DIAGNOSING JEPA WORLD MODELS WITH ACTION-CONDITIONED PREDICTIVE CONSISTENCY

Anonymous authors
Paper under double-blind review

ABSTRACT

Joint-embedding predictive architectures (JEPAs) learn world models that predict in a compact latent space rather than in pixels, reducing the pressure to model nuisance appearance. Yet this provides no guarantee against visual perturbations: they can still alter the encoded representation and affect subsequent action-conditioned predictions. Bisimulation captures this requirement precisely: two observations should be treated as the same state only when their action-conditioned consequences agree. Guided by this criterion, we introduce Action-Conditioned Predictive Consistency (ACPC), a diagnostic that measures how far a clean history and a visually perturbed view of it diverge after being rolled forward under the same action sequence. We prove that this divergence bounds perturbation-induced changes in multi-step prediction error and planner cost, i.e., the change in predicted cost caused by a visual perturbation under the same action sequence. Building on pairwise ACPC, we define two complementary measures: Invariance Radius (IR) summarizes clean–perturbed rollout spread, while Separation Rate (SR) checks whether different states remain distinguishable after rollout. Experiments on four visual control tasks show that pairwise ACPC predicts perturbation-induced changes in prediction error and planner cost. On LeWM, the joint IR–SR screen transfers across tasks, and both IR and SR remain informative under blur and resize. PLDM exhibits similar diagnostic trends under a different architecture.

1 INTRODUCTION

World models support control by predicting the outcomes of candidate actions (Hafner et al., 2025; Hansen et al., 2024). Joint-embedding predictive architectures (JEPAs) predict target representations in a latent space rather than reconstructing pixels, avoiding an explicit requirement to reproduce nuisance visual details (LeCun, 2022; Assran et al., 2023; Bardes et al., 2024; Van Assel et al., 2025; Littwin et al., 2024). Yet this design alone does not determine which visual differences matter for control. Such representations should be insensitive to task-irrelevant visual perturbations while preserving distinctions between situations that require different actions. This requirement echoes the intuition behind bisimulation, which defines state equivalence through action-conditioned consequences rather than visual appearance (Gelada et al., 2019; Zhang et al., 2021). However, initial representation distances alone do not show how a perturbation propagates through prediction. Over a multi-step rollout, the predictor may amplify or contract the initial representation difference. Encoder-level and single-step diagnostics do not directly measure this rollout-level effect. We therefore ask: when a clean history and its perturbed view are rolled forward under the same action sequence, how far apart are their predicted trajectories?

To measure this rollout-level effect, we introduce Action-Conditioned Predictive Consistency (ACPC). ACPC rolls a clean history and its perturbed view forward under the same action sequence and measures the distance between their predicted trajectories. Figure 1 provides a conceptual overview of how this pairwise distance is summarized by *Invariance Radius* (IR) and complemented by *Separation Rate* (SR). Low ACPC alone, however, does not rule out representational collapse: a constant representation would make every paired rollout identical. We therefore summarize ACPC values across histories with IR, where lower values indicate less sensitivity to visual perturbations. SR checks whether different states remain distinguishable after being rolled forward, with higher values indicating better separation.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

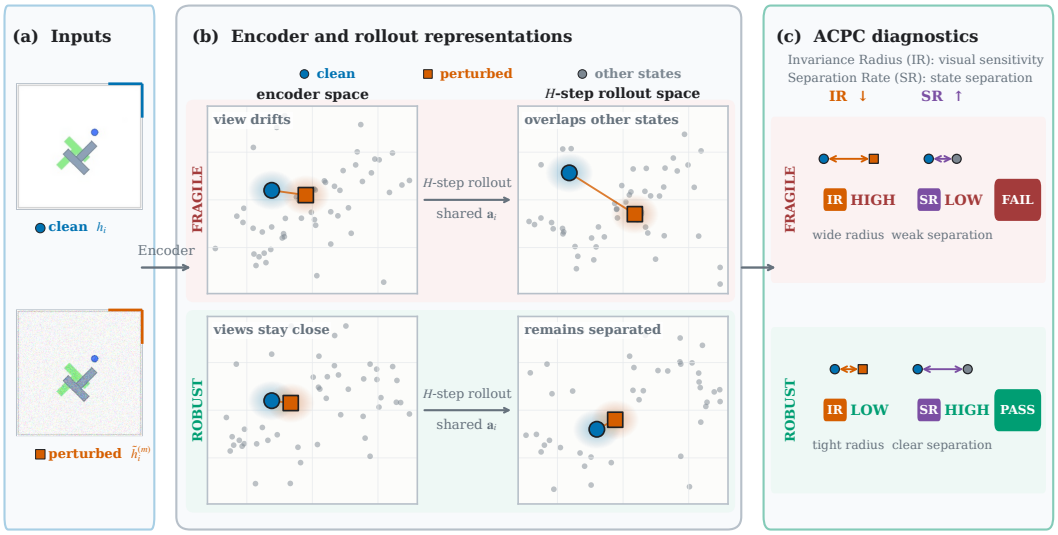


Figure 1: **Conceptual overview of ACPC, IR, and SR. (a)** A logged clean history and a visually perturbed view of it form the paired inputs. **(b)** A frozen world model encodes both views and rolls them forward for H steps under the same recorded action sequence. ACPC measures the distance between the two predicted trajectories. **(c)** IR summarizes perturbation-induced rollout spread across histories, while SR measures how often selected different-state rollouts remain separated beyond this spread.

We make three contributions. First, we introduce **ACPC**, which compares clean and perturbed rollout predictions under shared actions; IR summarizes sensitivity and SR checks state separation (Section 3). Second, we prove samplewise **bounds on prediction-error and planning-cost changes** over multi-step rollouts in which the same candidate action sequences are applied to both views (Propositions 1 and 2). These bounds require no distributional or smoothness assumptions. Third, we develop **ACPC-based checkpoint screening** using both IR and SR and evaluate the three proposed diagnostics on four tasks, three visual perturbations, and two architectures. On LeWM (Maes et al., 2026), ACPC is informative about perturbation-induced changes in prediction error and in the action sequences selected by the cross-entropy method (CEM (Kroese et al., 2006)). Across checkpoints, IR and SR provide non-equivalent views of perturbation sensitivity and state separation; their continuous values covary with retained control performance (Section 4).

2 Related Work

World Models. World models predict how an environment evolves under actions and use those predictions for control. DreamerV3 (Hafner et al., 2025) learns from imagined trajectories, while TD-MPC2 (Hansen et al., 2024) plans with learned latent dynamics. JEPAs learn representations by predicting targets in a latent space rather than reconstructing pixels (LeCun, 2022; Assran et al., 2023; Bardes et al., 2024; Assran et al., 2025). For goal-conditioned planning, PLDM (Sobal et al., 2025) learns action-conditioned latent dynamics from reward-free offline data using VICReg-inspired regularization. LeWM (Maes et al., 2026) is an end-to-end JEPA world model for action-conditioned latent prediction with a simple two-term objective: a next-step latent prediction loss and LeJEPA's SIGReg (Balestriero & LeCun, 2025), which serves as the sole anti-collapse regularizer. Related work studies collapse in self-predictive learning (Tang et al., 2023). VICReg (Bardes et al., 2022) prevents representation collapse with variance regularization and reduces feature redundancy with covariance regularization. VISReg (Wu et al., 2026) retains variance regularization but replaces VICReg's covariance regularization with Sliced-Wasserstein-based sketching. Complementary work studies the role of action conditioning (Khetarpal et al., 2025). Theoretical analyses ask which features latent prediction retains (Van Assel et al., 2025; Littwin et al., 2024). seq-JEPA (Ghaemi et al., 2025) further studies invariance and equivariance under known view transformations.

Robustness to Visual Perturbations. A common approach is to make learned representations insensitive to nuisance appearance changes during training. DrQ (Yarats et al., 2021) and DrQ-v2 (Yarats et al., 2022) use image augmentation to reduce visual overfitting in pixel-based control, while SODA (Hansen & Wang, 2021) decouples augmentation from policy learning and uses it to train invariant visual representations. ReOI (Chen et al., 2026) instead intervenes at test time, removing detected visual distractors before world-model prediction and then reintroducing them after prediction. Other methods modify what the world model is encouraged to represent. DreamerPro (Deng et al., 2022) replaces pixel reconstruction with prototypical representation learning to reduce sensitivity to background distractors. Denoised MDPs (Wang et al., 2022) explicitly separate control- and reward-relevant information from nuisance factors, while Sun et al. (2024) combine spatio-temporal masking with bisimulation-based representation learning to suppress exogenous visual noise. These approaches reflect a common principle: reducing the influence of visual differences that do not affect control while preserving distinctions that do. Bisimulation formalizes this principle by treating states as behaviorally similar when the same actions lead to similar rewards and transition dynamics (Gelada et al., 2019; Zhang et al., 2021; Shimizu & Tomizuka, 2025). Recent JEPA-based work adapts this idea to reward-free planning by defining state similarity from transition dynamics alone (Toso et al., 2026). In a related training-time approach, MWM (Yan et al., 2026) uses post-training to improve action-conditioned consistency over multi-step world-model predictions.

