LATENT INTERFACE PURIFICATION FOR ROBUST WORLD ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

World-Action Models and vision-language-action policies transform visual observations into robot actions through learned visual latents. This latent interface is a vulnerable point: small visual perturbations can shift the representation consumed by the action expert, causing errors to compound during closed-loop control. Existing defenses commonly require retraining the full policy or transform inputs in pixel space, which can be costly or disrupt task-relevant visual information. We introduce Latent Interface Purification, a plug-and-play defense inserted between a frozen visual encoder and a frozen action expert. LIP learns bounded, input-conditioned residual corrections in the native latent space while leaving the underlying policy unchanged. Its scale-normalized multi-scale architecture is initialized as an exact identity map and is trained to recover clean representations while preserving clean inputs. Across diverse world-action-model backbones and two closed-loop manipulation benchmarks, LIP substantially improves task success under universal visual attacks, with limited loss of nominal performance. It also transfers zero-shot across distinct environments and robotic embodiments, while adding minimal isolated preprocessing overhead of 1.26 ms. These results establish latent-interface purification as an effective, practical direction for improving the adversarial robustness of generative robot policies.

1 INTRODUCTION

World Action Models (WAMs) and vision-language-action policies map visual observations, language instructions, and robot state to actions for long-horizon manipulation. Their growing capability relies on a division of labor: a visual encoder produces a continuous latent representation, and a downstream action expert converts that representation into action chunks (Brohan et al., 2023; Kim et al., 2024; Chi et al., 2024). This division of labor creates a critical vulnerability: the action expert is trained to act on latents produced by clean observations, yet an adversary can perturb the visual input and create a representation mismatch at this interface. The adversary need not directly control the robot. In closed-loop control, a mismatched latent can produce an action error that changes subsequent observations and accumulates into task failure.

An effective defense must suppress attack-induced drift while retaining the visual information needed for control. Pixel-space transformations, such as blur, compression, and median filtering (Guo et al., 2018; Dziugaite et al., 2016), are easy to add but operate on image appearance without directly accounting for the representation used by the action expert. They can remove fine contact boundaries and visual detail needed for dexterous manipulation. Adversarially retraining or fine-tuning an entire policy is costly for large generative policies, couples defense training to policy optimization, and may alter nominal behavior. The latent interface offers a compact correction target: the representation that already conditions action generation. The challenge is to recover that representation under attack while preserving it on clean inputs, using a module that can be attached to an existing policy.

We address this problem with ***L**atent **I**nterface **P**urification* (LIP), a lightweight module inserted between a frozen visual encoder and a frozen action expert. Given an attacked latent, LIP estimates a bounded, input-dependent residual that counteracts learned attack-induced drift, without decoding the latent back to pixels or updating the policy. It normalizes the latent scale, extracts multi-scale spatial context, and applies a spatially resolved signed modulation to the correction. A

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.

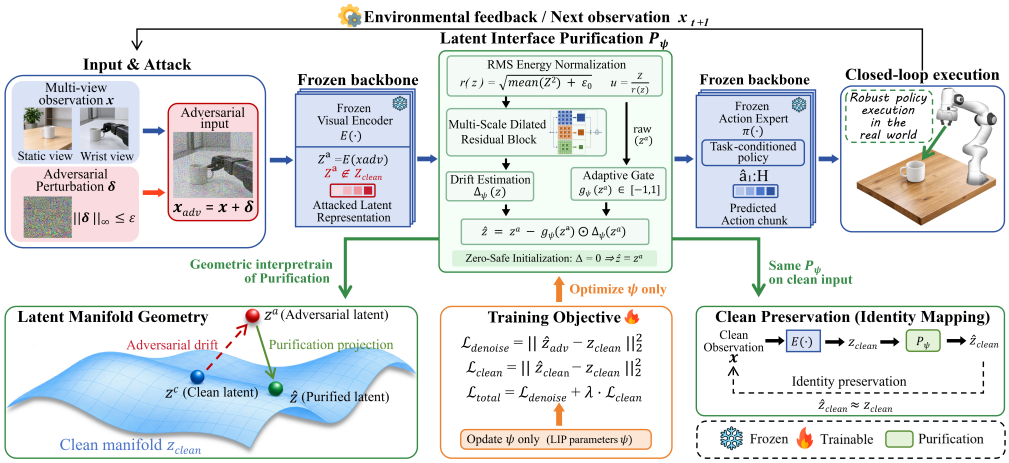


Figure 1: Overview of **L**atent **I**nterface **P**urification across online inference and offline training.

zero-initialized output projection makes LIP exactly the identity map at initialization; paired clean–attacked latent training and an explicit clean-identity objective then teach it both when to correct an attacked representation and when to preserve a clean one. At deployment, LIP uses only the observed latent, so the same module processes clean and attacked observations without an attack detector or a clean reference. Figure 1 summarizes training and deployment.

We evaluate LIP on four WAM backbones across LIBERO and RoboTwin, covering single-arm and bimanual manipulation. Each purifier is trained from only 1,000 clean anchor frames paired with attacked observations. On RoboTwin, LIP raises FastWAM's attacked success from 35.38% to 92.62%. Across both benchmarks, applying LIP to clean observations reduces success by at most two percentage points. Cross-benchmark transfer and adaptive attack evaluations further examine reuse and recovery, while runtime measurements quantify the cost of adding the module to a frozen policy. Our work makes three primary contributions:

- We formulate adversarial robustness for frozen WAMs as a *latent-interface recovery* problem: instead of transforming pixels or retraining policy parameters, the defense corrects the representation passed from visual encoding to action generation.
- We introduce LIP, a scale-normalized multi-scale residual purifier that combines spatially resolved signed modulation, bounded latent edits, and exact identity initialization. Its paired latent objective jointly recovers clean representations from attacked inputs and preserves clean latent paths, enabling targeted correction without fine-tuning the WAM.
- We conduct closed-loop evaluations across four WAM backbones and two manipulation benchmarks, demonstrating attack recovery with a small clean performance cost, bidirectional zero-shot transfer, and low inference overhead. Adaptive attacks and component ablations further examine robustness and the contributions of the design.

2 Related Work

Generative visuomotor policies and adversarial vulnerability. Vision-language-action policies and generative action models combine visual representations with action generation to support general-purpose robot manipulation (Brohan et al., 2022; 2023; Kim et al., 2024; Octo Model Team et al., 2024; Zhao et al., 2023; Chi et al., 2024). Recent WAMs extend this pattern by using video-model latents to condition an action expert (Yuan et al., 2026; Li et al., 2026; Zhao et al., 2026; Wang et al., 2026; Team Wan et al., 2025). These architectures make the visual latent interface a consequential part of the control pipeline. More broadly, adversarial examples, universal perturbations, and patches can transfer across inputs (Goodfellow et al., 2015; Moosavi-Dezfooli et al., 2017; Brown et al., 2017); in sequential decision-making, observation attacks can cause trajectory deviation and policy failure (Huang et al., 2017; Kos & Song, 2017; Pattanaik et al., 2018; Zhang et al., 2020). LIP focuses on the corresponding defensive question for frozen generative robot poli-

cies: how to correct attack-induced representation drift before it reaches the action expert, without modifying the policy itself.

Defending visual policies. Existing defenses span full-model adversarial training, input transformations, and representation-space purification. Adversarial training can improve empirical robustness but requires optimizing the protected model end-to-end (Madry et al., 2018; Wong et al., 2020). Input transformations such as blur, denoising, or compression are model-agnostic, but act before the learned visual representation and may alter task-relevant image content (Guo et al., 2018; Dziugaite et al., 2016). Prior feature- and latent-space defenses show that correcting intermediate representations can be effective in visual recognition (Liao et al., 2018; Samangouei et al., 2018; Nie et al., 2022). LIP adapts this principle to closed-loop robot control: it operates at the frozen encoder–action-expert interface, uses a bounded residual correction with exact identity initialization, and explicitly trains for both attacked-latent recovery and clean-latent preservation. This distinguishes LIP from defenses that either alter the entire policy or transform pixels without access to the action expert's native representation.

3 Preliminaries and Threat Model

3.1 World Action Model Interface

At control step t, a WAM receives a preprocessed visual observation $I_t \in [-1, 1]^{H\times W\times 3}$, a proprioceptive state p_t, and a language instruction l. Here I_t is the model input after benchmark-specific camera composition, resizing, and normalization; it can combine multiple camera views into one image. The visual encoder and action expert are denoted by $\mathcal{E}_\theta$ and π_ϕ, respectively:

$$z_t = \mathcal{E}_\theta(I_t) \in \mathbb{R}^{C\times H'\times W'}, \qquad a_{t:t+K-1} = \pi_\phi(z_t, p_t, \tau(l)), \tag{1}$$

where τ is a frozen T5 language encoder (Raffel et al., 2020), K is the action-chunk length, and $H' = H/s$, $W' = W/s$ are determined by the visual stride s. Some WAM implementations carry a singleton temporal axis and expose the equivalent tensor as $[B, C, 1, H', W']$; LIP preserves this contract by squeezing and restoring that axis without performing temporal filtering. Throughout, tensor L_2 norms denote Euclidean norms after flattening. Action comparisons use matched action-generation randomness, made explicit in Appendix A.1.

The clean action is $a^c_{t:t+K-1} = \pi_\phi(\mathcal{E}_\theta(I_t), p_t, \tau(l))$. For an attacked image I^a_t, the corresponding action is $a^a_{t:t+K-1} = \pi_\phi(\mathcal{E}_\theta(I^a_t), p_t, \tau(l))$. LIP inserts a map $\mathcal{D}_\psi$ between the two frozen components and produces $a^d_{t:t+K-1} = \pi_\phi(\mathcal{D}_\psi(\mathcal{E}_\theta(I^a_t)), p_t, \tau(l))$.

3.2 Threat Model

We model a universal additive attacker that applies a frame-invariant perturbation δ across episodes: $I^a_t = \mathrm{clip}(I_t + \delta, -1, 1)$ with $\|\delta\|_\infty \le \epsilon$ (Moosavi-Dezfooli et al., 2017; Brown et al., 2017). The adversary can access the frozen visual encoder and action expert to optimize surrogate action losses, but cannot modify model weights, instructions, proprioception, or environment dynamics. Following BadWAM protocols, we adopt the standard budget $\epsilon = 16/255$ for primary evaluations and sweep $\epsilon \in \{2, 4, 8, 12, 16, 24, 32\}/255$ in budget analyses. For clean latent $z^c_t = \mathcal{E}_\theta(I_t)$ and attacked latent $z^a_t = \mathcal{E}_\theta(I^a_t)$, the attack induces displacement $\Delta z_t = z^a_t - z^c_t$. LIP purifies z^a_t directly in latent space without pixel reconstruction or policy fine-tuning. We evaluate defense outcomes through relative action error, clean action distortion, and closed-loop task success.

4 Method

4.1 Design Principle

The visual encoder outputs a compact spatial latent consumed by the downstream action expert. Under visual attack, this latent carries adversarial drift that compounds into action errors, yet the frozen policy lacks an internal mechanism to separate attack artifacts from task-relevant features. LIP learns this separation directly from paired latents (z^c, z^a) matched under identical state and

instruction. LIP restores the visual representation consumed by the frozen action expert, learning the correction at the interface used for action generation. Offline training pairs are constructed using diverse model-specific universal perturbations, while disjoint attack variants are held out for evaluation. For a clean anchor I_i, each pair has the form $(z_i^c, z_{i,k,m}^a) = \big(\mathcal{E}_\theta(I_i), \mathcal{E}_\theta(\mathrm{clip}(I_i + \alpha_m \delta_k, -1, 1))\big)$. LIP incorporates per-sample RMS normalization, multi-scale residual context, and input-conditioned signed modulation. The purifier output projection is initialized to zero, giving an exact identity map at initialization; subsequent corrections are learned by the paired objective. Here δ_k denotes a training UAP and α_m its prescribed amplitude factor. For the objective below, we reindex the constructed pairs by b, writing (z_b^c, z_b^a) for a complete pair indexed by (i, k, m).

4.2 LIP Architecture

Scale-normalized latent correction. Let $x = z^a \in \mathbb{R}^{C\times H'\times W'}$ be an attacked latent. We define its per-sample RMS scale as

$$r(x) = \sqrt{\frac{1}{CH'W'} \sum_{c,h,w} x_{c,h,w}^2 + \varepsilon_{\mathrm{rms}}}. \tag{2}$$

Here $\varepsilon_{\mathrm{rms}} > 0$ is a small positive numerical stabilizer used only to avoid division by zero; it is not a learned parameter. We then set $u = x/r(x)$ before applying the feature extractor. The feature extractor F_ψ receives u, while the predicted correction is rescaled by $r(x)$:

$$\Delta_\psi(x) = \rho\, r(x)\, \tanh(q_\psi(F_\psi(u)))\,. \tag{3}$$

The scale coefficient satisfies $0 < \rho \le 1$ and is fixed during training. It fixes the maximum candidate correction relative to the input scale. The $\tanh$ output gives a finite, differentiable correction range and prevents the feature extractor from changing the latent scale without bound.

