VSAD: VALUATION-SENSITIVITY ATTACKS AND ROBUST DEFENSE FOR LATENT WORLD-MODEL PLANNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Model-based reinforcement learning (MBRL) achieves high sample efficiency by planning in the latent space of a learned world model. However, planners select actions by ranking candidate trajectories according to their predicted returns, making them vulnerable to local valuation errors.We show that these errors can be amplified by the planning process and identify valuation sensitivity as a characteristic signature of fragile high-value trajectories. Specifically, trajectories that receive spuriously high predicted returns exhibit substantial changes in valuation under small perturbations of the input observation, which correspondingly alter the initial latent state. They are also frequently associated with abnormal control patterns, such as action jitter and excessive impact, whereas successful trajectories exhibit more stable valuations. We further demonstrate that this sensitivity can be exploited at test time. In the white-box setting, bounded latent-state perturbations manipulate trajectory rankings and promote unreliable action sequences into the planner's elite set. In the black-box setting, NES-based observation perturbations directly alter the target planner's trajectory selection and induce similar control degradation. Based on this finding, we propose a training-free, model-agnostic defense that re-evaluates each candidate trajectory under multiple perturbed observations and ranks trajectories using a conservative score combining mean predicted return, return variance, and action smoothness. The defense suppresses trajectories whose high scores depend on unstable value estimates, without modifying the world model or requiring additional training data. Experiments show that it substantially reduces the effectiveness of both white-box and black-box attacks while preserving normal performance in clean environments. These results establish valuation sensitivity as a mechanistic signature of fragile trajectories, an exploitable attack surface, and a practical signal for robust planning. Code is available here.

1 INTRODUCTION

Model-based reinforcement learning (MBRL) improves sample efficiency by planning in a learned latent space (Jang et al., 2022; Reed et al., 2022; Tom et al., 2020; Black et al., 2024). Recent world-model methods (Hansen et al., 2024; Hafner et al., 2023) sample candidate action sequences at each decision step, roll them forward through the world model, estimate their returns, and execute the best one. Despite substantial architectural differences across planners, this always reduces to the same operation: ranking candidate trajectories by predicted return.

That operation is a fundamental vulnerability. A world model trained on finite data cannot be accurate everywhere; the planner, searching actively for high-value sequences, selectively amplifies whatever local errors exist (Arumugam et al., 2018; Jiang et al., 2015). This raises two questions: can an external adversary systematically exploit the same mechanism, and do overestimated trajectories carry an identifiable signature? Existing work has not addressed either question directly. Adversarial attacks on RL agents largely target the outputs of model-free policies, or influence control indirectly, rather than exploiting local value-estimation instability to manipulate rankings (Schott et al., 2024a; Cui et al., 2024; Pattanaik et al., 2017). Uncertainty-aware planning instead relies on

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.

ensembles or dedicated uncertainty estimators, at the cost of extra computation and retraining (Chua et al., 2018; Vlastelica et al., 2023; Pinto et al., 2017).

To answer these questions, we examine how candidate-trajectory values change under small input perturbations, and make three observations. Trajectories that receive high predicted returns yet fail in the environment frequently exhibit action jitter and excessive cumulative impact. Their predicted returns are systematically overestimated, leaving a clear gap to true returns. Most importantly, these high returns are not stable: under small perturbations to the initial latent state, their valuations fluctuate far more than those of successful trajectories. We accordingly identify valuation sensitivity as a mechanistic signature of fragile, overestimated trajectories: it explains where the failures come from, and simultaneously points to an exploitable attack surface and a usable signal for defense.

Based on this finding, we work along two directions, attack and defense, comprising three components. First, we design a white-box latent-space perturbation attack. Bounded perturbations are injected into intermediate latent states during candidate evaluation, directly targeting the world model's local sensitivity to manipulate the relative ranking of candidate trajectories and promote unreliable action sequences into the planner's elite set. This setting serves as a controlled mechanistic tool for characterizing the exploitation mechanism itself. Second, we propose a black-box observation perturbation attack, in which the attacker can only submit perturbed observations and observe the resulting actions, and searches via natural evolution strategies (NES (Wierstra et al., 2014; Mohamed, 2025)) for observations that drive the planner toward abnormal action templates. Under both settings, the resulting ranking changes cause action drift, trajectory jitter, and reduced task success, indicating that valuation sensitivity is not merely correlated with failure but is a genuinely exploitable mechanism in planning. Finally, we propose a defense applied at test time that needs no training and no model-specific design. We re-evaluate each candidate trajectory under multiple small perturbations and combine the mean predicted return with two penalties, one for return variance and one for action smoothness. The resulting conservative score replaces the original predicted return during ranking and elite selection. The defense requires no modification to the world model, no retraining, and no attack labels, and thus provides a unified response to both latent-space and observation-space perturbations.

Our contributions are threefold:

- We identify valuation sensitivity as a signature of fragile trajectories. These trajectories receive high predicted returns but perform poorly in the real environment, and they show unstable value estimates and abnormal control patterns under small perturbations.
- We develop two attacks targeting this mechanism: bounded latent-state perturbations in the white-box setting and NES-based observation perturbations in the black-box setting. Across the evaluated planners, they achieve attack success rates of 72.3%–81.6% and reduce task success to 12.4%–25.6%. Both attacks manipulate trajectory rankings and degrade control performance.
- We propose a stability-aware trajectory score that combines return statistics under perturbed observations and an action-smoothness penalty. The defense requires no model modification or additional training. Across the evaluated planners, it reduces the attack success rate to 5.5%–9.3% under white-box attacks and 18.5%–20.4% under black-box attacks, while largely preserving clean performance.

2 Related Work

2.1 World Models and Latent-Space Planning

World models support decision-making by predicting environment dynamics, rewards, and values in a compact latent space. Dreamer-style methods optimize policies and value functions using imagined trajectories, with DreamerV3 (Hafner et al., 2023) substantially improving robustness across tasks and domains. The TD-MPC family (Hansen et al., 2022b; 2024) instead combines an implicit world model with model predictive control. At each decision step, TD-MPC2 (Hansen et al., 2024) samples action sequences, estimates their returns from predicted rewards and terminal values, and updates its planning distribution accordingly. Recent methods such as IRIS (Hansen et al., 2022a) and UniZero (Pu et al., 2024) further employ Transformer architectures to model long-term dependencies and latent dynamics, extending world-model learning to more complex prediction and

planning problems. Despite their architectural and algorithmic differences, these methods rely on predicted returns to determine the relative preference among potential behaviors. In explicit planners such as TD-MPC, candidate trajectories are directly ranked and selected according to their predicted returns; in Dreamer-style methods, the same preference is induced indirectly through imagined policy optimization. Local model errors can therefore alter trajectory preferences and be amplified by the planning process. Although the effects of model error on planning and value estimation are well recognized, their systematic exploitation to manipulate trajectory rankings remains an underexplored adversarial threat.

