Manuscript submitted to *Security and Safety*, For Peer Review Only, Vol. x, xxxxxxx (2026)
https://doi.org/10.1051/sands/xxxxxxx
sands.edpsciences.org

Research Article OPEN ACCESS

Section Category

Breaking Flow-Matching Embodied Policies with Trajectory-Coupled Adversarial Perturbations

Siyuan Pan[1], Rong Guo[1], Mengxiang Liu[1], and Ruilong Deng[1]*

[1] *State Key Laboratory of Industrial Control Technology and College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China*

Abstract Flow-matching Vision-Language-Action (VLA) policies generate actions through τ-indexed ordinary differential equation (ODE) rollouts, exposing intermediate velocity-field structure that endpoint-only adversarial objectives do not directly target. We study this mechanism on the evaluated $\pi_{0.5}$ policy in LIBERO and propose the Tau-Path Coupled Attack (TPCA), a white-box visual attack that couples executed-window endpoint displacement with velocity-field divergence along the flow rollout. The study is deliberately scoped as a single-architecture mechanism analysis rather than a general claim about all flow-matching policies. Under matched compute against Visual-PGD, TPCA and endpoint-only optimization both produce near-saturated task failure at 3/255, while TPCA yields larger action-space endpoint displacement. As the perturbation budget tightens, the task-level gap becomes visible: at 1/255, TPCA induces a Drop of 0.38 whereas endpoint-only optimization induces 0.17, with over +129% larger action-space endpoint displacement. Cross-task checks on two additional LIBERO spatial tasks show consistent action-space endpoint-displacement advantages for the evaluated setting, while task-failure advantages are budget- and task-dependent. These results support a bounded conclusion: flow-rollout coupling provides additional optimization signal for the evaluated $\pi_{0.5}$ policy, but broader architectural generalization and defense effectiveness require future extensive studies.

Keywords Embodied intelligence security, vision-language-action model, flow matching, visual adversarial attack, robot policy robustness

Citation S. Pan, R. Guo, M. Liu, and R. Deng. Breaking Flow-Matching Embodied Policies with Trajectory-Coupled Adversarial Perturbations. Security and Safety 2026; **x**: xxxxxxx. https://doi.org/10.1051/sands/xxxxxxx

1 Introduction

Embodied intelligence connects perception, reasoning, and physical action in a closed loop: a robot observes its workspace through cameras, receives a task instruction in natural language, and must translate both into a sequence of motor commands executed in the real world. Early systems decomposed this pipeline into hand-engineered modules for object detection, motion planning, and low-level control. Vision-Language-Action (VLA) policies collapse these stages into a single learned model (§2.2 surveys the landscape). Robotics Transformers such as RT-1 [1] and RT-2 [2] demonstrated that a single transformer can map visual observations and language goals to actions at scale. Diffusion Policy [3] introduced conditional denoising as the action-generation mechanism, and affordance-grounded planners such as SayCan [4] and PaLM-E [5] integrated high-level language reasoning with low-level execution. A consequence of this end-to-end design is that any corruption of the visual input, whether accidental or adversarial, can propagate directly into physical motion [6, 7]. The π-series of VLA policies [8] takes this integration one step further by adopting flow matching [9] as the action-generation mechanism. Instead of decoding action

* Corresponding author (email: dengruilong@zju.edu.cn)

tokens in a single forward pass, the policy solves an ODE that integrates a learned velocity field from noise to a final action over a continuous index $\tau \in [0, 1]$. Each intermediate integration step produces a velocity vector that steers the evolving action trajectory.

The security of these systems, however, remains poorly characterized. Deep networks are known to be sensitive to imperceptible input perturbations [10, 11], and optimization-based attacks reliably alter model outputs under tight budgets [12, 13]—a vulnerability that extends to sequential decision-making [14, 15]. In the embodied intelligence domain, VLA policies exhibit severe fragility under distributional shifts and adversarial inputs [6], and these vulnerabilities can manifest as unsafe or degraded trajectories in collaborative-robot settings [7]. Contextual backdoor triggers can propagate through multimodal planning stacks [16], compounding the risk. Yet, as noted in recent landscape surveys [17, 18], existing security evaluations rely almost exclusively on binary task-success metrics and lack architecture-matched controls, leaving open whether observed vulnerabilities are mechanism-specific or generic perturbation effects.

This gap is especially relevant for flow-matching policies such as the π-series [8], which generate actions through τ-indexed ODE rollout dynamics rather than single-pass decoding. The intermediate velocity fields traversed during this rollout constitute a plausible attack surface, yet no prior work has examined whether directly targeting rollout geometry changes attack behavior relative to endpoint-only optimization under matched budgets. We investigate this question through a trajectory-aware adversarial framework that couples endpoint displacement with velocity-field divergence along the rollout. Because our empirical evidence is based on $\pi_{0.5}$ and LIBERO spatial tasks, we treat the study as a focused mechanism analysis rather than evidence that all flow-matching VLA policies share the same vulnerability profile.

This paper makes three contributions:

1. We propose a flow-aware visual attack objective that couples executed-window endpoint displacement with τ-path velocity divergence, producing larger action-space deviations than endpoint-only optimization under strictly matched compute on the evaluated $\pi_{0.5}$ policy.
2. We provide a budget-regime analysis across three perturbation levels (3/255, 2/255, 1/255). The Drop advantage is visible in the constrained-budget regime, while action-space endpoint-displacement advantages persist even when task failure is near-saturated at 3/255.
3. We establish a strict white-box evaluation protocol with verified gradient flow and explicit clean-success filtering, and report continuous action-space diagnostics—chunk-endpoint shift and flow-path shift—alongside binary Drop.

The remainder of the paper is organized as follows. Section 2 reviews adversarial attacks and embodied security. Section 3 formulates the threat model and presents the TPCA objective. Section 4 describes the experimental setup. Section 5 reports results, including the budget-regime analysis, ablation, and cross-task validation. Section 6 discusses implications and limitations, and Section 7 concludes.

2 Related work

Adversarial machine learning has been studied for over a decade. The attack toolkit is well established, and so are the evaluation pitfalls that once allowed inflated robustness claims to stand unchallenged [19, 20]. Embodied security is much newer, emerging only as VLA policies became capable enough to attract serious adversarial scrutiny [17, 18]. At the level of attack mechanics, the two fields are not that different: both use bounded perturbations optimized via gradients to induce failure without detectable input changes. The difference is in what gets attacked. Classical attacks target a single forward pass—one decision, one gradient signal. Closed-loop embodied policies couple visual observation to physical execution across replanning steps, where trajectory deviation below the binary failure threshold may still matter for execution quality, even though this paper does not measure contact forces, collision rates, or real-robot injury risk. The following subsections review state-of-the-art progress in both domains and identify the research gap this paper addresses.

2.1 Adversarial attacks

Optimization-based attacks

The discovery that deep networks are vulnerable to imperceptible input perturbations [10] prompted systematic study of worst-case examples. FGSM (Fast Gradient Sign Method) [11] showed that a single gradient step suffices against undefended models; iterative refinement via margin-based objectives (C&W [12], EAD [21]) and momentum accumulation [13] subsequently became the standard for stress-testing robustness. Universal perturbations [22]—single input-agnostic patterns that transfer across samples—revealed that vulnerability is not instance-specific. On the black-box side, substitute-model transfer [23], decision-boundary walks [24], single-pixel queries [25], and distribution-learned perturbations [26] demonstrated that gradient access is not strictly necessary. Physical-world synthesis of perturbations that survive viewpoint and lighting changes [27] extended the threat to camera-observed scenes—the setting most relevant to embodied policies.

Evaluation rigor and defensive baselines

Reliable measurement of robustness has proven as difficult as achieving it. Obfuscated gradients [19] and detection methods that fail under adaptive attack [28] repeatedly inflated reported defenses, prompting calls for standardized evaluation: feature-squeezing baselines [29] and parameter-free attack ensembles such as AutoAttack [20] partially addressed reproducibility, but inconsistent compute budgets across studies remain a source of incomparable results. On the defense side, PGD (Projected Gradient Descent)-based adversarial training [30] is the strongest empirical approach, with replay-based [31] and single-step [32] variants reducing its cost and TRADES [33] formalizing the accuracy–robustness trade-off. Certified methods based on randomized smoothing [34] offer provable ℓ_2 guarantees at a significant accuracy penalty. A recurring lesson from this literature is that evaluation protocol determines conclusions as much as the attack itself. Our matched-compute, multi-metric design follows from this observation.

