ATTACKING ACTION DISCRIMINABILITY IN LATENT WORLD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Latent world models support model-based control by learning action-conditioned state transitions in representation space. A key requirement of such models is that their predicted dynamics faithfully reflect the consequences of different actions, allowing downstream planners to distinguish alternative future trajectories. Many existing attacks are tied to specific action sequences or trajectories, whereas multiple alternative action sequences may reach the same goal, making persistence across plan variations and repeated replanning challenging. We instead target the action dependence of the latent dynamics itself, aiming to produce perturbations that are less tied to particular action sequences and remain persistent across multi-step rollout and replanning. Specifically, our attack operates at two complementary levels: action sensitivity suppression and multi-step discrepancy propagation. For the former, we construct action ensembles and directly reduce the dispersion of their predicted latent states, weakening the model's sensitivity to action variations. To further suppress the propagation of action-induced discrepancies over future rollout steps, we derive a finite-horizon sensitivity bound and reduce the dominant sensitivity of the latent transition dynamics. Additionally, we steer the attacked predictions away from their clean counterparts to induce erroneous rather than merely insensitive predictions. The proposed attack is evaluated on multiple JEPA-style latent world models across two control environments, demonstrating that suppressing action sensitivity in latent dynamics can effectively disrupt future-state prediction and downstream planning.

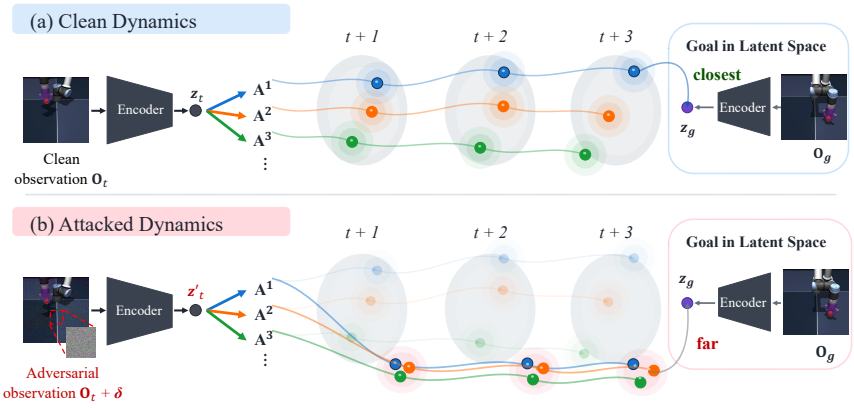


Figure 1: Illustration of clean and adversarial latent dynamics. Given different action sequences $\mathbf{A}$, clean dynamics preserve distinct action-conditioned trajectories that guide planning toward the goal, whereas our attack makes different action outcomes less distinguishable and drives the rollout away from the clean dynamics and the goal.

1 INTRODUCTION

World models (WMs) (Ha & Schmidhuber, 2018; Hafner et al., 2019b; 2023; Bruce et al., 2024; Zhou et al., 2024) have emerged as a powerful paradigm for learning environment dynamics, with applications in robotics, embodied intelligence, autonomous systems, and model-based control.

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.

Among different world-modeling paradigms, latent world models (Hafner et al., 2019b; 2023; Zhou et al., 2024) have become increasingly popular. Latent world models avoid direct pixel-level prediction by modeling environment dynamics in a learned representation space (Hafner et al., 2019b; 2023). Among them, JEPA-style world models directly predict future latent representations conditioned on actions, without reconstructing future observations (Zhou et al., 2024; Maes et al., 2026; Sobal et al., 2026). During model-based planning, multiple candidate action sequences are rolled out through the learned latent dynamics, and the planner selects the sequence whose predicted future representations are closest to the goal (Rubinstein & Kroese, 2004; Williams et al., 2017), as shown in Figure 1 (a). However, this design incurs a distinct vulnerability: downstream decisions depend on whether the learned latent dynamics can faithfully preserve the differences induced by different actions throughout the rollout. If adversarial perturbations weaken the dependence of latent predictions on candidate actions, the planner may fail to distinguish their consequences, leading to erroneous decisions even when the perturbation to the original observation is small.

Recent studies have begun to reveal adversarial vulnerabilities in world models. However, their applicability to JEPA-style latent world models remains limited in two distinct ways. First, attacks designed for generative world models (Guo et al., 2026a; Shen et al., 2026) often rely on attack surfaces such as diffusion denoising dynamics or auxiliary physical-conditioning signals, and therefore do not directly transfer to reconstruction-free latent predictors. Second, more generic attacks (Zhang et al., 2026; Duan et al., 2026) that can be adapted to predictive latent models target internal representation or prediction errors, or planner-specific goals. However, degrading a particular prediction or candidate trajectories does not necessarily lead to persistent planning failure. In model-based control, multiple action sequences may provide viable paths toward the same goal, and iterative planners repeatedly resample candidates. As a result, an attack tied to a specific prediction or candidate set may be bypassed by alternative plans or corrected during subsequent replanning.

To overcome these limitations, we target the action sensitivity of latent dynamics rather than individual trajectories. We consider a set of locally varied actions around the reference action and optimize the perturbation to encourage their predicted latent futures to converge, thus suppressing the model's ability to distinguish different action consequences. However, reducing cross-action variation alone could collapse the predictions near the correct dynamics. Therefore, we additionally push each perturbed prediction away from its corresponding clean prediction, encouraging the collapsed futures to be dynamically incorrect. Since the perturbation is applied only to the initial observation, suppressing action sensitivity at the first few prediction steps alone may not be sufficient, as discrepancies between action-conditioned rollouts can amplify over subsequent predictions. To characterize this effect, we derive a finite-horizon sensitivity bound on the distance between latent rollouts induced by different action sequences, which identifies the spectral norm of the state-transition Jacobian $\sigma_{max}(J_z)$ as a dominant multiplicative factor. Motivated by this analysis, we thereby reduce the $\sigma_{max}(J_z)$ to mitigate the propagation and amplification of action-dependent differences over the prediction horizon. Building on these objectives, we propose **Action-Insensitive Dynamics Attack (AIDA)**, which reduces the world-model predictor's sensitivity to action variations, while steering the predicted futures away from the clean dynamics and maintaining the effect across multi-step rollouts, as illustrated in Figure 1(b). Overall, our main contributions are:

- We identify action sensitivity in latent dynamics as a critical yet underexplored attack surface for JEPA-style world models, highlighting its role in distinguishing candidate actions during planning.
- We propose AIDA, a novel adversarial attack that drives diverse action-conditioned predictions toward similarly erroneous latent futures by suppressing cross-action dispersion, deviating from clean dynamics, and attenuating discrepancy propagation over multi-step rollouts.
- We evaluate AIDA on multiple JEPA-style world models and control benchmarks, showing its effectiveness in degrading downstream planning across multi-step rollout and iterative replanning, with consistent improvements over representative baselines.