Diagnostics for World Models. Model accuracy does not necessarily predict control performance: one-step likelihood can be poorly correlated with downstream performance (Lambert et al., 2020). This objective mismatch has motivated decision-aware approaches to model learning and evaluation (Wei et al., 2024). More broadly, world-model evaluation should match the model's intended decision-making use rather than rely on prediction fidelity alone (Yu et al., 2026). Recent diagnostics examine different aspects of decision-relevant world-model behavior. ATM (Chen, 2026) compares action information in encoded environment transitions and model-predicted transitions. Inverse-dynamics probes measure how well actions can be recovered from learned latents (Yeom et al., 2026). At the rollout level, Schäfer et al. (2026) test whether long imagined rollouts respond to changes in physical dynamics rather than merely preserve kinematic patterns. Some methods compare predicted multi-step latent transitions with those produced by the environment (Vakalis, 2026), or compare predicted and true plan costs (You et al., 2026). Other work evaluates robustness through visual attacks. WMAttack (Guo et al., 2026) automatically searches for attacks on closed-loop world-model agents, while ARB4WM (Zhang et al., 2026) evaluates attacks targeting policy, value, and latent dynamics. Complementary reinforcement learning work provides formal robustness certificates: CARRL (Lütjens et al., 2020) certifies Q-based action choices under bounded observation perturbations, while CROP (Wu et al., 2022) certifies per-state actions and lower bounds on finite-horizon cumulative reward. Some existing diagnostics evaluate multi-step prediction fidelity, planning-cost accuracy, or robustness under visual attacks. ACPC instead measures perturbation-induced predictive drift through clean–perturbed rollout comparison and relates this drift to changes in prediction error and planner cost.

3 Action-Conditioned Predictive Consistency as a Diagnostic

A visual perturbation can change a world model's predicted trajectory. ACPC measures this change by rolling a clean history and its perturbed version forward under the same action sequence and comparing the two trajectories. We then show that it bounds perturbation-induced changes in prediction error and planner cost. To summarize ACPC across a checkpoint, IR summarizes the clean–perturbed rollout spread, while SR measures whether rollouts from different states remain distinguishable. SR can additionally serve as a check against representational collapse. We then combine relative IR and SR into a checkpoint screen whose thresholds are calibrated on source tasks and transferred unchanged to held-out tasks.

3.1 Pairwise ACPC

Given a clean history and its perturbed version, ACPC compares the rollouts predicted from them under the same action sequence. Let h denote the clean history and let $\tilde{h}$ be the result of applying the evaluated visual perturbation to h. The frozen encoder maps them to $z = E_\theta(h)$ and $\tilde{z} = E_\theta(\tilde{h})$.

Let $\mathbf{a} = (a_0, \ldots, a_{H-1})$ denote an action sequence of rollout horizon H, and let F_θ denote the frozen action-conditioned predictor. For each prediction step $k = 1, \ldots, H$, the two representations rolled forward under the first k actions are

$$\hat{z}_k = F_\theta^k(z, \mathbf{a}_{0:k-1}), \qquad \hat{\tilde{z}}_k = F_\theta^k(\tilde{z}, \mathbf{a}_{0:k-1}). \tag{1}$$

We compare these predictions in the projected latent space used to evaluate planner costs. Let Π denote the corresponding projection. To represent the full H-step rollout with a single vector, we assign nonnegative weights α_k, with $\sum_{k=1}^{H} \alpha_k = 1$, and concatenate the weighted projected predictions:

$$\bar{G}_{\mathbf{a}}(z) = \left[\sqrt{\alpha_1}\Pi(\hat{z}_1)^\top \quad \cdots \quad \sqrt{\alpha_H}\Pi(\hat{z}_H)^\top\right]^\top. \tag{2}$$

Action-Conditioned Predictive Consistency (ACPC) is the distance between the two weighted predicted rollouts:

$$\mathrm{ACPC}_H(h, \tilde{h}, \mathbf{a}) = \left\|\bar{G}_{\mathbf{a}}(E_\theta(h)) - \bar{G}_{\mathbf{a}}(E_\theta(\tilde{h}))\right\|_2 = \left(\sum_{k=1}^{H} \alpha_k \left\|\Pi(\hat{z}_k) - \Pi(\hat{\tilde{z}}_k)\right\|_2^2\right)^{1/2}. \tag{3}$$

Low ACPC means that the two predicted trajectories remain close; high ACPC means that they diverge. Encoder shift $\|z - \tilde{z}\|_2$ measures the difference before prediction, whereas ACPC measures predictive divergence over the rollout. Using the same action sequence holds the applied actions fixed in the comparison. ACPC measures rollout consistency, not prediction accuracy.

3.2 Prediction-Error Bounds

ACPC itself does not require access to the observed future. To relate ACPC to prediction error, let $Y_{\mathbf{a}}^H$ denote the observed future associated with action sequence $\mathbf{a}$, represented in the same weighted projected space as $\bar{G}_{\mathbf{a}}$. We define prediction error as the distance between a predicted rollout and $Y_{\mathbf{a}}^H$. Each clean–perturbed pair is constructed from the same underlying environment trajectory. Accordingly, both predictions are evaluated against the same observed future.

Proposition 1 (Prediction-error change bound)**.** *Define*

$$e_h = \left\|\bar{G}_{\mathbf{a}}(E_\theta(h)) - Y_{\mathbf{a}}^H\right\|_2, \qquad e_{\tilde{h}} = \left\|\bar{G}_{\mathbf{a}}(E_\theta(\tilde{h})) - Y_{\mathbf{a}}^H\right\|_2.$$

Then every paired sample satisfies

$$|e_{\tilde{h}} - e_h| \le \mathrm{ACPC}_H(h, \tilde{h}, \mathbf{a}). \tag{4}$$

This result follows directly from the reverse triangle inequality; the proof is given in Appendix A. The bound concerns the change in error, not absolute prediction accuracy. Both rollouts may therefore be inaccurate even when ACPC is small. In our experiments, the observed future is used only to evaluate the prediction-error change and is not required to compute ACPC (Section 4.4).

3.3 Planner-Cost Sensitivity

The previous result shows that ACPC bounds perturbation-induced changes in prediction error. We next ask whether the rollout divergence measured by ACPC also bounds changes in planner cost. In goal-conditioned planning, the planner ranks candidate action sequences by the predicted costs of their final states. A visual perturbation can shift these predicted endpoints and thereby change the candidate costs.

For each candidate action sequence $\mathbf{a}^j$, we roll the clean and perturbed histories forward under the same candidate sequence. At the final rollout step, we map the two predicted representations into the planner's latent space using Π:

$$x_j = \Pi(F_\theta^H(E_\theta(h), \mathbf{a}^j)), \qquad \tilde{x}_j = \Pi(F_\theta^H(E_\theta(\tilde{h}), \mathbf{a}^j)).$$

Thus, x_j and $\tilde{x}_j$ are the clean and perturbed versions of the H-th projected prediction appearing in the candidate-specific ACPC rollout.

The planner evaluates candidate j by how close its predicted endpoint is to a fixed goal embedding g. The same candidate therefore receives the clean and perturbed planner costs

$$C_j = \|x_j - g\|_2^2, \qquad \tilde{C}_j = \|\tilde{x}_j - g\|_2^2.$$

The difference $|\tilde{C}_j - C_j|$ measures how much the visual perturbation changes the planner's predicted cost for candidate j.

Proposition 2 (Planner-cost sensitivity). *Let* $A_j = \mathrm{ACPC}_H(h, \tilde{h}, \mathbf{a}^j)$. *If* $\alpha_H > 0$, *then*

$$|\tilde{C}_j - C_j| \le 2\sqrt{C_j}\frac{A_j}{\sqrt{\alpha_H}} + \frac{A_j^2}{\alpha_H}. \tag{5}$$

This result connects ACPC to changes in the planner's predicted cost under the same action sequence. The proof is given in Appendix A.

3.4 Checkpoint-Level IR and SR

Pairwise ACPC measures one clean–perturbed pair under one action sequence. To summarize perturbation sensitivity across a checkpoint, we define the *Invariance Radius* (IR), while the *Separation Rate* (SR) measures whether different states remain distinguishable after rollout.