From classifiers to generative trajectories

The attacks above target a single decision boundary. Generative models introduce a fundamentally different structure: denoising diffusion [35] and score-based SDEs [36] produce outputs through iterative integration, exposing intermediate states that can each serve as an optimization target. Flow matching [9] replaces stochastic denoising with deterministic ODE integration over a learned velocity field indexed by a continuous variable τ. For an attacker with gradient access, this means a loss can be defined at every integration step—not only at the terminal output—providing denser supervision per unit of perturbation budget. The path objective in this work exploits precisely this property.

2.2 Embodied security

Sequential-decision vulnerabilities

Adversarial threats to sequential policies differ qualitatively from the single-image setting. An adversarial opponent can destabilize a well-trained agent without any single action appearing anomalous [14]; training-time poisoning embeds persistent vulnerabilities in batch RL [15] and policy-teaching [37] pipelines. In embodied control, a trajectory deviation that does not cross a binary success threshold may still indicate degraded control quality—motivating evaluation metrics that capture continuous action-space displacement rather than pass/fail outcomes alone.

Policy architectures and the expanding attack surface

The attack surface of learned manipulation policies has expanded with each architectural generation. Early spatial-correspondence [38] and language-grounded [39] methods coupled perception with action; sequence-modeling formulations (Decision Transformer [40], Trajectory Transformer [41], Behavior Transformers [42]) then made action trajectories differentiable end-to-end. Language-conditioned planning broadened the interface through affordance grounding [4] and related methods [43–46].

The VLA paradigm consolidated these advances. Robotics Transformers (RT-1 [1], ACT [47]), cross-embodiment data unification [48], and vision-language backbones (PaLM-E [5], RT-2 [2]) established the infrastructure, while generative action heads based on conditional diffusion [3, 49] replaced fixed-vocabulary decoders. The ecosystem continues to grow across efficient [50, 51], multimodal [52, 53], 3D-aware [54, 55], and open-source [56–59] variants. Flow-matching policies, most prominently π_0 [8], represent the latest step, generating actions via τ-indexed ODE rollouts rather than token-level decoding.

Security evaluation gaps

Despite this rapid expansion, security analysis of VLA systems remains sparse. Landscape surveys identify robustness as a critical open problem [17, 18]; the few empirical studies reveal severe fragility under distributional shifts [6] and show that targeted perturbations can produce unsafe or degraded trajectories in collaborative-robot settings [7], directly implicating safety standards for human–robot interaction [60]. Contextual backdoor attacks on LLM-driven embodied agents [16] demonstrate that multimodal triggers propagate through planning stacks with minimal detection. Critically, existing evaluations report binary task-success against a single attack without architecture-matched controls, making it difficult to determine whether an observed failure is generic to gradient-based perturbation or specific to how the model generates actions. We address this gap within one evaluated flow-matching architecture through a controlled ablation design where Visual-PGD shares the full TPCA pipeline except for the flow-coupling term.

3 Flow-Aware Attack Approach

3.1 Problem formulation

3.1.1 Flow-matching dynamics and embodied attack surface

For a context tuple $c = (I, T, s)$ with visual observation I, language instruction T, and robot state s, a flow head evolves actions via a discretized ODE-like rollout:

$$a_{k+1} = a_k + \Delta\tau \cdot v_\theta(a_k, \tau_k, c), \quad k \in \{0, \dots, K-1\}. \tag{1}$$

Here v_θ is a learned velocity field conditioned on context and τ index. Endpoint extraction is performed from a designated rollout position consistent with deployment-time control. In contrast to autoregressive or tokenized action heads, this mechanism makes intermediate trajectory geometry a first-class object and therefore a potential attack target [3, 8, 9, 57].

For the evaluated flow-matching action head, the security implication is twofold (Figure 1). First, adversarial perturbations can accumulate across steps rather than only at a terminal decoder pass. Second, optimization signals can be injected through both endpoint and path channels, enabling attacks that are structurally aligned with model internals. This differs from generic endpoint losses used by many baseline attacks and motivates architecture-aware evaluation.

3.1.2 Threat model, constraints, and evaluation quantities

We assume an adversary can perturb visual observations but cannot modify model parameters, simulator physics, control firmware, or training data. Perturbations obey strict budgets:

$$\|\delta_I\|_\infty \le \varepsilon, \quad d_{\mathrm{RMS}}(I, I+\delta_I) \le \eta. \tag{2}$$

In our main setup, $\varepsilon \in \{3/255, 2/255, 1/255\}$ and $\eta = 0.02$, where d_{RMS} is computed on pixel values normalized to $[0, 1]$:

$$d_{\mathrm{RMS}}(I, I+\delta_I) = \sqrt{\frac{1}{3HW} \sum_{c=1}^{3} \sum_{h=1}^{H} \sum_{w=1}^{W} (\delta_I(c,h,w))^2}. \tag{3}$$

The primary endpoint is task-success drop:

$$\mathrm{Drop} = \mathrm{CleanSR} - \mathrm{AdvSR}. \tag{4}$$

We additionally report endpoint (EP) shift and path shift, enabling multi-criteria evaluation. Throughout the paper, we use $\delta_\infty \equiv \|\delta_I\|_\infty$ as shorthand for the measured L_∞ perturbation magnitude and $\delta_{\mathrm{RMS}} \equiv d_{\mathrm{RMS}}(I, I+\delta_I)$ for the measured root-mean-square (RMS) distance.

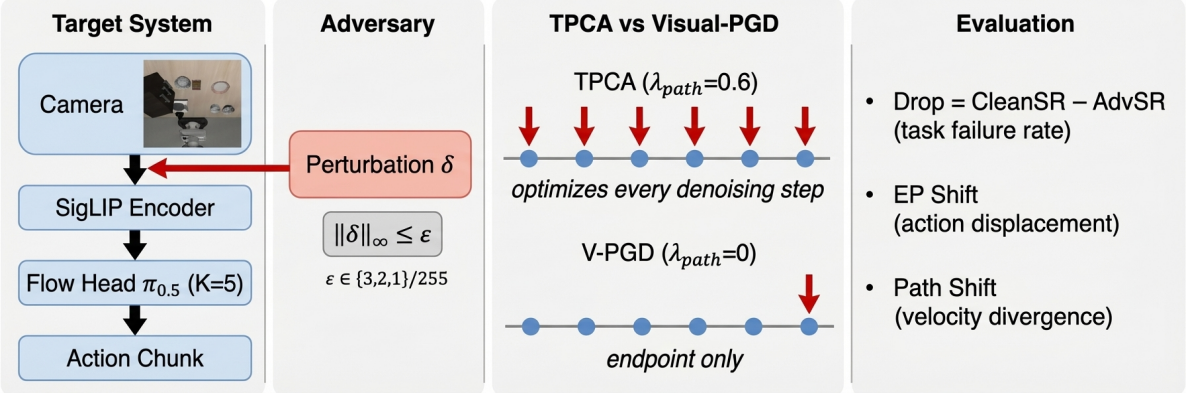

Figure 1. Threat model and method comparison. The adversary injects bounded perturbation δ into the camera input. TPCA optimizes against every denoising step of the τ-indexed ODE rollout (red arrows at all steps), while Visual-PGD targets only the endpoint action.

3.2 Tau-Path coupled objective

TPCA is designed as an *executed-window first* attack (Figure 2): at each replanning event the attack re-optimizes the current visual observation, targeting the controls that will actually be executed before the next replan. For LIBERO spatial task 1 this means $M = \texttt{replan\_steps} = 5$. The central methodological idea is coupling EP displacement with flow-trajectory velocity divergence, which provides gradient signal from every denoising step in the flow-matching rollout.

The TPCA objective uses an executed-window chunk objective with flow-aware path coupling as a first-class gradient source ($\lambda_{\text{path}} = 0.60$), along with margin shaping and alignment regularization. Visual-PGD uses the identical objective but with $\lambda_{\text{path}} = 0$, serving simultaneously as a competitive baseline and as a controlled ablation of the flow-coupling term. Both methods use the same number of PGD iterations (I=40) with no restarts, ensuring strictly matched gradient compute.