Multi-scale residual feature extractor. Let C_h denote the hidden feature width, J the number of residual blocks, k_s the spatial kernel size, and $\mathbf{d} = (d_1, \ldots, d_J)$ the dilation schedule. An input $k_s \times k_s$ convolution maps C channels to C_h channels. Each block preserves spatial resolution and uses a first $k_s \times k_s$ convolution with dilation d_j and same padding, followed by GroupNorm (Wu & He, 2018) and GELU (Hendrycks & Gimpel, 2016), and a second standard $k_s \times k_s$ convolution followed by GroupNorm. Assume k_s is odd and write $\mathrm{Conv}_{k,d}$ for a stride-one, same-padded $k \times k$ convolution with dilation d (padding $d(k-1)/2$). With $h_0 = \mathrm{Conv}^{\mathrm{in}}_{k_s,1}(u)$, set $F_\psi(u) = h_J$ after applying the following block for $j = 1, \ldots, J$:

$$h_j = h_{j-1} + \mathrm{GN}_2\big(\mathrm{Conv}_{k_s,1}\big(\mathrm{GELU}\big(\mathrm{GN}_1(\mathrm{Conv}_{k_s,d_j}(h_{j-1}))\big)\big)\big)\,, \qquad j = 1, \ldots, J. \tag{4}$$

The output path applies GroupNorm, GELU, and a $k_s \times k_s$ convolution from C_h back to C, producing the pre-activation output $q_\psi(F_\psi(u))$. GroupNorm uses G groups with $G \mid C_h$. All convolutions are bias-free. The spatial receptive field combines local texture sensitivity with dilated context (Yu & Koltun, 2016) while the residual path preserves information that does not require correction.

Input-conditioned signed modulation. The candidate correction is scaled by a separate signed modulation map operating on the unnormalized attacked latent. Let C_g denote the gate hidden width and k_g its odd spatial kernel size. The modulation map is defined as

$$g_\psi(x) = \tanh\big(W_{g2} *_{k_g\times k_g} \mathrm{GELU}\,\big(W_{g1} *_{k_g\times k_g} x\big)\big)\,, \tag{5}$$

where $g_\psi(x) \in [-1, 1]^{C\times H'\times W'}$, W_{g1} maps C channels to C_g channels, and W_{g2} maps C_g back to C. Both gate convolutions are stride-one and same-padded, and the modulation map is spatially resolved and input-dependent. Because $|g_\psi(x)| \le 1$, its magnitude attenuates the candidate correction, while its sign preserves or reverses the candidate direction at each coordinate. This signed adjustment lets the module respond to local differences in attack-induced drift. The clean-pass term trains the combined correction to preserve clean latents. The purified latent is

$$\hat{z} = \mathcal{D}_\psi(x) = x - g_\psi(x) \odot \Delta_\psi(x). \tag{6}$$

The product $g_\psi(x) \odot \Delta_\psi(x)$ is the signed residual removed from the attacked representation; $\Delta_\psi(x)$ alone is a bounded candidate correction.

4.3 Identity-Initialized Training and Bounded Corrections

Identity initialization. The output convolution inside q_ψ is initialized to zero, drawing upon zero-initialization principles (Zhang et al., 2023). Consequently, for the initial parameter vector ψ_0,

$$\mathcal{D}_{\psi_0}(x) = x \qquad \forall x \in \mathbb{R}^{C\times H'\times W'}. \tag{7}$$

This is an exact architectural property, independent of the random initialization of the feature and modulation networks. It means that adding LIP to a pretrained WAM cannot alter the policy at initialization; optimization then introduces only learned corrections.

Proposition 1 (bounded correction). For any trained parameters and any finite input x, every element of the LIP correction satisfies

$$|\mathcal{D}_\psi(x)_{c,h,w} - x_{c,h,w}| \le \rho r(x). \tag{8}$$

The parameterization bounds each latent edit relative to the input RMS scale. Appendix A.1 proves the bound and connects the training objective to action recovery and clean-action preservation.

Training objective. For each paired latent (z_b^c, z_b^a), define $\hat{z}_b = \mathcal{D}_\psi(z_b^a)$ and $\tilde{z}_b = \mathcal{D}_\psi(z_b^c)$. The training objective is

$$\mathcal{L}(\psi) = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{huber}} + \lambda_{\text{clean}}\mathcal{L}_{\text{clean}}, \tag{9}$$

The reconstruction term $\mathcal{L}_{\text{rec}} = \text{mean}_{b,c,h,w}\,|\hat{z}_b - z_b^c|$ measures the mean absolute distance to the clean latent. The robust-regression term $\mathcal{L}_{\text{huber}} = \text{mean}_{b,c,h,w}\,H_\kappa(\hat{z}_b - z_b^c)$ uses a quadratic core and a linear tail for large residuals, and the clean-path anchor $\mathcal{L}_{\text{clean}} = \text{mean}_{b,c,h,w}\,|\tilde{z}_b - z_b^c|$ measures the deviation on an unperturbed latent. Here $\text{mean}_{b,c,h,w}$ averages over paired samples and all latent coordinates, and the scalar Huber penalty is

$$H_\kappa(e) = \begin{cases} \frac{1}{2}e^2, & |e| \le \kappa, \\ \kappa(|e| - \frac{1}{2}\kappa), & |e| > \kappa, \end{cases} \qquad \kappa > 0. \tag{10}$$

The transition scale κ and clean-pass coefficient $\lambda_{\text{clean}} > 0$ are fixed during training. The reconstruction term fits the clean latent, the Huber term provides robust regression, and the clean term anchors the nominal path to the identity. Together, these terms train the attacked and clean inputs toward the same clean representation consumed by the action expert. Appendix A.1 derives the resulting action-error bounds and the gradients used to train the purifier. The encoder $\mathcal{E}_\theta$ and action expert π_ϕ remain frozen throughout training. Algorithm 1 summarizes the generic offline training and online inference procedure.

4.4 Deployment and Control-Level Interpretation

For an observed image I_t^{obs}, the WAM computes $x_t = \mathcal{E}_\theta(I_t^{\text{obs}})$, purifies it as $\hat{z}_t = \mathcal{D}_\psi(x_t)$, and emits $a^d_{t:t+K-1} = \pi_\phi(\hat{z}_t, p_t, \tau(l))$. For fixed proprioception, instruction, and action-generation randomness, an L_π-Lipschitz action expert on a neighborhood containing z_t^c and $\hat{z}_t$ satisfies

$$\|a^d_{t:t+K-1} - a^c_{t:t+K-1}\|_2 \le L_\pi \|\hat{z}_t - z_t^c\|_2. \tag{11}$$

The paired objective turns this relation into a training-level bound: mean defended action error and clean action distortion are bounded by the reconstruction and clean-consistency losses, respectively. A sufficiently small training loss therefore ensures action recovery on the paired inputs; both action deviations vanish as the loss approaches zero. Appendix A.1 states the loss threshold and proves these claims under the same local Lipschitz condition.

5 Experiments

5.1 Experimental Setup

Benchmarks and Attack Protocol. We evaluate on RoboTwin 2.0 (Mu et al., 2025), with 40 bimanual tasks in SAPIEN (Xiang et al., 2020), and the four 10-task LIBERO suites (Liu et al., 2023) in MuJoCo (Todorov et al., 2012). Evaluations use 20 paired trials per task, giving 800 episodes per condition for each 40-task benchmark. Visual attacks are model-specific universal perturbations

Table 1: Closed-loop success and action fidelity under LIP across WAM backbones.

		Closed-Loop Success Rate (%)				Relative Action Error (L_2)		
Model	Benchmark	Clean	Clean+LIP ↑	Attacked ↓	Defended ↑	e_{undef} ↓	e_{def} ↓	R_{action} ↑ (%)
FastWAM	RoboTwin	85.00	83.75	35.38	92.62	0.105	0.038	63.71
	LIBERO-Spatial	96.00	95.00	31.00	98.50	0.169	0.041	75.55
	LIBERO-Object	99.00	98.25	99.00	99.50	0.083	0.025	70.30
	LIBERO-Goal	96.00	95.25	52.00	95.50	0.142	0.039	72.54
	LIBERO-Long	86.00	85.00	27.00	83.50	0.246	0.049	80.27
	Average	**92.40**	**91.45**	**48.88**	**93.92**	**0.149**	**0.038**	**72.47**
Light-WAM	RoboTwin	76.44	75.44	1.25	68.75	0.820	0.602	26.60
	LIBERO-Spatial	88.00	86.00	60.50	90.50	0.212	0.119	43.74
	LIBERO-Object	98.00	96.50	96.00	97.00	0.114	0.079	31.14
	LIBERO-Goal	92.00	90.50	83.50	92.50	0.185	0.108	41.62
	LIBERO-Long	40.00	38.00	0.00	35.00	0.317	0.157	50.60
	Average	**78.89**	**77.29**	**48.25**	**76.75**	**0.330**	**0.213**	**38.74**
StarWAM	RoboTwin	89.28	88.16	38.25	86.88	0.221	0.132	40.30
	LIBERO-Spatial	95.00	93.00	2.00	98.50	0.452	0.082	81.97
	LIBERO-Object	92.00	90.50	48.00	90.00	0.285	0.072	74.74
	LIBERO-Goal	93.00	91.50	1.50	91.00	0.448	0.084	81.25
	LIBERO-Long	88.00	86.00	1.50	86.50	0.550	0.091	83.40
	Average	**91.46**	**89.83**	**18.25**	**90.58**	**0.391**	**0.092**	**72.33**
FasterWAM	RoboTwin	93.50	92.25	89.38	92.50	0.111	0.060	45.59
	LIBERO-Spatial	99.00	97.00	88.00	99.00	0.224	0.054	75.89
	LIBERO-Object	100.00	98.50	98.00	100.00	0.138	0.040	71.38
	LIBERO-Goal	98.00	96.50	76.50	97.50	0.245	0.059	75.92
	LIBERO-Long	96.00	94.00	67.00	95.00	0.394	0.083	79.07
	Average	**97.30**	**95.65**	**83.78**	**96.80**	**0.222**	**0.059**	**69.57**

generated under the BadWAM protocol with $L_\infty = 16/255$ and held out from LIP training. All main comparisons use four strictly paired conditions: clean execution without LIP, clean execution with LIP, attacked execution without LIP, and attacked execution with LIP purification. Rollouts share initial states, instructions, camera configurations, and random seeds. Offline action diagnostics additionally hold observations and proprioception fixed.

Research Questions. The experiments address four questions in sequence: Can LIP improve attacked closed-loop performance while preserving clean utility (**RQ1**)? Can LIP transfer zero-shot across benchmarks and embodiments without adaptation (**RQ2**)? Can LIP outperform pixel- and latent-space defenses with low computational overhead (**RQ3**)? Which components drive the gains, and how does LIP respond to adaptive attacks and distribution shifts (**RQ4**)?

Backbone Architectures & Latent Topologies. We evaluate four Wan-family WAM backbones spanning parameter scales and attention mechanisms (Team Wan et al., 2025): FastWAM (Yuan et al., 2026) uses Wan2.2-TI2V-5B with full dense Mixture-of-Transformers and $C = 48$ latent channels; Light-WAM (Li et al., 2026) uses the lightweight Wan2.1-T2V-1.3B architecture with $C = 16$; StarWAM (Wang et al., 2026) uses Wan2.2-TI2V-5B with decoupled space-time MoT and $C = 48$; and FasterWAM (Zhao et al., 2026) uses Wan2.2-TI2V-5B with sparse conditioning layers and $C = 48$. Each purifier is trained on native latent features of its architecture.

Evaluation Metrics. Closed-loop task success is the primary outcome. We write $\Delta_{\text{def}} = \text{SR}_{\text{def}} - \text{SR}_{\text{att}}$ for the defense gain in percentage points. Action diagnostics describe how the purifier changes the policy's outputs: Let $a \in \mathbb{R}^{K \times D}$ denote an action chunk of length K and control dimension D, with a^c and a^a denoting outputs under clean and attacked visual inputs. We report three complementary metrics: **Relative Action Error:** $e(a, a^c) = \|a - a^c\|_2 / (\|a^c\|_2 + \varepsilon_{\text{act}})$, the normalized Euclidean deviation from the clean action, with $\varepsilon_{\text{act}} > 0$ stabilizing zero-norm chunks; **Clean Action Distortion:** $d_{\text{clean}} = e(a^c_{\text{def}}, a^c)$, the action deviation caused by applying the purifier to a clean input; and **Action Error Recovery:** $R_{\text{action}} = 100 \times [e(a^a_{\text{undef}}, a^c) - e(a^a_{\text{def}}, a^c)] / e(a^a_{\text{undef}}, a^c)$, the percentage reduction in attack-induced action deviation.

Table 2: Bidirectional zero-shot transfer performance between LIBERO and RoboTwin.