To compute IR, we select n history windows from the logged data as anchors. Anchor i provides a clean history h_i and its recorded action sequence $\mathbf{a}_i$. We apply the visual perturbation M times to obtain $\tilde{h}_i^{(1)}, \ldots, \tilde{h}_i^{(M)}$ and compute ACPC for each clean–perturbed pair. Because latent distances can have different local scales across histories, we normalize ACPC by s_i, the median projected one-step clean motion over the next H transitions:

$$R_i^{(m)} = \frac{\mathrm{ACPC}_H(h_i, \tilde{h}_i^{(m)}, \mathbf{a}_i)}{s_i + \varepsilon_{\mathrm{norm}}}, \qquad \bar{R}_i = \frac{1}{M}\sum_{m=1}^{M} R_i^{(m)}. \tag{6}$$

The raw IR is the empirical q-quantile of these anchor-level sensitivities:

$$\mathrm{IR}_q^{\mathrm{raw}}(\theta) = Q_q\big(\{\bar{R}_i\}_{i=1}^n\big). \tag{7}$$

Low IR alone is insufficient, since a collapsed representation could also make clean and perturbed rollouts similar. SR therefore compares this perturbation-induced spread with the separation between different states. For each eligible anchor i, we select a nearby logged history h_i^{diff} with a different state-coordinate label and roll both h_i and h_i^{diff} forward under the anchor's recorded action sequence $\mathbf{a}_i$. Their normalized rollout distance is

$$D_i^{\mathrm{diff}} = \frac{\mathrm{ACPC}_H(h_i, h_i^{\mathrm{diff}}, \mathbf{a}_i)}{s_i + \varepsilon_{\mathrm{norm}}}. \tag{8}$$

Let $\mathcal{I}$ be the set of anchors for which such a different-state comparison is available. SR is the fraction of these pairs whose rollout distance exceeds raw IR by a margin δ:

$$\mathrm{SR}_{q,\delta}(\theta) = \frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \mathbf{1}\big[D_i^{\mathrm{diff}} > \mathrm{IR}_q^{\mathrm{raw}}(\theta) + \delta\big]. \tag{9}$$

We use $q = 0.90$ and $\delta = 0.10$. Lower IR indicates less sensitivity to the visual perturbation, while higher SR indicates that different states remain separated beyond the perturbation-induced spread. A collapsed representation can achieve low IR but will fail SR.

3.5 Checkpoint Screening

SR uses raw IR because it compares different-state separation with the perturbation radius within the same checkpoint. For comparisons across checkpoints, we instead normalize IR by a matched unaugmented reference. Let θ_0 denote the corresponding unaugmented checkpoint from the same task and model family under a matched training setup. We define

$$\mathrm{IR}_q^{\mathrm{rel}}(\theta; \theta_0) = \frac{\mathrm{IR}_q^{\mathrm{raw}}(\theta)}{\mathrm{IR}_q^{\mathrm{raw}}(\theta_0)}. \tag{10}$$

By construction, $\mathrm{IR}_q^{\mathrm{rel}}(\theta_0;\theta_0) = 1$, while values below 1 indicate lower perturbation sensitivity than the reference checkpoint.

We screen checkpoints jointly with relative IR and SR. A checkpoint passes if

$$\mathrm{IR}_q^{\mathrm{rel}}(\theta;\theta_0) \le \tau_{\mathrm{IR}} \qquad \text{and} \qquad \mathrm{SR}_{q,\delta}(\theta) \ge \tau_{\mathrm{SR}}.$$

This is a Boolean screen rather than a continuous ranking. We calibrate τ_{IR} and τ_{SR} using planning success on a set of source tasks, then keep both thresholds fixed when evaluating the remaining tasks. Planning success is used only for calibration and is not an input to the screen after the thresholds are fixed. Unlike the pairwise bounds above, this checkpoint-level screening rule is empirical.

4 Experiments

We evaluate the diagnostics at two levels. At the pair level, we test whether ACPC provides information about perturbation-induced changes in prediction error and planner cost. At the checkpoint level, we test whether IR and SR track control performance and whether this relationship transfers across tasks, evaluation noise levels, model architectures, and visual shifts.

4.1 Evaluation Protocol

Models, Tasks, and Evaluation. We evaluate four control tasks: TwoRoom navigation, PushT planar manipulation, Reacher arm control, and OGBench-Cube 3D manipulation. CEM plans five-step action sequences with the frozen world model, and performance is measured by the percentage of episodes that reach the task goal.

We use training-time Gaussian-noise augmentation as the main controlled axis for producing checkpoints with different responses to visual perturbations. For the main sweep, $\sigma_{\max}$ sets the upper bound of the training noise: each frame is perturbed with $\sigma_{\mathrm{train}} \sim \mathrm{Uniform}(0, \sigma_{\max})$. We compare one unaugmented checkpoint with eight augmented checkpoints at $\sigma_{\max} \in \{0.01, \ldots, 0.08\}$. At evaluation time, all checkpoints are tested with fixed noise $\sigma_{\mathrm{eval}} = 0.08$, while the goal image remains clean.

Diagnostic Protocol. We compute eight-step ACPC with uniform horizon weights and summarize checkpoint-level sensitivity using IR at the 90th percentile and SR as defined in Section 3.4. Blur and resize provide additional off-axis perturbations using the same diagnostic protocol. Detailed sampling settings and perturbation parameters are provided in the appendix.

Broad-Severity Protocol. We additionally vary both the training range $\sigma_{\max} \in \{0.1, 0.2, 0.3, 0.4, 0.5, 0.7, 1.0, 1.5\}$ and the evaluation severity $\sigma_{\mathrm{eval}} \in \{0.08, 0.16, 0.32, 0.64\}$ to test whether the diagnostics remain informative beyond the main setting. The diagnostic definition, normalization, and screening thresholds remain unchanged. Each task–condition pair uses three independent training runs, with each checkpoint evaluated using three seeds and 100 episodes per seed; the training run is the unit of replication.

Threshold Protocol. Let S_0 denote the success rate of the unaugmented checkpoint and $S_{\max}$ the highest success rate in the fixed nine-checkpoint sweep. A checkpoint with success rate S meets the robustness criterion if

$$S \ge S_0 + 0.8(S_{\max} - S_0).$$

We additionally require that its clean-observation success rate decreases by no more than five percentage points relative to the unaugmented checkpoint.

4.2 Gaussian-Noise Performance and IR–SR Diagnostics

Planning success varies with the training noise level $\sigma_{\max}$, and the best-performing range differs across tasks (Figure 2). Reacher is a special case: augmentation also increases clean success from 59.2% to 76–82%, so its higher success under noisy evaluation cannot be attributed to robustness alone.

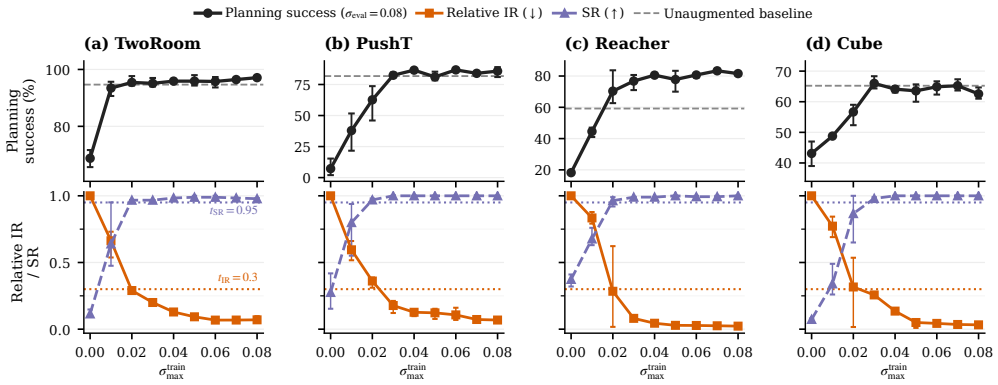


Figure 2: Checkpoint-level planning performance and diagnostics across the complete Gaussian-noise training sweep. Points are across-run means; error bars span the three training runs, and success-rate axes are scaled per task. Lower IR and higher SR are favorable.

Across tasks, every augmentation level that meets the success-rate criterion has lower IR and higher SR than its matched unaugmented reference, although neither diagnostic changes monotonically with $\sigma_{\max}$. Across the 108 LeWM checkpoint rows, 77 pass the IR threshold $\tau_{\mathrm{IR}} = 0.3$. Every checkpoint that passes the IR threshold also passes the SR threshold $\tau_{\mathrm{SR}} = 0.95$.

The evaluated PLDM sweeps show a similar qualitative pattern: better control performance is associated with lower IR and higher SR (Appendix C). Both diagnostics also remain informative under the tested blur and resize shifts (Appendix G).

4.3 IR and SR at Broader Evaluation Noise Levels

We use the broad-severity experiment to test whether checkpoints that pass the relative-IR threshold also preserve separation between different-state rollouts.