The default executed-window objective is

$$\mathcal{J}_{\text{main}} = \mathcal{L}_{\text{exec}} + \lambda_{\text{margin}}\mathcal{R}_{\text{margin}} + w_{\text{path}}\,\mathcal{L}_{\text{path}}, \tag{5}$$

where the adaptive path weight w_{path} gates the flow coupling term based on current endpoint displacement (see below), and $\lambda_{\text{path}} = 0.60$ in the mainline. The executed-window term is

$$\mathcal{L}_{\text{exec}} = \sum_{h=1}^{M} \omega_h \left\| a^{\text{adv}}_{\text{chunk},h} - a^{\text{clean}}_{\text{chunk},h} \right\|_2, \qquad \sum_h \omega_h = 1, \tag{6}$$

$$\mathcal{L}_{\text{path}} = \sum_{k=1}^{K} \tilde{w}_k \left\| v^{\text{adv}}_k - v^{\text{clean}}_k \right\|_2, \tag{7}$$

$$\mathcal{R}_{\text{align}} = \cos\left(a^{\text{adv}}_{\text{eval}} - a^{\text{clean}}_{\text{eval}},\ \sum_{k=1}^{K} \tilde{w}_k \left(v^{\text{adv}}_k - v^{\text{clean}}_k \right) \right), \tag{8}$$

$$\mathcal{R}_{\text{margin}} = \sigma\left(\frac{\mathcal{L}_{\text{exec}} - \tau_{\text{hint}}}{T} \right), \tag{9}$$

$$w_{\text{path}} = \lambda_{\text{path}} \left[g_{\text{floor}} + (1 - g_{\text{floor}})\,\sigma\big((\mathcal{L}_{\text{exec}} - \tau_{\text{hint}})/T\big) \right], \tag{10}$$

$$\tilde{w}_k = \frac{w_k \cdot \mathbf{1}(k \in \mathcal{W})}{\sum_j w_j \cdot \mathbf{1}(j \in \mathcal{W})}. \tag{11}$$

Here w_k=$1/K$ are uniform base weights over the K denoising steps, $\mathcal{W}$ is the active tau window for the path term (set to all K steps in every reported experiment), $\sigma(\cdot)$ is the sigmoid function, T is a temperature coefficient, and τ_{hint} is the target executed-window shift hint.

The adaptive path weight w_{path} gates the flow-coupling term so that path optimization activates only after endpoint displacement reaches a threshold, preventing the path term from consuming perturbation budget before a task-relevant displacement is established. The gate floor $g_{\text{floor}} = 0.40$ ensures a baseline path contribution even early in optimization. Budget compliance is enforced via strict dual projection ($\|\delta_I\|_\infty \leq \varepsilon$ and $\|\delta_I\|_{\text{rms}} \leq \eta$) applied after each PGD step, rather than as a soft penalty term. In the mainline configuration, $M = 5$ and the default profile is `uniform_chunk`, so the objective focuses on the first executed window rather than future controls that will be overwritten by closed-loop replanning.

The implementation also supports alignment and curvature regularizers:

$$\mathcal{J}_{\text{full}} = \mathcal{J}_{\text{main}} + \lambda_{\text{align}}\mathcal{R}_{\text{align}} + \lambda_{\text{curv}}\mathcal{R}_{\text{curv}}, \tag{12}$$

where $\mathcal{R}_{\text{align}}$ encourages directional consistency between the endpoint displacement and the aggregate velocity deviation, and $\mathcal{R}_{\text{curv}}$ measures trajectory curvature differences between clean and attacked rollouts. These regularizers are used as implementation-level conditioning terms rather than independent claims of robustness or attack success.

In all reported experiments, the path term operates over all K denoising steps with uniform weighting. Adaptive window pruning—where per-step sensitivity scores $g_k = \|\nabla_{\delta_I}\|v_k^{\text{adv}} - v_k^{\text{clean}}\|_2\|_2$ would select a subset of steps—is not used in Tables 2–5 or any reported figure, and is left for future investigation of compute-constrained attack variants.

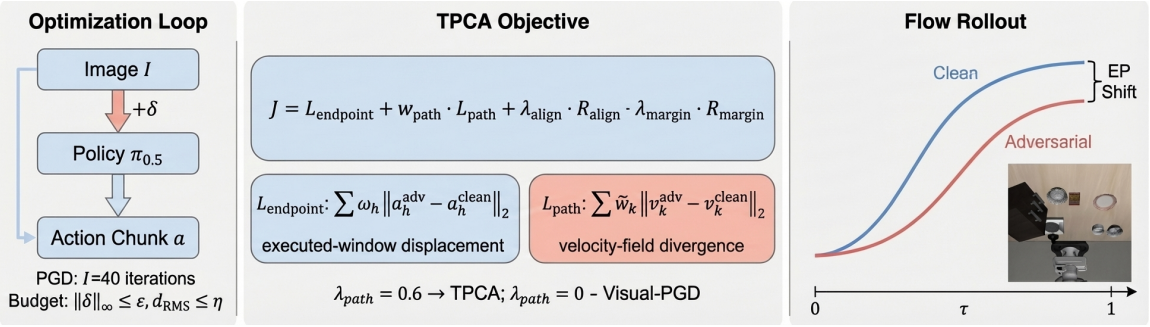

Figure 2. TPCA optimization pipeline. Left: PGD perturbation loop with dual budget projection. Center: the objective couples endpoint displacement ($\mathcal{L}_{\text{endpoint}}$) with velocity-field divergence ($\mathcal{L}_{\text{path}}$); setting λ_{path}=0 recovers Visual-PGD. Right: the τ-indexed ODE rollout, where adversarial (red) and clean (blue) rollout trajectories diverge. Adaptive window pruning is illustrated as a possible future acceleration component and is not used in the reported experiments.

3.3 Optimization protocol and deployment alignment

TPCA unifies evaluation and auxiliary schedules (uniform tau over $[0, 1]$), collapsing both rollouts into a single differentiable pass that halves peak GPU memory and enables operation on 24 GB GPUs. The strict white-box runtime uses K=5 flow steps, step size α=0.007, and strict dual visual-budget projection. Both TPCA and Visual-PGD use I=40 projected PGD iterations per replan step with normalized gradient momentum ($\tilde{g} = g/\mathbb{E}[|g|]$, $v \leftarrow \beta v + \tilde{g}$, β=0.5), M=5, λ_{align}=0.20, λ_{margin}=1.20, λ_{curv}=0.05, gripper-focus weight 1.60, and gate floor g_{floor}=0.40. The only difference between methods is the flow-path coupling weight: λ_{path}=0.60 for TPCA and λ_{path}=0 for Visual-PGD. Budget sensitivity experiments (§5.3) sweep $\varepsilon \in \{3/255, 2/255, 1/255\}$ while keeping the optimization protocol fixed. All settings are validated through gradient diagnostics (100% non-zero gradient rate across all seeds).

The theoretical motivation draws on Lipschitz dynamics: if v_θ is Lipschitz in action and context, endpoint error obeys a Gronwall-type bound with multiplicative amplification across rollout steps. The path term can provide additional intermediate supervision at every denoising step. The budget sweep tests whether this additional signal changes empirical behavior relative to Visual-PGD, the λ_{path}=0 control.

3.4 Objective geometry and compute complexity

Geometrically, endpoint loss drives a global displacement vector in action space while path loss constrains local tangent-field mismatch along rollout trajectories, approximating a first-order control objective over both position and velocity consistency. Consider a smooth relaxation of the task-failure boundary, $\tilde{b}(a) = \sigma\big((\|a - a^{\text{clean}}\|_2 - \tau_{\text{eval}})/T\big)$. The change in $\tilde{b}$ around iterate t satisfies

$$\Delta\tilde{b}_t \approx \nabla_a \tilde{b}\big|_{a_t}^{\top} \Delta a_t, \qquad \Delta a_t \propto \nabla_\delta \mathcal{J}(\delta_t). \tag{13}$$

Path-coupled terms broaden the support of useful gradients by introducing intermediate constraints, improving finite-budget crossing probability when endpoint-only gradients are sparse.

In white-box mode, both endpoint-only and path-coupled attacks are $\mathcal{O}(IK)$ where I is outer iterations and K rollout steps. We monitor objective decomposition term-by-term using Eq. 5 for post-hoc attribution.

