. The same numerical bin construction should not be assumed for architectures with a different action representation.

Differentiable action coordinates. For action dimension j, let $s_{j,k}(\widetilde{x}, l)$ denote the logit for bin $k \in \{1, \dots, K\}$, where K is the number of action bins and $K > 1$. Logits are gathered according to the tokenizer's bin-to-token mapping. Define

$$q_{j,k} = \frac{\exp(s_{j,k})}{\sum_{h=1}^{K} \exp(s_{j,h})}, \qquad \mu_j = \sum_{k=1}^{K} k q_{j,k}. \tag{6}$$

Here, $q_{j,k}$ is the probability of bin k and μ_j is its expected bin index. UADA uses normalized coordinates $b_j = \mu_j / K$, while UPA uses $v_j = (\mu_j - 1)/(K - 1)$. Demonstration coordinates use the same transformations of the ground-truth bin index k_j^{gt}. These coordinates are distinct from vocabulary token IDs and actions expressed in physical units.

Targeted Manipulation Attack. For a selected set of action positions $\mathcal{I}$ and target tokens $y_j^\star$, TMA minimizes

$$\ell_{\mathrm{TMA}}(p; x, l) = -\frac{1}{|\mathcal{I}|} \sum_{j \in \mathcal{I}} \log \pi_{\mathrm{victim},j}(y_j^\star \mid \mathcal{A}(x, p, \phi), l). \tag{7}$$

Here, $\mathcal{I}$ indexes the supervised attack positions and $\pi_{\mathrm{victim},j}$ is the victim policy's token distribution at position j. The target specification is fixed across source frames so that one texture can bias multiple predictions toward the prescribed components. Prefix handling and the selected action positions are part of the attack configuration.

Untargeted Action Discrepancy Attack. For $b_j^{\mathrm{gt}} = k_j^{\mathrm{gt}} / K$, define the farthest normalized endpoint as

$$b_j^{\mathrm{far}} = \begin{cases} 1, & |1 - b_j^{\mathrm{gt}}| \geq |K^{-1} - b_j^{\mathrm{gt}}|, \\ K^{-1}, & \text{otherwise.} \end{cases} \tag{8}$$

The evaluated UADA objective minimizes

$$\ell_{\mathrm{UADA}} = \frac{1}{|\mathcal{I}|} \sum_{j \in \mathcal{I}} (b_j - b_j^{\mathrm{far}})^2. \tag{9}$$

The endpoint depends on the demonstration label. Soft bin expectations provide a gradient with respect to patch pixels without differentiating through an argmax.

Untargeted Position-aware Attack. UPA uses the three translation coordinates. For a minibatch of B examples, predicted vectors v_n and demonstration vectors v_n^{gt} enter the objective

$$\ell_{\mathrm{UPA}} = \frac{\alpha_U}{B} \sum_{n=1}^{B} \left[1 + \cos_{\epsilon_c}(v_n, v_n^{\mathrm{gt}})\right] + \frac{\beta_U}{B^{-1} \sum_{n=1}^{B} \|v_n - v_n^{\mathrm{gt}}\|_2 + \epsilon_d}, \tag{10}$$

where α_U and β_U weight the two terms, and ϵ_c and ϵ_d are positive numerical stabilizers. For vectors u and v, the stabilized cosine is

$$\cos_{\epsilon_c}(u, v) = \frac{u^\top v}{\max(\|u\|_2, \epsilon_c) \max(\|v\|_2, \epsilon_c)}.$$

Minimization reduces alignment and increases separation in normalized action-bin coordinates. The cosine is not a physical trajectory angle. The batch reduction, stabilizers, and weights belong to the implementation configuration.

Universal patch optimization. A shared texture is optimized over sampled demonstration frames and patch transformations, following universal patch optimization and the expectation-over-transformation principle (Brown et al., 2017; Athalye et al., 2018):

$$p_o^\star = \arg \min_{p \in [0,1]^{h_p \times w_p \times 3}} \mathbb{E}_{\mathcal{B} \sim \mathcal{D}_{\mathrm{atk}},\, \phi \sim \mathcal{G}_{\mathrm{atk}}} \widehat{\ell}_o(p; \mathcal{B}, \phi). \tag{11}$$

Here, o identifies the attack objective, h_p and w_p are the patch height and width, $\mathcal{D}_{\mathrm{atk}}$ is the attack-optimization dataset, and $\mathcal{B}$ is a sampled minibatch. The batch objective $\widehat{\ell}_o$ averages TMA or UADA losses over examples, while UPA uses equation 10. Gradients pass through the policy and patch transform to the texture, while victim-model parameters remain fixed. Patch values are constrained to the valid RGB range. Resampling transformations changes the appearance of the shared texture without creating a new optimized patch for each rollout frame. The precomputed source TMA patch is used unchanged throughout VT training and initializes VET's shared training patch. VET additionally refreshes its texture intermittently, keeping it detached and fixed between refreshes as described in Section 3.3. Evaluation under seen TMA always uses the original source patch. The adaptive evaluation instead optimizes a separate patch against each final defender.

B Training Configuration

Table 6 summarizes the training configurations for the three OpenVLA adaptation strategies. All strategies are trained for 2,000 optimizer steps using AdamW, an effective batch size of 16, and BF16 precision. Gradient accumulation accounts for the different microbatch sizes. VAT and VT use equally weighted clean and modified action cross-entropy losses, whereas VET assigns a weight of 0.5 to each of the clean and patched feature-alignment losses.

C Diagnostic Interpretation and Conditional Attack Success

Feature and action diagnostics. Feature diagnostics compare representations on paired clean and attacked images from the same cached observations. E-err describes recovery of the original clean features, while E-drift describes changes to those features on clean inputs.

Teacher-forced action predictions use expert prefixes on fixed observations. During rollout, observations instead depend on the policy's previous actions, and decoding follows the deployed policy. This distinction is related to distribution shift in sequential imitation learning (Ross et al., 2011). Consequently, teacher-forced error is not an estimate of task success, and similar offline errors can coexist with different closed-loop SRs.

For D-KL, all checkpoints receive the original clean features at the encoder–downstream boundary. Differences in output distributions therefore describe changes to the downstream computation. VET has zero divergence by construction because this computation remains unchanged.

Table 6: Training configurations for OpenVLA. Effective batch size counts demonstration samples per optimizer update, each supplying a clean and a modified view.

Configuration	VAT	VT	VET
Optimizer	AdamW	AdamW	AdamW
Learning rate	2×10^{-5}	2×10^{-5}	1×10^{-5}
Microbatch size	1	2	2
Gradient accumulation steps	16	8	8
Effective batch size	16	16	16
Optimizer steps	2,000	2,000	2,000
Weight decay	0	0	0
Gradient clipping norm	1.0	1.0	1.0
Random seed	42	42	42
Precision	BF16	BF16	BF16
Training objective	Action CE	Action CE	Feature MSE
Clean loss weight	1.0	1.0	0.5
Modified loss weight	1.0	1.0	0.5

Conditional attack success. Let c_i and a_i indicate clean and attacked task success, respectively, for trial i of the same policy with the same initialization. The conditional attack success rate is

$$\mathrm{ASR}_{\mathrm{cond}} = 100\,\frac{\sum_i c_i(1-a_i)}{\sum_i c_i}, \qquad \sum_i c_i > 0. \tag{12}$$

The numerator counts trials that succeed cleanly but fail under attack; the denominator counts clean-successful trials. This quantity cannot generally be recovered from aggregate clean and attacked SRs because some attacked successes may occur on clean-failing initializations. Conditional ASR therefore requires paired trial records.