2.2 Adversarial Attacks on Reinforcement Learning

Recent attacks on reinforcement-learning agents have progressed beyond one-step observation perturbations toward stealthier and temporally structured threats. BadRL (Cui et al., 2024) introduces a sparse targeted backdoor attack that activates malicious behavior at only a small number of critical time steps, thereby manipulating long-horizon decisions while largely preserving benign policy performance. SHIFT (Sun et al., 2026) instead use a diffusion model to generate adversarial states consistent with the underlying state distribution while reducing the cumulative return of the target policy. Recent surveys (Schott et al., 2024b) further identify observation perturbation, training-data poisoning, backdoor injection, and environment manipulation as major attack surfaces in deep reinforcement learning, although most existing studies still focus on the action outputs or task returns of model-free policies. As world-model-based embodied agents become increasingly prominent, the attack surface has expanded from explicit policy networks to internal prediction and planning components. Recent studies perturb latent world-model representations to induce erroneous future predictions and policy failures, or poison training data to covertly manipulate latent dynamics and downstream control behavior (Zhang et al., 2026; Riaño et al., 2026). However, existing methods do not directly exploit local value-estimation instability to manipulate candidate-trajectory rankings. Our work formulates the attack as trajectory-ranking manipulation and studies how bounded perturbations promote low-quality trajectories into the planner's elite set.

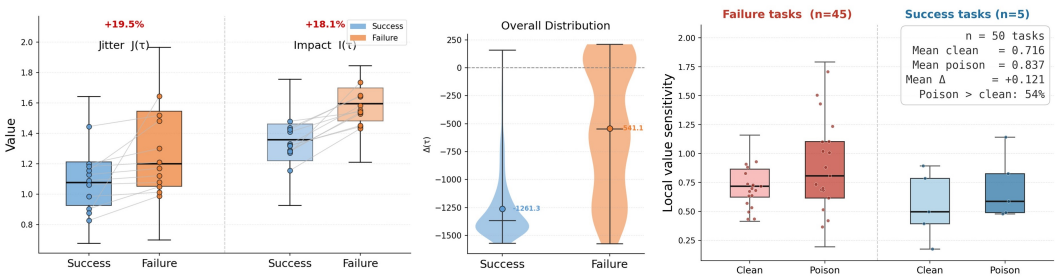
2.3 Robust and Uncertainty-Aware Planning

Learned dynamics models often incur larger prediction errors in regions with limited data coverage or under distribution shift. Consequently, many model-based reinforcement learning methods explicitly account for model uncertainty during planning. PETS (Chua et al., 2018) propagates predictive uncertainty through probabilistic dynamics ensembles and trajectory sampling, evaluating candidate action sequences across multiple model predictions. More recent methods further incorporate different forms of uncertainty into risk-aware planning. RAMBO-RL (Vlastelica et al., 2023) distinguishes epistemic from aleatoric uncertainty in trajectory-sampling MPC to balance active exploration and risk avoidance, while COPlanner (Wang et al., 2024) combines conservative model rollouts with optimistic exploration in the real environment to reduce the influence of model error on policy learning. These approaches primarily rely on probabilistic models, ensembles, or learned uncertainty estimators to modify planning or training objectives. A related line of work studies how planners exploit inaccuracies in learned models. Planner overfitting occurs when optimization discovers action sequences with high predicted returns in the model but poor performance in the true environment, and prior work mitigates this behavior by restricting or regularizing the plan space (Arumugam et al., 2018). RePo (Zhu et al., 2023) regularizes posterior predictability to improve the robustness of visual world-model representations to task-irrelevant distribution shifts. TD-MPC2 performs scalable latent-space planning with an implicit world model and MPPI, focusing primarily on training stability and robustness across tasks. In contrast, our method measures the local valuation sensitivity of candidate trajectories within a fixed world model and uses this signal to adjust trajectory scores at test time, without model ensembles or retraining.

3 Preliminaries and Threat Model

3.1 World-Model Planning

We consider a model-based reinforcement learning agent that performs receding-horizon planning with a learned world model. Let o_t denote the observation at time t, g an optional task identifier, and z_t the corresponding latent state. The observation encoder maps the current observation to the



(a) (b) (c)

Figure 1: Trajectory fragility analysis. (a) Action jitter and impact for successful vs. failed trajectories, showing failures exhibit erratic control. (b) Predicted vs. true return, revealing that failed trajectories receive spuriously high predictions. (c) Return sensitivity under small latent perturbations, showing attack-promoted trajectories are far less stable than normal ones.

latent space as $z_t = h_\theta(o_t, g)$. The world model consists of a latent dynamics model, a reward head, and a value head. Given a latent state and an action, the dynamics model predicts the next latent state, while the reward and value heads estimate the immediate reward and the future return. For a candidate action sequence $\tau = (a_0, \ldots, a_{H-1})$, the model recursively generates latent states from an initial state z_0 and assigns the predicted score

$$R(\tau; z_0, g) = \sum_{h=0}^{H-1} \gamma^h r_\theta(z_h, a_h, g) + \gamma^H V_\theta(z_H, g), \qquad z_{h+1} = d_\theta(z_h, a_h, g). \tag{1}$$

The terminal value $V_\theta(z_H, g)$ estimates the return beyond the planning horizon, allowing the planner to evaluate finite-horizon rollouts while retaining information about longer-term outcomes.

At each decision step, the planner samples candidate action sequences, evaluates them using $R(\tau; z_t, g)$, and updates its action-sampling distribution. For example, CEM selects the highest-scoring candidates as an elite set, whereas MPPI assigns return-dependent weights to candidate sequences. These procedures can be represented by a generic planning operator: $\tau_t^* = \mathcal{P}(z_t; R)$. The agent executes only the first action $a_t^* = [\tau_t^*]_0$, receives the next observation, and replans from the resulting latent state. Consequently, errors in the predicted scores can alter the relative ranking of candidate trajectories and thereby change the selected action.

3.2 Threat Model

Attacker objective. The attacker aims to manipulate the planner's trajectory ranking by exploiting valuation sensitivity, causing low-quality trajectories to be selected over trajectories that perform better in the true environment. The attack targets the model-based evaluation process without modifying environment dynamics, reward functions, or world-model parameters.

Attacker capabilities. We consider two capability settings. In the white-box setting, the attacker accesses the victim's world-model parameters and can introduce bounded perturbations δ ($\|\delta\|_\infty \leq \epsilon$) into latent states during planning. This setting serves as a mechanistic analysis tool to characterize the exploitation mechanism. In the black-box setting, the attacker can only query the victim agent and observe its actions. The attacker applies bounded perturbations to input observations, which indirectly influence the latent states produced by the victim's encoder.

Defender capabilities. The defender can intervene at test time by re-evaluating candidate trajectories under perturbed latent states and modifying the scoring function used for ranking. However, the defender cannot retrain the world model or access attack labels.

4 Valuation Sensitivity of Fragile Trajectories

A learned world model cannot be uniformly accurate, particularly in regions poorly covered by training data. When used by a planner that actively searches for high-value sequences, local model errors can be amplified, leading to the selection of trajectories that appear promising in the model but fail in the true environment (Arumugam et al., 2018; Wu et al., 2022). We investigate whether such fragile trajectories exhibit identifiable characteristics.