3.5 Detection considerations and scope

A natural follow-up question is whether rollout-level inconsistency can be used for runtime screening. One possible diagnostic is step-count consistency, which would compare action outputs under different ODE integration step counts and flag inputs whose predictions vary strongly across rollout resolutions. We do not formalize or evaluate this screening rule as a validated defense result in this study: the low-budget regimes studied in the main experiments (1/255–3/255), adaptive attackers that explicitly minimize the screening score, and false-positive calibration across tasks require a dedicated evaluation. We therefore treat step-count consistency only as a prospective diagnostic and leave validated defense design to future work. The tau-window pruning idea mentioned in Section 3 is also an optimization-side acceleration idea, not a detector, and it is not used in the reported experiments.

4 Experimental Setup

4.1 Models, benchmark, protocol, and compute setting

The evaluation target is $\pi_{0.5}$ with an OpenPI-compatible LIBERO protocol (Table 1). We focus on deep analysis of a single flow-matching architecture rather than breadth across multiple models; consequently, all architectural claims are restricted to the evaluated $\pi_{0.5}$ implementation and should be tested on additional flow-matching or diffusion-policy systems before being generalized. Seed sets are reported only after passing the clean-success, zero-fallback, and budget-validity checks.

Table 1. Experimental setup and evaluation protocol

Item	Configuration
Task benchmark	LIBERO spatial task 1 (OpenPI-compatible, primary); tasks 2, 3 (cross-task)
Main model	$\pi_{0.5}$ flow-based action head (OpenPI), K=5 flow steps
Architecture scope	Single-architecture mechanism study of $\pi_{0.5}$; broader flow-policy generalization is outside scope
Seeds	Candidate seeds 7–16; valid n reported per condition after pre-specified CleanSR ≥ 0.8 filtering; 3 trials per seed
Perturbation budget	$\varepsilon \in \{3/255, 2/255, 1/255\}$; $d_{\text{RMS}} \leq 0.02$ (vision-only)
Primary metrics	CleanSR, AdvSR, Drop (= CleanSR − AdvSR), EP Shift, Path Shift
Attack config	I=40 PGD iters, α=0.007, λ_{path}=0.60, λ_{margin}=1.20, momentum β=0.5
Compute	4× RTX 4090 (24 GB), parallel seed execution

All reported attack results use the valid seed set shown in the corresponding table. Both TPCA and Visual-PGD use identical I=40 PGD iterations (no restarts), ensuring strictly matched gradient compute. The only difference is the flow-path coupling term (λ_{path}=0.60 for TPCA, λ_{path}=0 for Visual-PGD). Budget compliance is verified per-seed with zero-violation tolerance.

Task selection follows a "primary-plus-validation" policy: we use LIBERO spatial task 1 as the primary benchmark (stable CleanSR = 1.0, clear non-zero baseline Drop) and LIBERO spatial tasks 2 and 3 as cross-task validation targets (§5.5). All three tasks share the same $\pi_{0.5}$ policy backbone, isolating task-specific variation from architecture-level effects. We keep LIBERO/OpenPI-compatible tasks as the centerline and avoid introducing additional benchmark families to prevent compute dilution and interpretation ambiguity.

4.2 Baselines, implementation details, and statistical reporting

The main comparison includes TPCA, Visual-PGD, CW-L2 (visual), and Random-Noise. Visual-PGD uses the same executed-window chunk objective as TPCA (with `optimize_chunk_endpoint=true`) but sets λ_{path}=0, disabling flow-path coupling. Both TPCA and Visual-PGD use I=40 PGD iterations with no restarts, ensuring strictly matched gradient compute—the only difference is the flow-coupling switch. CW-L2 uses the same endpoint objective with an L2 penalty, and Random-Noise applies uniform random perturbations within the budget.

All methods use identical budget envelopes and endpoint extraction conventions, operating through differentiable PyTorch internals with verified gradient flow; any run with non-zero fallback rate is excluded.

The statistical protocol reports mean and standard deviation over up to n=10 independent seeds (7–16) with 3 episode trials per seed; the actual number of valid seeds per condition is n=8−9 after clean-success filtering. To prevent baseline policy instability from inflating the adversarial Drop metric, we predefine a clean-success filter and exclude seeds with CleanSR < 0.8 before attack comparison. Across conditions, 1–2 of 10 candidate seeds are typically excluded (most commonly seeds 8 and 9, whose CleanSR=0.6−0.8 reflects task-specific policy variance rather than attack effects). We use exact paired permutation tests (enumerating all 2^n sign-flip permutations) for pairwise method comparisons on continuous metrics (EP shift, path shift), and sign-consistency checks for directional claims. We emphasize directional consistency across seeds to reduce cherry-picking risk. Tables report the resulting valid n for every condition. Metrics include CleanSR, AdvSR, Drop, chunk-endpoint shift, and flow-path shift; gains are interpreted only for conditions with matched valid seed pairs.

Confidence intervals are computed from empirical seed-level variance, and all interval-based interpretations are qualified by direction-consistency checks. For seed-level gains $\Delta_b^{(s)} = \text{Drop}_{TPCA}^{(s)} - \text{Drop}_b^{(s)}$, we report both mean gain and sign ratio

$$\Gamma_b = \frac{1}{S} \sum_{s=1}^{S} \mathbf{1}(\Delta_b^{(s)} > 0), \tag{14}$$

where S is the number of evaluation seeds. A gain is considered directionally stable only when $\Gamma_b = 1$.

Implementation-level controls include deterministic seed wiring, immutable config snapshots, and controlled metric extraction from unified attack summaries.

4.3 Threat scenarios and evaluation checklist

We evaluate three attacker goals: (i) manipulation deviation (wrong-object grasping or placement), (ii) trajectory degradation (path dynamics deviate despite endpoint success), and (iii) mechanism diagnostics (how endpoint and path-space quantities differ under matched compute). Each run passes a reproducibility checklist: budget validation, matched endpoint extractors, fixed seed lists, deterministic aggregation, and complete result reporting. Flow-specific terms (path shift, velocity-field divergence) are interpreted as mechanism-analysis quantities specific to the evaluated τ-indexed ODE rollout, not universal measures of physical harm or robustness across all VLA families. Both methods are evaluated under identical optimization configurations to isolate the contribution of flow-aware coupling.

Table 2. Main results at the primary budget (ε=3/255). All rows use n=9 valid seeds after excluding CleanSR < 0.8. Both TPCA and Visual-PGD reach near-complete task failure, while CW-L2 and Random-Noise produce zero Drop under this implementation. Under identical gradient compute (I=40, no restarts), TPCA produces larger action-space endpoint and rollout-path diagnostics, with the EP Shift difference significant under the paired permutation test.

Method	Valid seeds	Drop	EP Shift	Path Shift	p-value	EP Wins
TPCA (ours)	9	**0.98±0.06**	**1.91±0.24**	**1.98±0.09**	0.004	8/9
Visual-PGD	9	0.98±0.06	1.41±0.11	1.52±0.08		
CW-L2(visual)	9	0.00	<0.1	—	—	—
Random-Noise	9	0.00	<0.1	—	—	—

Both TPCA and Visual-PGD use I=40 PGD iterations with no restarts (strictly matched compute); the only difference is λ_{path}. Values reported as mean ± std across the valid primary-task seed set. Valid seeds are selected before attack comparison using the pre-specified CleanSR $\geq$ 0.8 criterion over candidate seeds 7–16; the table reports the resulting seed count rather than treating CleanSR as an attack-specific outcome. Drop denotes the attack-induced reduction in task success relative to clean performance. EP Shift = mean chunk-endpoint L_2 displacement. Path Shift = mean velocity-field divergence along τ rollout. EP Shift and Path Shift are action-space policy-output diagnostics and should not be interpreted as physical contact, collision, or harm measurements. p-value from exact paired permutation test (2^9=512 permutations). CW-L2 is retained as a diagnostic baseline; in this implementation, its L_2 penalty suppresses effective perturbation strength below the task-failure threshold. "—" indicates metrics not applicable (CW-L2 and Random produce zero Drop, so paired comparison and Path Shift are undefined).