2 Related Works

2.1 World Models

World models learn dynamics of the environment from observations, enabling agents to anticipate future states and support planning and decision making (Hafner et al., 2019a; Hu et al., 2023;

Wang et al., 2024). Among recent approaches, latent predictive world models directly model future representations in latent space, avoiding explicit reconstruction of future observations. The Joint-Embedding Predictive Architecture (JEPA) provides a representative formulation, learning representations by predicting target embeddings from contextual representations (Assran et al., 2023). DINO-WM (Zhou et al., 2024) is built on pretrained DINOv2 representations, learning action-conditioned predictions of future visual features for zero-shot planning. PLDM (Sobal et al., 2026) further demonstrates the effectiveness of JEPA-style latent dynamics for planning from reward-free offline trajectories, particularly emphasizing generalization under suboptimal data. V-JEPA 2-AC (Assran et al., 2025) scales latent predictive modeling by adapting a large self-supervised video model to action-conditioned robotic dynamics, enabling zero-shot planning on physical robots. LeWM (Maes et al., 2026) more recently showed that such joint-embedding world models can be trained stably end-to-end directly from pixels, without relying on pretrained visual encoders.

At test time, these models can be used for planning by rolling out candidate action sequences in the learned latent space and evaluating their predicted outcomes against a target representation. Planning is commonly formulated in a model predictive control (MPC) framework, where an action sequence is optimized over a finite horizon, and a short action prefix is executed. The procedure is repeated from the newly observed state (Hafner et al., 2019b; Hansen et al., 2022). The finite-horizon optimization can be solved using sampling-based methods such as the Cross-Entropy Method (CEM) (De Boer et al., 2005) and Model Predictive Path Integral (MPPI) (Williams et al., 2017) control. CEM iteratively samples candidate action sequences, retains low-cost elites, and refits the sampling distribution, whereas MPPI updates a nominal action sequence through cost-weighted averaging over sampled action sequences. Such planning procedures continually explore and refine diverse candidate action sequences during optimization. As a result, attacks tied to particular action sequences or predicted trajectory may weaken under candidate resampling or replanning. This motivates attacks that remain effective across action variations and repeated planning.

2.2 Adversarial Attacks on World Models

Adversarial attacks introduce imperceptible perturbations onto the input that cause the model to produce incorrect outputs (Goodfellow et al., 2014; Madry et al., 2017). Recently, growing attention has been devoted to adversarial vulnerabilities in world models (Guo et al., 2026a; Zhang et al., 2026; Shen et al., 2026; Guo et al., 2026b). BadWorld (Shen et al., 2026) attacks autoregressive visual world models by disrupting their denoising dynamics and further employs trajectory-adaptive optimization to generate perturbations that generalize across varying control sequences. Hallucination-Driven Policy Failure (Zhang et al., 2026) perturbs latent representations and exploits temporal dependencies in world-model-based agents to propagate adversarial effects across time and induce persistent policy errors. Nevertheless, these attacks are primarily designed for generative world models or policy-driven world-model agents, and their objectives cannot be directly applied to planner-based latent predictive world models. More recently, TRAP (Duan et al., 2026) explicitly considers the planning process and attacks the relative ranking of candidate trajectories. TRAP targets planner decisions by suppressing high-value trajectories through a tail-aware ranking objective. However, its optimization is tied to the candidate trajectories sampled in the current planning stage, whereas iterative planning may continually introduce new action sequences and updated representations. As a result, the attack may not persist across subsequent planning iterations, motivating perturbations that generalize across varying action candidates.

3 Methodology

3.1 Preliminaries

World model planning. We consider model-based control settings that use a learned world model to predict the consequences of candidate actions in a latent space. At time step t, the agent receives a visual observation $\mathbf{O}_t$, optionally accompanied by a proprioceptive state $\mathbf{p}_t$, and a goal observation $\mathbf{O}_g$. An observation encoder E_θ maps them into latent representations,

$$\mathbf{z}_t = E_\theta(\mathbf{O}_t, \mathbf{p}_t), \qquad \mathbf{z}_g = E_\theta(\mathbf{O}_g), \tag{1}$$

where $\mathbf{p}_t$ is omitted for models that operate only on visual observations. A sampling-based planner such as CEM optimizes an action sequence $\mathbf{A}_t$ through K_p iterative updates. At each iteration, it

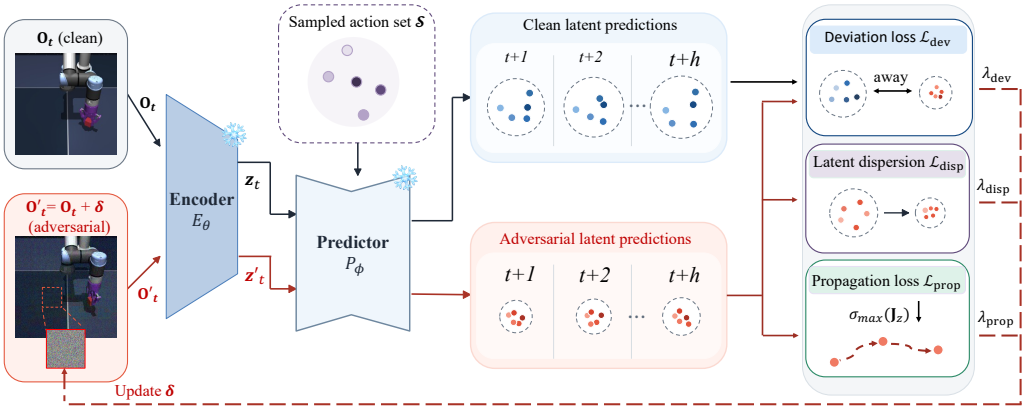


Figure 2: Overview of our AIDA framework. We perturb the first observation and obtain adversarial latent predictions under sampled action sequences. The perturbation is optimized with three complementary objectives: a deviation loss $\mathcal{L}_{\text{dev}}$ that pushes adversarial predictions away from clean dynamics, a latent-dispersion loss $\mathcal{L}_{\text{disp}}$ that reduces the variation among predictions induced by different actions, and a propagation loss $\mathcal{L}_{\text{prop}}$ that attenuates action-dependent discrepancies over multi-step rollouts.

samples N candidate sequences and rolls them out through the world model. The resulting terminal latent states against the goal state are evaluated by distance, and the obtained candidate costs are used to update the action distribution, which is used to sample a new set of candidates for the next iteration. This **sample–evaluate–update** cycle is repeated for K_p iterations. After optimization, only the first few action blocks are executed; if the goal is not reached, the planner repeats the procedure from the new observation under MPC.

Threat Model. The objective is to induce erroneous latent predictions that mislead the downstream planner into selecting suboptimal actions. The attack operates under a white-box setting, with access to the parameters and gradients of the world model. Perturbations $\boldsymbol{\delta}$ are applied only to the visual observation $\mathbf{O}_t$ under an ℓ_∞ constraint, leaving the goal observation, proprioceptive state, and actions unchanged. Our method does not require access to sampled candidate sequences or internal optimization process.

Method Overview. As demonstrated in Figure 2, the perturbed observation $\mathbf{O}'$ is encoded and passed to the predictor together with sampled action variations, producing a set of adversarial latent predictions. We optimize the perturbation in MI-FGSM (Dong et al., 2018) style with three objectives: reducing the dispersion among adversarial latent predictions, mitigating the propagation of action-dependent discrepancies, and increasing deviation from clean predictions. Although the predictor generally operates on a short observation context, we use a single input frame in the following formulation for simplicity.

3.2 Action Insensitivity

Sampling-based planners evaluate many candidate actions, and multiple action choices may lead to the same goal. Thus, attacks tailored to a fixed set of action sequences can be bypassed by alternative candidates. More fundamentally, effective planning relies on the world model producing distinct future predictions that reflect the consequences of different actions. We therefore seek to weaken this action dependence by reducing prediction variability across action candidates.