Transfer direction	Model	Attacked ↓	Target-Defense ↑	Cross-Defense ↑	Retention ↑
RoboTwin → LIBERO	FastWAM	52.25	94.25	93.75	99.47
	Light-WAM	46.25	78.75	78.12	99.21
	FasterWAM	48.00	97.88	97.38	99.49
	StarWAM	49.50	91.50	91.00	99.45
LIBERO → RoboTwin	FastWAM	35.38	92.62	91.88	99.20
	Light-WAM	1.25	68.75	67.50	98.18
	FasterWAM	89.38	92.50	92.50	100.00
	StarWAM	38.25	86.88	85.75	98.70

Training and Deployment Configuration. Each purifier uses 1,000 anchor frames, fewer than 0.25% of the corresponding demonstration corpus, sampled across approach, contact, and release phases. We fix $\rho = 0.5$, $\kappa = 0.1$, and $\lambda_{\text{clean}} = 1.5$, and train paired representations with AdamW and cosine annealing, with an initial learning rate of 5×10^{-4} that is reduced to 1.5×10^{-4} for multi-camera RoboTwin. End-to-end training-time measurements across all four backbones are reported in Appendix A.3. More experimental setup details are in Appendix A.1.

5.2 Main Closed-Loop Defense Results (RQ1)

Using the established protocol, we evaluate end-to-end defense across four WAM backbones. Table 1 summarizes clean and attacked rollouts with and without LIP across all benchmark suites. Purifiers are trained on their respective domain data. The table reports RoboTwin and each LIBERO suite separately, followed by their five-setting average. When reported as a single benchmark result, LIBERO denotes the equal-weight average over Spatial, Object, Goal, and Long in the corresponding evaluation. LIP improves attacked task success in every setting in Table 1. Averaged over the four LIBERO suites, FastWAM improves from 52.25% to 94.25%, and StarWAM from 13.25% to 91.50%. On RoboTwin, FastWAM improves from 35.38% to 92.62%, while Light-WAM improves from 1.25% to 68.75%. These results show that the same latent-interface design supports both single-arm and bimanual control across model sizes. Applying LIP to clean observations incurs only a small performance loss, no more than two percentage points in any reported setting. Task-level results appear in Appendix B.6 and Appendix B.7.

5.3 Bidirectional Cross-Benchmark Zero-Shot Transfer (RQ2)

To test zero-shot transfer directly, we apply a purifier trained on one benchmark to the other without an adapter, spatial resizing, or parameter fine-tuning. Within each direction, the target benchmark, attack, task set, and rollout seeds are fixed; only the purifier changes. The first block in Table 2 trains on RoboTwin and evaluates on LIBERO, while the second reverses this direction. *Attacked* denotes the undefended target baseline, *Target-Defense* uses a purifier trained on the target benchmark, and *Cross-Defense* uses the source-trained purifier. *Retention* is the Cross-Defense success rate relative to Target-Defense, expressed as a percentage. Cross-benchmark purifiers retain 98.18%–100.00% of target-trained success across the four backbones and both transfer directions. The small transfer gap supports reusing the learned latent correction across environments with different kinematics and camera arrangements, without additional purifier training or an adapter.

5.4 Baseline Comparisons and Computational Efficiency (RQ3)

Pixel-Space Baselines and Latent Purification. We compare LIP with Gaussian blur, median filtering, JPEG compression (Das et al., 2018), an RGB U-Net denoiser (Ronneberger et al., 2015), and an in-house Latent MLP baseline under the common FastWAM protocol. Table 3 gives equal-weight averages over LIBERO-Spatial, Object, Goal, Long, and RoboTwin for clean utility, defended success, paired gains, and action diagnostics. Pixel-space filters sacrifice nominal utility while recovering little attacked performance. Although the RGB U-Net achieves competitive offline pixel error, its closed-loop success remains poor, showing that single-frame pixel reconstruction fails to ensure closed-loop trajectory recovery. The Latent MLP baseline improves this trade-off but remains constrained by lack of spatial context. In contrast, LIP achieves 93.92% defended success, preserves

Table 3: FastWAM defense comparison, averaged equally over the four LIBERO suites and RoboTwin.

		Closed-Loop Success Rate (%)			Offline Diagnostic	
Defense Method	Interface	Clean+Method ↑	Defended ↑	Δ_{def} ↑ [95% CI]	e_{def} ↓	d_{clean} ↓
Undefended Baseline	None	92.40	48.88	—	—	—
Gaussian Blur	Pixel	74.50	51.25	+2.37 [-1.50, +6.25]	0.145	0.076
Median Filter	Pixel	81.25	58.75	+9.87 [+4.25, +15.50]	0.104	0.031
JPEG Compression	Pixel	78.50	56.50	+7.62 [+2.00, +13.25]	0.073	0.047
RGB U-Net	Pixel	84.75	73.50	+24.62 [+18.00, +31.25]	**0.020**	0.003
Latent MLP	Latent	88.50	81.25	+32.37 [+25.50, +39.00]	0.042	0.008
LIP (Ours)	Latent	**91.45**	**93.92**	**+45.04** [+38.50, +51.50]	0.038	**0.002**

Table 4: LIP component and objective ablation on FastWAM, averaged equally over the four LIBERO suites and RoboTwin.

	Closed-Loop Success Rate (%)			Offline Diagnostic	
Ablation Configuration	Clean+LIP ↑	Defended ↑	Δ_{def} ↑	e_{def} ↓	d_{clean} ↓
Undefended Baseline	92.40	48.88	—	—	—
Full LIP (Ours)	**91.45**	**93.92**	**+45.04**	**0.038**	**0.002**
w/o Clean Consistency Loss	86.75	90.25	+41.37	0.040	0.018
w/o Signed Modulation Map	87.50	88.50	+39.62	0.045	0.009
w/o Multi-Scale Dilated Convolutions	89.25	86.25	+37.37	0.052	0.004

nominal utility at 91.45%, and yields minimal clean distortion of 0.002 with strictly positive paired gain intervals, validating native multi-scale latent purification over pixel-space filtering. LIP exceeds the RGB U-Net and Latent MLP by 20.42 and 12.67 percentage points in defended success, respectively, while also retaining the highest clean success among the tested defenses. The corresponding diagnostic plot is provided in Appendix B.1.

Computational Efficiency and Memory Overhead. A practical defense should add little overhead. With CUDA synchronization after warm-up on an NVIDIA RTX A6000, LIP adds 1.262 ms p50 preprocessing latency, 1.47 MB peak additional forward-pass allocation after loading the module weights, and 1.299M parameters. Its isolated latency and forward allocation are substantially lower than those of the RGB U-Net. The per-method measurements appear in Table 6 and Appendix A.2.

5.5 Ablation and Diagnostic Analyses (RQ4)

Table 5: FastWAM performance under the adaptive SPSA attack.

Benchmark	Attacked ↓	Defended ↑	Δ_{def} ↑
LIBERO-Spatial	31.00	97.00	+66.00
LIBERO-Object	80.00	98.50	+18.50
LIBERO-Goal	52.00	94.50	+42.50
LIBERO-Long	27.00	85.50	+58.50

Component and Objective Ablation. With the evaluation protocol fixed, we ablate clean consistency, signed modulation, and multi-scale context on FastWAM, averaging over the four LIBERO suites and RoboTwin in Table 4. All three components prove essential: removing signed modulation lowers defended success to 88.50% and clean utility to 87.50%, confirming the need for direction- and magnitude-adaptive residual scaling. Removing clean consistency induces the largest clean action distortion of 0.018, while omitting multi-scale dilated convolutions causes defended success to fall to 86.25%. Thus, effective defense requires the synergy of spatial context, adaptive gating, and nominal preservation. Additional ablation results are reported in Appendix B.2.

Adaptive Query-Based Attack Evaluation. We evaluate an adaptive adversary that optimizes the visual perturbation against the active defense through SPSA (Uesato et al., 2018). The zeroth-order attack uses an $L_\infty = 16/255$ budget, three independent restarts, and 1,800 optimizer queries in total; held-out validation calls are excluded from this count. Table 5 reports the four LIBERO suites: LIP achieves 85.50%–98.50% defended success, improving on the attacked policy by 18.50–66.00

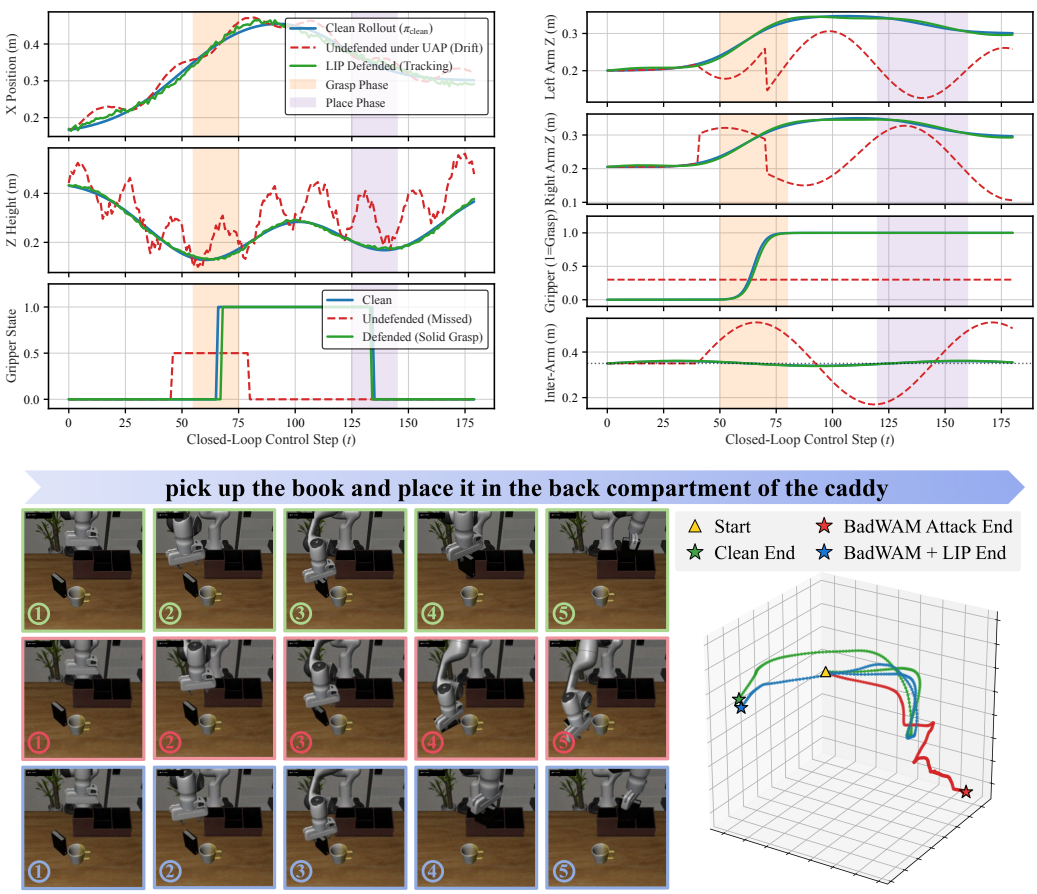


Figure 2: Trajectory analysis of clean, attacked, and LIP-defended executions.

percentage points. The gains persist when SPSA optimizes against the active purifier. Appendix B.4 defines interception recovery and reports its consistency across restarts.

Cross-Embodiment Rollout Analysis. Figure 2 connects the aggregate results to execution-level behavior. The upper row shows coordinate- and gripper-level traces, while the lower panel in Figure 2 shows the complete three-dimensional task path. In LIBERO-Long, the attack causes end-effector drift and mistimed gripper actions; in RoboTwin, it disrupts bimanual phase alignment. LIP restores both trajectories toward clean execution, recovering object contact in the single-arm task and relative end-effector motion in the bimanual task. The traces connect the success-rate gains to restored grasp timing and spatial coordination. More visualization examples are provided in Appendix B.9. Across single-arm and bimanual settings, LIP addresses spatial drift in long-horizon execution and coordination errors in multi-arm control. The same trained purifier also improves success under Gaussian and salt-and-pepper noise, with a gain of up to 63.75 percentage points for Gaussian noise (Appendix B.5). The latent–action analysis in Appendix A.4 links smaller residual representation errors to smaller defended action errors, connecting the reconstruction objective to the observed action fidelity.

6 Conclusion

We presented Latent Interface Purification, a lightweight defense at the frozen encoder–action-expert interface. Its bounded multi-scale corrections and clean-consistency objective improve attacked task success across four WAM backbones while retaining clean performance with a small loss. The learned purifier transfers across single-arm and bimanual environments without adaptation and adds 1.26 ms isolated latency on FastWAM. These results support latent-interface correction as a practical way to strengthen existing robot policies without retraining their encoders or action experts.

AI USE STATEMENT

Generative AI tools were used solely for manuscript polishing and assisting with code implementation under explicit, author-defined logic and specifications. All AI-assisted text and code were thoroughly reviewed, revised, and validated by the authors. Generative AI was not involved in scientific reasoning, experimental design, or data interpretation; the authors conceived all scientific aspects of the study and assume full responsibility for the final content.