5 Results and Analysis

5.1 Setup sanity and reproducibility checks

Before comparing attack strength, we validate three constraints: (i) uniform budget compliance across methods, (ii) consistent rollout schedule between clean and attacked endpoints, and (iii) deterministic metric extraction from seed-level logs. The clean-reference endpoint extractor is verified to be identical across methods, and every excluded seed is accounted for through the clean-success filter reported in the tables.

Gradient flow is validated via runtime diagnostics: every run logs the fraction of non-zero gradient components per optimization step. The mainline TPCA configuration achieves 100% non-zero gradient rate across all seeds, confirming that the differentiable PyTorch path provides valid optimization signals throughout the attack. LIBERO spatial task 1 is selected as the primary benchmark because it produces stable CleanSR = 1.0 across all seeds.

5.2 Main results under matched compute

Table 2 reports the matched-compute comparison at the primary budget ε=3/255. Both TPCA and Visual-PGD use identical optimization configurations (I=40 PGD iterations, no restarts)—the only difference is the flow-path coupling term (λ_{path}=0.60 vs 0). This design controls compute fairness: within the evaluated implementation, differences between TPCA and Visual-PGD isolate the effect of enabling the flow-path coupling term.

Both gradient-optimized methods achieve Drop $\approx$ 0.98 (near-complete task failure) across valid seeds, while CW-L2 and Random-Noise achieve Drop $\approx$ 0. We include CW-L2 not as a competitive baseline but as a diagnostic control: its L_2 penalty concentrates perturbation energy into a sparse pixel subset, suppressing the effective L_∞ magnitude below the threshold needed to cross the task-failure boundary. This confirms that the budget alone does not guarantee task failure—effective gradient exploitation with sufficient L_∞ utilization is required. These results indicate that the $\pi_{0.5}$ flow-matching policy is susceptible to gradient-optimized white-box visual attacks at moderate perturbation budgets.

While Drop is identical, TPCA produces +36% larger mean EP Shift (1.91 ± 0.24 vs 1.41 ± 0.11, p=0.004, 8/9 paired seeds positive) and larger Path Shift (1.98 ± 0.10 vs 1.52 ± 0.09), showing that flow-aware coupling increases rollout-level action-space displacement even when task-level Drop is near-saturated. Visual-PGD's non-zero Path Shift is an incidental byproduct of endpoint perturbation propagating through the ODE rollout, not a directly optimized quantity.

5.3 Budget-regime analysis

Table 3. Budget-regime analysis: TPCA vs Visual-PGD across three perturbation budgets under strictly matched compute (I=40, no restarts). The EP Shift advantage increases as budget tightens. At 2/255 and 1/255, TPCA also achieves higher Drop, while at 3/255 task failure is near-saturated for both optimized attacks.

ε	Method	Drop	EP Shift	Path Shift	p	EP Wins	n
3/255	TPCA (ours)	0.98±0.06	**1.91±0.24**	**1.98±0.09**	.004	8/9	9
	Visual-PGD	0.98±0.06	1.41±0.11	1.52±0.08			
2/255	TPCA (ours)	**0.91±0.14**	**1.43±0.24**	**1.54±0.16**	.002	9/9	9
	Visual-PGD	0.87±0.13	0.94±0.21	1.05±0.19			
1/255	TPCA (ours)	**0.38±0.21**	**0.48±0.10**	**0.64±0.08**	.004	8/8	8
	Visual-PGD	0.17±0.16	0.21±0.04	0.26±0.03			

Both methods use identical configurations (I=40 PGD iterations, no restarts) at all budget levels; the only difference is λ_{path}. Seeds with CleanSR < 0.8 are excluded. p-values from exact paired permutation tests on EP Shift. At 1/255, TPCA achieves 2.2× the Drop of Visual-PGD while producing +129% larger EP Shift on valid paired seeds. Across all budgets, 25/26 paired seeds favor TPCA in EP Shift.

Table 3 shows that the action-space endpoint-displacement advantage of flow-aware coupling increases as perturbation budget tightens. Task-level Drop is near-saturated at 3/255, but TPCA achieves higher Drop at the constrained 2/255 and 1/255 regimes.

At 3/255 (near-saturation regime), both methods achieve Drop ≈ 0.98 but TPCA produces +36% larger EP Shift (1.91 vs 1.41, p=0.004, n=9). At 2/255 (transition regime), TPCA maintains Drop = 0.91 while Visual-PGD weakens to 0.87, with EP Shift advantage widening to +52% (1.43 vs 0.94, p=0.002, 9/9 paired seeds). At 1/255 (sub-threshold regime), the gap is most pronounced: TPCA achieves Drop = 0.38 while Visual-PGD manages only 0.17—a 2.2× difference—with +129% EP Shift advantage (0.48 vs 0.21, p=0.004, 8/8 paired seeds).

The EP Shift advantage increases monotonically (+36% → +52% → +129%; Figure 3), consistent with the hypothesis that flow-aware coupling extracts more displacement per unit of perturbation budget. Below this near-saturation regime (2/255 and 1/255), TPCA achieves both higher Drop and larger action-space endpoint displacement—a dual advantage not visible from Drop alone at 3/255. The 1/255 case is particularly relevant for threat modeling: at a budget roughly one-eighth of the standard adversarial-ML benchmark, flow-aware attacks retain partial effectiveness while endpoint-only optimization becomes largely ineffective (Drop = 0.17).

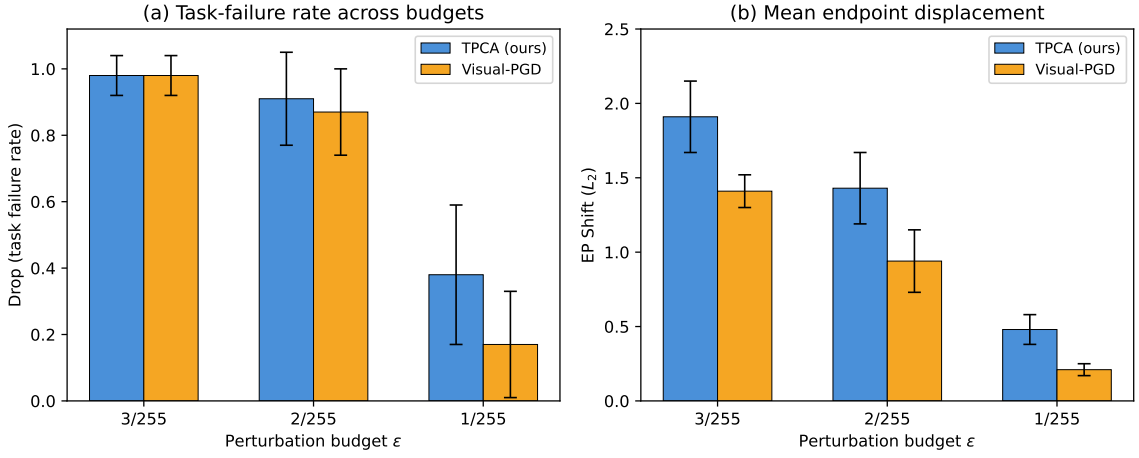

Figure 3. Budget-regime analysis. (a) Task-failure rate (Drop) across three perturbation budgets. At 3/255 both methods are near-saturated; at 2/255 and 1/255 TPCA achieves higher Drop. (b) Endpoint displacement (EP Shift) across budgets. TPCA's advantage widens monotonically as budget tightens. Both methods use identical I=40 iterations with no restarts.

Figure 4 visualizes the monotonic trend in relative EP Shift advantage. The pattern is consistent with a gradient-density hypothesis: intermediate velocity-field supervision provides denser optimization signal that becomes increasingly valuable as the perturbation budget constrains the feasible search space.