To reduce action sensitivity, we enforce similar latent predictions across multiple symmetrically perturbed action sets sampled around the ground-truth sequence. Specifically, given a ground-truth action sequence $\mathbf{A}_g \in \mathbb{R}^{h_a \times (dd_a)}$, where h_a denotes the horizon action blocks and each block contains d environment actions of dimension d_a, we set $\boldsymbol{\epsilon}_{e,1} = 0$, while the remaining perturbations are organized into antithetic Gaussian pairs: $\boldsymbol{\epsilon}_{e,2r} = \boldsymbol{\xi}_{e,r}$ and $\boldsymbol{\epsilon}_{e,2r+1} = -\boldsymbol{\xi}_{e,r}$, where $\boldsymbol{\xi}_{e,r} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. One action sequence in action set e will be:

$$\mathbf{A}_{e,q} = \mathrm{clip}\left(\mathbf{A}_g + b\boldsymbol{\epsilon}_{e,q},\, a_{\min}, a_{\max}\right), \tag{2}$$

where b controls the sampling scale. The antithetic construction provides more balanced coverage, reducing directional bias from finite sampling and encouraging the collapse objective to capture action variations more symmetrically. Together with $\mathbf{A}_g$, these actions form one sampled action set $\mathcal{S}$ with Q action sequences, among which we encourage the predicted latent states to pull close to each other. Since a single set captures only a limited subset of possible action variations, we sample N_{eot} independent sets $\{\mathcal{S}_e\}_{e=1}^{N_{eot}}$ and average the objective across them.

For each $\mathcal{S}_e$, the dispersion of its predicted latent states around their group-wise mean is measured. Given an adversarial observation $\mathbf{O}'_t = \mathbf{O}_t + \boldsymbol{\delta}$, predictions by P_ϕ corresponding to each action sequence $\hat{\mathbf{z}}'_{e,q}$ are first obtained, then the prediction center within each set $\bar{\mathbf{z}}'_e$ can be computed:

$$\hat{\mathbf{z}}'_{e,q} = P_\phi(E_\theta(\mathbf{O}'_t), \mathbf{A}_{e,q}), \qquad \bar{\mathbf{z}}'_e = \frac{1}{Q}\sum_{\mathbf{A}_{e,q}\in\mathcal{S}_e} \hat{\mathbf{z}}'_{e,q}. \tag{3}$$

The group-wise latent dispersion objective is defined as

$$\mathcal{L}_{\mathrm{disp}} = \frac{1}{N_{\mathrm{eot}}}\sum_{e=1}^{N_{\mathrm{eot}}}\left(\sum_{\mathbf{A}_{e,q}\in\mathcal{S}_e}\left\|\mathrm{vec}\left(\hat{\mathbf{z}}'_{e,q} - \bar{\mathbf{z}}'_e\right)\right\|_2^2\right)^{\frac{1}{2}}. \tag{4}$$

However, reducing latent dispersion alone may leave the collapsed predictions close to their clean counterparts. Therefore, perturbed predictions are encouraged to deviate from clean predictions. For each action sequence $\mathbf{A}_{e,q}$, the predictions $\hat{\mathbf{z}}'_{e,q}$ obtained from the $\mathbf{O}'_t$ are pushed away from those $\hat{\mathbf{z}}_{e,q}$ obtained from the $\mathbf{O}_t$:

$$\mathcal{L}_{\mathrm{dev}} = -\frac{1}{N_{\mathrm{eot}}}\sum_{e=1}^{N_{\mathrm{eot}}}\left(\sum_{q=1}^{Q}\left\|\mathrm{vec}\big(\hat{\mathbf{z}}'_{e,q} - \hat{\mathbf{z}}_{e,q}\big)\right\|_2^2\right)^{\frac{1}{2}}. \tag{5}$$

The latent dispersion loss $\mathcal{L}_{\mathrm{disp}}$ compacts adversarial predictions induced by different actions, while the deviation loss $\mathcal{L}_{\mathrm{dev}}$ pushes the resulting prediction cluster away from the clean dynamics. Together, they reduce action-induced variation in adversarial predictions while preventing collapsed predictions from remaining within the benign latent neighborhood.

3.3 Multi-Step Sensitivity Attenuation

While the preceding objective directly reduces the discrepancy among predictions induced by different actions, it does not explicitly account for how such discrepancies propagate over a multi-step rollout. This is particularly relevant in our setting, where the perturbation is applied only to the initial observation, while its effect needs to persist through latent transitions. To characterize this, we analyze how action-induced latent discrepancies evolve over a finite prediction horizon.

If given a latent state $\mathbf{z}_t$ and a next-step environment action $\mathbf{a}_t$, the next latent state is predicted as $\hat{\mathbf{z}}_{t+1} = P_\phi(\mathbf{z}_t, \mathbf{a}_t)$. Consider two candidate action sequences starting from the same initial latent context. Using a local first-order approximation of the predictor, the discrepancy between two latent rollouts induced by different action can be locally bounded as

$$\left\|\hat{\mathbf{z}}^{(1)}_{t+1} - \hat{\mathbf{z}}^{(2)}_{t+1}\right\|_2 \lesssim \|\mathbf{J}_{z,t}\|_2 \left\|\mathbf{z}^{(1)}_t - \mathbf{z}^{(2)}_t\right\|_2 + \|\mathbf{J}_{a,t}\|_2 \left\|\mathbf{a}^{(1)}_t - \mathbf{a}^{(2)}_t\right\|_2, \tag{6}$$

where

$$\mathbf{J}_{z,t} = \frac{\partial P_\phi(\mathbf{z}_t, \mathbf{a}_t)}{\partial \mathbf{z}_t}, \qquad \mathbf{J}_{a,t} = \frac{\partial P_\phi(\mathbf{z}_t, \mathbf{a}_t)}{\partial \mathbf{a}_t}, \tag{7}$$

characterizing the local sensitivities of the transition to the latent context and action input, respectively. Since both rollouts start from the same latent ($\left\|\mathbf{z}^{(1)}_t - \mathbf{z}^{(2)}_t\right\|_2 = 0$), recursively applying this relation from the initial state $\mathbf{z}_t$ to an arbitrary rollout step $t+h$ gives

$$\left\|\mathbf{z}^{(1)}_{t+h} - \mathbf{z}^{(2)}_{t+h}\right\|_2 \lesssim \sum_{i=0}^{h-1}\left(\prod_{j=i+1}^{h-1}\rho_{z,t+j}\right)\rho_{a,t+i}\left\|\mathbf{a}^{(1)}_{t+i} - \mathbf{a}^{(2)}_{t+i}\right\|_2, \tag{8}$$

where

$$\rho_{z,t} = \|\mathbf{J}_{z,t}\|_2 = \sigma_{max}(\mathbf{J}_{z,t}), \qquad \rho_{a,t} = \|\mathbf{J}_{a,t}\|_2 = \sigma_{max}(\mathbf{J}_{a,t}), \tag{9}$$

as the spectral norm of a matrix equals its largest singular value $\sigma_{max}(\cdot)$.

The resulting bound shows that action-induced discrepancies are propagated through successive latent transitions, with the spectral norm of the state-transition Jacobian ρ_z acting as a dominant multiplicative factor. This motivates additionally suppressing the local transition sensitivity under perturbed rollout, so that discrepancies induced by different actions grow more slowly over future prediction steps. Since the predictor cannot be modified, we aim to steer the perturbed latent rollout toward locally less sensitive regions with smaller ρ_z.