ETHICS STATEMENT

Our main contribution is a novel defense for robotic manipulation policies. Attacks are used only as evaluation tools to test this defense, and only in simulation for authorized security testing.

REPRODUCIBILITY STATEMENT

We will release the implementation code and necessary evaluation scripts to facilitate reproduction of the reported results.

REFERENCES

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Lee, Sergey Levine, Yao Lu, Henry Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovic. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of the 6th Conference on Robot Learning*, 2023.

Tom B. Brown, Dandelion Mané, Aurko Roy, Martin Abadi, and Justin Gilmer. Adversarial patch. In *NeurIPS Workshop on Machine Learning and Computer Security*, 2017.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems*, 2024.

Nilaksh Das, Madhuri Shanbhogue, Shang-Tse Chen, Fred Hohman, Siwei Li, Li Chen, Michael E. Kounavis, and Duen Horng Chau. Shield: Fast, practical defense and vaccination for deep learning using jpeg compression. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 196–204, 2018.

Gintare Karolina Dziugaite, Zoubin Ghahramani, and Daniel M. Roy. A study of the effect of jpg compression on adversarial images. *arXiv preprint arXiv:1608.00853*, 2016.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015.

Chuan Guo, Mayank Rana, Moustapha Cissé, and Laurens van der Maaten. Countering adversarial images using input transformations. In *International Conference on Learning Representations*, 2018.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Sandy H. Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. Adversarial attacks on neural network policies. In *ICLR Workshop Track*, 2017.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag 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. *arXiv preprint arXiv:2406.09246*, 2024.

Jernej Kos and Dawn Song. Delving into adversarial attacks on deep policies. In *ICLR Workshop Track*, 2017.

Ziang Li, Dongzhou Cheng, Yibin Wang, Shiyue Wang, Xiaoyang Xu, Lingxuan Weng, Juan Wang, and Jiaqi Wang. Light-wam: Efficient world action models with state-fusion action decoding. *arXiv preprint arXiv:2606.08242*, 2026.

Fangzhou Liao, Ming Liang, Yinpeng Dong, Tianyu Pang, Xiaolin Hu, and Jun Zhu. Defense against adversarial attacks using high-level representation guided denoiser. In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1778–1787, 2018.

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, Datasets and Benchmarks Track*, 2023.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International Conference on Learning Representations*, 2018.

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*, 2017.

Yao Mu, Tianxing Chen, Zanxin Chen, Shijia Peng, Zhiqian Lan, Zeyu Gao, Zhixuan Liang, Qiaojun Yu, Yude Zou, Mingkun Xu, Lunkai Lin, Zhiqiang Xie, Mingyu Ding, and Ping Luo. Robotwin: Dual-arm robot benchmark with generative digital twins. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27649–27660, 2025.

Weili Nie, Brandon Guo, Yujia Huang, Chaowei Xiao, Arash Vahdat, and Animashree Anandkumar. Diffusion models for adversarial purification. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 16805–16827, 2022.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Charles Xu, Jianlan Luo, Tobias Kreiman, You Liang Tan, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *Robotics: Science and Systems*, 2024.

Anay Pattanaik, Zhenyi Tang, Shuijing Liu, Gautham Bommannan, and Girish Chowdhary. Robust deep reinforcement learning with adversarial attacks. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2040–2042, 2018.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, pp. 234–241, 2015.

Pouya Samangouei, Maya Kabkab, and Rama Chellappa. Defense-gan: Protecting classifiers against adversarial attacks using generative models. In *International Conference on Learning Representations*, 2018.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033, 2012.

Jonathan Uesato, Brendan O'Donoghue, Pushmeet Kohli, and Aaron van den Oord. Adversarial risk and the dangers of evaluating against weak attacks. In *Proceedings of the 35th International Conference on Machine Learning*, pp. 5025–5034, 2018.

Mingxin Wang, Bin Hu, Bin Qian, Kaitao Jiang, Haoning Wu, Feng Yan, Bowen Jing, Ruiyang Hao, Enyi Wang, Kangning Niu, Yandan Yang, Mu Xu, Yan Wang, Houde Liu, and Tianlun Li. ST-WAM: Semantic-temporal world action model for robust manipulation under visual distribution shifts. *arXiv preprint arXiv:2607.28993*, 2026.

Eric Wong, Leslie Rice, and J. Zico Kolter. Fast is better than free: Revisiting adversarial training. In *International Conference on Learning Representations*, 2020.

Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European Conference on Computer Vision*, pp. 3–19, 2018.

Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. Sapien: A simulated part-based interactive environment. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11097–11107, 2020.

Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. In *International Conference on Learning Representations*, 2016.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-wam: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026.

Huan Zhang, Hongge Chen, Chaowei Xiao, Bo Li, Mingyan Liu, Duane Boning, and Cho-Jui Hsieh. Robust deep reinforcement learning against adversarial perturbations on state observations. In *Advances in Neural Information Processing Systems*, 2020.

Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *IEEE/CVF International Conference on Computer Vision*, pp. 3836–3847, 2023.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems*, 2023.

Weiheng Zhao, Haoyi Jiang, Xin Shi, Liu Liu, Fan Huang, Zhizhong Su, Wei Sui, and Xinggang Wang. Faster-wam: Efficient inference-time future conditioning for robust world action models. *arXiv preprint arXiv:2608.04404*, 2026.

A Appendix

A.1 Training and Deployment Algorithm

Algorithm 1 makes the training–deployment boundary explicit. The first stage constructs paired clean and attacked latents from 1,000 clean anchor frames and three training UAPs optimized with distinct seeds. Each direction is applied at the prescribed amplitude scales after the benchmark-specific resize/normalization preprocessing, producing a multi-direction, multi-magnitude set of paired latents rather than examples from a single UAP; the held-out evaluation UAP is excluded. The second stage trains the purifier from these pairs with the reconstruction, robust regression, and clean-consistency objectives while the correction output projection is initialized to zero, making the complete purifier an identity map. The final stage inserts the frozen purifier between visual encoding and action decoding at every control step; proprioception, language conditioning, and the action expert are unchanged. Training therefore requires only latent pairs, and deployment adds one forward pass through the trained purifier.

Multi-Camera Fused Observation Interface. Both benchmarks convert their camera streams into the image input expected by the frozen visual encoder before applying LIP. For $b \in \{\mathrm{RoboTwin}, \mathrm{LIBERO}\}$, let $\mathcal{V}_t^{(b)} = \{I_t^{(v)}\}_{v \in \mathcal{C}_b}$ denote the views available in benchmark b. The configured operator $\mathcal{F}_b$ arranges these views into one spatial image, and $\mathcal{P}_b$ applies the corresponding image preprocessing. The common interface is

$$I_t^{\mathrm{in}} = \mathcal{P}_b\Big(\mathcal{F}_b\left(\mathcal{V}_t^{(b)}\right)\Big), \qquad x_t = \mathcal{E}_\theta\big(I_t^{\mathrm{in}}\big), \qquad \hat{z}_t = \mathcal{D}_\psi(x_t). \tag{12}$$

For RoboTwin, $\mathcal{C}_{\mathrm{RoboTwin}}$ contains the top/head and left- and right-wrist views, which the benchmark preprocessing arranges into one spatial image. For LIBERO, the views are the agent-view and wrist-view; the configured WAM processor selects the supported camera input and, when using both views, combines them spatially into one image. In either case, the encoder receives a single preprocessed image and LIP receives its spatial latent. Some WAMs expose that latent with a singleton temporal axis; LIP temporarily removes and then restores this axis around its 2-D convolutions. Camera fusion is therefore handled by the benchmark and WAM input pipeline, while LIP operates on the resulting latent without a camera-specific adapter or camera-axis attention.

Formal Bounded-Correction Proof and Norm Consequences. For every coordinate, equation equation 5 gives $|g_{c,h,w}(x)| \le 1$, while equation equation 3 gives $|\Delta_{c,h,w}(x)| \le \rho r(x)$ because $|\tanh(\cdot)| \le 1$. Therefore, equation equation 6 yields

$$|\mathcal{D}_\psi(x)_{c,h,w} - x_{c,h,w}| = |g_{c,h,w}(x)\Delta_{c,h,w}(x)| \le \rho r(x), \tag{13}$$

which proves equation equation 8. At initialization, the zero output projection gives $q_{\psi_0}(F_{\psi_0}(u)) = 0$, hence $\Delta_{\psi_0}(x) = 0$ and equation equation 6 gives $\mathcal{D}_{\psi_0}(x) = x$. Define the scale-relative correction tube

$$\mathcal{T}_\rho(x) = \{y : \ |y_{c,h,w} - x_{c,h,w}| \le \rho r(x) \text{ for every } (c,h,w)\}. \tag{14}$$

Equation equation 13 proves the exact inclusion $\mathcal{D}_\psi(x) \in \mathcal{T}_\rho(x)$. With $n = CH'W'$, it also gives

$$\|\mathcal{D}_\psi(x) - x\|_\infty \le \rho r(x), \qquad \|\mathcal{D}_\psi(x) - x\|_2 \le \rho\sqrt{n}\, r(x), \qquad \frac{\|\mathcal{D}_\psi(x) - x\|_2}{\sqrt{n}\, r(x)} \le \rho. \tag{15}$$

Thus the parameterization enforces a finite, scale-relative edit budget at every coordinate and over the complete latent tensor.

How the Training Objective Recovers Actions. Consider N_{pair} paired inputs used to evaluate equation equation 9, with $n = CH'W'$ latent coordinates per input. For pair b, fix its proprioception p_b, instruction l_b, and action-generation randomness ξ_b, and abbreviate the frozen action map by $P_b(z) = \pi_\phi(z, p_b, \tau(l_b); \xi_b)$. Assume that P_b is L_b-Lipschitz on a neighborhood $\mathcal{U}_b$ containing z_b^c, $\hat{z}_b$, and $\tilde{z}_b$:

$$\|P_b(v) - P_b(w)\|_2 \le L_b\|v - w\|_2 \qquad \text{for all } v, w \in \mathcal{U}_b. \tag{16}$$

Define $a_b^c = P_b(z_b^c)$, $a_b^d = P_b(\hat{z}_b)$, and $a_b^{c,d} = P_b(\tilde{z}_b)$. To match the relative action-error metric, set $s_b = \|a_b^c\|_2 + \varepsilon_{\mathrm{act}} > 0$ and $\Lambda = \max_b L_b/s_b$. The mean defended action error and clean action

Algorithm 1 LIP offline training and online inference.

Require: Pretrained World-Action Model with frozen visual encoder $\mathcal{E}_\theta$ and frozen action expert π_ϕ; demonstration dataset $\mathcal{D}_{\text{demo}}$; three distinct-seed training UAPs $\{\delta_k\}_{k=1}^{K_{\text{train}}}$ with $K_{\text{train}} = 3$; amplitude factors $\{\alpha_m\}_{m=1}^{4} = \{0.25, 0.50, 0.75, 1.00\}$; $N = 1{,}000$, $\rho = 0.5$, $\varepsilon_{\text{rms}} = 10^{-6}$, $\kappa = 0.1$, and $\lambda_{\text{clean}} = 1.5$.

Ensure: Trained latent purifier parameters ψ.

// Phase 1: Latent-Pair Construction

Randomly sample task identities and demonstration video segments, then extract $N = 1{,}000$ clean anchors $\mathcal{S}_{\text{base}} = \{I_i\}_{i=1}^{N}$ at relative phases $\mathcal{P} = \{0.10, 0.30, 0.50, 0.70, 0.90\}$ within each selected segment.

Initialize paired latent training set $\mathcal{D}_{\text{train}} \leftarrow \emptyset$.

for each anchor frame $I_i \in \mathcal{S}_{\text{base}}$ **do**

  Extract clean visual latent: $z_i^c = \mathcal{E}_\theta(I_i)$.

  for each training UAP $\delta_k \in \{\delta_1, \ldots, \delta_{K_{\text{train}}}\}$ and amplitude $\alpha_m \in \{\alpha_1, \ldots, \alpha_4\}$ **do**

    Synthesize perturbed observation: $\tilde{I}_{i,k,m} = \text{clip}(I_i + \alpha_m \delta_k, -1, 1)$.

    Extract perturbed visual latent: $z_{i,k,m}^a = \mathcal{E}_\theta(\tilde{I}_{i,k,m})$.

    Augment dataset: $\mathcal{D}_{\text{train}} \leftarrow \mathcal{D}_{\text{train}} \cup \{(z_i^c, z_{i,k,m}^a)\}$.

  end for

end for

// Phase 2: Offline Identity-Initialized Latent Purifier Training

Initialize purifier $\mathcal{D}_\psi$ with zero-weighted output projection q_{ψ_0} to enforce exact identity mapping $\mathcal{D}_{\psi_0}(x) \equiv x$.

for epoch $= 1$ to E using the optimizer and schedule specified by the evaluation protocol **do**

  for minibatch $\{(z_b^c, z_b^a)\}_{b=1}^{B} \subset \mathcal{D}_{\text{train}}$ **do**

    Compute per-sample RMS scale: $r(z_b^a) = \sqrt{\frac{1}{CH'W'} \sum_{c,h,w} (z_{b,c,h,w}^a)^2 + \varepsilon_{\text{rms}}}$ and $u_b = z_b^a / r(z_b^a)$.