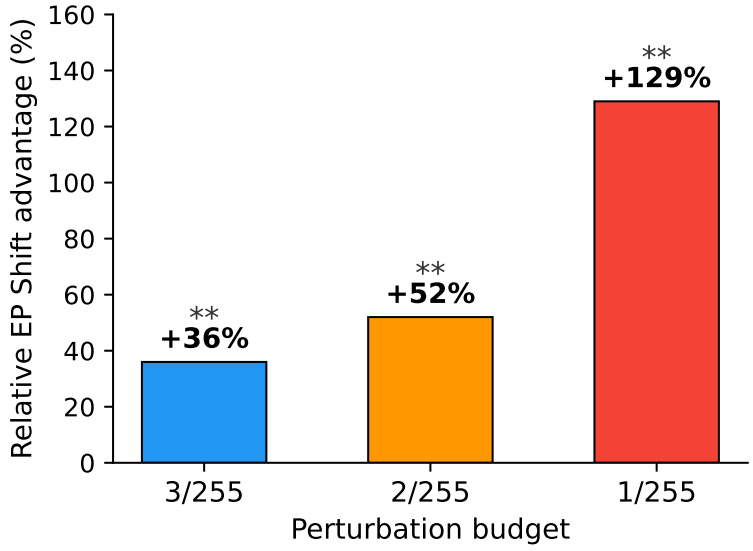

Figure 4. Relative EP Shift advantage (%) of TPCA over Visual-PGD. The advantage increases monotonically from +36% at 3/255 to +129% at 1/255. $*\,p<0.01$; $**\,p<0.005$; all survive Bonferroni correction.

Figure 5 shows per-seed EP Shift and Drop at 1/255, the most informative budget level. TPCA produces higher EP Shift on every seed (8/8), and achieves non-zero Drop on 6/8 seeds versus only 4/8 for Visual-PGD.

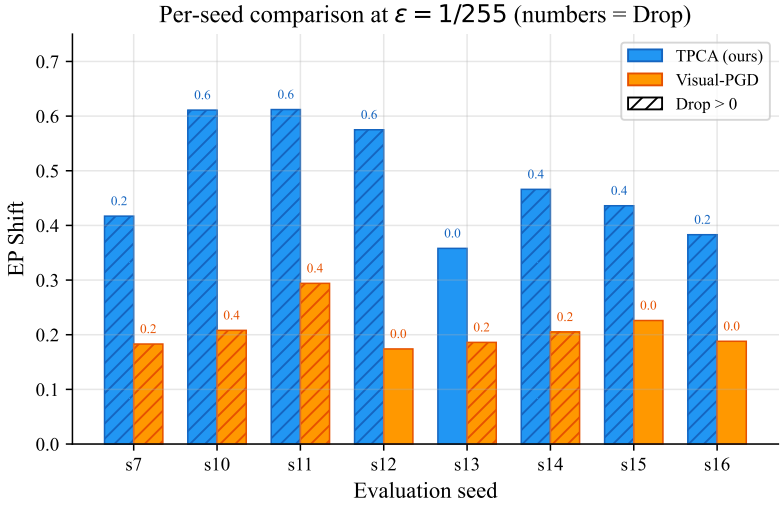

Figure 5. Per-seed comparison at ε=1/255. Bar height = EP Shift; number above each bar = Drop. Hatched bars (//) indicate non-zero Drop. TPCA (blue) produces higher EP Shift on every seed (8/8). Visual-PGD achieves Drop = 0 on 4/8 seeds while TPCA achieves Drop = 0 on only 2/8.

5.4 Ablation study on flow-path coupling

The budget-regime analysis (Table 3) includes a controlled switch ablation: Visual-PGD uses the identical optimization pipeline as TPCA (I=40 iterations, same momentum, same projection) with the sole difference being λ_{path}=0. The widening EP gap across budgets (+36% → +52% → +129%) directly quantifies the marginal value of flow-path coupling. At the near-saturated 3/255 level the path term improves EP but not Drop; at tighter budgets it improves *both*, indicating that the gradient density provided by intermediate velocity-field supervision becomes increasingly important when the perturbation budget constrains the feasible optimization landscape.

We fix $\lambda_{\text{path}} = 0.60$ via pilot runs on seeds 0–6 (disjoint from the evaluation set 7–16) to avoid hyperparameter overfitting. Table 4 provides a direct switch ablation at 2/255 showing the effect of enabling the selected path-coupling weight. Prior experiments with reduced iterations (I=3) at higher budgets show that optimization depth is necessary: Drop collapses to ∼0.07 even with path coupling enabled, confirming that the path term amplifies but does not replace sufficient optimization depth.

Table 4. Direct switch ablation of flow-path coupling at ε=2/255 (I=40, no restarts). The sole difference between the two configurations is whether the selected path-coupling weight is disabled (λ_{path}=0) or enabled (λ_{path}=0.60); this is not a λ_{path} sweep or sensitivity analysis.

Configuration	λ_{path}	Drop	EP Shift	n
Visual-PGD	0.0	0.87±0.13	0.94±0.21	9
TPCA (ours)	0.6	**0.91±0.14**	**1.43±0.24**	9

5.5 Cross-task robustness check

To test whether the observed behavior is limited to the primary task, we repeat the budget-regime analysis on LIBERO spatial tasks 2 and 3 at 2/255 and 1/255 under matched compute (I=40, no restarts). Table 5 summarizes the results.

Table 5. Cross-task validation at 2/255 and 1/255 on LIBERO spatial tasks 2 and 3. EP Shift advantage is directionally consistent across all four conditions (32/32 paired seeds). At 2/255, TPCA achieves higher mean Drop on both tasks. At 1/255, task failure is near zero for both methods on these cross-tasks, but TPCA maintains +71−94% larger EP Shift.

ε	Task	Method	Drop	EP Shift	EP Wins	n
2/255	2	TPCA	**0.35±0.13**	**0.63±0.10**	8/8	8
		V-PGD	0.22±0.16	0.39±0.09		
	3	TPCA	0.47±0.24	**0.92±0.17**	8/8	8
		V-PGD	0.43±0.23	0.54±0.12		
1/255	2	TPCA	0.02±0.07	**0.29±0.01**	8/8	8
		V-PGD	0.00±0.00	0.17±0.03		
	3	TPCA	0.05±0.09	**0.35±0.06**	8/8	8
		V-PGD	0.00±0.00	0.18±0.03		

All conditions use n=8 paired seeds with CleanSR ≥ 0.8. All EP Shift p-values < 0.004 (exact paired permutation tests). EP Wins = paired seeds where TPCA EP Shift exceeds Visual-PGD.

Three patterns hold across the cross-tasks. First, at 2/255 TPCA achieves higher mean Drop on both tasks: 0.35 vs 0.22 on task 2 and 0.47 vs 0.43 on task 3, consistent with the primary-task finding that flow-aware coupling improves task-failure rate at constrained budgets. Second, EP Shift advantage is directionally consistent across all four cross-task conditions (32/32 paired seeds positive, all p<0.004). Third, at 1/255 Visual-PGD achieves Drop ≈ 0 on both cross-tasks, while TPCA maintains +71−94% larger EP Shift—indicating that flow-aware coupling produces measurable action-space deviation below the task-failure threshold on these additional tasks. This does not by itself establish cross-architecture generality.

6 Discussion

This section interprets the results within their empirical scope. The evidence supports a mechanism-level finding for the evaluated $\pi_{0.5}$ policy: coupling endpoint displacement with flow-rollout velocity divergence changes the optimization behavior relative to endpoint-only Visual-PGD. It does not establish that every flow-matching VLA policy, diffusion policy, or real robot will exhibit the same vulnerability profile.

6.1 Attack mechanism and effectiveness

At 3/255, both optimized methods are near-saturated at Drop ≈ 0.98, so binary task failure alone cannot distinguish their severity. The budget sweep breaks that symmetry. At 2/255, TPCA retains Drop=0.91 versus 0.87 for Visual-PGD; at 1/255, the ratio reaches 2.2× (0.38 vs 0.17). The monotonic widening of the EP Shift gap (+36% → +52% → +129%) as ε decreases from 3/255 to 1/255 is consistent with a gradient-density explanation: when the feasible perturbation set becomes small, intermediate velocity-field comparisons can provide useful optimization signal that endpoint-only objectives lack. We state this as an empirical interpretation, not a proof that path coupling will dominate on every flow-based policy.

6.2 Interpreting trajectory diagnostics

EP Shift and Path Shift are action-space diagnostics. In the LIBERO manipulation domain, the first three action dimensions correspond to end-effector translation commands, so larger EP Shift indicates a larger commanded displacement relative to the clean rollout. These metrics are more interpretable than feature-space distances, but they are not direct measurements of contact force, collision probability, object damage, or human-safety risk. A high EP Shift should therefore be read as evidence of larger policy-level action deviation, not as a calibrated physical-harm score. Validating the connection between these diagnostics and real-robot safety outcomes remains future work.