Let the spectral norm of the latent Jacobian of one transition from sampled actions be denoted as $\rho_z^{e,q} = \sigma_{max}(\mathbf{J}_z^{e,q})$. Explicitly constructing $\rho_z^{e,q}$ is prohibitively expensive because of the high dimensionality of the latent representation. Hence, we estimate its dominant singular value using power iteration without materializing the full Jacobian. Starting from a normalized vector $\mathbf{v}_0$, each iteration alternates between Jacobian-vector and vector-Jacobian products as:

$$\mathbf{u}_k = \frac{\mathbf{J}_z^{e,q}\mathbf{v}_k}{\|\mathbf{J}_z^{e,q}\mathbf{v}_k\|_2}, \qquad \mathbf{v}_{k+1} = \frac{\left(\mathbf{J}_z^{e,q}\right)^\top \mathbf{u}_k}{\left\|\left(\mathbf{J}_z^{e,q}\right)^\top \mathbf{u}_k\right\|_2}. \tag{10}$$

After K iterations, the largest singular value can be approximated by:

$$\widehat{\rho}_z^{e,q} = \|\mathbf{J}_z^{e,q}\mathbf{v}_K\|_2 . \tag{11}$$

Then, a latent propagation loss with a threshold is used to suppress the amplification along the adversarial rollout:

$$\mathcal{L}_{\text{prop}} = \frac{1}{N_{eot}Q}\sum_{e=1}^{N_{eot}}\sum_{q=1}^{Q}\left[\max\left(\widehat{\rho}_z^{e,q} - \tau, 0\right)\right]^2 . \tag{12}$$

Here, $\tau < 1$ serves as a reference sensitivity margin that provides optimization pressure toward lower local sensitivity, rather than assuming that all transitions can be driven into a contractive regime. In practice, though $\widehat{\rho}_z$ may remain above one, reducing its magnitude can still help attenuate the multiplicative propagation term in Eq. 8.

3.4 Overall Objective and Perturbation Optimization

Building on the three objectives introduced above, our attack jointly reduces latent dispersion across sampled actions, drives perturbed predictions away from their clean counterparts, and attenuates the propagation of action-dependent discrepancies. The overall attack loss is obtained by combining these three losses with weighting coefficients:

$$\mathcal{L}_{\text{all}} = \lambda_{\text{disp}}\mathcal{L}_{\text{disp}} + \lambda_{\text{dev}}\mathcal{L}_{\text{dev}} + \lambda_{\text{prop}}\mathcal{L}_{\text{prop}}. \tag{13}$$

The perturbation is updated following the momentum iterative fast gradient sign method (MI-FGSM) (Dong et al., 2018) with step size α:

$$m_i = \mu m_{i-1} + \frac{\nabla_{\boldsymbol{\delta}}\mathcal{L}_{\text{all}}}{\text{mean}(|\nabla_{\boldsymbol{\delta}}\mathcal{L}_{\text{all}}|)}, \qquad \boldsymbol{\delta}^{i+1} = \text{Clip}_{\mathcal{B}_\epsilon}\{\boldsymbol{\delta}^i - \alpha \cdot \text{sign}\,(m_i)\}, \tag{14}$$

where ∇ denotes the gradient operator and $\text{sign}(\cdot)$ returns the element-wise sign of the gradient. $\text{Clip}_{\mathcal{B}_\varepsilon}$ restricts $\boldsymbol{\delta}$ inside the boundary of the ℓ_p-norm ball of radius ε.

4 Experiments

4.1 Experimental Setup

Evaluated models and environments. We evaluate our attack on three representative latent predictive world models, including LeWM (Maes et al., 2026), DINO-WM (Zhou et al., 2024) and PLDM (Sobal et al., 2026), on two robotic manipulation benchmarks, PushT (Chi et al., 2025) and OGBench-Cube (Park et al., 2025). For DINO-WM, we use the standard variant on PushT and

Table 1: Attack performance across three latent predictive world models on OGBench-Cube and PushT. We report success rate (SR) and attack success rate (ASR) on both datasets. For OGBench-Cube, FGD denotes the mean final goal distance (cm); for PushT, FPE and FAE denote the mean final position error and final angle error (°), respectively. DINO-WM refers to DINO-WM on PushT and DINO-WM-noprop on OGBench-Cube.

World Model	Method	OGBench-Cube			PushT			
		SR ↓	ASR ↑	FGD ↑	SR ↓	ASR ↑	FPE ↑	FAE ↑
LeWM	Clean	79.00	–	5.42	84.00	–	27.68	6.54
	TRAP	76.00	15.52	6.08	78.60	13.35	29.03	6.17
	UADA/TMA	38.00	58.02	10.21	13.20	85.71	**280.80**	23.45
	Ours	**24.20**	**73.28**	**14.82**	**8.40**	**91.57**	257.37	**24.40**
DINO-WM	Clean	80.00	–	4.93	41.40	–	40.25	14.39
	TRAP	79.60	13.25	4.95	18.60	67.15	52.42	20.64
	UADA/TMA	55.60	40.25	7.73	20.60	69.08	57.52	24.98
	Ours	**29.60**	**68.25**	**11.47**	**14.40**	**75.85**	**81.08**	**31.41**
PLDM	Clean	45.00	–	9.17	71.60	–	110.95	11.85
	TRAP	46.40	28.00	9.08	68.40	10.89	99.50	11.66
	UADA/TMA	26.60	62.67	11.70	13.60	82.96	269.00	25.94
	Ours	**23.20**	**65.78**	**13.26**	**8.80**	**89.39**	**319.65**	**25.97**

the no-proprioception variant (DINO-WM-noprop) on OGBench-Cube due to the availability of released pretrained checkpoints. For both benchmarks, we construct 500 valid start–goal pairs from pre-collected expert trajectories after excluding cases that already satisfy the success criterion at initialization. For OGBench-Cube, we further retain only pairs successfully solved by the clean LeWM planner, resulting in a higher clean success rate and providing a clearer basis for comparing different attack methods. No such clean-success filtering is applied to PushT.

Compared methods. TRAP (Duan et al., 2026) manipulates the relative ranking of candidate trajectories to mislead world-model planning. We adapt its original patch-based formulation to an ℓ_∞-bounded full-image perturbation while retaining its planner-dependent trajectory-ranking objective. UADA and TMA (Wang et al., 2025) from the attack on vision-language-action models are also considered. We construct adversarial action sequences by replacing expert actions with maximally deviated valid actions while keeping the original goal latent fixed, and optimize the input perturbation so that these adversarial actions are predicted to reach the goal.

Implementation details. We follow the temporal sampling used during world-model training. During attack, each trajectory segment contains 4 observation frames sampled at a frame skip of 5 and the corresponding three action blocks, each consisting of 5 environment actions. The first observation is perturbed, while remaining three observations serve as clean context for teacher-forced prediction. The corresponding action sequence read from the dataset is used as the ground-truth sequence $\mathbf{A}_g$. During attack optimization, subsequent observations and actions remain unchanged. The perturbation budget is $\varepsilon = 16$, and $\boldsymbol{\delta}$ is updated for 20 iterations with $\mu = 0.5, \alpha = 1$. For action ensemble, $N_{eot} = 2, Q = 5, b = 0.55$, and the ensembles are resampled every five attack iterations. Power iteration $K = 6$, and $\tau = 0.9$ in Eq. 12. loss weights $(\lambda_{\text{disp}}, \lambda_{\text{dev}}, \lambda_{\text{prop}})$ are $(1, 2, 1)$ for OGBench-Cube and $(1, 1, 1)$ for PushT. Unless otherwise specified, we use the original planning configuration released.