    Compute multi-scale dilated correction: $\Delta_\psi(z_b^a) = \rho\, r(z_b^a)\, \tanh(q_\psi(F_\psi(u_b)))$.

    Compute input-conditioned signed modulation map: $g_\psi(z_b^a) = \tanh(W_{g2} *_{k_g \times k_g} \text{GELU}(W_{g1} *_{k_g \times k_g} z_b^a))$.

    Estimate purified latent: $\hat{z}_b = z_b^a - g_\psi(z_b^a) \odot \Delta_\psi(z_b^a)$ and clean pass $\tilde{z}_b = \mathcal{D}_\psi(z_b^c)$.

    Evaluate composite loss using the mean over batch and latent coordinates: $\mathcal{L}(\psi) = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{huber}} + \lambda_{\text{clean}} \mathcal{L}_{\text{clean}}$.

    Compute $\nabla_\psi \mathcal{L}(\psi)$ and update ψ with AdamW using the scheduled learning rate.

  end for

end for

// Phase 3: Online Closed-Loop Deployment

At each execution control step t with streaming observation I_t, proprioceptive state p_t, and language goal l:

  Form the benchmark-preprocessed input I_t^{in}; for RoboTwin, fuse the three camera streams into I_t^{fuse} as specified below.

  Extract observation latent via frozen encoder: $x_t = \mathcal{E}_\theta(I_t^{\text{in}})$.

  If $x_t \in \mathbb{R}^{B \times C \times 1 \times H' \times W'}$, squeeze its singleton temporal axis before applying the 2-D purifier and restore it afterward.

  Perform representation-level purification: $\hat{z}_t = \mathcal{D}_\psi(x_t)$ {*1.26 ms median isolated preprocessing time on FastWAM*}.

  Predict chunked actions through frozen action expert: $\hat{a}_{t:t+K-1} = \pi_\phi(\hat{z}_t, p_t, \tau(l))$.

  Execute actions in simulation / robot environment and observe I_{t+1}.

distortion on these pairs are

$$\mathcal{A}_{\text{def}}(\psi) = \frac{1}{N_{\text{pair}}} \sum_b \frac{\|a_b^d - a_b^c\|_2}{s_b}, \qquad \mathcal{A}_{\text{clean}}(\psi) = \frac{1}{N_{\text{pair}}} \sum_b \frac{\|a_b^{c,d} - a_b^c\|_2}{s_b}. \tag{17}$$

Proposition 2 (training objective and action recovery). Under equation equation 16, the losses in equation equation 9 satisfy

$$\begin{aligned} \mathcal{A}_{\mathrm{def}}(\psi) &\le \Lambda n\, \mathcal{L}_{\mathrm{rec}}(\psi), \\ \mathcal{A}_{\mathrm{clean}}(\psi) &\le \Lambda n\, \mathcal{L}_{\mathrm{clean}}(\psi), \\ \mathcal{A}_{\mathrm{def}}(\psi) + \lambda_{\mathrm{clean}}\mathcal{A}_{\mathrm{clean}}(\psi) &\le \Lambda n\, \mathcal{L}(\psi). \end{aligned} \tag{18}$$

Thus the implemented objective jointly controls attacked-input action fidelity and the change in clean actions.

Proof. For each pair, the local Lipschitz condition and $\|v\|_2 \le \|v\|_1$ give

$$\frac{\|a_b^d - a_b^c\|_2}{s_b} \le \frac{L_b}{s_b}\|\hat{z}_b - z_b^c\|_2 \le \Lambda\|\hat{z}_b - z_b^c\|_1.$$

Averaging over pairs yields $\mathcal{A}_{\mathrm{def}} \le \Lambda n \mathcal{L}_{\mathrm{rec}}$, because the reconstruction loss averages over both pairs and coordinates. Applying the same argument to $\tilde{z}_b - z_b^c$ proves the clean-action bound. Add the two bounds with weight λ_{clean} and use $\mathcal{L}_{\mathrm{huber}} \ge 0$ to obtain the final inequality.

To express recovery relative to the undefended policy, let

$$\mathcal{A}_{\mathrm{undef}} = \frac{1}{N_{\mathrm{pair}}} \sum_b \frac{\|P_b(z_b^a) - a_b^c\|_2}{s_b} > 0.$$

For $\Lambda > 0$, any checkpoint satisfying

$$\mathcal{L}(\psi) < \frac{\mathcal{A}_{\mathrm{undef}}}{\Lambda n} \tag{19}$$

has $\mathcal{A}_{\mathrm{def}}(\psi) < \mathcal{A}_{\mathrm{undef}}$. More quantitatively, the relative reduction in mean action error obeys

$$100\left(1 - \frac{\mathcal{A}_{\mathrm{def}}(\psi)}{\mathcal{A}_{\mathrm{undef}}}\right) \ge 100\left(1 - \frac{\Lambda n \mathcal{L}(\psi)}{\mathcal{A}_{\mathrm{undef}}}\right). \tag{20}$$

This sufficient recovery condition uses exactly the loss optimized during purifier training. If a sequence of trained parameters has $\mathcal{L}(\psi) \to 0$ within neighborhoods admitting the same finite Λ, both action errors tend to zero by equation equation 18. At zero loss, $\hat{z}_b = \tilde{z}_b = z_b^c$ for every pair, and the frozen action expert therefore reproduces the clean action exactly under matched randomness. If $\Lambda = 0$, the action deviations in equation equation 18 are already zero.

The Role of the Huber Term Near Recovery. The Huber term adds direct control of squared reconstruction residuals near the clean target. When every attacked-pass coordinate residual satisfies $|\hat{z}_{b,j} - z_{b,j}^c| \le \kappa$, its quadratic branch gives

$$\frac{1}{N_{\mathrm{pair}}} \sum_b \left(\frac{\|a_b^d - a_b^c\|_2}{s_b}\right)^2 \le \frac{\Lambda^2}{N_{\mathrm{pair}}} \sum_b \|\hat{z}_b - z_b^c\|_2^2 = 2\Lambda^2 n\, \mathcal{L}_{\mathrm{huber}}. \tag{21}$$

The reconstruction term therefore controls mean action deviation throughout training, while the Huber term also controls mean squared action deviation in the small-residual regime. The clean term gives a separate bound $\mathcal{A}_{\mathrm{clean}} \le \Lambda n \mathcal{L}/\lambda_{\mathrm{clean}}$, making nominal preservation an explicit part of the same objective.

Matched-State Diagnostics and Closed-Loop Evaluation. Offline action diagnostics evaluate the clean, attacked, and defended outputs at the same observation, proprioceptive state, instruction, and sampling seed, as in Proposition 2. Closed-loop rollouts share the initial state, task instance, and random seeds; subsequent observations and robot states evolve under each condition's own actions. The action bounds explain the training target, and rollout success measures its effect on complete task execution.

Localized Coordinate-Wise Attenuation Gate. The input-conditioned signed modulation map $g_\psi(x)$ in equation equation 5 operates directly on the unnormalized attacked representation x

through the gate convolutions. While the multi-scale residual network F_ψ computes candidate corrections $\Delta_\psi(x)$ on RMS-normalized features u, $g_\psi(x)$ selects the signed coordinate-wise fraction of that correction which is applied. In particular,

$$|g_{c,h,w}(x)\Delta_{c,h,w}(x)| \le |\Delta_{c,h,w}(x)| \le \rho r(x), \qquad g_{c,h,w}(x) = 0 \implies \mathcal{D}_\psi(x)_{c,h,w} = x_{c,h,w}. \tag{22}$$

Thus the gate gives exact zero correction at a suppressed coordinate and permits corrections approaching the full candidate magnitude as $|g|$ approaches one, while its sign preserves or reverses the candidate direction. The clean-consistency loss imposes the corresponding output-level identity condition on clean latents, namely $\mathcal{D}_\psi(z^c) = z^c$ at its zero-loss solution.

Loss Derivatives and Purifier Training. Write the flattened attacked-pass residual as $e_b = \mathcal{D}_\psi(z_b^a) - z_b^c$ and the clean-pass residual as $f_b = \mathcal{D}_\psi(z_b^c) - z_b^c$. For $\ell(e) = |e| + H_\kappa(e)$, the scalar derivative is

$$\ell'(e) = \begin{cases} \operatorname{sign}(e) + e, & 0 < |e| \le \kappa, \\ (1+\kappa)\operatorname{sign}(e), & |e| > \kappa. \end{cases} \tag{23}$$

At zero, $\partial\ell(0) = [-1, 1]$ and we select the zero subgradient. Let $v(e)$ apply this derivative coordinate-wise and define the purifier Jacobians $J_b^a = \partial\mathcal{D}_\psi(z_b^a)/\partial\psi$ and $J_b^c = \partial\mathcal{D}_\psi(z_b^c)/\partial\psi$. The gradient supplied to the optimizer is

$$\nabla_\psi \mathcal{L} = \frac{1}{N_{\text{pair}} n} \sum_b \left[(J_b^a)^\top v(e_b) + \lambda_{\text{clean}} (J_b^c)^\top \operatorname{sign}(f_b) \right], \tag{24}$$

using $\operatorname{sign}(0) = 0$. This expression directly connects the paired targets to updates of the feature extractor, correction projection, and signed modulation branch.

For each nonzero residual, $v(e)$ has the sign of e and magnitude $1 + \min\{|e|, \kappa\}$. The L_1 term maintains a pull toward the clean target even for small residuals, and Huber adds residual-dependent feedback with a bounded contribution for large errors. Its curvature is one for $0 < |e| < \kappa$, while the combined penalty retains the L_1 kink at zero. The clean-pass gradient simultaneously penalizes edits to nominal inputs. Minibatch AdamW uses these gradients to fit the shared purifier; loss reduction tightens the joint action-error bound in equation equation 18. The recovery threshold in equation equation 19 specifies when that bound ensures an improvement over the undefended actions on the paired inputs.

Phase-Balanced Anchor Frame Sampling. All benchmarks use the same two-level sampling rule. Each purifier is trained with 1,000 anchor frames (fewer than 0.25% of the available demonstration data): task identities and demonstration video segments are selected randomly within the corresponding benchmark, and, conditional on each selected segment, frames are extracted at the fixed relative phases $\mathcal{P} = \{10\%, 30\%, 50\%, 70\%, 90\%\}$. Random task/segment selection supplies task diversity, whereas the fixed phase set gives every selected video segment balanced temporal coverage across approach, contact/grasp, and release/stabilization. This rule is shared by every benchmark and purifier training set.

A.2 Latency Breakdown Diagnostic

We report defense-only preprocessing separately from the complete frozen-policy forward pass, because the latter also includes VAE encoding and the backbone action head. In Table 6, p50 latency is the median synchronized module time after warm-up, peak forward allocation is the increase in peak allocated GPU memory above the baseline measured after module weights have been loaded, and the parameter column counts trainable defense parameters. Parameter storage is reported separately through the parameter count. All methods use the same FastWAM backbone and the input representation appropriate to their pixel- or latent-space interface.

LIP requires 1.262 ms median isolated preprocessing time and 1.47 MB peak additional forward allocation, compared with 7.240 ms and 208.53 MB for the RGB U-Net. These measurements quantify the runtime cost after weight loading. Figure 3 places this defense-only cost against the complete forward pass: across all four WAM backbones, LIP accounts for only 0.07%–0.14% of total forward time. The small runtime share supports adding LIP to the frozen policy's inference path. Isolated module latency and full-pipeline timing are measured separately.

Table 6: Defense-only preprocessing cost on FastWAM.

Method	p50 latency ↓ (ms)	Peak forward allocation ↓ (MB)	Parameters ↓
Gaussian blur	0.285	2.33	0
Median filter	0.727	15.45	0
JPEG ($Q = 75$)	4.935	1.15	0
RGB U-Net	7.240	208.53	0.222M
Latent MLP	0.191	1.84	0.091M
LIP (Ours)	**1.262**	**1.47**	1.299M

Table 7: End-to-end purifier training time on LIBERO and RoboTwin. Ratios are RoboTwin divided by LIBERO.