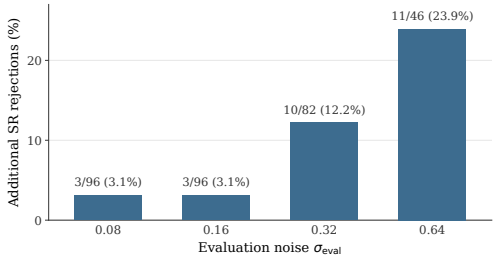


Figure 3: **SR provides a distinct fixed-threshold constraint beyond relative IR.** Among checkpoints satisfying $\mathrm{IR}^{\mathrm{rel}} \leq 0.3$, each bar shows the percentage rejected by $\mathrm{SR} < 0.95$; annotations give the rejection count and the number passing IR.

Among checkpoints that pass the relative-IR threshold, SR rejects 3/96 at $\sigma_{\mathrm{eval}} = 0.08$ and 3/96 at $\sigma_{\mathrm{eval}} = 0.16$. The rejection count increases to 10/82 and 11/46 at the two larger evaluation noise levels (Figure 3; Appendix D). Thus, passing IR alone does not guarantee preserved state separation, especially as evaluation noise increases; SR provides an additional check on this property.

4.4 ACPC and Prediction-Error Change

We test whether eight-step ACPC under the recorded actions helps explain the perturbation-induced prediction-error change $d = |e_{\tilde{h}} - e_h|$. We use $H = 8$, the longest horizon with a complete logged future for every anchor, and analyze the unaugmented checkpoint from each task and training run.

We estimate d with ridge regression using 16-fold group cross-validation: 15 trajectory groups are used for training and the remaining group for testing, with all samples from the same trajectory kept in one split (Appendix E).

The Base model uses perturbation severity, encoder distance, and one-step ACPC as the covariates for regression. Base+Control$_8$ additionally uses the best-performing eight-step ACPC computed with zeroed, swapped, or temporally shuffled actions. Base+ACPC$_8$ instead uses eight-step ACPC under the recorded actions. These comparisons test whether multi-step ACPC adds information beyond simpler diagnostics and whether that information depends on the recorded action sequence.

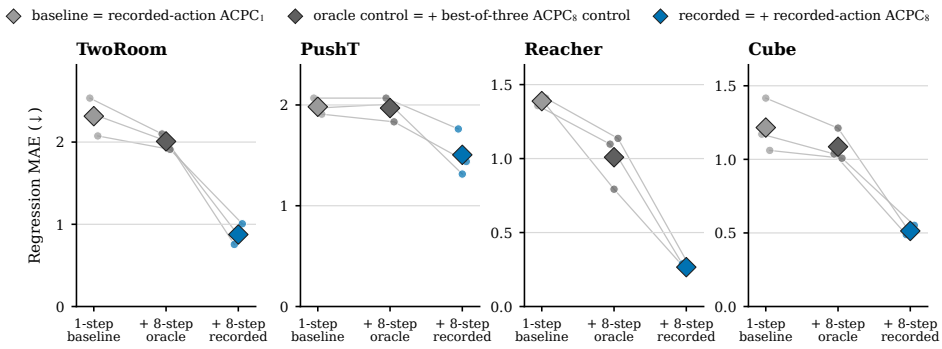


Figure 4: Cross-validated MAE for predicting the error drift d on the unaugmented checkpoints; lower is better.

Base+ACPC$_8$ achieves the lowest cross-validated MAE in all 12 task–run cells (Figure 4). Its MAE is $55.9 \pm 4.7\%$ lower than Base and $51.3 \pm 3.5\%$ lower than Base+Control$_8$ (mean $\pm$ sample standard deviation across the three runs). These results show that eight-step ACPC under the recorded actions provides information about prediction-error change beyond both the simpler diagnostics and the tested destroyed-action controls.

4.5 ACPC and the Cost of CEM Plan Changes

A visual perturbation may cause CEM to select a different action sequence, but a plan change alone does not indicate how consequential that change is. We therefore test whether planner-horizon ACPC helps predict the additional planner cost associated with selecting the perturbed-history plan.

Let $\mathbf{a}$ and $\tilde{\mathbf{a}}$ denote the final plans selected from the clean and perturbed histories. We measure $(C_h(\tilde{\mathbf{a}}) - C_h(\mathbf{a}))_+$, the extra cost assigned by the clean model to the perturbed-history plan, and refer to it as *CEM selection regret*, where $(x)_+ = \max(x, 0)$. This is a single-decision model cost, not cumulative regret or simulator return.

We fit two ridge regression models to predict $\log(1 + \text{selection regret})$. The baseline regression uses perturbation severity, training condition, one-step ACPC, and the clean cost gap between the best and second-best candidates. The expanded regression adds five-step ACPC, which measures perturbation-induced divergence over the full CEM planning horizon. For each candidate pool, ACPC is computed for every candidate action sequence and summarized by its 90th percentile. Within each training run, both models are trained on three tasks and tested on the held-out task, rotating across all four tasks. Appendix I gives the CEM protocol, budget, and sample counts.

Adding five-step ACPC lowers cross-task test MAE by 15.2%, with lower error in all 12 task–run test cases (Figure 5). Thus, planner-horizon ACPC provides information about the cost of perturbation-induced plan changes beyond immediate perturbation sensitivity and the clean candidate-cost gap.

4.6 Cross-Task Checkpoint Screening

We select the IR–SR thresholds using a subset of tasks (source tasks) and apply them unchanged to the remaining tasks (held-out tasks). For each held-out task, checkpoints are ordered by training noise level $\sigma_{\max}$. We compare the lowest $\sigma_{\max}$ at which the screen passes with the lowest $\sigma_{\max}$ at

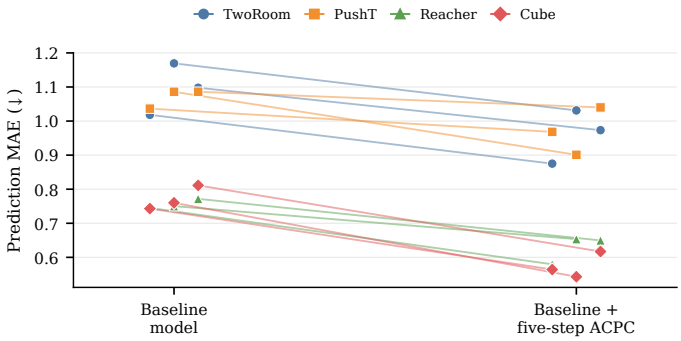


Figure 5: Adding planner-horizon ACPC improves prediction of the extra model cost associated with changes in the plan selected by adaptive CEM.

which planning performance meets the success-rate criterion. With two or three source tasks, these two points differ by 0.5 grid steps on average and by at most one step, where one step corresponds to $\Delta\sigma_{\max} = 0.01$.

We also evaluate the screen as a binary classifier over the held-out checkpoints, where passing the success-rate criterion defines the positive class. The screen achieves balanced accuracy 0.900, precision 0.913, and recall 0.953; balanced accuracy averages the correct-classification rates for checkpoints that do and do not meet the success-rate criterion. Full results are reported in Table 9 in Appendix F.

5 Discussion and Limitations

Scope and limitations. ACPC diagnoses how visual perturbations change predicted rollouts and planner costs; it does not modify CEM. Our planner experiment therefore tests whether ACPC reflects the extra predicted cost of a perturbation-induced plan change, not whether ACPC-guided planning improves control. The IR–SR screen has additional requirements: relative IR needs an unaugmented reference, IR normalization uses observed clean transitions, and SR requires state labels to form different-state pairs. SR is therefore limited to the tested state distinctions, and clean performance must be checked separately. Gaussian noise is evaluated at multiple levels, whereas blur and resize are tested at only one level each. We also do not evaluate natural changes in background, lighting, camera pose, or occlusion. The selected IR threshold is at the boundary of the tested range, so larger values remain untested.

6 Conclusion

ACPC measures how much a clean observation and its visually perturbed counterpart diverge after both are rolled forward under the same action sequence. We show that this divergence bounds changes in multi-step prediction error and predicted planning costs. IR and SR serve complementary diagnostic roles: IR measures clean–perturbed rollout spread, while SR checks whether different states remain distinguishable after rollout. On LeWM, the IR–SR checkpoint screen transfers across tasks, and both diagnostics remain informative under blur and resize. Across the evaluated checkpoints, LeWM and PLDM show similar trends in both IR and SR.

AI USE STATEMENT

Generative AI tools were used to assist with language editing and polishing, information retrieval, code generation, and visualization and analysis of experimental results. The authors reviewed all AI-assisted outputs and checked the code and analyses for correctness. The authors take full responsibility for the final content of the submission.

REPRODUCIBILITY STATEMENT

The appendix specifies tasks, perturbations, checkpoints, seeds, sampling, normalization, thresholds, and aggregation. The anonymous supplementary material includes the code and machine-readable results needed to reproduce the reported figures and tables.

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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 15619–15629, 2023. doi: 10.1109/CVPR52729.2023.01499.