Evaluation metrics. For both benchmarks, we report success rate (SR) and attack success rate (ASR). ASR is computed as the percentage of clean-successful trials that are turned into failures by the attack. Additionally, terminal-state errors are measured to capture detailed performance degradation. For OGBench-Cube, we use the mean final goal distance (FGD) in centimeters. For PushT, we report the mean final position error (FPE) and mean final angle error (FAE) in degrees.

4.2 Main results

Comparison in OGBench-Cube task. OGBench-Cube evaluates 3D robotic manipulation, requiring the agent to grasp and move a cube toward a specified target position. As shown in Table 1, our

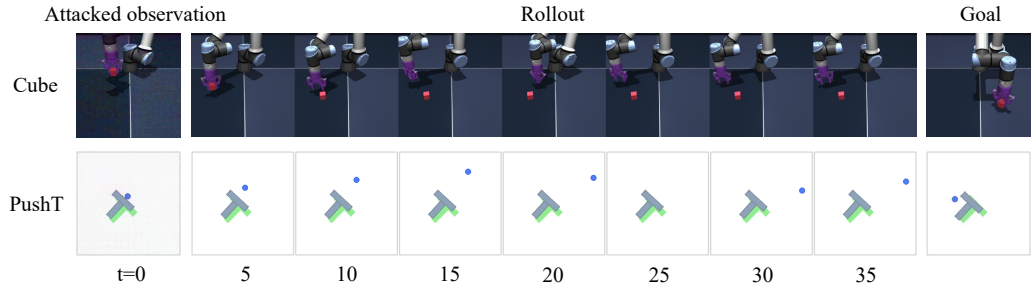


Figure 3: Visualization of rollout on OGBench-Cube and PushT starting from observations attacked by our method. The latent predictions from LeWM with CEM planner are decoded.

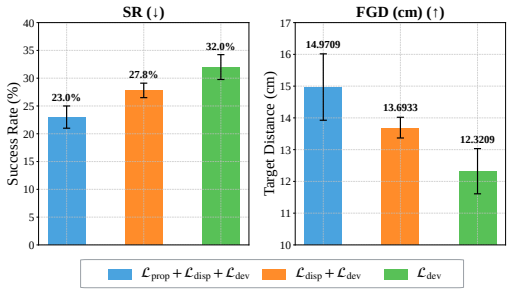


Figure 4: Ablation study of the proposed loss components.

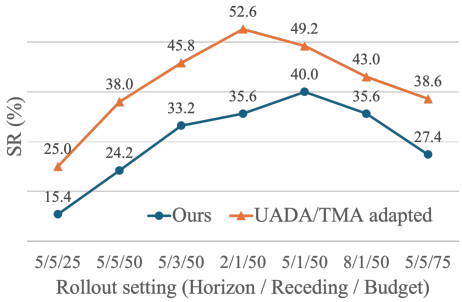


Figure 5: Success rate under different rollout configurations.

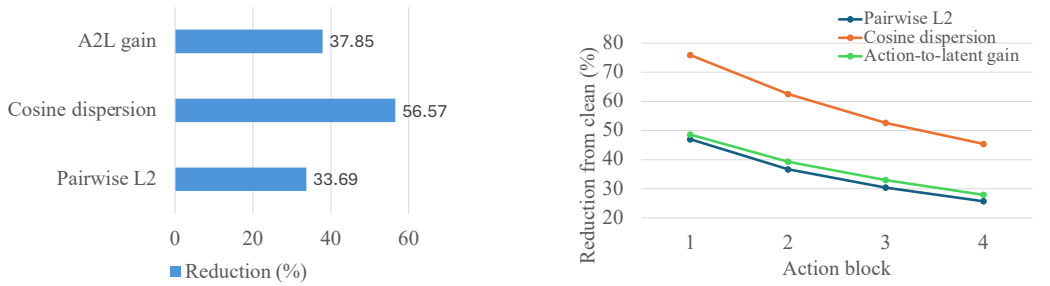
attack consistently causes the strongest degradation on three world models. Specifically, on LeWM, compared to the strongest baseline UADA/TMA, our method achieves a 13.8 percentage point lower SR, a 15.26 point higher ASR, and a 4.61 higher FGD. DINO-WM shows a similar trend, where our method achieves the lowest SR, outperforming the second-best UADA/TMA by 26.0 percentage points. On PLDM, ours similarly obtains the lowest SR (23.2%), highest ASR (65.78%), and largest FGD (13.26). TRAP shows only limited degradation across three world models, likely due to its dependence on specific candidate trajectories.

Comparison in PushT task. Complementing the 3D robotic setting of OGBench-Cube, PushT considers planar contact-rich manipulation requiring coordinated control of object position and orientation. Our method remains consistently effective under this control setting. For example, on LeWM, it achieves the lowest SR (8.4%) and highest ASR (91.57%), improving over UADA/TMA by 4.8 and 5.86 percentage points. Although UADA/TMA attains a slightly larger FPE, both methods already induce substantial positional deviation. TRAP achieves markedly higher attack effectiveness on DINO-WM with PushT than in the other settings, possibly suggesting that its trajectory-specific objective is more compatible with this particular setting.

Visualization. We perturb the initial observation and set the goal to the dataset state 25 environment steps (5 action blocks) ahead, using LeWM with CEM for rollout on OGBench-Cube and PushT. As shown in Figure 3, starting from the adversarial initial observations, the rollouts exhibit repeated or locally oscillatory motions, and the planner fails to make consistent progress toward the goal even on simple cases.

4.3 Ablation Studies and Additional Experiments

Ablation studies. We conduct the ablation study on a subset of OGBench-Cube with LeWM, reporting the mean and standard deviation over five random seeds. The results in Figure 4 show that each objective provides a complementary contribution to the attack. Introducing $\mathcal{L}_{\text{disp}}$ decreases the SR by 4.2 percentage points and increases the FGD by 1.37 cm, demonstrating the benefit of re-



(a) Overall reduction of action sensitivity after our attack.

(b) Block-wise reduction along the latent rollout.

Figure 6: Analysis of action sensitivity reduction.

ducing the dispersion of action-dependent prediction. Adding $\mathcal{L}_{\text{prop}}$ brings another 4.8 percentage points reduction in SR and a 1.28 cm increase in FGD, displaying the effectiveness of attenuating the propagation of action-dependent discrepancies. Although variation exists across random seeds, the error bars show that the overall improvement remains consistent across the ablation stages. The full configuration with all components yields the best results.

Robustness across different rollout configurations. We further evaluate the attack robustness when varying the planning horizon, receding horizon, and evaluation budget. The planning horizon sets the number of future action blocks considered in each planning round, while the receding horizon controls replanning frequency by specifying how many action blocks are executed before each replanning. The evaluation budget limits the maximum number of environment steps. It can be observed in Figure 5 that attack effectiveness decreases moderately with more frequent replanning or larger evaluation budgets, as both provide the planner with more opportunities to recover from the initially perturbed observation. Nevertheless, our attack consistently achieves lower SR and remains relatively stable.