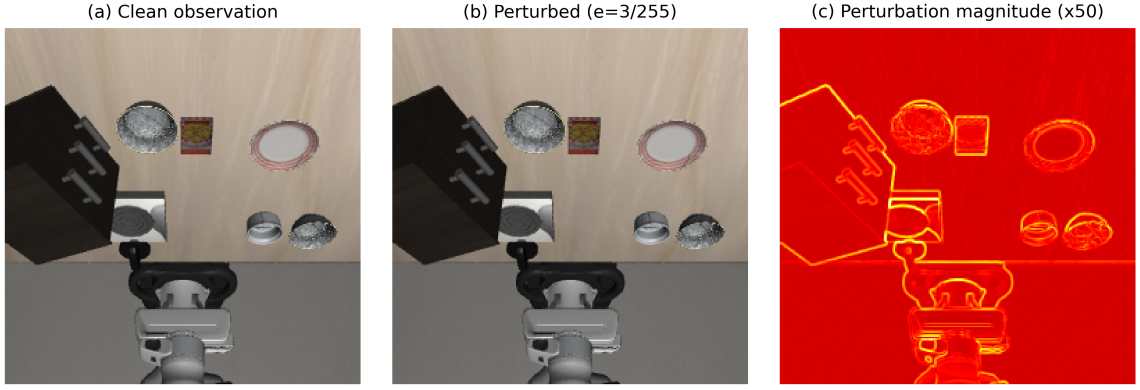

Figure 6. Simulated LIBERO spatial task 1 observation under adversarial perturbation (ε=3/255). Clean and perturbed camera observations are visually similar, while the amplified perturbation map shows where the bounded perturbation concentrates.

6.3 Scope, natural perturbations, and defenses

The experiments use optimized digital perturbations in simulation. They should not be equated with natural corruptions such as Gaussian noise, motion blur, compression artifacts, lighting variation, or camera smudges. Such corruptions may interact with the perception stack differently and require separate evaluation. The results also do not validate a deployable defense. Step-count consistency screening, input sanitization, adversarial training, and adaptive attacks against such defenses are plausible follow-up directions, but we do not report defense performance numbers. Finally, the study evaluates one OpenPI-compatible $\pi_{0.5}$ policy on LIBERO spatial tasks. Testing additional flow-matching action heads, diffusion policies such as Diffusion Policy, and real-robot deployments is necessary before making broad claims about embodied systems outside this benchmark.

7 Conclusion

TPCA couples executed-window endpoint displacement with velocity-field divergence along the τ-indexed rollout of an evaluated flow-matching embodied policy. On $\pi_{0.5}$ in LIBERO, the matched-compute budget sweep shows that task-level Drop is near-saturated for both TPCA and Visual-PGD at 3/255, while TPCA produces larger action-space endpoint displacement. As the budget tightens, the difference becomes visible in task failure: at 1/255, TPCA reaches Drop = 0.38 compared with 0.17 for Visual-PGD, with +129% larger EP Shift. Cross-task checks support consistent action-space endpoint-displacement advantages on two additional LIBERO spatial tasks, although task-level Drop remains budget- and task-dependent.

The main conclusion is therefore bounded. For the evaluated $\pi_{0.5}$ policy, flow-rollout path coupling provides additional optimization signal beyond endpoint-only Visual-PGD, especially under constrained visual budgets. The evidence does not yet establish universal vulnerability of all flow-matching policies, does not validate defenses, and does not measure real-world physical harm. Future work should test additional flow-matching and diffusion-style action heads, evaluate natural corruptions and adaptive defenses, and connect action-space diagnostics to real-robot safety outcomes.

8 Ethics, Disclosure, and Reproducibility

This work is conducted for defensive security research. All experiments are simulation-based with no real-robot harmful deployment. We disclose methods and aggregate results for robustness evaluation while withholding operationally optimized attack presets until corresponding mitigations are available. Reproducibility artifacts include fixed seed protocols, configuration snapshots, and script-based table/figure generators. Public release should follow responsible disclosure norms.

Acknowledgments

The authors thank Dr. Haoming Liu from Zhejiang University for his helpful feedback on embodied AI security evaluation practice.

Funding

This work was supported in part by the National Natural Science Foundation of China under Grant 62293503, 62293500, 62293502; in part by the Frontier Technologies R&D Program of Jiangsu under Grant BF2024065; in part by the State Key Laboratory of Industrial Control Technology under Grant ICT2024A13, ICT2025C04; in part by the Xiaomi Foundation; and in part by the Fundamental Research Funds for the Central Universities under Grant 226-2025-00165.

Conflicts of Interest

The authors declare no competing interests.

Data Availability

The original data used in this study are available from the corresponding author upon reasonable request. All valid seed counts used in the main comparisons are reported in the paper.

Authors' Contributions

Siyuan Pan conceived the threat model, designed the method, implemented the attack pipeline, conducted the experiments, performed the statistical analysis, and drafted the manuscript. Rong Guo contributed to discussions on experimental design and provided feedback on the evaluation protocol. Mengxiang Liu supervised the research, advised on methodological refinement, and revised the manuscript. Ruilong Deng acquired funding, provided overall guidance on the research direction, and revised the manuscript.

References

[1] 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. In *Robotics: Science and Systems (RSS)*, 2023.

[2] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Peter Florence, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning (CoRL)*, 2023.

[3] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems (RSS)*, 2023.

[4] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as I can, not as I say: Grounding language in robotic affordances. In *Conference on Robot Learning (CoRL)*, pages 287–318, 2022.

[5] Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. PaLM-E: An embodied multimodal language model. In *International Conference on Machine Learning (ICML)*, pages 8469–8488, 2023.

[6] Zhiqiang Wang, Zhehua Zhou, Jiayang Song, Yuheng Huang, Zhan Shu, and Lei Ma. VLATest: Testing and evaluating vision-language-action models for robotic manipulation. *Proc. ACM Softw. Eng.*, 2(FSE):1615–1638, 2025.

[7] Yunpeng Huang, Zhenyu Wang, Zhixiong Wan, Yu Tian, Hang Xu, Yanjie Han, et al. ANNIE: Be careful of your robots. *arXiv preprint arXiv:2509.03383*, 2025.

[8] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, et al. π_0: A vision-language-action flow model for general robot control. In *Robotics: Science and Systems (RSS)*, 2025.

[9] Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

[10] 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 (ICLR)*, 2014.

[11] Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations (ICLR)*, 2015.

[12] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *IEEE Symposium on Security and Privacy (S&P)*, 2017.

[13] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[14] Adam Gleave, Michael Dennis, Neel Kant, Cody Wild, Sergey Levine, and Stuart J. Russell. Adversarial policies: Attacking deep reinforcement learning. In *International Conference on Learning Representations (ICLR)*, 2020.

[15] Yuzhe Ma, Xuezhou Zhang, Wen Sun, and Jerry Xiaojin Zhu. Policy poisoning in batch reinforcement learning and control. In *Neural Information Processing Systems (NeurIPS)*, 2019.

[16] Aishan Liu, Yuguang Zhou, Xianglong Liu, Tianyuan Zhang, Siyuan Liang, Jiakai Wang, et al. Compromising LLM-driven embodied agents with contextual backdoor attacks. *IEEE Transactions on Information Forensics and Security*, 2025.

[17] Yueen Ma, Zixing Song, Yuzheng Zhuang, Jianye Hao, and Irwin King. A survey on vision-language-action models for embodied AI. *arXiv preprint arXiv:2405.14093*, 2024.

[18] Kento Kawaharazuka, Jungwoo Oh, Jun Yamada, Ingmar Posner, and Yuke Zhu. Vision-language-action models for robotics: A review towards real-world applications. *IEEE Access*, 2025.

[19] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning (ICML)*, pages 274–283, 2018.

[20] Francesco Croce and Matthias Hein. Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In *International Conference on Machine Learning (ICML)*, pages 2206–2216, 2020.

[21] Pin-Yu Chen, Yash Sharma, Huan Zhang, Jinfeng Yi, and Cho-Jui Hsieh. EAD: Elastic-net attacks to deep neural networks via adversarial examples. In *AAAI Conference on Artificial Intelligence*, 2018.