Mido Assran, Adrien Bardes, David Fan, Quentin Garrido, Russell Howes, Mojtaba Komeili, Matthew Muckley, Ammar Rizvi, Claire Roberts, Koustuv Sinha, Artem Zholus, Sergio Arnaud, Abha Gejji, Ada Martin, Francois Robert Hogan, Daniel Dugas, Piotr Bojanowski, Vasil Khalidov, Patrick Labatut, Francisco Massa, Marc Szafraniec, Kapil Krishnakumar, Yong Li, Xiaodong Ma, Sarath Chandar, Franziska Meier, Yann LeCun, Michael Rabbat, and Nicolas Ballas. V-JEPA 2: Self-supervised video models enable understanding, prediction and planning. arXiv preprint arXiv:2506.09985, 2025. URL https://arxiv.org/abs/2506.09985.

Randall Balestriero and Yann LeCun. LeJEPA: Provable and scalable self-supervised learning without the heuristics. arXiv preprint arXiv:2511.08544, 2025. URL https://arxiv.org/abs/2511.08544.

Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-invariance-covariance regularization for self-supervised learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=xm6YD62D1Ub.

Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. Revisiting feature prediction for learning visual representations from video. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=QaCCuDfBk2.

Jiaheng Chen. ATM: Action-consistency transfer matrix for diagnosing and improving latent world models. arXiv preprint arXiv:2606.09028, 2026. URL https://arxiv.org/abs/2606.09028.

Yuxin Chen, Jianglan Wei, Chenfeng Xu, Boyi Li, Masayoshi Tomizuka, Andrea Bajcsy, and Ran Tian. Reimagination with test-time observation interventions: Distractor-robust world model predictions for visual model predictive control. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2026. URL https://re-oi.github.io/.

Fei Deng, Ingook Jang, and Sungjin Ahn. DreamerPro: Reconstruction-free model-based reinforcement learning with prototypical representations. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 4956–4975. PMLR, 2022. URL https://proceedings.mlr.press/v162/deng22a.html.

Carles Gelada, Saurabh Kumar, Jacob Buckman, Ofir Nachum, and Marc G. Bellemare. DeepMDP: Learning continuous latent space models for representation learning. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2170–2179. PMLR, 2019. URL https://proceedings.mlr.press/v97/gelada19a.html.

Hafez Ghaemi, Eilif B. Muller, and Shahab Bakhtiari. seq-JEPA: Autoregressive predictive learning of invariant-equivariant world models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, pp. 32943–32973, 2025. doi: 10.52202/085713-1104.

Zhixiang Guo, Siyuan Liang, Shi Fu, Cheng Guo, András Balogh, Márk Jelasity, and Dacheng Tao. WMAttack: Automated attack search for adversarial evaluation of world-model agents. arXiv preprint arXiv:2605.23220, 2026. URL https://arxiv.org/abs/2605.23220.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, 2025. doi: 10.1038/s41586-025-08744-2.

Nicklas Hansen and Xiaolong Wang. Generalization in reinforcement learning by soft data augmentation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13611–13617, 2021. doi: 10.1109/ICRA48506.2021.9561103.

Nicklas Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2024. URL https://openreview.net/forum?id=Oxh5CstDJU.

Michael F. Hutchinson. A stochastic estimator of the trace of the influence matrix for laplacian smoothing splines. *Communications in Statistics - Simulation and Computation*, 18(3):1059–1076, 1989. doi: 10.1080/03610918908812806.

Khimya Khetarpal, Zhaohan Daniel Guo, Bernardo Avila Pires, Yunhao Tang, Clare Lyle, Mark Rowland, Nicolas Heess, Diana L. Borsa, Arthur Guez, and Will Dabney. A unifying framework for action-conditional self-predictive reinforcement learning. In *Proceedings of the 28th International Conference on Artificial Intelligence and Statistics*, volume 258 of *Proceedings of Machine Learning Research*, pp. 181–189. PMLR, 2025. URL https://proceedings.mlr.press/v258/khetarpal25a.html.

Dirk P. Kroese, Sergey Porotsky, and Reuven Y. Rubinstein. The cross-entropy method for continuous multi-extremal optimization. *Methodology and Computing in Applied Probability*, 8(3):383–407, 2006. doi: 10.1007/s11009-006-9753-0.

Nathan Lambert, Brandon Amos, Omry Yadan, and Roberto Calandra. Objective mismatch in model-based reinforcement learning. In *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, volume 120 of *Proceedings of Machine Learning Research*, pp. 761–770. PMLR, 2020. URL https://proceedings.mlr.press/v120/lambert20a.html.

Yann LeCun. A path towards autonomous machine intelligence. OpenReview preprint, 2022. URL https://openreview.net/forum?id=BZ5a1r-kVsf. Version 0.9.2.

Etai Littwin, Omid Saremi, Madhu Advani, Vimal Thilak, Preetum Nakkiran, Chen Huang, and Joshua Susskind. How JEPA avoids noisy features: The implicit bias of deep linear self distillation networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 91300–91336, 2024. doi: 10.52202/079017-2897.

Björn Lütjens, Michael Everett, and Jonathan P. How. Certified adversarial robustness for deep reinforcement learning. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1328–1337. PMLR, 2020. URL https://proceedings.mlr.press/v100/lutjens20a.html.

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. URL https://arxiv.org/abs/2603.19312.

Salah Rifai, Pascal Vincent, Xavier Muller, Xavier Glorot, and Yoshua Bengio. Contractive auto-encoders: Explicit invariance during feature extraction. In Lise Getoor and Tobias Scheffer (eds.), *Proceedings of the 28th International Conference on Machine Learning*, pp. 833–840. Omnipress, 2011. URL https://icml.cc/2011/papers/455_icmlpaper.pdf.

Finn Rasmus Schäfer, Korbinian Moller, Yuan Gao, Christian Oefinger, Sebastian Schmidt, and Johannes Betz. Imagined rollouts are kinematic, not dynamic: A diagnosis of long-horizon world-model failure. arXiv preprint arXiv:2607.05966, 2026. URL https://arxiv.org/abs/2607.05966.

Yutaka Shimizu and Masayoshi Tomizuka. Bisimulation metric for model predictive control. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2025. URL https://openreview.net/forum?id=F07ic7huE3.

Uladzislau Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, Tim G. J. Rudner, and Yann LeCun. Learning from reward-free offline data: A case for planning with latent dynamics models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, pp. 43905–43941, 2025. doi: 10.52202/085713-1465.

Ruixiang Sun, Hongyu Zang, Xin Li, and Riashat Islam. Learning latent dynamic robust representations for world models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 47234–47260. PMLR, 2024. URL https://proceedings.mlr.press/v235/sun24n.html.

Yunhao Tang, Zhaohan Daniel Guo, Pierre Harvey Richemond, Bernardo Avila Pires, Yash Chandak, Rémi Munos, Mark Rowland, Mohammad Gheshlaghi Azar, Charline Le Lan, Clare Lyle, András György, Shantanu Thakoor, Will Dabney, Bilal Piot, Daniele Calandriello, and Michal Valko. Understanding self-predictive learning for reinforcement learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 33632–33656. PMLR, 2023. URL https://proceedings.mlr.press/v202/tang23d.html.

Leonardo F. Toso, Davit Shadunts, Yunyang Lu, Nihal Sharma, Donglin Zhan, Nam H. Nguyen, and James Anderson. Learning invariant visual representations for planning with joint-embedding predictive world models. arXiv preprint arXiv:2602.18639, 2026. URL https://arxiv.org/abs/2602.18639.

Donna Vakalis. Operator-on-F complements value-equivalence: A planning-time diagnostic for latent world models. arXiv preprint arXiv:2607.04464, 2026. URL https://arxiv.org/abs/2607.04464.

Hugues Van Assel, Mark Ibrahim, Tommaso Biancalani, Aviv Regev, and Randall Balestriero. Joint-embedding vs reconstruction: Provable benefits of latent space prediction for self-supervised learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, pp. 21897–21937, 2025. doi: 10.52202/085713-0740.

Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL https://www.jmlr.org/papers/v9/vandermaaten08a.html.

Tongzhou Wang, Simon S. Du, Antonio Torralba, Phillip Isola, Amy Zhang, and Yuandong Tian. Denoised MDPs: Learning world models better than the world itself. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 22591–22612. PMLR, 2022. URL https://proceedings.mlr.press/v162/wang22c.html.

Ran Wei, Nathan Lambert, Anthony D. McDonald, Alfredo Garcia, and Roberto Calandra. A unified view on solving objective mismatch in model-based reinforcement learning. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=tQVZgvXhZb.

Fan Wu, Linyi Li, Zijian Huang, Yevgeniy Vorobeychik, Ding Zhao, and Bo Li. CROP: Certifying robust policies for reinforcement learning through functional smoothing. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=HOjLHrlZhmx.

Haiyu Wu, Randall Balestriero, and Morgan Levine. VISReg: Variance-invariance-sketching regularization for JEPA training. arXiv preprint arXiv:2606.02572, 2026. URL https://arxiv.org/abs/2606.02572.