Feature dispersion analysis. We further examine whether our attack indeed reduces the latent dispersion induced by diverse action sequences. For each clean observation and adversarial one resulted from our method, we randomly sample five action sequences and roll them out for four action blocks. As shown in Figure 6a, our attack consistently reduces all three dispersion-related metrics, with cosine dispersion decreasing by 56.57%. Pairwise L2 measures the average Euclidean separation among latent predictions, while cosine dispersion further captures their directional discrepancy, and action-to-latent gain (A2L gain) quantifies the latent variation induced per unit action variation. This indicates that predictions become substantially more concentrated after our attack. Figure 6b shows that the reduction gradually decreases with rollout horizon but remains evident at the fourth block, indicating persistent effects over multi-step latent prediction.

Additional experiments. Section A provides additional implementation details for both our method and the adapted baselines, together with detailed hyperparameter studies of our method. We further evaluate the performance of adversarial images on a different planner, analyze the effect of varying perturbation budgets ε, and report preliminary results under simple defense strategies. Additional theoretical derivations are also included to complement the main text. We also provide more visualizations of our attack and baselines.

5 Conclusion

In this work, we investigate the robustness of latent predictive world models from the perspective of action-conditioned latent dynamics. Without relying on any planner, our attack perturbs only the initial observation to induce distorted latent futures that remain less discriminative across varying actions and persist over multi-step prediction. Experiments on multiple world models and manipulation benchmarks show that such perturbations can consistently degrade downstream planning across different planning settings. These results reveal the action dependence of latent dynamics as a critical vulnerability in model-based control and suggest that robust world models should preserve reliable action-conditioned structure under adversarial observations.

AI USE STATEMENT

In this work, we did not use generative AI tools for tasks requiring mandatory disclosure. We have not used generative AI tools to develop theoretical models, formulate mathematical claims, design experiments, or interpret results, and the remaining mandatory-disclosure tasks are not applicable to this work. Additionally, we used generative AI tools to create or edit software code and to proofread the manuscript for grammar, clarity, and readability. We have reviewed all AI-assisted work. The authors reviewed and tested AI-assisted code for correctness and verified that the suggested revisions preserved the technical meaning and accuracy of the manuscript. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies adversarial vulnerabilities in latent predictive world models and world-model-based planning. By revealing such vulnerabilities, our findings aim to raise awareness and encourage the development of their robustness and security. All experiments are conducted in simulation environments using publicly available models and benchmarks.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the proposed method, optimization objectives in section 3, with implementation details in section 4.1. Additional implementation details, hyperparameter settings, baseline configurations, and theoretical derivations included are in the appendix. Code will be released to facilitate reproduction of the reported results.

REFERENCES

Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629. IEEE, 2023.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, et al. V-jepa 2: Self-supervised video models enable understanding, prediction and planning. *arXiv preprint arXiv:2506.09985*, 2025.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first international conference on machine learning*, 2024.

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. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Pieter-Tjerk De Boer, Dirk P Kroese, Shie Mannor, and Reuven Y Rubinstein. A tutorial on the cross-entropy method. *Annals of operations research*, 134(1):19–67, 2005.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9185–9193. IEEE, 2018.

Siyuan Duan, Ke Zhang, and Xizhao Luo. Trap: Tail-aware ranking attack for world-model planning. *arXiv preprint arXiv:2605.01950*, 2026.

Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.

Zhixiang Guo, Siyuan Liang, Andras Balogh, Noah Lunberry, Rong-Cheng Tu, Mark Jelasity, and Dacheng Tao. When world models dream wrong: Physical-conditioned adversarial attacks against world models. *arXiv preprint arXiv:2602.18739*, 2026a.

Zhixiang Guo, Siyuan Liang, Shi Fu, Cheng Guo, Andras Balogh, Mark Jelasity, and Dacheng Tao. Wmattack: Automated attack search for adversarial evaluation of world-model agents. *arXiv preprint arXiv:2605.23220*, 2026b.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2(3):440, 2018.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019a.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019b.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. *arXiv preprint arXiv:2203.04955*, 2022.

Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. *arXiv preprint arXiv:1706.06083*, 2017.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworld-model: Stable end-to-end joint-embedding predictive architecture from pixels. *arXiv preprint arXiv:2603.19312*, 2026.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International Conference on Learning Representations*, volume 2025, pp. 94937–94982, 2025.

Reuven Y Rubinstein and Dirk P Kroese. *The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation, and machine learning*, volume 133. Springer, 2004.

Linghui Shen, Mingyue Cui, and Xingyi Yang. Badworld: Adversarial attacks on world models. *arXiv preprint arXiv:2606.16519*, 2026.

Uladzislau Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, Tim GJ Rudner, and Yann LeCun. Learning from reward-free offline data: A case for planning with latent dynamics models. *Advances in Neural Information Processing Systems*, 38:43905–43941, 2026.

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

Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. Drivedreamer: Towards real-world-drive world models for autonomous driving. In *European conference on computer vision*, pp. 55–72. Springer, 2024.

Grady Williams, Andrew Aldrich, and Evangelos A Theodorou. Model predictive path integral control: From theory to parallel computation. *Journal of Guidance, Control, and Dynamics*, 40 (2):344–357, 2017.

Junjian Zhang, Hao Tan, Ruonan Li, Aiping Li, and Zhaoquan Gu. Adversarial attacks against world models: Hallucination-driven policy failure. *Applied Sciences*, 16(11):5484, 2026.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. Dino-wm: World models on pre-trained visual features enable zero-shot planning. *arXiv preprint arXiv:2411.04983*, 2024.

A APPENDIX

A.1 IMPLEMENTATION DETAILS

Table 2: Hyperparameter of the adapted TRAP.

parameters	eps	steps	tail-frac	beta	λ_{tail}	λ_{sign}	λ_{mag}	λ_{tv}	λ_{L2}
value	16.0	80	0.4	0.5	1.0	1.0	0.5	1×10^{-3}	1×10^{-5}

Table 3: Hyperparameter of the adapted UADA/TMA.

parameters	eps	alpha	steps	decay	random-start
value	16	1	20	0.5	True

All experiments are conducted on NVIDIA RTX A6000 GPUs. We use a single GPU for adversarial example generation and two GPUs for downstream planning evaluation.

Unless otherwise specified, we follow the official evaluation configurations of the corresponding world models and planners. For planning, we use 300 candidate action sequences with a planning horizon of 5 action blocks. The receding horizon is set to 5, and the maximum evaluation budget is 50 environment steps. For each evaluation episode, the goal observation is selected 25 environment steps after the initial observation.

For the adapted baselines, we retain their original objectives while adjusting some attack details to our latent predictive world-model setting. We perform lightweight hyperparameter tuning while following the recommended settings from the original papers whenever applicable. The resulting hyperparameters for TRAP are summarized in Table 2. We similarly adapt UADA and TMA to our setting, with their attack configurations reported in Table 3.