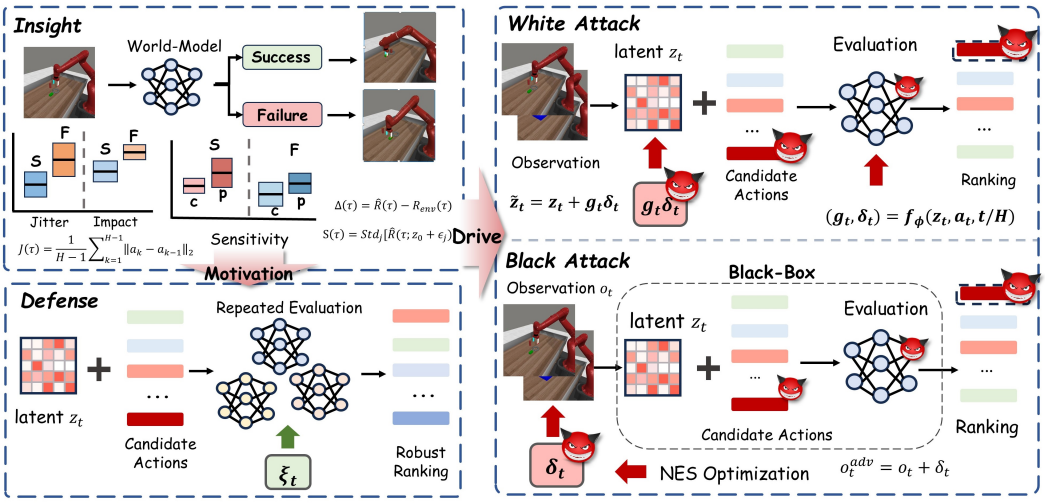


Figure 2: Overview of the proposed framework. Candidate trajectories are ranked by model-predicted return, which is manipulated by latent-space (white-box) or observation-space (black-box) perturbations, and stabilized at test time by conservative re-evaluation under perturbed states.

Abnormal control patterns in failures. We begin by analyzing clean-environment failures where the planner selects trajectories with high predicted returns that nevertheless fail. Figure 1(a) visualizes the action jitter $J(\tau) = \frac{1}{H-1}\sum_{k=1}^{H-1}\|a_k - a_{k-1}\|_2$ and control impact (cumulative action magnitude) for successful versus failed trajectories. Failed trajectories exhibit significantly higher jitter and impact, suggesting the planner favors erratic control patterns that do not contribute to task completion. Figure 1(b) further shows that these abnormal trajectories receive spuriously high predicted returns from the model, creating a valuation gap $\Delta(\tau) = \hat{R}(\tau) - R_{\text{env}}(\tau)$ where model predictions systematically overestimate true performance.

Valuation sensitivity as a mechanistic signature. We next examine whether overestimation is a global property of an action sequence or depends locally on the initial latent state. For each trajectory τ, we evaluate its predicted return from perturbed initial states $z_0 + \epsilon_j$ where $\|\epsilon_j\|$ is small, and compute the empirical sensitivity: $S(\tau) = \text{Std}_j\left[\hat{R}(\tau; z_0 + \epsilon_j)\right]$.

Figure 1(c) reveals that overestimated failure trajectories exhibit substantially higher return variance under perturbations than successful trajectories, whose valuations remain stable. This separation demonstrates that valuation sensitivity, the instability of predicted returns under small state perturbations, serves as a discriminative signature of fragile trajectories. These observations motivate both our attack and defense strategies: if high predicted returns can be destabilized by small perturbations, then (i) an attacker can exploit this sensitivity to manipulate trajectory rankings, and (ii) a defender can detect and suppress unstable trajectories by measuring their valuation sensitivity.

5 Method

We now present our method for exploiting and defending against valuation sensitivity in latent world-model planners. Figure 2 provides an overview of our framework, which consists of three components: a white-box latent-space perturbation attack that exposes the exploitation mechanism (Section 5.1), a black-box observation-only attack that reaches the same instability through the encoder interface (Section 5.2), and a stability-aware test-time defense that re-evaluates candidate trajectories under perturbed states and ranks them with a conservative score (Section 5.3). Both attacks leave the victim's world model, reward function, and environment dynamics untouched; the defense likewise requires no retraining and no attack labels.

5.1 White-Box Latent-Space Perturbation Attack

To investigate whether valuation sensitivity can be exploited to alter trajectory rankings, we consider a white-box attack with privileged access to the planner's latent rollouts. Beyond knowing the

model parameters and gradients, the attacker can inject bounded perturbations into intermediate latent states during candidate evaluation. We use this setting as a controlled mechanism study. The world-model parameters and environment remain unchanged, and the attacker does not directly overwrite trajectory scores, elite-set membership, or executed actions. At each imagined step h, an action-conditioned module produces a perturbation and an activation gate:

$$(\delta_h, g_h) = f_\phi(z_h, a_h, h/H), \qquad \tilde{z}_h = z_h + g_h \delta_h, \qquad \|\delta_h\|_\infty \le \epsilon, \quad g_h \in [0,1]. \tag{2}$$

Here, z_h and a_h denote the current imagined state and candidate action. The gate controls the intervention strength. Since the module conditions on each candidate's state and action, its interventions need not correspond to a single observation perturbation.

We train the module to promote an abnormal action template τ_b, such as hold, jitter, or retreat, relative to a reference sequence τ_g generated by the policy prior. Let $\hat{R}_\phi$ denote the predicted return with the perturbation module active. The training objective is

$$\mathcal{L} = \mathbb{E}_{z_0}\left[\max\left(0, m + \hat{R}_\phi(\tau_g; z_0) - \hat{R}_\phi(\tau_b; z_0)\right)\right] + \alpha_{\text{short}}\mathcal{L}_{\text{short}} + \lambda_\delta \mathbb{E}_h\|\delta_h\|_2^2 + \lambda_g \mathbb{E}_h[g_h]. \tag{3}$$

The first term encourages the abnormal sequence to outrank the reference by a margin $m > 0$. The short-term term $\mathcal{L}_{\text{short}}$ is the negative expected discounted sum of predicted rewards along τ_b, encouraging favorable intermediate reward estimates. The remaining terms penalize perturbation magnitude and gate activation. Only the attack module is trained; the victim world model remains frozen.

During inference, the module perturbs latent states within candidate rollouts, altering the predicted returns used by the planner's ordinary ranking and distribution updates. These changes can promote abnormal sequences into the elite set and induce persistent action deviations through receding-horizon planning. This experiment tests exploitability under bounded internal interventions; it neither establishes a certified worst-case bound nor implies that the same interventions are realizable through observation changes. The next section examines the complementary observation-only black-box setting.