Model	LIBERO (s) ↓	RoboTwin (s) ↓	Ratio ↓
FastWAM	303.5	368.9	1.22×
Light-WAM	411.6	425.2	1.03×
StarWAM	331.9	377.5	1.14×
FasterWAM	318.2	290.4	0.91×
Mean per model	341.3	365.5	1.07×

A.3 Purifier Training-Time Efficiency

Table 7 complements the inference-time measurements with end-to-end purifier training cost. For each model and benchmark, training uses 1,000 anchor frames, three model-specific training UAPs, and four perturbation amplitudes $\{0.25, 0.50, 0.75, 1.00\}$, yielding 12,000 clean–attacked latent pairs. The independently optimized test UAP is held out and excluded from training. Each run uses 10 epochs, batch size 32, AdamW, cosine annealing, and the composite purification loss. The per-model timer includes model loading, latent-pair extraction, all training epochs, and checkpoint saving. Both benchmark runs are performed on RTX A6000 GPUs in identical states; the learning rate is 5×10^{-4} for LIBERO and 1.5×10^{-4} for multi-camera RoboTwin. Mean per-model training time is 341.3 s on LIBERO and 365.5 s on RoboTwin. Notably, this training cost is very small compared with the cost of training WAM or adversarially training it.

A.4 Latent–Action Relationship Analysis

We examine how latent recovery relates to action fidelity using 20 paired FastWAM evaluations. Pairing keeps the task, instruction, environment state, attack condition, and rollout seed aligned while comparing the clean, attacked, and purified representations. For each pair we measure latent drift $\|z^a - z^c\|_2$, correction magnitude $\|\hat{z} - z^a\|_2$, residual latent distance $\|\hat{z} - z^c\|_2$, and the corresponding action deviations. Their sample means are 0.7015, 0.3649, and 0.4342, respectively.

The three scatter relationships connect representation drift, correction, and action fidelity: latent drift correlates with attacked action error ($r = 0.7011$, $p = 5.74 \times 10^{-4}$), correction magnitude correlates with action-error reduction ($r = 0.5976$, $p = 5.39 \times 10^{-3}$), and residual purified-to-clean distance is associated with residual defended action error ($r = 0.5354$, $p = 0.015$). A descriptive linear path fit gives a drift-to-correction slope of 0.1747 and a correction-to-action-reduction slope of 2.0814. Their product is $0.1747 \times 2.0814 \simeq 0.3636$, with a bootstrap 95% interval of $[0.1246, 0.5814]$, summarizing the positive association along the fitted path. Figure 4 illustrates the conceptual path diagram, while Figure 5 presents the empirical regression diagnostics in Figure 5a and the corresponding clean, attacked, and purified latent geometry in Figure 5b. These results show that smaller residual latent distances accompany smaller defended action errors, supporting reconstruction of the action expert's input as a useful training target.

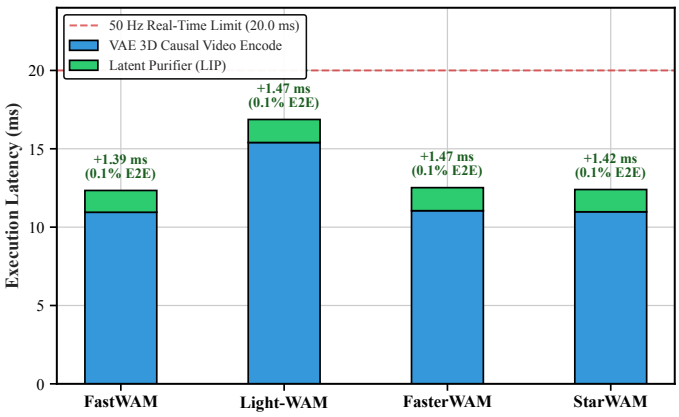


Figure 3: LIP's share of backbone forward-pass time.

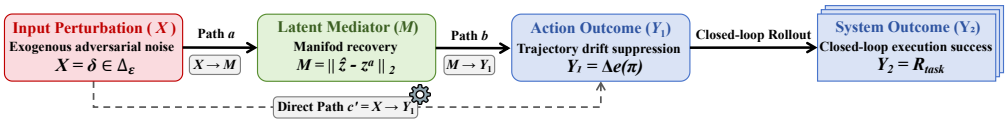


Figure 4: Conceptual relationships between latent perturbation, correction, action deviation, and task success.

B Supplementary Experimental Analyses

B.1 Baseline Diagnostic Visualization

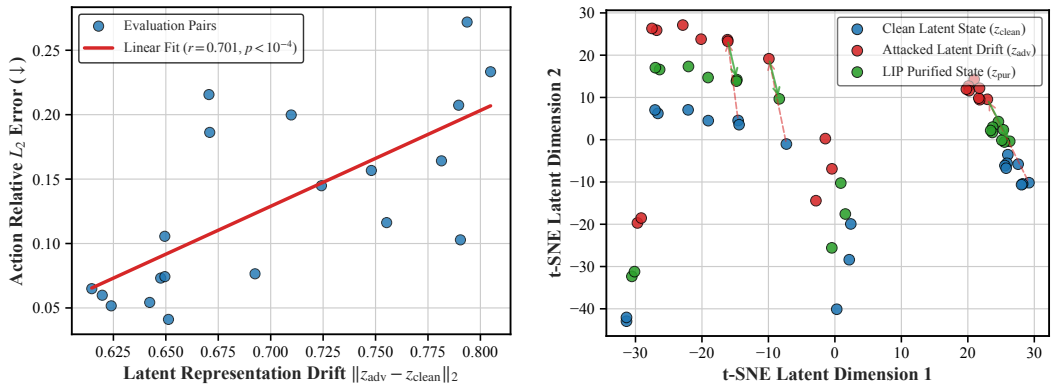
Figure 6 complements Table 3 with fixed-gate, random-gate, and clean-latent oracle controls, averaged equally over the four LIBERO suites and RoboTwin. The horizontal axis is clean action distortion and the vertical axis is attacked action error, so the lower-left region is the desirable trade-off. LIP's full input-conditioned design reaches substantially lower attacked error than the plotted practical alternatives while keeping clean distortion moderate; the oracle marks the optimistic limit obtained when the clean latent is supplied directly. The separation between these points also explains why a favorable single-frame diagnostic alone does not guarantee closed-loop task recovery: a defense must reduce attacked deviation without moving clean actions too far from the original policy. The plotted controls are an expanded offline diagnostic; Table 3 remains the source for the exact end-to-end baseline comparison.

B.2 Training Frame-Budget Ablation

Figure 7 shows defended success as the training budget increases from 300 to 1,500 frames under the same LIBERO attack and rollout protocol. In the plotted curves, the largest increase occurs for Light-WAM, with a smaller increase for FastWAM; StarWAM changes little and FasterWAM already starts at a high success level. Beyond the default 1,000 frames, the curves are non-monotonic and vary only modestly by backbone. This observed saturation supports 1,000 frames as a practical accuracy–data-cost trade-off. Relative to 300 frames, the 1,000-frame points change by $+2.42$, $+7.75$, $+0.58$, and -0.82 percentage points for FastWAM, Light-WAM, StarWAM, and FasterWAM, respectively. We use 1,000 frames as a common operating point that captures most of the observed gains at low data cost.

B.3 Complete Perturbation-Budget Results

We sweep universal-perturbation budgets from $\epsilon = 0$ to $32/255$ with the trained LIP component fixed throughout the sweep. Paired rollouts keep the task and seed matched at each budget. Figure 8 shows the LIBERO relative action-error curves for FastWAM and Light-WAM backbones: attacked



(a) Latent–action correlations (b) Latent manifold geometry

Figure 5: Latent–action relationships and latent geometry in paired FastWAM evaluations. Panel (a) shows correlations with dashed least-squares fits between latent drift, correction magnitude, and action deviations. Panel (b) displays the t-SNE manifold of clean, attacked, and LIP-purified latents, where arrows depict latent correction vectors.

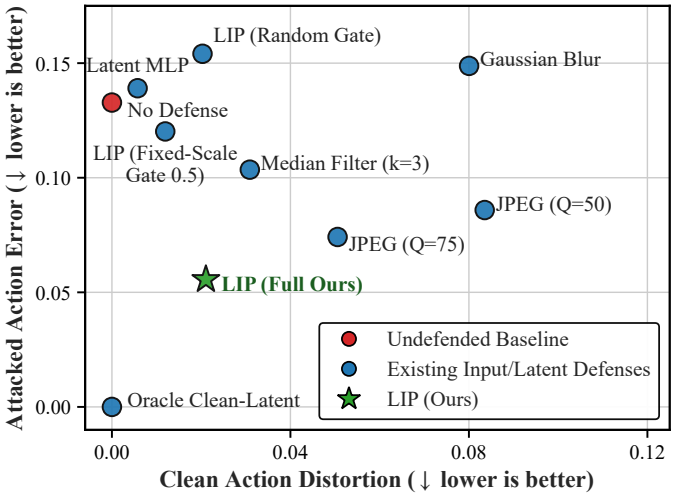


Figure 6: Clean-distortion versus attacked-error diagnostics for FastWAM defenses, averaged equally over the four LIBERO suites and RoboTwin.

error grows steadily with perturbation strength, whereas the defended curves rise more slowly and remain close to their nominal levels at weak budgets. Table 8 reports FastWAM results averaged equally over the four LIBERO suites and RoboTwin.

Across these five settings, Table 8 reports mean closed-loop success and relative action error. At the standard $16/255$ budget, LIP raises success from 48.88% under attack to 93.92%; at the largest $32/255$ budget, it still retains 79.50% success while the undefended policy falls to 8.75%. Using the synchronized action-error series, the reduction is 74.29% at the standard budget and remains positive at every nonzero budget, ranging from 71.56% to 81.48%. The agreement between the action-level and task-level trends indicates that purification improves the closed-loop control signal rather than only an offline representation metric. The clean row reports success with and without the always-on purifier.

As a complementary check, Figure 9 evaluates a freshly optimized universal perturbation at every budget instead of rescaling one fixed perturbation. Results are averaged equally over the four LIBERO suites and RoboTwin. The defended action-error curve remains below the undefended

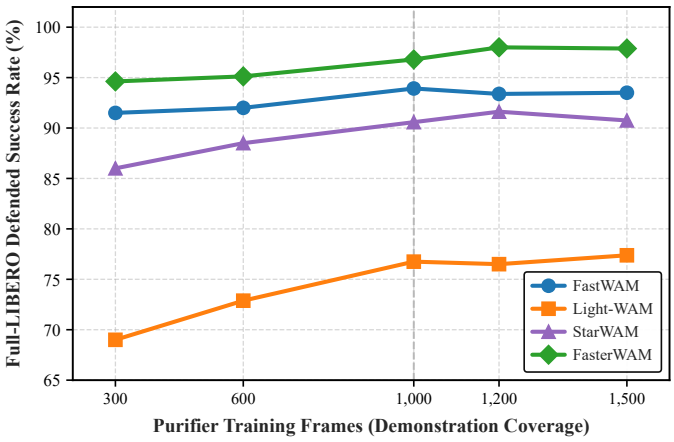


Figure 7: Defended LIBERO success versus training-frame budget for the four WAM backbones.

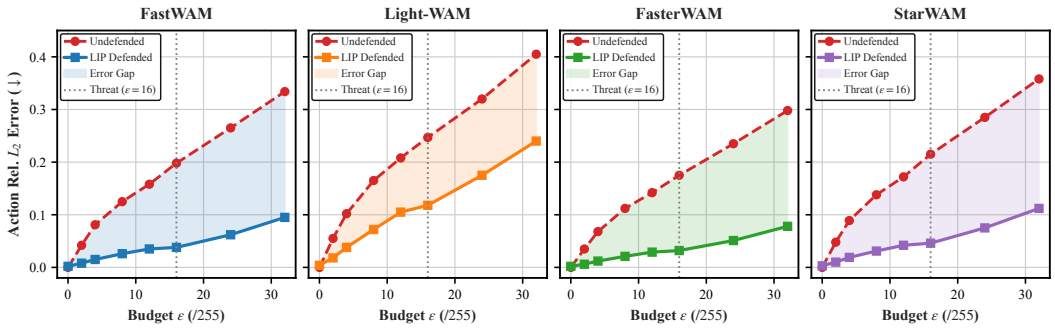


Figure 8: Relative action L_2 error versus perturbation budget for FastWAM and Light-WAM on LIBERO.

curve throughout this independently optimized sweep. The separation persists when each budget has its own optimized attack, supporting recovery across both perturbation direction and strength.

B.4 Adaptive Query-Based Attack Details

The adaptive stress test optimizes the perturbation against the active defense through SPSA rather than reusing a perturbation trained without LIP. It uses the same $16/255$ constraint, three independent restarts, and 600 optimizer queries per restart (1,800 total, excluding held-out validation calls). The suite-level results in Table 5 show positive gains of 18.50–66.00 percentage points, with defended success of 85.50%–98.50%. Figure 10a visualizes success under the adaptive attack. The corresponding offline diagnostic reduces relative action error from 0.142 to 0.058, linking the rollout gains to improved action fidelity under defense-aware optimization.

Interception Recovery Across Restarts. For each restart r, let S_r^c, S_r^a, and S_r^d denote task success under clean, attacked, and defended execution with matched evaluation conditions. The attack-induced performance loss is $A_r = S_r^c - S_r^a$, and the improvement from defense is $B_r = S_r^d - S_r^a$. For $A_r > 0$, define

$$R_{\text{int}}^{(r)} = 100\,\frac{B_r}{A_r} = 100\,\frac{S_r^d - S_r^a}{S_r^c - S_r^a}. \tag{25}$$

An interception recovery of 100% means that defended success reaches the clean baseline; values above 100% mean that it exceeds that baseline. This task-success metric measures the fraction of attack-induced performance loss recovered by LIP. The separate R_{action} metric measures the reduction in relative action error.