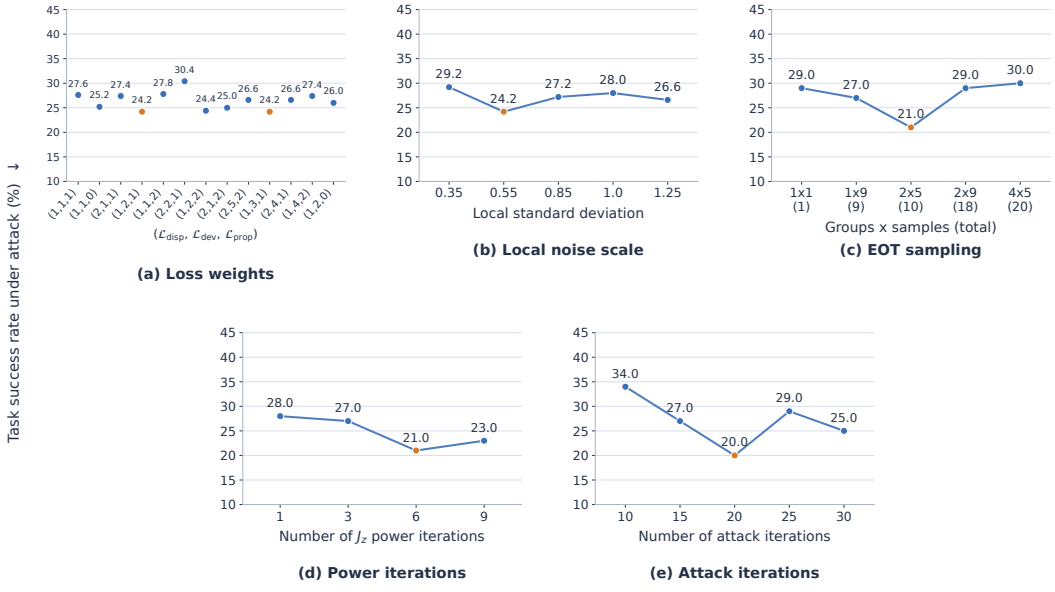


Figure 7: Hyperparameter study of our attack on OGBench-Cube. We sweep different configurations to attack LeWM on n samples of OGBench-Cube. Orange marks the lowest observed task success rates in each sweep. $n = 500$ in (a-b), while $n = 100$ in (c-e).

Hyperparameter study of our method. We study the sensitivity of our attack to several key hyperparameters. Figure 7 (a) and (b) are evaluated on the full 500-sample set, while Figure 7 (c)-(e) use a 100-sample subset for efficient tuning. For the loss weights, the attack remains relatively stable across different combinations, with $(\mathcal{L}_{disp}, \mathcal{L}_{dev}, \mathcal{L}_{prop}) = (1, 2, 1)$ achieving the lowest success

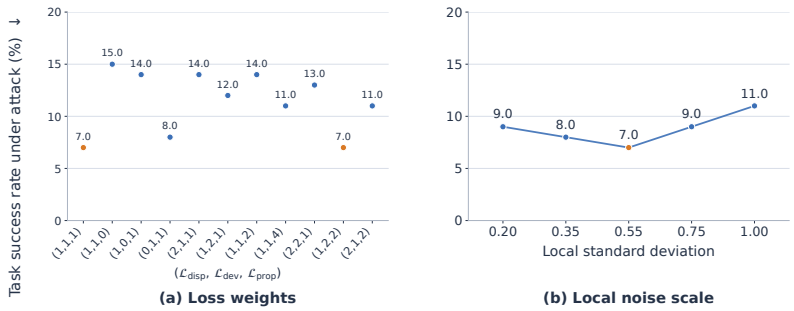


Figure 8: Hyperparameter study of our attack on PushT. We tune the loss weight and the local standard deviation of action sampling on 500 samples of PushT. Orange marks the lowest observed task success rates in each sweep.

Table 4: Attack performance of LeWM with MPPI as planner on OGBench-Cube and PushT. We report success rate (SR) on both datasets. For OGBench-Cube, FGD denotes the mean final goal distance (cm); for PushT, FPE and FAE denote the mean final position error and final angle error (°), respectively.

World Model	Method	OGBench-Cube		PushT		
		SR ↓	FGD ↑	SR ↓	FPE ↑	FAE ↑
LeWM	UADA/TMA	20.40	14.10	9.40	**284.23**	**24.60**
	Ours	**17.80**	**15.71**	**6.60**	215.35	23.94

rate of 24.2%. Varying the action sampling scale shows a similar trend. For EOT sampling, using two groups with five samples per group substantially improves the attack over both fewer samples and larger sampling sets, suggesting that organizing action samples into multiple independent EOT groups is more effective than simply increasing the number of samples within a single group. Further increasing the total number of samples does not yield additional gains. Increasing the number of power iterations improves the estimation of the dominant singular value up to 6 iterations. Finally, the attack is most effective with 20 optimization iterations, as using either fewer or more iterations results in weaker performance. We also slightly tune the loss weights and the local standard deviation during action sampling. According to Figure 8, weights on PushT are set to $(1, 1, 1)$. The hyperparameter study on PushT and main evaluation use different parallel evaluation setups, leading to different random execution orders and minor numerical variations, while the relative trends remain consistent.

A.2 Transferability across Planners

Our main experiments use CEM as the downstream planner. To examine whether the attack effectiveness depends on a specific planning algorithm, we further evaluate the adversarial observations using MPPI. We compare our method with UADA and TMA under the same MPPI evaluation protocol. As shown in Table 4, both attacks remain effective under MPPI, while our method consistently achieves lower SR (17.8% and 6.6%). The attacks are generally stronger under MPPI than under CEM. A possible explanation is that MPPI updates the nominal action sequence using cost-weighted averaging over all sampled trajectories, making its update more directly affected when adversarial perturbations broadly distort the predicted latent outcomes and reduce action discriminability.

A.3 Attack Performance across Different Perturbation Budgets

We additionally evaluate different perturbation budgets on a subset of OGBench-Cube, and the results are reported in Table 5. Across different perturbation budgets, our method consistently achieves lower SR and higher FGD than the strong baseline UADA/TMA. The strongest attack is observed at $\varepsilon = 16$. Further increasing ε does not bring additional gains, suggesting that attack effectiveness is not strictly monotonic with the perturbation budget.

Table 5: Attack performance under different perturbation budgets ε.

ε	Method	SR↓	FGD↑
8	UADA/TMA	48.0%	9.49
	Ours	**31.0%**	**13.97**
12	UADA/TMA	**35.0%**	10.60
	Ours	**35.0%**	**14.51**
16	UADA/TMA	38.0%	9.25
	Ours	**22.0%**	**16.27**
20	UADA/TMA	37.0%	10.43
	Ours	**26.0%**	**15.69**
24	UADA/TMA	42.0%	9.82
	Ours	**26.0%**	**14.82**

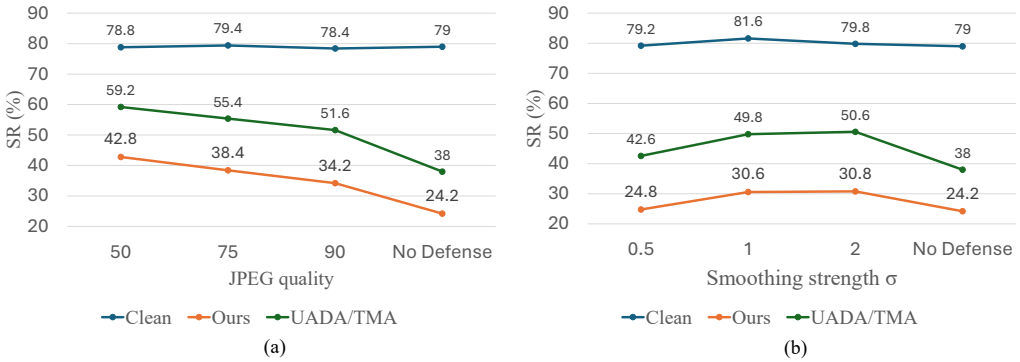


Figure 9: Attack performances under preprocessing defenses.