5.2 Black-Box Observation Perturbation Attack

The latent-space attack assumes privileged access to the planner's internal rollout. We now consider a black-box setting in which the attacker cannot access the model parameters, latent states, gradients, predicted returns, or MPPI trajectories. The attacker can only submit perturbed observations to the planner and observe its output actions. At decision step t, the attacker first queries the planner with the clean observation o_t and obtains the corresponding action. It then selects an abnormal action template, such as hold, jitter, or retreat, and uses the resulting action a_t^{bad} as the attack target. The attacker searches for a bounded observation perturbation that makes the planner output an action close to this target:

$$\min_{\|\delta_t\|_\infty \le \epsilon} \mathcal{L}_{\text{bb}}(\delta_t) = \left\|\pi_{\text{plan}}(o_t + \delta_t) - a_t^{\text{bad}}\right\|_2. \tag{4}$$

Since gradients and internal scores are unavailable, we optimize this objective with Natural Evolution Strategies (NES). At each iteration, the attacker evaluates the planner at antithetically perturbed observations $o_t + \delta_t + \sigma u_i$ and $o_t + \delta_t - \sigma u_i$, where u_i are random search directions. The resulting loss differences provide a zeroth-order search direction, after which the perturbation is updated by a projected step:

$$\delta_t \leftarrow \Pi_{\|\delta\|_\infty \le \epsilon}\left(\delta_t - \alpha \ \text{sign}\left(\widehat{\nabla}_{\delta_t}\mathcal{L}_{\text{bb}}\right)\right). \tag{5}$$

The attack uses only action feedback and does not require access to the planner's predicted returns or candidate trajectories. When the attack is inactive, the clean observation is passed to the planner. When activated, the optimized observation is submitted and the resulting action is executed in the environment.

The perturbed observation changes the latent state produced by the victim's encoder and may consequently alter candidate valuation and trajectory selection. Repeating the attack across decision steps can induce persistent action drift, jitter, or retreat behavior under receding-horizon planning. This experiment evaluates whether the instability identified by the latent-space analysis can also be reached through an observation-only interface. Its effectiveness depends on the perturbation and query budgets, which we report together with the attack results.

5.3 VALUATION-STABILITY DEFENSE

The white-box and black-box attacks exploit instability in the planner's trajectory evaluation. We therefore introduce a training-free, test-time defense that evaluates each candidate trajectory under multiple small perturbations and penalizes candidates whose predicted returns are highly sensitive. Let A_i denote the i-th candidate action sequence. The planner evaluates A_i from the clean latent state and from several perturbed latent states, obtaining predicted returns $\{V_i^{(k)}\}_{k=1}^{K}$. We compute the mean return and its standard deviation, and use them to define a conservative trajectory score:

$$S_i = \mathrm{mean}_k\, V_i^{(k)} - \beta_{\mathrm{sens}}\, \mathrm{std}_k\, V_i^{(k)} - \lambda_{\mathrm{smooth}} P_i, \qquad P_i = \frac{1}{H-1} \sum_{h=1}^{H-1} \|a_{i,h} - a_{i,h-1}\|_2^2. \quad (6)$$

Here, β_{sens} controls the penalty for valuation sensitivity, while $\lambda_{\mathrm{smooth}}$ controls the action-smoothness penalty. The clean evaluation is included among the perturbed evaluations, so the defense retains the original return objective while discouraging unstable candidates.

The conservative score S_i replaces the original predicted return during candidate ranking and elite selection. In MPPI, the action distribution is updated using weights proportional to $\exp(\kappa S_i)$, where κ is the temperature coefficient. Thus, trajectories with high nominal values but unstable evaluations or abrupt action changes receive lower influence on the next planning distribution. The defense does not modify the world-model parameters, reward function, or environment dynamics, and requires no additional training data or attack labels. In the white-box setting, it reduces the influence of trajectories whose values are artificially increased by latent-state perturbations. In the black-box setting, it mitigates the downstream effect of observation perturbations after they alter the planner's root latent state. We evaluate both robustness and clean performance to measure this trade-off.

6 EXPERIMENTS

6.1 EXPERIMENTAL SETUP

Victim Models. We evaluate pretrained TD-MPC2-48M, TD-MPC2-317M, and NEWT (Hansen et al., 2026) models. TD-MPC2 selects actions through latent-space trajectory optimization within a learned implicit world model. All model parameters remain frozen during evaluation.

Dataset. We conduct experiments on all 50 robotic manipulation tasks in Meta-World, including grasping, pushing, pulling, and assembly. All tasks use 39-dimensional observations and 4-dimensional continuous actions. Each episode contains 200 environment steps with an action repeat of 2, corresponding to 100 control decisions. We run 50 episodes for each task and evaluation setting. More experimental details are provided in the Appendix A.

Evaluation Metrics. We report task success rate (**Success**), cumulative episode reward (**Reward**), and attack success rate (**ASR**). Following TD-MPC2, Success is determined by the success indicator at the *final step* of an episode, while Reward is the sum of environment rewards over the entire episode. Success and Reward are first averaged over episodes within each task and then averaged equally across the 50 Meta-World tasks. We define ASR using *micro aggregation*, which measures the proportion of clean-successful episodes that fail under attack:

$$\mathrm{ASR} = \frac{\sum_{j=1}^{50} \sum_{i=1}^{N_j} c_{j,i}\,(1 - a_{j,i})}{\sum_{j=1}^{50} \sum_{i=1}^{N_j} c_{j,i}} \times 100\%, \quad (7)$$

where $c_{j,i}, a_{j,i} \in \{0, 1\}$ denote whether the i-th paired episode of task j succeeds under the clean and attacked conditions, respectively, and N_j is the number of paired episodes for task j. A higher ASR indicates a more effective attack. We report ASR only when $\sum_{j=1}^{50} \sum_{i=1}^{N_j} c_{j,i} > 0$.

6.2 MAIN EXPERIMENTAL RESULTS

Attack and Defense Results. Table 1 shows that both white-box and black-box attacks substantially degrade all three world-model planners, with undefended ASRs ranging from 72.3% to 81.6%. The proposed defense reduces these rates to 5.5%–9.3% under white-box attacks and 18.5%–20.4% under black-box attacks, accompanied by substantial improvements in task success and cumulative

Table 1: Performance and robustness of world-model planners on 50 Meta-World tasks under clean, white-box, and black-box conditions. S denotes task success rate (%), R denotes cumulative environment reward, and A denotes attack success rate (%).

Victim Model	Defense	Clean		White-box			Black-box		
		S↑	R↑	S↑	R↑	A↓	S↑	R↑	A↓
TD-MPC2-48M	No Defense	86.1	1456.4	12.4	437.4	81.6	19.6	440.2	80.1
	DD	81.5	1438.8	20.7	531.0	78.3	20.0	525.7	79.5
	Q-Ensemble	83.7	1473.6	47.4	993.2	52.1	43.7	953.0	56.2
	Ours	86.3	1467.8	84.1	1455.5	8.8	80.1	1426.3	18.5
TD-MPC2-317M	No Defense	91.2	1564.0	19.2	468.9	80.0	20.0	446.2	79.0
	DD	84.4	1527.3	23.8	515.3	75.7	21.7	524.8	78.8
	Q-Ensemble	88.1	1542.7	42.5	912.6	54.2	38.5	813.6	61.2
	Ours	91.3	1578.3	87.1	1506.4	5.5	74.6	1473.6	19.4
NEWT	No Defense	92.3	1653.4	22.4	532.8	76.1	25.6	582.5	72.3
	Ours	92.8	1662.6	88.4	1649.0	9.3	70.5	1589.4	20.4

reward. Clean performance remains close to that of the original planners. These results demonstrate consistent robustness gains across the evaluated model sizes and architectures, although a larger performance gap remains under black-box attacks.