Table 8: FastWAM results across perturbation budgets, averaged equally over the four LIBERO suites and RoboTwin; boldface marks the standard budget.

	Closed-Loop Success Rate (%)			Relative Action Error (L_2)		
Budget ϵ	Attacked ↓	Defended ↑	Δ_{def} ↑ [95% CI]	e_{undef} ↓	e_{def} ↓	R_{action} ↑ (%)
0/255 (Clean)	92.40	91.45	-0.95 [-4.12, +1.38]	-	-	-
2/255	88.50	89.25	+0.75 [-2.15, +3.60]	0.042	0.008	**+80.95**
4/255	81.25	91.50	+10.25 [+6.50, +14.20]	0.081	0.015	**+81.48**
8/255	70.00	92.75	+22.75 [+18.10, +27.25]	0.125	0.026	**+79.20**
12/255	58.50	93.25	+34.75 [+29.80, +39.60]	0.158	0.035	**+77.85**
16/255	48.88	**93.92**	**+45.04** [+40.37, +48.00]	0.149	0.038	**+74.29**
24/255	18.50	87.25	+68.75 [+63.50, +73.80]	0.265	0.062	**+76.60**
32/255	8.75	79.50	**+70.75** [+65.20, +75.90]	0.334	0.095	**+71.56**

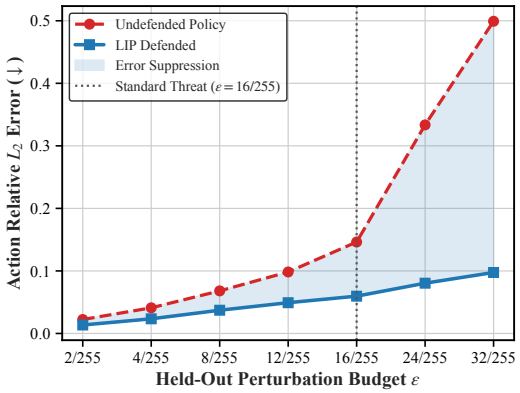


Figure 9: Action error under independently optimized universal perturbations, averaged equally over the four LIBERO suites and RoboTwin.

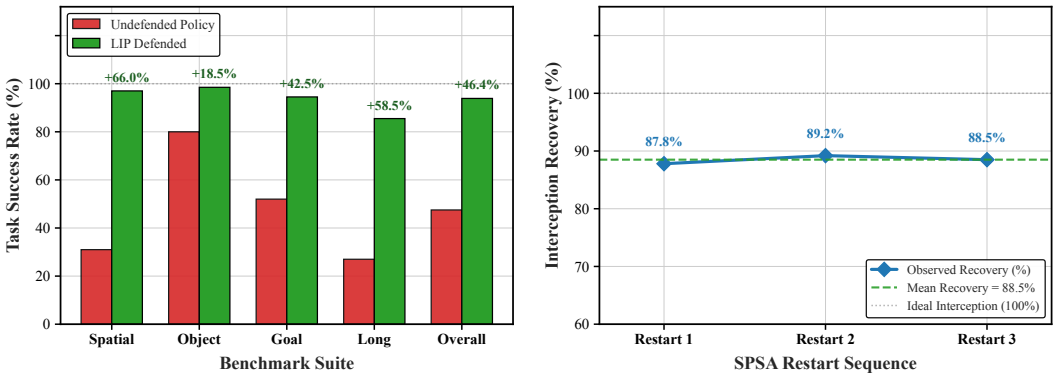
Figure 10b reports interception recovery of 87.8%, 89.2%, and 88.5% across the three restarts. LIP thus recovers a similar fraction of lost task performance across independently initialized adaptive attacks, with a mean of 88.5%.

B.5 Complete Natural-Corruption Sweep

We evaluate five natural-corruption families at five severity levels on eight fixed LIBERO tasks, with ten trials per task and condition. Thus each cell contains 80 undefended and 80 defended rollouts, corresponding to 160 condition-level observations per cell. The same trained purifier is used across corruption types and severities without per-corruption retuning in this comparison, and the corruption is applied to every control-step RGB observation on both cameras. Table 9 preserves the two raw success rates, while panel (a) of Figure 11 visualizes their percentage-point difference. The action-error views below use the same paired conditions but report representation-level fidelity rather than task completion alone.

Table 9 makes the heatmap interpretable without recovering the two underlying rates from their difference. The undefended column is the original policy evaluated on the same corrupted observation, not the nominal clean-input baseline. Gaussian noise produces the largest recoverable success gaps, reaching a 63.75-point gain at an intermediate severity, while salt-and-pepper noise provides substantial recovery at lower and middle severities. Motion blur stays close to parity, and brightness or contrast shifts sometimes favor the undefended policy; the clearest transfer gains are therefore concentrated in the noise families.

Table 10 provides the raw relative action-error reductions summarized by panel (b) of Figure 11 and the absolute-error plot in Figure 12. At severity five, LIP reduces relative action error by 78.87% for Gaussian noise and 51.24% for salt-and-pepper noise, while the photometric and blur families show



(a) Suite success

(b) Restart recovery

Figure 10: Adaptive SPSA evaluation on LIBERO under the stated query budget. Panel (a) compares undefended and LIP-defended suite success; panel (b) shows interception recovery across independent attack restarts.

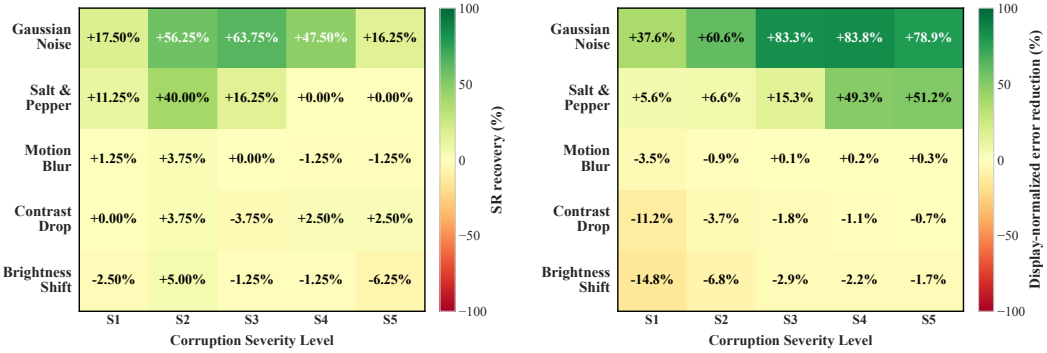
Table 9: Raw closed-loop success rates under natural corruptions on FastWAM. "Undefended" denotes the original policy on each corrupted input.

Corruption	Severity	Undefended SR (%)	LIP SR (%)	Δ (pp)
Gaussian noise	1	67.50	85.00	+17.50
Gaussian noise	2	26.25	82.50	+56.25
Gaussian noise	3	0.00	63.75	+63.75
Gaussian noise	4	0.00	47.50	+47.50
Gaussian noise	5	0.00	16.25	+16.25
Motion blur	1	93.75	95.00	+1.25
Motion blur	2	91.25	95.00	+3.75
Motion blur	3	92.50	92.50	0.00
Motion blur	4	95.00	93.75	-1.25
Motion blur	5	96.25	95.00	-1.25
Salt-and-pepper	1	30.00	41.25	+11.25
Salt-and-pepper	2	0.00	40.00	+40.00
Salt-and-pepper	3	0.00	16.25	+16.25
Salt-and-pepper	4	0.00	0.00	0.00
Salt-and-pepper	5	0.00	0.00	0.00
Brightness shift	1	93.75	91.25	-2.50
Brightness shift	2	81.25	86.25	+5.00
Brightness shift	3	86.25	85.00	-1.25
Brightness shift	4	82.50	81.25	-1.25
Brightness shift	5	77.50	71.25	-6.25
Contrast drop	1	92.50	92.50	0.00
Contrast drop	2	88.75	92.50	+3.75
Contrast drop	3	90.00	86.25	-3.75
Contrast drop	4	78.75	81.25	+2.50
Contrast drop	5	45.00	47.50	+2.50

smaller or sign-changing effects. The action-error heatmap divides the motion-blur, brightness-shift, and contrast-drop rows by 10 only for visual comparability because their undefended errors are substantially smaller; Table 10 retains every unmodified measurement. Taken together, the success and action-level views show that, within the tested set, relative-error reductions remain positive across all Gaussian levels (37.61%–83.81%) and all salt-and-pepper levels (5.62%–51.24%). In contrast, motion blur ranges from -35.01% to $+3.46\%$, brightness shift from -147.95% to -16.74%, and contrast drop from -111.69% to -6.68%. Thus the strongest gains are specific to the tested noise families, while smooth photometric shifts remain a challenge for the current purifier.

Table 10: Raw relative action-error reduction under natural corruptions on FastWAM; positive values indicate improvement.

Corruption Type	Severity-1 Reduction ↑	Severity-5 Reduction ↑	Range over Levels 1–5
Gaussian noise	**+37.61%**	**+78.87%**	+37.61% to +83.81%
Motion blur	-35.01%	**+3.46%**	-35.01% to +3.46%
Salt-and-pepper	**+5.62%**	**+51.24%**	+5.62% to +51.24%
Brightness shift	-147.95%	-16.74%	-147.95% to -16.74%
Contrast drop	-111.69%	-6.68%	-111.69% to -6.68%



(a) SR recovery.

(b) Action-error reduction.

Figure 11: Natural-corruption results on FastWAM: SR recovery (left) and relative action-error reduction (right).

B.6 LIBERO Per-Task Recovery

Figure 13 complements the aggregate LIBERO comparison with task-level evidence for all 40 tasks. The figure reports the clean, attacked, and LIP-defended success rates for each task from the paired four-condition ledger. The attacked bars expose task-dependent degradation, whereas the green bars show broad recovery across spatial, object, and goal tasks with a lower tail on the long-horizon suite. These results demonstrate consistent defense efficacy across diverse task semantics, object geometries, and horizon lengths.

B.7 RoboTwin Per-Task Recovery

Figure 14 resolves the FastWAM aggregate on RoboTwin into the same 40 bimanual tasks used by the closed-loop benchmark. The mean success sequence is 85.00% clean, 35.38% under attack, and 92.62% with LIP. Recovery is distributed rather than driven by a small subset: 39 of the 40 tasks reach at least 75% defended success and 33 reach at least 90%. These task-level results show that the aggregate gain is accompanied by high defended success across a broad range of bimanual skills.

B.8 Paired Bootstrap Uncertainty

Figure 15 summarizes the defended-minus-attacked success-rate gains with 2,000 paired task-cluster bootstrap replicates. The points are the estimates reported by the frozen four-condition ledger, and the horizontal intervals are its 95% confidence intervals. All LIBERO intervals and three of the four RoboTwin intervals lie above zero; the FasterWAM–RoboTwin interval crosses zero, matching its comparatively small aggregate gain. This uncertainty view complements the point estimates in Table 1 without introducing another evaluation condition.

B.9 Qualitative Rollout Visualizations

Figure 16 shows two additional Light-WAM evaluator rollouts from LIBERO: task 0 and task 3. For each task, the two rows display the attacked rollout without LIP and the LIP-defended rollout, re-

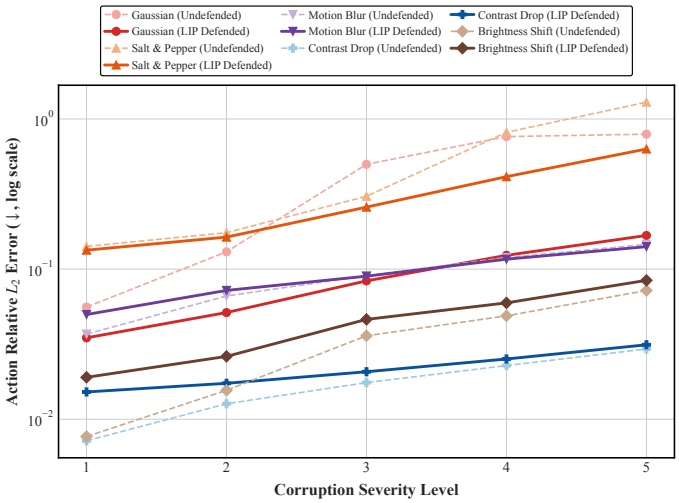


Figure 12: Absolute relative action error under natural corruptions on FastWAM; solid and dashed lines denote LIP and undefended policies.