[22] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[23] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, Somesh Jha, Z. Berkay Celik, and Ananthram Swami. Practical black-box attacks against machine learning. In *ACM Asia Conference on Computer and Communications Security (AsiaCCS)*, 2017.

[24] Wieland Brendel, Jonas Rauber, and Matthias Bethge. Decision-based adversarial attacks: Reliable attacks against black-box machine learning models. In *International Conference on Learning Representations (ICLR)*, 2018.

[25] Jiawei Su, Danilo Vasconcellos Vargas, and Kouichi Sakurai. One pixel attack for fooling deep neural networks. *IEEE Transactions on Evolutionary Computation*, 23(5):828–841, 2019.

[26] Yandong Li, Lijuan Li, Liqiang Wang, Tong Zhang, and Boqing Gong. NATTACK: Learning the distributions of adversarial examples for an improved black-box attack on deep neural networks. In *International Conference on Machine Learning (ICML)*, 2019.

[27] Anish Athalye, Logan Engstrom, Andrew Ilyas, and Kevin Kwok. Synthesizing robust adversarial examples. In *International Conference on Machine Learning (ICML)*, 2018.

[28] Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *AISec@CCS*, 2017.

[29] Weilin Xu, David Evans, and Yanjun Qi. Feature squeezing: Detecting adversarial examples in deep neural networks. In *Network and Distributed System Security Symposium (NDSS)*, 2018.

[30] 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 (ICLR)*, 2018.

[31] Ali Shafahi, Mahyar Najibi, Amin Ghiasi, Zheng Xu, John P. Dickerson, Christoph Studer, Larry S. Davis, Gavin Taylor, and Tom Goldstein. Adversarial training for free! In *Neural Information Processing Systems (NeurIPS)*, pages 3353–3364, 2019.

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

[33] Hongyang Zhang, Yaodong Yu, Jiantao Jiao, Eric Xing, Laurent El Ghaoui, and Michael I. Jordan. Theoretically principled trade-off between robustness and accuracy. In *International Conference on Machine Learning (ICML)*, 2019.

[34] Jeremy M. Cohen, Elan Rosenfeld, and J. Zico Kolter. Certified adversarial robustness via randomized smoothing. In *International Conference on Machine Learning (ICML)*, 2019.

[35] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[36] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations (ICLR)*, 2021.

[37] Amin Rakhsha, Goran Radanovic, Rati Devidze, Xiaojin Zhu, and Adish Singla. Policy teaching via environment poisoning: Training-time adversarial attacks against reinforcement learning. In *International Conference on Machine Learning (ICML)*, 2020.

[38] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan M. Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, and Johnny Lee. Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning (CoRL)*, pages 726–747, 2021.

[39] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. CLIPort: What and where pathways for robotic manipulation. In *Conference on Robot Learning (CoRL)*, 2022.

[40] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. In *Neural Information Processing Systems (NeurIPS)*, pages 15084–15097, 2021.

[41] Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. In *Neural Information Processing Systems (NeurIPS)*, pages 1273–1286, 2021.

[42] Nur Muhammad Shafiullah, Zichen Jeff Cui, Ariuntuya Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning k modes with one stone. In *Neural Information Processing Systems (NeurIPS)*, 2022.

[43] Dhruv Shah, Błażej Osiński, Brian Ichter, and Sergey Levine. LM-Nav: Robotic navigation with large pre-trained models of language, vision, and action. In *Conference on Robot Learning (CoRL)*, pages 492–504, 2022.

[44] Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Language models as zero-shot planners: Extracting actionable knowledge for embodied agents. In *International Conference on Machine Learning (ICML)*, 2022.

[45] Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[46] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.

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

[48] Open X-Embodiment Collaboration. Open X-Embodiment: Robotic learning datasets and RT-X models. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903, 2024.

[49] Xiao Ma, Sumit Patidar, Iain Haughton, and Stephen James. Hierarchical diffusion policy for kinematics-aware multi-task robotic manipulation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024.

[50] Jiahang Liu, Menglin Liu, Zongli Wang, Peng An, Xueqian Li, Kun Zhou, et al. RoboMamba: Efficient vision-language-action model for robotic reasoning and manipulation. In *Neural Information Processing Systems (NeurIPS)*, 2024.

[51] Junjie Wen, Yichen Zhu, Jinming Li, Minjie Zhu, Zhiyuan Tang, Kun Wu, et al. TinyVLA: Toward fast, data-efficient vision-language-action models for robotic manipulation. *IEEE Robotics and Automation Letters*, 2025.

[52] An-Chieh Cheng, Yandong Ji, Zhuoran Yang, Xiaofeng Zou, Jan Kautz, Erdem Biyik, et al. NaVILA: Legged robot vision-language-action model for navigation. In *Robotics: Science and Systems (RSS)*, 2025.

[53] Zhiyuan Zhou, Yichen Zhu, Minjie Zhu, Junjie Wen, Ning Liu, Zhiyuan Xu, et al. ChatVLA: Unified multimodal understanding and robot control with vision-language-action model. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 5377–5395, 2025.

[54] Zixian Wu, Yifan Zhou, Xin Xu, Zhenyu Wang, and Hang Yan. MoManipVLA: Transferring vision-language-action models for general mobile manipulation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

[55] Chengmao Li, Junjie Wen, Yiming Peng, Yuzhuo Peng, Fan Feng, and Yichen Zhu. PointVLA: Injecting the 3D world into vision-language-action models. *arXiv preprint arXiv:2503.07511*, 2025.

[56] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Pannag Lam, Pannag Sanketi, et al. OpenVLA: An open-source vision-language-action model. In *Conference on Robot Learning (CoRL)*, 2024.

[57] Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Chelsea Finn, and Sergey Levine. FAST: Efficient action tokenization for vision-language-action models. In *Robotics: Science and Systems (RSS)*, 2025.

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

[59] Yanjiang Guo, Jianke Zhang, Xiaoyu Chen, Xiang Ji, Yen-Jen Wang, Yucheng Hu, et al. Improving vision-language-action model with online reinforcement learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[60] International Organization for Standardization. ISO/TS 15066:2016—robots and robotic devices—collaborative robots. Technical report, ISO, Geneva, 2016.

Siyuan Pan received the Bachelor of Engineering degree from the National University of Defense Technology, Changsha, China, in 2024. He is currently a master's student at the College of Control Science and Engineering, Zhejiang University. His research interests include embodied intelligence security, vision-language-action models, and adversarial robustness.

Rong Guo received the Bachelor of Engineering degree from North China Electric Power University, Beijing, China, in 2023. She is currently a combined Master's-Ph.D. student at the College of Control Science and Engineering, Zhejiang University. Her research interests include HVDC, CPS and cybersecurity.

Mengxiang Liu received the B.Sc. degree in Automation from Tongji University, Shanghai, in 2017 and the Ph.D. degree in Cyberspace Security from Zhejiang University, Hangzhou, in 2022. He is currently a Lecturer in Cyber-Physical Systems Security with the School of Computer Science, University of Bristol, Bristol, UK. His research interests include cyber-physical systems security, smart grid, and industrial control systems security.

Ruilong Deng received the B.Sc. and Ph.D. degrees both in Control Science and Engineering from Zhejiang University, Hangzhou, Zhejiang, China, in 2009 and 2014, respectively. He was a Research Fellow with Nanyang Technological University, Singapore, from 2014 to 2015; an AITF Postdoctoral Fellow with the University of Alberta, Edmonton, AB, Canada, from 2015 to 2018; and an Assistant Professor with Nanyang Technological University, from 2018 to 2019. Currently, he is a Professor with the College of Control Science and Engineering, Zhejiang University; and a Deputy Director of the State Key Laboratory of Industrial Control Technology. His research interests include smart grid, cyber security, and control systems. Dr. Deng serves/served as an Associate Editor for IEEE Transactions on Smart Grid, IEEE Power Engineering Letters, IEEE/CAA Journal of Automatica Sinica, and IEEE/KICS Journal of Communications and Networks, and a Guest Editor for IEEE Transactions on Cloud Computing, IEEE Transactions on Emerging Topics in Computing, IEEE Journal of Emerging and Selected Topics in Industrial Electronics, and IET Cyber-Physical Systems: Theory & Applications. He also serves/served as a Symposium Chair for IEEE SmartGridComm, IEEE ICPS, and IEEE GLOBECOM.