Comparison with Test-Time Defenses. We compare our method with Denoising Defense (DD) and Q-Ensemble on both TD-MPC2 model sizes. DD uses stochastic perturbations and conservative decisions to mitigate value-estimation errors, while Q-Ensemble uses disagreement among multiple Q-functions as an uncertainty signal. Implementation details are provided in the Appendix B. As shown in Table 1, DD offers limited protection, whereas Q-Ensemble achieves larger improvements but still leaves substantial vulnerability. Our method achieves higher success rates and rewards and lower ASRs than both baselines under both attack settings. For example, under white-box attack on TD-MPC2-48M, it achieves a success rate of 84.1% and a reward of 1455.5, compared with 47.4% and 993.2 for Q-Ensemble. The consistent advantage on TD-MPC2-317M further supports its effectiveness relative to the evaluated defenses.

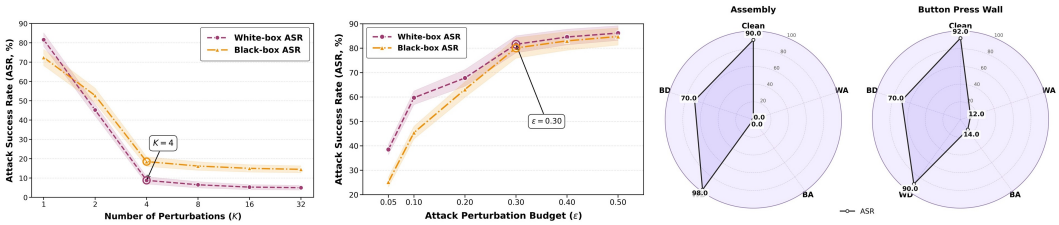
Table 2: Standard and defense-aware white-box attacks on TD-MPC2-48M. S: success (%); R: reward; A: ASR (%).

Setting	No Defense			Ours		
	S↑	R↑	A↓	S↑	R↑	A↓
Standard WB	12.4	437.4	81.6	84.1	1455.5	8.8
Adaptive WB	10.8	421.5	84.6	77.2	1373.1	20.3

Defense-Aware White-Box Attack. We further evaluate a defense-aware white-box attacker that has access to the defense scoring rule and accounts for both sensitivity and action-smoothness penalties during optimization. As shown in Table 2, adapting the attack to the defense increases its ASR from 8.8% to 20.3% and decreases the defended success rate from 84.1% to 77.2%. Nevertheless, under this adaptive attack, the defended planner substantially outperforms the undefended planner in both success rate (77.2% vs. 10.8%) and reward (1373.1 vs. 421.5). Thus, the defense remains beneficial against the evaluated adaptive attacker, although its protection is reduced.

6.3 Ablation Studies

We examine the contributions of the scoring components in Table 3. Implementation details are provided in the Appendix C. Mean-Only ranks candidates by their mean predicted return under perturbations. The + Sensitivity and + Smoothness variants each add the corresponding penalty to Mean-Only, while Full includes both penalties. Adding the sensitivity penalty yields a larger robustness gain than adding the smoothness penalty alone. On TD-MPC2-48M, the white-box ASR decreases from 53.1% with Mean-Only to 24.4% with sensitivity and 42.2% with smoothness; the



(a) (b) (c)

Figure 3: Analysis of defense parameters and task robustness. (a) White-box and black-box attack success rates under different numbers of perturbations K; (b) attack success rates under different perturbation budgets ϵ; (c) comparison of task success rates on Assembly and Button Press Wall before and after defense.

Table 3: Ablation of the defense scoring components on TD-MPC2. Mean-Only uses the mean predicted return under perturbations; + Sensitivity and + Smoothness add the respective penalty to Mean-Only, while Full includes both penalties. S denotes task success rate (%), R denotes cumulative environment reward, and A denotes attack success rate (%).

Victim Model	Variant	Clean		White-box			Black-box		
		S↑	R↑	S↑	R↑	A↓	S↑	R↑	A↓
TD-MPC2-48M	Mean-Only	85.7	1463.6	45.8	983.7	53.1	32.6	846.3	65.4
	+ Sensitivity	86.1	1468.1	75.2	1282.9	24.4	68.3	1124.5	30.2
	+ Smoothness	85.3	1431.6	53.7	1105.3	42.2	43.6	953.9	54.7
	Full	**86.3**	**1467.8**	**84.1**	**1455.5**	**8.8**	**80.1**	**1426.3**	**18.5**
TD-MPC2-317M	Mean-Only	89.7	1573.2	40.1	924.7	57.3	30.5	772.6	68.3
	+ Sensitivity	91.2	1561.7	72.7	1263.9	24.8	67.3	1027.4	30.1
	+ Smoothness	90.3	1536.5	50.2	985.4	47.3	42.6	943.7	51.7
	Full	**91.3**	**1578.3**	**87.1**	**1506.4**	**5.5**	**74.6**	**1473.6**	**19.4**

corresponding black-box ASRs decrease from 65.4% to 30.2% and 54.7%, respectively. Combining both penalties further improves robustness and achieves the highest attacked success rates and rewards among the evaluated variants. The same pattern holds for TD-MPC2-317M. These results suggest valuation sensitivity provides the stronger individual robustness signal, while action smoothness contributes additional protection when combined with it.

7 Conclusion

This work studies how local value-estimation errors distort candidate-trajectory rankings in latent world-model planners. We find that trajectories with high predicted returns but poor real-environment performance exhibit substantially larger valuation changes under small latent perturbations, and are frequently associated with abnormal action jitter. Based on this, we develop a white-box latent intervention attack, a black-box NES observation attack, and a training-free test-time defense that re-evaluates candidates under perturbations and ranks them by combined return mean, variance, and action smoothness. Experiments show this substantially reduces attack success rates while largely preserving clean performance.

Limitations. The white-box attack requires direct modification of internal latent states and is mainly a mechanistic tool. The black-box NES attack has a high query cost and unclear real-world feasibility. And the proposed stability score provides no formal robustness guarantee. Future work will study low-query, physically realizable attacks and evaluate the approach across broader world-model and planning architectures.

REQUIRED: AI USE STATEMENT

During the preparation of this manuscript, the authors used AI-based tools to assist with language editing, organization of the manuscript, and parts of the software implementation. AI tools were also used during the early-stage exploration of method design to suggest alternative formulations and implementation strategies. All final methodological decisions, theoretical claims, experimental protocols, source-code changes, and interpretations of results were made and reviewed by the authors. The authors independently verified the mathematical formulations, implementation correctness, experimental results, citations, and compliance with the ICLR submission requirements. AI tools were not used to generate or fabricate experimental data, evaluation results, or citations. The authors take full responsibility for the content of this submission.