A.4 Performance against Defenses

We further evaluate robustness against two simple input preprocessing defenses, JPEG compression and Gaussian smoothing (3×3 kernel), using 500 adversarial images generated by our method and UADA/TMA. As shown in Figure 9, both defenses partially recover the success rate (SR). However, our method remains consistently more effective. Under JPEG compression, its SR ranges from 34.2% to 42.8%, compared with 51.6%–59.2% for UADA/TMA. Under Gaussian smoothing, our SR remains below 31%, while UADA/TMA reaches up to 50.6%. These results suggest that our perturbations are less affected by these preprocessing defenses.

A.5 Detailed Derivation of Finite-Horizon Latent Sensitivity

We provide additional details for the multi-step sensitivity analysis in Section 3.3. Consider two latent rollouts induced by different action sequences, and define their latent and action discrepancies at rollout step k as

$$\Delta \mathbf{z}_{t+k} = \mathbf{z}^{(1)}_{t+k} - \mathbf{z}^{(2)}_{t+k}, \qquad \Delta \mathbf{a}_{t+k} = \mathbf{a}^{(1)}_{t+k} - \mathbf{a}^{(2)}_{t+k}. \tag{15}$$

For the action-conditioned predictor $\mathbf{z}_{t+k+1} = P_\phi(\mathbf{z}_{t+k}, \mathbf{a}_{t+k})$, a local first-order approximation gives

$$\Delta \mathbf{z}_{t+k+1} \approx \mathbf{J}_{z,t+k} \Delta \mathbf{z}_{t+k} + \mathbf{J}_{a,t+k} \Delta \mathbf{a}_{t+k}, \tag{16}$$

where

$$\mathbf{J}_{z,t+k} = \frac{\partial P_\phi}{\partial \mathbf{z}_{t+k}}, \qquad \mathbf{J}_{a,t+k} = \frac{\partial P_\phi}{\partial \mathbf{a}_{t+k}}. \tag{17}$$

More precisely, the first-order expansion includes a higher-order residual term, which is omitted under the local first-order approximation. Taking the L_2 norm and applying the triangle inequality together with $\|\mathbf{A}\mathbf{x}\|_2 \le \|\mathbf{A}\|_2 \|\mathbf{x}\|_2$, we obtain

$$\|\Delta \mathbf{z}_{t+k+1}\|_2 \lesssim \rho_{z,t+k}\|\Delta \mathbf{z}_{t+k}\|_2 + \rho_{a,t+k}\|\Delta \mathbf{a}_{t+k}\|_2, \tag{18}$$

where

$$\rho_{z,t+k} = \|\mathbf{J}_{z,t+k}\|_2, \qquad \rho_{a,t+k} = \|\mathbf{J}_{a,t+k}\|_2.$$

Since the two rollouts start from the same initial latent state, $\Delta \mathbf{z}_t = 0$. The first rollout step therefore satisfies

$$\|\Delta \mathbf{z}_{t+1}\|_2 \lesssim \rho_{a,t}\|\Delta \mathbf{a}_t\|_2. \tag{19}$$

For the second step,

$$\|\Delta \mathbf{z}_{t+2}\|_2 \lesssim \rho_{z,t+1}\rho_{a,t}\|\Delta \mathbf{a}_t\|_2 + \rho_{a,t+1}\|\Delta \mathbf{a}_{t+1}\|_2. \tag{20}$$

Recursively applying the same relation gives, for an arbitrary rollout step $t + h$,

$$\|\Delta \mathbf{z}_{t+h}\|_2 \lesssim \sum_{i=0}^{h-1} \left(\prod_{j=i+1}^{h-1} \rho_{z,t+j} \right) \rho_{a,t+i}\|\Delta \mathbf{a}_{t+i}\|_2. \tag{21}$$

The factor $\rho_{a,t+i}\|\Delta \mathbf{a}_{t+i}\|_2$ characterizes the discrepancy directly introduced by different actions, whereas $\prod_{j=i+1}^{h-1} \rho_{z,t+j}$ determines how this discrepancy is propagated through the latent dynamics. Thus, ρ_z repeatedly modulates the discrepancy over later prediction steps. Since our $\mathcal{L}_{\text{disp}}$ attends to the local action sensitivity already, we further regularize the state-transition sensitivity ρ_z. Just as Eq. 9, directly computing it would require explicitly constructing a high-dimensional Jacobian. Hence, we estimate its dominant singular value using power iteration.

Starting from a normalized vector $\mathbf{v}_0$, we iteratively compute Eq. 10. The alternating JVP/VJP updates converge toward the dominant singular directions of $\mathbf{J}_z$, allowing its spectral norm to be estimated without explicitly constructing the Jacobian.

A.6 More Visualizations

We provide additional qualitative visualizations for all three attack methods on DINO-WM with OGBench-Cube and on PLDM with PushT. During each rollout, we visualize one frame every five environment steps, corresponding to one action block. Due to space limitations, we show only the first 35 steps (7 action blocks), and terminate the visualization early if the goal is reached.

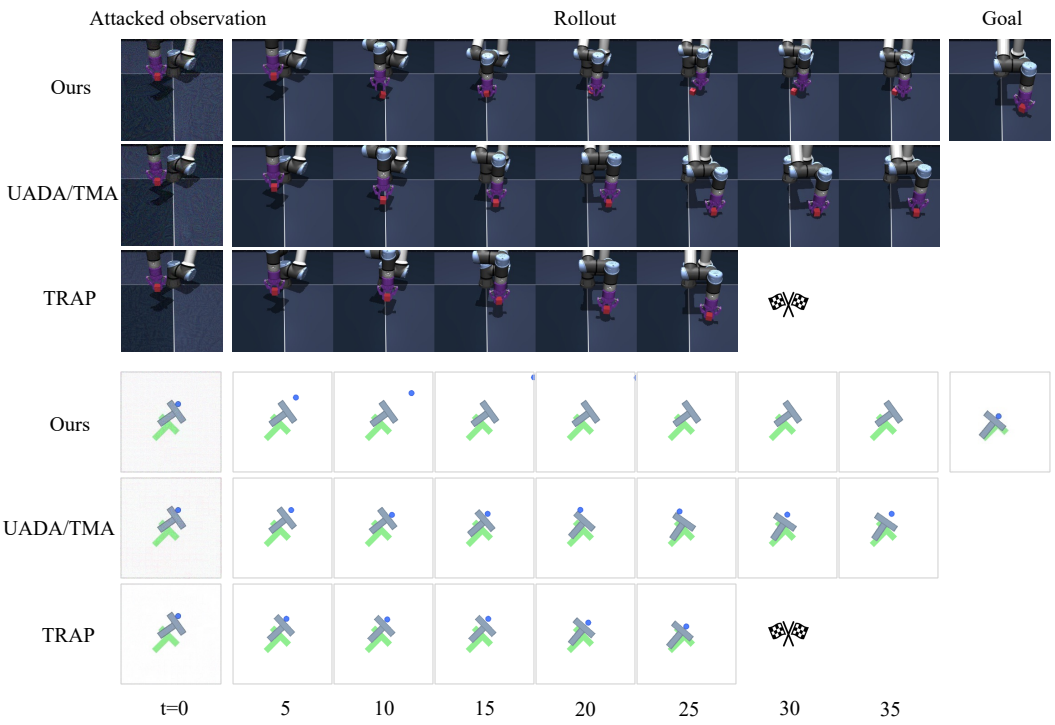


Figure 10: Qualitative visualizations for three attack methods.