Han Yan, Zishang Xiang, Zeyu Zhang, and Hao Tang. MWM: Mobile world models for action-conditioned consistent prediction. arXiv preprint arXiv:2603.07799, 2026. URL https://arxiv.org/abs/2603.07799.

Denis Yarats, Ilya Kostrikov, and Rob Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=GY6-6sTvGaf.

Denis Yarats, Rob Fergus, Alessandro Lazaric, and Lerrel Pinto. Mastering visual continuous control: Improved data-augmented reinforcement learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2022. URL https://openreview.net/forum?id=_SJ-_yyes8.

Jewon Yeom, Hanseul Kim, Jeongjae Park, Sungmok Jung, Jaejin Lee, and Taesup Kim. What makes video world model latents action-relevant: Prediction over reconstruction. arXiv preprint arXiv:2606.07687, 2026. URL https://arxiv.org/abs/2606.07687.

Hanzhe You, Yonggang Zhang, Maohao Ran, Zhiqin Yang, Zhenyuan Zhang, Wei Xue, Jun Song, Xinmei Tian, and Yike Guo. A control theory of predictability in latent world models. arXiv preprint arXiv:2607.10362, 2026. URL https://arxiv.org/abs/2607.10362.

Yang Yu, Shiyuan Zhang, Yifei Sheng, Haoxiang Ren, and Haoxin Lin. How should world models be evaluated for embodied decision-making? a decision-making-centric position. arXiv preprint arXiv:2606.15032, 2026. URL https://arxiv.org/abs/2606.15032.

Amy Zhang, Rowan T. McAllister, Roberto Calandra, Yarin Gal, and Sergey Levine. Learning invariant representations for reinforcement learning without reconstruction. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2021. URL https://openreview.net/forum?id=-2FCwDKRREu.

Junjian Zhang, Hao Tan, Ruonan Li, Dong Zhu, Aiping Li, and Zhaoquan Gu. ARB4WM: An adversarial robustness benchmark for world models in continuous control. arXiv preprint arXiv:2606.16605, 2026. URL https://arxiv.org/abs/2606.16605.

A PROOFS

Proof of Proposition 1. Let $u = \bar{G}_{\mathbf{a}}(E_\theta(h))$, $v = \bar{G}_{\mathbf{a}}(E_\theta(\tilde{h}))$, and $y = Y^H_{\mathbf{a}}$. Both errors use the same target y in the weighted rollout space of Equation (3). The reverse triangle inequality gives

$$|e_{\tilde{h}} - e_h| = \big|\, \|v - y\|_2 - \|u - y\|_2 \,\big| \tag{11}$$

$$\leq \|(v - y) - (u - y)\|_2 = \|v - u\|_2 = \mathrm{ACPC}_H(h, \tilde{h}, \mathbf{a}). \tag{12}$$

The one-sided error increase satisfies the same bound because $(e_{\tilde{h}} - e_h)_+ \leq |e_{\tilde{h}} - e_h|$.

Proof of Proposition 2. For a single candidate, omit the index. Let $r = \|\tilde{x} - x\|_2$ and $A = \mathrm{ACPC}_H(h, \tilde{h}, \mathbf{a})$. Since r is the final-step displacement in ACPC, its definition gives

$$\alpha_H r^2 \leq A^2.$$

Thus, $r \leq A/\sqrt{\alpha_H}$. The difference between the two squared costs factors as

$$|\tilde{C} - C| = \big|\|\tilde{x} - g\|_2^2 - \|x - g\|_2^2\big| \tag{13}$$

$$= \big|\langle \tilde{x} - x, \tilde{x} + x - 2g\rangle\big| \tag{14}$$

$$\leq \|\tilde{x} - x\|_2 \, \|\tilde{x} + x - 2g\|_2 \tag{15}$$

$$\leq r\big(2\|x - g\|_2 + r\big) \tag{16}$$

$$\leq 2\sqrt{C}\,\frac{A}{\sqrt{\alpha_H}} + \frac{A^2}{\alpha_H}. \tag{17}$$

The first inequality follows from Cauchy–Schwarz, and the second follows from the triangle inequality. The last inequality uses $r \le A/\sqrt{\alpha_H}$ and $\|x - g\|_2 = \sqrt{C}$.

Tightness of the prediction-error bound. For the prediction-error bound, let u be a unit vector, set the common target to $Y = 0$, and let the two predicted rollouts be su and tu, where $s, t \ge 0$. Then

$$\big|\, \|tu\|_2 - \|su\|_2 \,\big| = \|tu - su\|_2,$$

so Equation (4) holds with equality.

B Local-Geometry Case-Study Definitions

This supplementary case study visualizes how augmentation contracts perturbed views relative to nearby clean histories. We use 128 logged PushT histories. For each history, we generate one clean view and 18 perturbed views, with six draws at each standard deviation in $\{0.01, 0.04, 0.08\}$. Each of the four qualitative t-SNE panels uses a separate deterministic projection (van der Maaten & Hinton, 2008), so coordinates are not compared across panels.

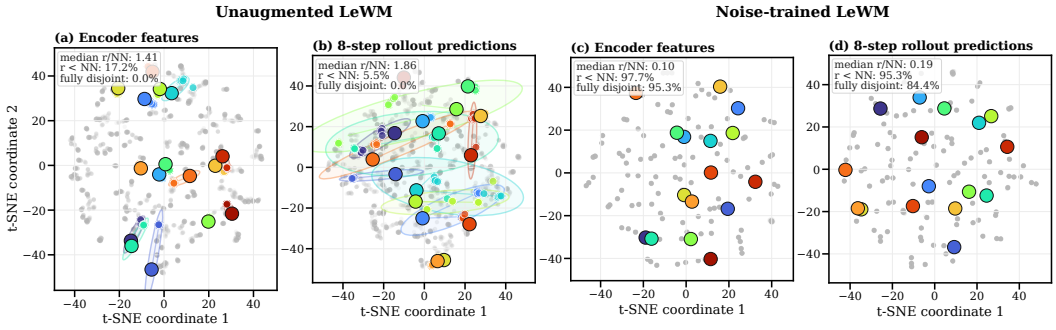


Figure 6: Local geometry for matched unaugmented (left) and $\sigma_{\max} = 0.08$ PushT checkpoints (right), at the encoder and after eight rollout steps. Markers show clean histories and perturbed views; summaries are computed in the original 192-dimensional space, not in t-SNE.

Without augmentation, median r/NN is 1.41 at the encoder and 1.86 after rollout, with no fully disjoint perturbation clouds. With augmentation, the ratios fall to 0.10 and 0.19, and the disjoint fractions rise to 95.3% and 84.4%. This is descriptive geometry, not evidence of task-semantic separation or planning robustness.

We evaluate two representation spaces: the encoder output, denoted by $\mathcal{V} = \mathrm{enc}$, and the representation after eight autoregressive rollout steps, denoted by $\mathcal{V} = \mathrm{roll}$. The rollout horizon is $H = 8$ (Section 3.1). For history i, let $v_{i,\mathcal{V}}^{(0)}$ be its clean representation and let $v_{i,\mathcal{V}}^{(m)}$, $m = 1, \dots, M$ with $M = 18$, be its perturbed representations. In the rollout space, the clean and perturbed views of the same history use the same recorded action sequence. We then define the sampled visual radius, the distance to the nearest other clean history, and their ratio:

$$r_{i,\mathcal{V}}^{\mathrm{vis}} = \max_{1 \le m \le M} \left\| v_{i,\mathcal{V}}^{(m)} - v_{i,\mathcal{V}}^{(0)} \right\|_2,$$

$$d_{i,\mathcal{V}}^{\mathrm{NN}} = \min_{j \ne i} \left\| v_{i,\mathcal{V}}^{(0)} - v_{j,\mathcal{V}}^{(0)} \right\|_2, \qquad \rho_{i,\mathcal{V}}^{\mathrm{local}} = \frac{r_{i,\mathcal{V}}^{\mathrm{vis}}}{d_{i,\mathcal{V}}^{\mathrm{NN}}}. \tag{18}$$

The figures abbreviate $\rho_{i,\mathcal{V}}^{\mathrm{local}}$ as r/NN. A value below one means that every sampled perturbation moves the representation by less than the distance from its clean anchor to any other clean anchor. A value of at least one means that at least one perturbation displacement is as large as the nearest-clean distance. This comparison uses only distance magnitudes: it does not show that a perturbed representation approaches or overlaps another history, and the nearest clean history need not differ across a task-relevant state boundary.