REFERENCES

Dilip Arumugam, David Abel, Kavosh Asadi, Nakul Gopalan, Christopher Grimm, Jun Ki Lee, Lucas Lehnert, and Michael L Littman. Mitigating planner overfitting in model-based reinforcement learning. *arXiv preprint arXiv:1812.01129*, 2018.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. pi0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.

Jing Cui, Yufei Han, Yuzhe Ma, Jianbin Jiao, and Junge Zhang. Badrl: Sparse targeted backdoor attack against reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 11687–11694, 2024.

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

Nick Hansen, Hao Su, and Xiaolong Wang. Td-mpc2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, volume 2024, pp. 47376–47405, 2024.

Nick Hansen, Hao Su, and Xiaolong Wang. Learning massively multitask world models for continuous control. In *International Conference on Learning Representations*, volume 2026, pp. 5048–5091, 2026.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. In *International Conference on Machine Learning*, 2022a. URL `https://api.semanticscholar.org/CorpusID:247318709`.

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

Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *conference on Robot Learning*, pp. 991–1002. PMLR, 2022.

Nan Jiang, Alex Kulesza, Satinder Singh, and Richard Lewis. The dependence of effective planning horizon on model accuracy. In *Proceedings of the 2015 international conference on autonomous agents and multiagent systems*, pp. 1181–1189, 2015.

Nachaat Mohamed. Artificial intelligence and machine learning in cybersecurity: a deep dive into state-of-the-art techniques and future paradigms. *Knowledge and Information Systems*, 67(8): 6969–7055, 2025.

Anay Pattanaik, Zhenyi Tang, Shuijing Liu, Gautham Bommannan, and Girish Chowdhary. Robust deep reinforcement learning with adversarial attacks. *arXiv preprint arXiv:1712.03632*, 2017.

Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. In *International conference on machine learning*, pp. 2817–2826. PMLR, 2017.

Yuan Pu, Yazhe Niu, Zhenjie Yang, Ji-Yuan Ren, Hongsheng Li, and Yu Liu. Unizero: Generalized and efficient planning with scalable latent world models. *Trans. Mach. Learn. Res.*, 2025, 2024. URL https://api.semanticscholar.org/CorpusID:270559382.

Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022.

Roberto Riaño, Gorka Abad, Stjepan Picek, and Aitor Urbieta. When the world lies: Backdoor attacks on latent world models for downstream control. *arXiv preprint arXiv:2609.15781*, 2026.

Lucas Schott, Josephine Delas, Hatem Hajri, Elies Gherbi, Reda Yaich, Nora Boulahia-Cuppens, Frederic Cuppens, and Sylvain Lamprier. Robust deep reinforcement learning through adversarial attacks and training: A survey. *arXiv preprint arXiv:2403.00420*, 2024a.

Lucas Schott, Joséphine Delas, Hatem Hajri, Elies Gherbi, Reda Yaich, Nora Cuppens-Boulahia, Frédéric Cuppens, and Sylvain Lamprier. Robust deep reinforcement learning through adversarial attacks and training : A survey. *ArXiv*, abs/2403.00420, 2024b. URL https://api.semanticscholar.org/CorpusID:268201718.

Xiaolin Sun, Feidi Liu, Zhengming Ding, and Zizhan Zheng. Diffusion guided adversarial state perturbations in reinforcement learning. *Advances in Neural Information Processing Systems*, 38: 43780–43815, 2026.

Brown Tom, Mann Benjamin, Ryder Nick, Subbiah Melanie, Dhariwal Prafulla, Neelakantan Arvind, Shyam Pranav, Sastry Girish, Askell Amanda, Agarwal Sandhini, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33(1):1877–1901, 2020.

Marin Vlastelica, Sebastian Blaes, Cristina Pineri, and Georg Martius. Mind the uncertainty: risk-aware and actively exploring model-based reinforcement learning. *arXiv preprint arXiv:2309.05582*, 2023.

Xiyao Wang, Ruijie Zheng, Yanchao Sun, Ruonan Jia, Wichayaporn Wongkamjan, Huazhe Xu, and Furong Huang. Coplanner: Plan to roll out conservatively but to explore optimistically for model-based rl. In *International Conference on Learning Representations*, volume 2024, pp. 17089–17110, 2024.

Daan Wierstra, Tom Schaul, Tobias Glasmachers, Yi Sun, Jan Peters, and Jürgen Schmidhuber. Natural evolution strategies. *The Journal of Machine Learning Research*, 15(1):949–980, 2014.

Zifan Wu, Chao Yu, Chen Chen, Jianye Hao, and Hankz Hankui Zhuo. Plan to predict: Learning an uncertainty-foreseeing model for model-based reinforcement learning. *Advances in Neural Information Processing Systems*, 35:15849–15861, 2022.

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.

Chuning Zhu, Max Simchowitz, Siri Gadipudi, and Abhishek Gupta. Repo: Resilient model-based reinforcement learning by regularizing posterior predictability. *Advances in Neural Information Processing Systems*, 36:32445–32467, 2023.

APPENDIX

A EXPERIMENTAL SETUP

White-Box Attack Settings. The white-box attack experiments were conducted on three pretrained world models: TD-MPC2-48M, TD-MPC2-317M, and NEWT. All models were evaluated on the complete set of 50 Meta-World manipulation tasks. We ran 50 evaluation episodes per task. Except for the model size and the corresponding pretrained checkpoint, the three models used the same attack and evaluation configuration. The attack module was optimized for 3,000 steps using a state buffer of 512 samples and a batch size of 64. The latent-state perturbation budget was set to $\epsilon = 0.3$. During attack training, 10,000 environment steps were used to collect candidate states. No fixed minimum reward threshold was imposed. Instead, candidate states were selected according to the empirical reward distribution, retaining states in the top 30% of observed returns. The margin in the ranking objective was set to 0.25, and the coefficient of the short-horizon reward term was set to 0.15. The perturbation-magnitude regularization coefficient was 10^{-4}, and the gate-activation regularization coefficient was 10^{-3}. The attack module was optimized with Adam using a learning rate of 10^{-3}. All remaining planning and environment configurations followed the original settings of the corresponding models.

Black-Box Attack Settings The black-box attack experiments were conducted on three pretrained world models: TD-MPC2-48M, TD-MPC2-317M, and NEWT. All models were evaluated on the complete set of 50 Meta-World tasks, with 50 attacked episodes per task. Except for the model size and the corresponding pretrained checkpoint, the three models used the same black-box attack configuration. The attack objective was set to `targeted_bad`, which directs the planner toward a predefined abnormal action pattern through observation-space perturbations. The observation perturbation budget was set to $\epsilon = 0.1$ under the ℓ_∞ norm. The attack used Natural Evolution Strategies (NES) for zeroth-order optimization and was updated for five NES steps. Each update used eight search samples. The standard deviation of the search noise was set to 0.01, and the update step size was set to 0.015. The attack was applied at every eligible control decision step, corresponding to `attack_every=1`. The reward-based trigger threshold was set to 2.0, while the step-based trigger threshold was set to 999. Thus, under this configuration, attack activation was primarily determined by the reward-based condition rather than by the step-based condition. The same attack and trigger configurations were used for all three victim models.