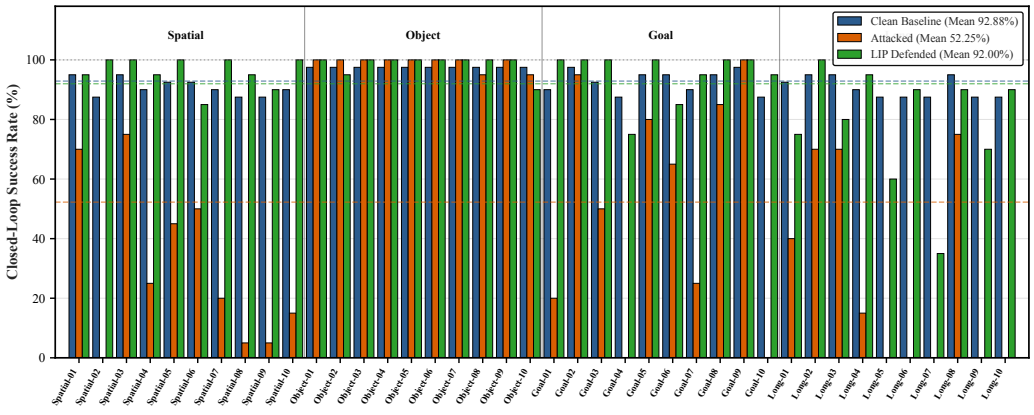


Figure 13: LIBERO FastWAM success across 40 tasks under clean, attacked, and LIP-defended conditions.

spectively. The simulator recorder saves its clean rendering rather than the perturbed policy tensor; therefore, we render the exact held-out UAP at its evaluation magnitude on each displayed camera panel in both rows. We sample the same relative rollout positions because successful episodes terminate once the task-completion condition is met, whereas failed episodes run to the evaluation horizon. The visualization complements the aggregate success rates and trajectory traces: the undefended policy visibly drifts away from its object-interaction sequence, while LIP preserves task progress through completion.

Figure 17 provides complementary bimanual RoboTwin evidence with StarWAM on *Grab Roller* and *Blocks Ranking Size*. Each panel shows the three real policy views (head, left wrist, and right wrist); the red and green rows are matched failed attacked and successful LIP-defended episodes, respectively, with the exact held-out UAP rendered on every displayed view. The matched views show how LIP restores task progress and coordination across the two arms.

Figure 18 shows three FastWAM LIBERO-Long rollouts with five time-ordered agent-view frames and the corresponding three-dimensional end-effector paths. Clean and LIP-defended executions complete the tasks, while attacked executions fail; the paired frames and trajectories make this execution-level contrast clear.

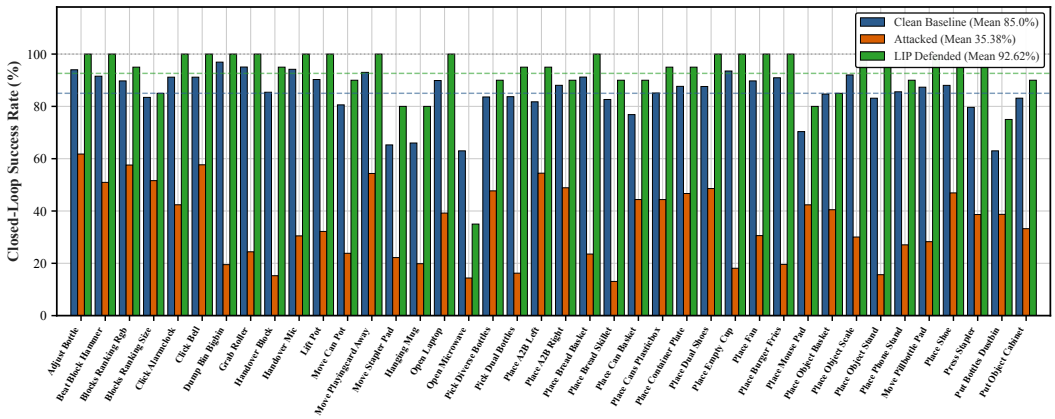


Figure 14: FastWAM closed-loop success across RoboTwin tasks under attack and LIP defense.

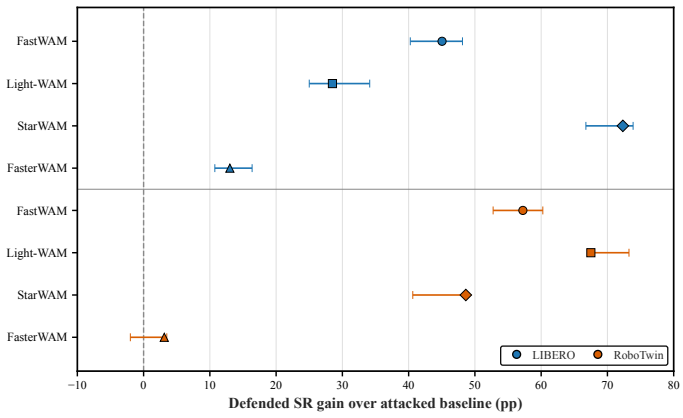


Figure 15: Paired task-cluster bootstrap intervals for defended-minus-attacked success gains.

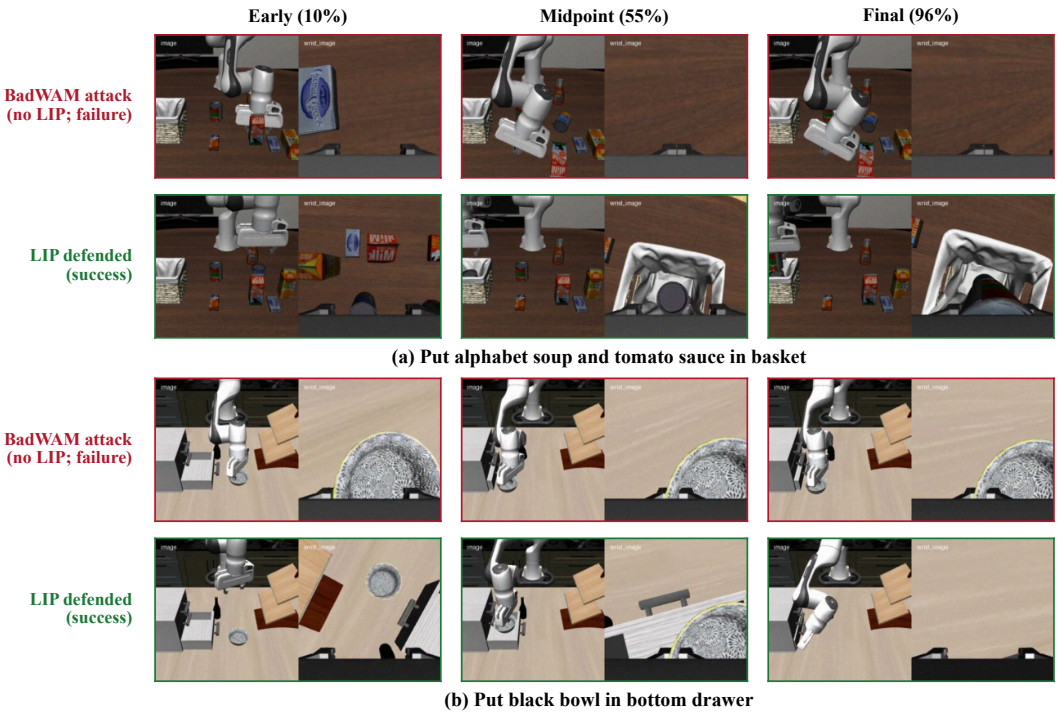


Figure 16: Qualitative Light-WAM rollouts on LIBERO: Attacked (red) and LIP-defended (green).

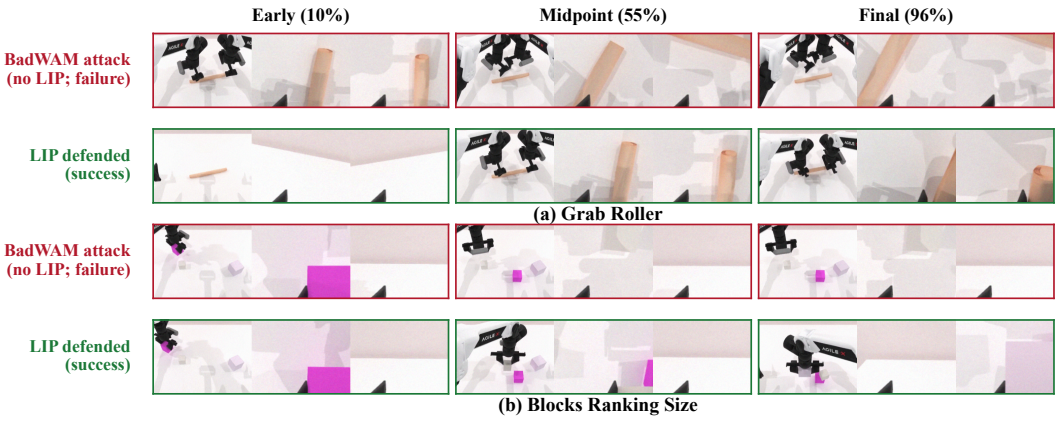


Figure 17: Qualitative StarWAM rollouts on RoboTwin: Attacked (red) and LIP-defended (green).

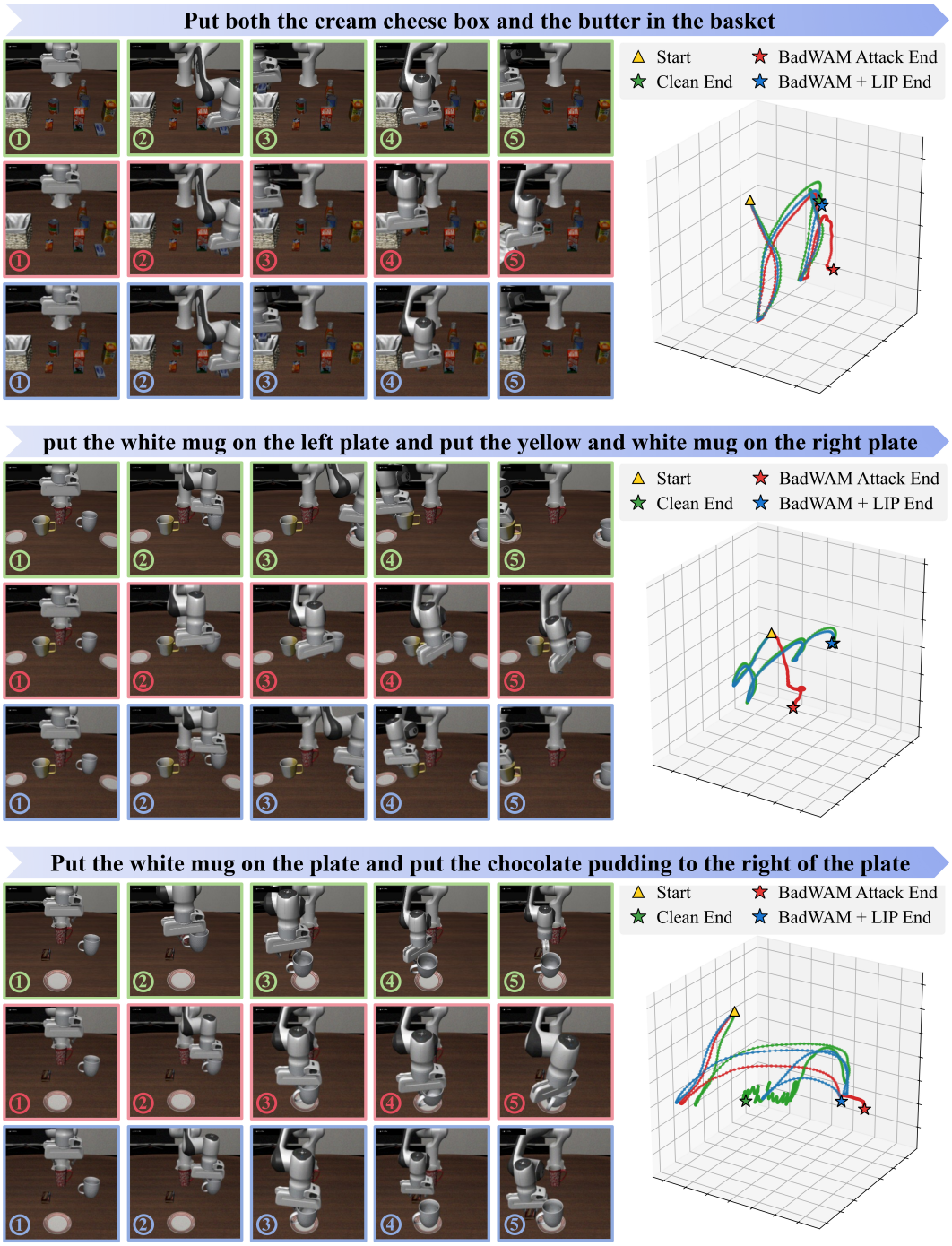


Figure 18: Qualitative rollout results of FastWAM on representative tasks. Each example shows time-ordered observations and 3D end-effector trajectories under clean, BadWAM-attacked, and LIP-defended execution.