The r/NN ratio considers only the perturbation radius of anchor i. To account for the perturbation radii of both histories, we also compare their enclosing balls. The ball around anchor i is *fully disjoint* if it does not intersect the ball around any other anchor:

$$\left\| v_{i,\mathcal{V}}^{(0)} - v_{j,\mathcal{V}}^{(0)} \right\|_2 > r_{i,\mathcal{V}}^{\mathrm{vis}} + r_{j,\mathcal{V}}^{\mathrm{vis}} \quad \text{for every } j \neq i. \tag{19}$$

Across the 128 anchors, we report three summaries: the median r/NN ratio, the fraction with $r_{i,\mathcal{V}}^{\mathrm{vis}} < d_{i,\mathcal{V}}^{\mathrm{NN}}$, and the fraction satisfying the fully disjoint condition in Equation (19). The second summary compares each perturbation radius with the distance to the nearest other clean anchor. The third also accounts for the perturbation radius around every other anchor. All three summaries are computed in the original high-dimensional representation; overlap between the qualitative t-SNE ellipses is not used.

These summaries are descriptive and depend on the sampled histories and perturbation draws. They compare the encoder output and the final rollout step but do not examine intermediate predictions. They also do not bound changes in prediction error or planning cost; those theoretical links are provided by pairwise ACPC in Section 3.1.

C Evaluation and Diagnostic Protocol

This appendix specifies the fixed evaluation and diagnostic protocol used in the reported results.

Evaluation grid. We evaluate LeWM on PushT, TwoRoom, Reacher, and Cube. Each checkpoint is evaluated with seeds 42, 43, and 44, using 100 episodes per seed. The primary evaluation adds Gaussian noise with standard deviation 0.08 to the history observations while keeping the goal image clean. For each task, the training grid contains one unaugmented checkpoint and eight checkpoints trained with full-sequence Gaussian-noise augmentation at $\sigma_{\max} \in \{0.01, 0.02, \ldots, 0.08\}$.

PLDM protocol. PLDM is evaluated on the same four tasks and nine-checkpoint training grid as LeWM, using the same evaluation seeds and number of logged histories. Its diagnostics use the same clean–perturbed pairing, eight-step rollout horizon, and SR definition. In the PLDM plots, raw IR is divided by the value of the unaugmented PLDM checkpoint from the same task and training run.

PLDM results. To test whether the diagnostics are specific to LeWM, we apply the same ACPC, IR, and SR definitions to PLDM, which uses a different architecture and training recipe. The evaluated PLDM sweeps show a similar qualitative pattern: better control performance is associated with lower IR and higher SR (Table 1).

Table 1: **PLDM diagnostics across the augmented training grid.** Values are three-run means; brackets give their range across all eight augmented levels.

Task	Planning success (%) unaug. → augmented range	Relative IR augmented range	SR unaug. → augmented range
TwoRoom	67.1±8.0→[95.7, 97.7]	[0.071, 0.440]	0.082→[0.967, 1.000]
PushT	10.2±5.9→[20.9, 70.3]	[0.070, 0.866]	0.228→[0.439, 1.000]
Reacher	53.0±29.7→[63.0, 82.2]	[0.166, 0.893]	0.650→[0.887, 1.000]
Cube	45.9±1.7→[50.6, 53.9]	[0.043, 0.742]	0.170→[0.637, 1.000]

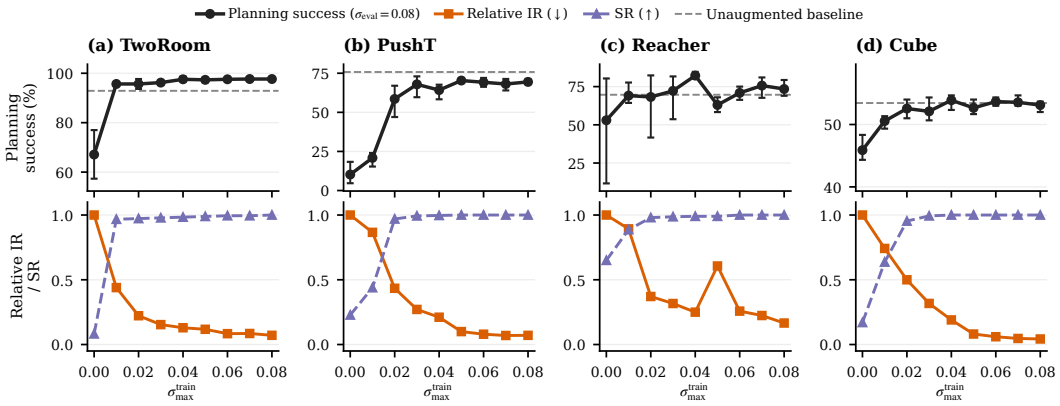


Figure 7: Complete PLDM Gaussian-noise training sweep over three runs. Curves show three-run means and success bars span the across-run range. The summary in Table 1 includes every augmented level rather than selecting a recovery subset.

Success-rate criterion. For each task and training run, let S_0 be the noisy-observation success rate of the unaugmented checkpoint, and let $S_{\max}$ be the highest noisy-observation success rate observed in the fixed nine-checkpoint sweep. A checkpoint meets the success-rate criterion if its noisy-observation success rate is at least

$$S_0 + 0.8(S_{\max} - S_0)$$

and its clean-observation success rate is no more than five percentage points below that of the unaugmented checkpoint. At the task level, we call an augmentation level recovered only when this criterion holds in at least two of the three training runs. $S_{\max}$ is used only to scale this recovery label; every level meeting the criterion remains part of the reported range.

IR protocol. We compute IR separately for each task, training run, and checkpoint. To express rollout divergence relative to the normal motion in each history, we first compute a clean-motion scale. For anchor i, let $u_{i,0}^{\text{obs}}$ be the projected embedding of the final history observation, and let $u_{i,1:H}^{\text{obs}}$ be the projected embeddings of the next H observed clean frames. We define

$$s_i = Q_{0.50}\left(\left\{\|u_{i,k}^{\text{obs}} - u_{i,k-1}^{\text{obs}}\|_2\right\}_{k=1}^{H}\right). \tag{20}$$

Thus, s_i is the median displacement between consecutive clean observations, including the transition from the final history observation to the first future observation.

For each anchor, we generate multiple perturbation draws using the visual shift being evaluated, with five draws for Gaussian noise. For the Gaussian-noise experiments in Section 4.2, we use $\sigma = 0.08$; the blur and resize experiments apply their corresponding shifts. For every draw, we compute eight-step ACPC with uniform weights $\alpha_k = 1/H$ and normalize it by $s_i + \varepsilon_{\text{norm}}$. We average the normalized values across draws for each anchor and then take the 90th percentile (q90) across anchors. The result is raw IR in Equation (7).

SR compares different-state distances with raw IR from the same checkpoint. For the sweep plots, raw IR is divided by the value of the corresponding unaugmented checkpoint, as defined in Equation (10). Only the LeWM cross-task analysis uses relative IR for threshold selection. We use $\varepsilon_{\text{norm}} = 10^{-8}$ in Equation (6).

Across all 108 LeWM checkpoints from the four tasks in the main Gaussian-noise sweep, zero clean-motion scale occurred only in PushT, at one of the 100 anchors per checkpoint. For this anchor, the denominator reduces to $\varepsilon_{\text{norm}}$. Excluding it changed relative IR by at most 0.034, with only one checkpoint crossing each of the two fixed IR thresholds examined (0.1 and 0.3). These crossings occurred near the threshold boundaries. The 90th-percentile aggregation prevents this single extreme value from directly determining IR.

Table 2 reports the results for the tested rollout horizons and summary quantiles.

Table 2: Horizon and quantile sensitivity of the IR reduction at fixed evaluation noise $\sigma = 0.08$. Each entry is the median, over the three training runs, of the ratio between the IR of the checkpoint trained at the highest augmentation level ($\sigma_{\max} = 0.08$) and that of the unaugmented checkpoint; lower means a larger reduction. q80, q90, and q95 denote the 80th, 90th, and 95th percentiles, respectively. These values are descriptive and do not retune any threshold.

	$q = 0.90$ by horizon				$H = 8$ by quantile		
Task	H1	H2	H4	H8	q80	q90	q95
TwoRoom	0.05	0.04	0.05	0.06	0.06	0.06	0.06
PushT	0.06	0.07	0.06	0.09	0.07	0.09	0.09
Reacher	0.02	0.03	0.03	0.02	0.02	0.02	0.02
Cube	0.04	0.03	0.04	0.04	0.03	0.04	0.04

SR protocol. SR follows Equation (9). We first define a local neighborhood using the standardized logged endpoint states. Let $d_{0.35}$ be the 35th percentile of all pairwise distances between different endpoints—that is, the q35 of all off-diagonal Euclidean distances. This cutoff keeps the comparisons local while retaining an eligible neighbor for most anchors; Table 3 reports the resulting counts.