White-Box Defense Settings. The white-box defense experiments were conducted on the same three models: TD-MPC2-48M, TD-MPC2-317M, and NEWT. Evaluation was performed on all 50 Meta-World tasks, with 50 episodes per task. The defense used the temporal robust-MPPI configuration. For each candidate trajectory, four latent-state perturbation samples were used for re-evaluation. The latent-noise scale was set to 0.030, the valuation-sensitivity penalty coefficient was set to 2.0, and the action-smoothness penalty coefficient was set to 0.05. The defense start fraction was set to 0.5. For the defense configuration, the attack module was trained for 500 steps, and 1,000 warm-up environment steps were used for candidate-state collection. The white-box attack budget targeted by the defense was set to $\epsilon = 0.3$, consistent with the attack experiments. The same defense hyperparameters were used for all three models; the model size was varied only through the corresponding pretrained checkpoint.

Black-Box Defense Settings. The black-box defense experiments were conducted on three pretrained world models: TD-MPC2-48M, TD-MPC2-317M, and NEWT. All models were evaluated on the complete set of 50 Meta-World tasks, with 50 attacked episodes per task. Except for the model size and the corresponding pretrained checkpoint, the three models used the same experimental configuration. Model compilation was disabled during evaluation. The black-box attack parameters used in the defense experiments were identical to those used in the standalone black-box attack experiments. Thus, apart from enabling the defense, no attack-side parameter was changed in the black-box defense experiments, ensuring a direct comparison between the undefended and defended settings. The black-box defense used a robust planning configuration based on observation-space perturbations. The defense mode was set to `observation`, and eight observation perturbation samples were used to re-evaluate candidate planning trajectories. The observation-noise scale was set to 0.01, the observation-based valuation-sensitivity penalty coefficient was set to 4.0, and the

action-smoothness penalty coefficient was set to 0.05. The defense score combined predicted-return statistics under observation perturbations with an action-smoothness constraint to re-rank candidate trajectories.

Black-Box Computational Cost. We additionally measured the end-to-end computational cost in the black-box setting. On the `mt80` benchmark with the `TD-MPC2-48M` model, we used an observation-space defense with 8 perturbed observations, an observation-noise scale of $\sigma_{\text{obs}} = 0.01$, $\beta_{\text{obs}} = 4.0$, and $\lambda_{\text{smooth}} = 0.05$. The black-box NES attack used 5 NES update steps with 8 search samples per step, and $\sigma_{\text{NES}} = 0.01$. Under this configuration, a complete evaluation of one episode takes approximately 10 minutes when both the attack and the defense are enabled. For comparison, the undefended black-box attack takes approximately 6 minutes per episode, whereas clean evaluation of the model takes approximately 1 minute per episode. These measurements correspond to end-to-end wall-clock time, including target-planner queries, NES optimization, and the repeated evaluations under multiple observation perturbations used by the defense.

B Baseline Test-Time Defenses

We compare our method with two test-time defense baselines, namely Denoising Defense (DD) and Q-Ensemble. Both baselines operate on the frozen pretrained TD-MPC2 model and require neither additional training data nor modifications to the environment dynamics. For a fair comparison, all methods use the same model checkpoint, planning horizon, MPPI population size, number of planning iterations, number of elites, temperature coefficient, and attack budget. The baselines differ only in how candidate trajectories are evaluated or selected.

B.1 Denoising Defense

Denoising Defense (DD) mitigates planning errors by applying stochastic perturbations to the current latent state and making decisions based on multiple evaluations. Let z_t denote the latent state at the current decision step, and let $\{\epsilon_k\}_{k=1}^{K}$ denote independently sampled perturbations. For a candidate action sequence A_i, DD evaluates the predicted return under each perturbed latent state:

$$V_{i,k} = R(A_i; z_t + \epsilon_k), \qquad k = 1, \ldots, K. \tag{8}$$

The resulting values are aggregated using the DD decision rule to obtain a denoised value estimate for each candidate trajectory:

$$\widetilde{V}_i = \mathcal{A}_{\text{DD}}\left(V_{i,1}, \ldots, V_{i,K}\right), \tag{9}$$

where $\mathcal{A}_{\text{DD}}$ denotes the aggregation and conservative decision procedure used by Denoising Defense. The denoised values $\widetilde{V}_i$ replace the nominal trajectory values during candidate selection, while the remaining MPPI planning procedure is kept unchanged.

DD reduces the influence of isolated valuation errors by averaging or aggregating predictions obtained from locally perturbed latent states. In this way, candidate trajectories that are sensitive to stochastic perturbations are less likely to dominate the planning distribution. Unlike our method, however, DD does not explicitly decompose the aggregated prediction into a mean-return term and a valuation-sensitivity term, nor does it include an explicit action-smoothness penalty. It therefore serves as a denoising-based robustness baseline rather than a sensitivity-aware trajectory-ranking method.

B.2 Q-Ensemble Defense

Q-Ensemble uses the disagreement among multiple Q-functions as an uncertainty signal. TD-MPC2 maintains multiple Q-function heads trained with the same latent representation but different bootstrap targets and stochastic training realizations. For a candidate trajectory A_i, we evaluate its terminal action value using all M Q-functions:

$$Q_i^{(m)} = Q^{(m)}\left(z_{i,H}, a_{i,H}\right), \qquad m = 1, \ldots, M. \tag{10}$$

The ensemble mean and disagreement are computed as

$$\overline{Q}_i = \frac{1}{M} \sum_{m=1}^{M} Q_i^{(m)}, \tag{11}$$

and

$$D_i = \mathrm{Std}_m \left[Q_i^{(m)} \right]. \tag{12}$$

We rank candidate trajectories using the uncertainty-aware score

$$S_i^{\mathrm{ens}} = \overline{Q}_i - \beta_{\mathrm{ens}} D_i, \tag{13}$$

where β_{ens} controls the strength of the ensemble disagreement penalty. The resulting score replaces the original terminal value during candidate ranking and MPPI distribution updates. This encourages the planner to prefer trajectories whose value estimates are consistent across Q-functions and to suppress trajectories associated with large ensemble disagreement.

Q-Ensemble differs from our method in the source of its uncertainty signal. Q-Ensemble measures disagreement among multiple Q-functions at a fixed latent state, whereas our method measures the sensitivity of the full predicted return to perturbations of the root latent state. Thus, Q-Ensemble captures value-function disagreement, while our method targets local, input-dependent instability in trajectory valuation and additionally accounts for action smoothness.

C Ablation Variants

Table 3 ablates the individual components of the proposed test-time defense score. All variants use the same TD-MPC2 model, candidate action sequences, and MPPI planning procedure, and differ only in how candidate trajectories are scored. In all cases, each candidate trajectory is evaluated from multiple perturbed root latent states to obtain its predicted returns.

Mean-Only. Mean-Only ranks candidate trajectories solely according to their mean predicted return across the perturbed latent states. This variant can reduce the effect of errors from a single state evaluation, but it does not explicitly penalize valuation instability or impose any constraint on the smoothness of the action sequence. Mean-Only therefore measures the benefit of repeated perturbed evaluation and simple return aggregation alone.

+ Sensitivity. + Sensitivity augments Mean-Only with a valuation-sensitivity penalty. Specifically, we measure the variation of the predicted return for each candidate trajectory across the perturbed latent states and penalize candidates with large return variance. This variant isolates the contribution of the valuation-sensitivity signal to robustness.

+ Smoothness. + Smoothness augments Mean-Only with an action-smoothness penalty. The smoothness cost is computed from the changes between consecutive actions in a candidate sequence, reducing the score of trajectories that contain abrupt action changes or high-frequency jitter. This variant evaluates the contribution of action regularization to suppressing abnormal control patterns.

Full. Full combines the mean predicted return, the valuation-sensitivity penalty, and the action-smoothness penalty. It therefore evaluates candidate trajectories according to their nominal predicted performance, valuation stability, and control smoothness, and corresponds to the complete defense proposed in this work.

Conceptually, Mean-Only focuses on the average performance under perturbed evaluations, + Sensitivity focuses on the stability of the predicted value, and + Smoothness focuses on the temporal regularity of the action sequence. Comparing these variants allows us to disentangle the contributions of repeated evaluation, valuation sensitivity, and action smoothness, and to examine their complementarity in the full defense.

D ATTACK TEMPLATE DESIGNS

To examine how different malicious action templates affect attack performance, we compare three attack configurations: Adaptive-only, Hybrid Templates, and Full Attack. All configurations use the same victim model, perturbation budgets, optimization procedure, and evaluation protocol, and differ only in how the malicious action templates are constructed.

D.1 ADAPTIVELY LEARNED MALICIOUS TEMPLATES

In the Adaptive-only configuration, the attack module learns malicious action templates directly from data and does not use manually designed hold, jitter, or retreat templates. Specifically, we collect reference behaviors from states near successful task completion and train the attack module to construct action sequences that induce the planner to deviate from normal control behavior. The learned templates can therefore adapt to the action distribution of the victim model and to the current task state, without assuming a predefined abnormal control pattern.

For the white-box attack, the learned malicious templates guide the latent-state perturbation module to increase the predicted value of malicious action sequences relative to normal reference sequences. For the black-box attack, the templates are projected into single-step target actions, and NES is used to search for observation perturbations that make the planner output actions close to the selected target. Adaptive-only therefore measures the attack capability obtained solely from learning state- and model-dependent malicious templates.

D.2 MANUALLY DESIGNED MALICIOUS TEMPLATES

In addition to the adaptive templates, we introduce several manually designed templates with explicit control semantics to represent common abnormal behaviors:

- Hold/slow-down: scales subsequent actions toward a small fraction of the reference actions, causing the agent to stop or slow down;
- Jitter: adds alternating directional perturbations to the reference actions, producing high-frequency action oscillations;
- Retreat: applies an action in the opposite direction of the nominal control along selected action dimensions, causing the agent to move away from the target;
- Stop: sets subsequent actions to zero, preventing further task progress.

These templates do not directly modify the environment state, reward function, or victim model. Instead, they serve as optimization targets that guide the attack toward candidate trajectories exhibiting specific abnormal behaviors. In the black-box setting, because the attacker observes and controls only the action at the current decision step, trajectory-level templates are projected into single-step action targets, and NES is used to search for the corresponding adversarial observations.

D.3 TEMPLATES EXPERIMENT SETTING

Adaptive-only In the Adaptive-only configuration, the attack module learns malicious action templates from data and does not use the manually designed hold, jitter, retreat, or stop templates introduced in this work. Specifically, the attack module learns action sequences from states near successful task completion that can induce the planner to deviate from its nominal behavior. The resulting templates are adapted to the behavior distribution of the victim model and the current task state.

Hybrid Templates Hybrid Templates combines adaptively learned templates with manually designed templates. This setting combines malicious actions learned from task states and victim behavior with predefined abnormal control patterns, allowing us to evaluate the effect of using both types of templates simultaneously.

Importantly, Full Attack (Ours) is not an additional combination of Adaptive-only and manually designed templates. Full Attack denotes the complete attack configuration used in our final method,

Table 4: Ablation of attack-template designs on TD-MPC2-48M. S denotes task success rate (%), R denotes cumulative environment reward, and A denotes attack success rate (%). Arrows indicate stronger attack performance under attacked conditions.

Attack Variant	Clean		White-box			Black-box		
	S	**R**	**S↓**	**R↓**	**A↑**	**S↓**	**R↓**	**A↑**
Adaptive-only	84.2	1473.5	57.4	1182.6	40.5	53.8	1243.1	44.1
Hybrid Templates	83.1	1436.2	32.6	963.7	63.5	30.3	982.0	68.2
Full Attack (Ours)	86.1	1456.4	12.4	437.4	81.6	19.6	440.2	80.1

in which the attack targets are constructed exclusively from the malicious templates designed by us. It does not include the adaptively learned templates. Specifically, Full Attack uses the designed hold, jitter, retreat, and stop templates together with the corresponding template-selection mechanism and attack optimization procedure.

D.4 Analysis of Results

As shown in Table 4, all three attack configurations substantially degrade the victim's performance, but their attack strengths differ. Adaptive-only, which uses only adaptively learned malicious templates, achieves ASRs of 40.5% and 44.1% under the white-box and black-box settings, respectively. This demonstrates that adaptive templates can effectively exploit the planner's valuation sensitivity.

Hybrid Templates combines adaptive templates with manually designed templates and increases the white-box and black-box ASRs to 63.5% and 68.2%, respectively. This indicates that manually specified behavioral priors improve the attack's ability to induce abnormal behaviors such as stopping, jittering, and retreating.

Importantly, Full Attack (Ours) uses only the manually designed hold, jitter, retreat, and stop templates proposed in this work; it does not include the adaptively learned templates. Full Attack achieves the strongest results, with white-box and black-box ASRs of 81.6% and 80.1%, while reducing task success to 12.4% and 19.6%, respectively. Compared with Hybrid Templates, it improves the white-box and black-box ASRs by 18.1 and 11.9 percentage points, showing that our designed templates can manipulate trajectory selection more consistently than the hybrid configuration.

Full Attack also preserves strong clean performance, achieving a task success rate of 86.1% and a cumulative reward of 1456.4. This confirms that the attack templates do not affect normal planning under clean inputs. Overall, adaptive templates can discover effective attack behaviors, whereas the manually designed templates proposed in this work provide clearer behavioral priors and stronger attack performance.