For each anchor i, we select the nearest history $j(i)$ that has a different endpoint-state label and lies within distance $d_{0.35}$. If no such history exists, the anchor is excluded from SR. We roll both histories forward under the anchor's recorded action sequence $\mathbf{a}_i$. This sequence was not generally executed by the neighboring history, but using it for both rollouts prevents action differences from affecting the comparison. Their normalized rollout distance is

$$D_i^{\text{diff}} = \frac{\left\| \bar{G}_{\mathbf{a}_i}(E_\theta(h_i)) - \bar{G}_{\mathbf{a}_i}(E_\theta(h_{j(i)})) \right\|_2}{s_i + \varepsilon_{\text{norm}}}. \tag{21}$$

The pair counts as separated when D_i^{diff} is greater than raw q90 IR plus the fixed margin 0.10. Prediction errors in the neighbor's rollout under the anchor's actions may increase the measured separation.

For pairing, the endpoint state is taken after the eight observed future steps and therefore records the state reached under that trajectory's own actions. The dataset preprocessor standardizes each state coordinate, and neighborhood distances use the full standardized state vector. Endpoint-state labels are constructed by median splits of the task-specific coordinates listed in Table 3, using the fixed batch of 100 endpoints generated with anchor seed 9101. Because the logged endpoints, labels, and neighborhoods are fixed before checkpoint evaluation, the selected pairs and eligible-anchor counts are identical across checkpoints and training runs.

Table 3: Endpoint-state labels used to construct SR pairs. The state is taken at the eighth observed future step, and its coordinates are standardized using full-dataset statistics. Counts report eligible and skipped histories among the fixed 100 anchors. These labels define operational partitions for the diagnostic; they are not semantic or action-relevance annotations.

Task	Logged state field (dim.)	Label construction $\ell(y)$	Eligible / skipped
TwoRoom	`pos_agent` (2)	Median split of the standardized agent x coordinate (`pos_agent[0:1]`).	61 / 39
PushT	`state` (7)	Three-bit label from median splits of standardized block x, block y, and block angle (`state[2:5]`).	98 / 2
Reacher	`observation` (6)	Three-bit label from median splits of the two standardized joint velocities and their Euclidean norm (`observation[4:6]`).	100 / 0
Cube	`observation` (28)	Three-bit label from median splits of the first two standardized coordinates and $\|r\|_2$, where $r = \bar{o}_{0:3} - \bar{o}_{25:28}$.	100 / 0

Threshold grids. For the LeWM cross-task evaluation, we search

$$t_{\text{IR}} \in \{0.05, 0.075, 0.10, 0.15, 0.20, 0.30\}, \qquad t_{\text{SR}} \in \{0.80, 0.85, 0.90, 0.95\}.$$

For each nonempty subset of tasks used for threshold selection, we evaluate every threshold pair. We first minimize the mean number of augmentation-grid steps between the first checkpoint accepted by the IR–SR screen and the first checkpoint that meets the success-rate criterion. Remaining ties are resolved by, in order, fewer early acceptances, fewer late acceptances, higher balanced accuracy, a smaller t_{IR}, and a larger t_{SR}.

The selected thresholds are then applied unchanged to the tasks that were not used for their selection. Success-rate labels from those tasks and all blur or resize results are excluded from threshold selection.

Reproducibility. The commands used to rebuild the paper figures and tables are documented in `paper1/scripts/README.md` and `tools/README_paper1.md`. The released machine-readable summaries are sufficient for the reader-facing aggregation and plotting steps. Recomputing checkpoint-level diagnostics additionally requires the released checkpoints and datasets. The summaries record the training seeds, evaluation seeds, and protocol hashes used for each result.

Table 4: Recovery ranges in the original Gaussian-noise training sweep (nine checkpoints per task: no augmentation plus eight noise levels; Figure 2 shows every level). The final column lists the contiguous training-noise ranges that meet the prespecified success-rate criterion in Section 4.1. It summarizes an accepted range rather than selecting a single training strength.

Task	Unaugmented success (%)	Training levels meeting criterion
TwoRoom	68.8	0.01–0.08
PushT	7.2	0.03–0.08
Reacher	18.2	0.03–0.08
Cube	43.1	0.03–0.07

D Additional Multi-Severity Analysis

This section gives the aggregation and sensitivity details behind Section 4.3. The broad-severity experiment is analyzed separately from the original narrow sweep. It keeps the original thresholds and pair construction fixed and uses continuous retention rather than defining a new performance band.

The same 3 TwoRoom checkpoints account for every additional SR rejection among IR-passing rows at $\sigma_{\mathrm{eval}} = 0.08$ and 0.16: 1 has $\sigma_{\max} = 1.0$ and 2 have $\sigma_{\max} = 1.5$. At $\sigma_{\mathrm{eval}} = 0.32$, these rejections comprise 4 TwoRoom, 5 PushT, and 1 Reacher checkpoints; at 0.64, they comprise 5 TwoRoom and 6 PushT checkpoints. No Cube checkpoint passes IR while failing SR. The conditional rejections thus span three tasks at 0.32 and two at 0.64, although some checkpoint rows recur across severity slices and the conditional denominator also changes. All low-severity rejections fall on TwoRoom, whose SR catalog uses one median split and has the fewest eligible anchors (61 of 100; Table 3); this concentration should be read within the scope of the tested pair catalog.

Table 5: Fixed-threshold decision decomposition in the separate broad-severity experiment. Each row partitions the same 96 trained checkpoints at one evaluation severity. "IR pass, SR fail" is the additional rejection shown in Figure 3. The reverse disagreement appears at the two higher severities, so neither condition subsumes the other.

σ_{eval}	Pass both	IR pass, SR fail	IR fail, SR pass	Fail both
0.08	93	3	0	0
0.16	93	3	0	0
0.32	72	10	5	9
0.64	35	11	5	45

The conditions also disagree in the other direction: at both $\sigma_{\mathrm{eval}} = 0.32$ and 0.64, 5 and 5 rows, respectively, pass SR but fail relative IR. The reverse disagreement in Table 5 is expected to be possible because the conditions use different references. Relative IR divides the raw perturbation radius by the unaugmented checkpoint's radius at the same severity. SR instead compares that raw

radius with clean different-state distances from the checkpoint under evaluation. The fixed-threshold results therefore cannot be reproduced by nesting one condition inside the other.

For each task and training run, the grid contains eight training-noise maxima and four evaluation severities. Let x_{ij} denote any diagnostic or behavioral quantity at training level i and evaluation severity j. We remove the two additive main effects using

$$\widetilde{x}_{ij} = x_{ij} - \overline{x}_{i\cdot} - \overline{x}_{\cdot j} + \overline{x}_{\cdot\cdot}$$

Spearman correlations are computed over the 32 centered cells within a single task and training run. We then average the 3 training-run correlations within each task and weight the 4 tasks equally. The task ranges in Table 6 summarize heterogeneity; they are not confidence intervals. Double-centering leaves $(8-1)(4-1) = 21$ interaction degrees of freedom and imposes row- and column-sum constraints, so the 32 cells are not treated as independent observations.

Table 6: Descriptive Spearman associations with same-checkpoint retention (success under noise minus clean success) in the separate broad-severity experiment. Within each task and training run, all quantities are double-centered over the 8×4 training–evaluation severity grid. Training-run correlations are then averaged within each task, and tasks receive equal weight. The adjusted column reports partial Spearman correlation after residualizing the rank-transformed diagnostic and outcome against the rank-transformed exposure ratio $-\sigma_{\mathrm{eval}}/\sigma_{\max}$ within each task–run block. Ranges contain the four task means, not confidence intervals. The exposure ratio is a simple metadata comparator, not an exhaustive baseline, and these associations do not establish checkpoint-selection accuracy.

Quantity (favorable direction)	Direct ρ	Task range	Adjusted ρ	Task range
$-\sigma_{\mathrm{eval}}/\sigma_{\max}$	0.787	[0.662, 0.887]	–	–
$-\mathrm{IR}^{\mathrm{rel}}$	0.796	[0.671, 0.909]	0.458	[0.070, 0.667]
SR	0.826	[0.756, 0.900]	0.532	[0.225, 0.694]

We next relate the continuous diagnostics to same-checkpoint retention, defined as success under noise minus clean success. After accounting for the simple exposure ratio $-\sigma_{\mathrm{eval}}/\sigma_{\max}$, the correlations remain positive when tasks are weighted equally: 0.458 for IR and 0.532 for SR. The adjusted IR association is near zero on PushT (0.070), so the result is not uniform across tasks. These associations do not establish better checkpoint selection or replace evaluation of nominal model quality.

On PushT, the exposure comparator's direct task-level correlation exceeds both diagnostics. This is why we use it as an explicit design-metadata comparator and do not interpret the small differences among the direct aggregate values as diagnostic superiority.

Using stress success divided by same-checkpoint clean success instead of their percentage-point difference gives the same qualitative result. The equal-task mean direct correlations are 0.813 for lower relative IR and 0.839 for higher SR; after accounting for the exposure ratio they are 0.560 and 0.626, respectively. Every clean-success estimate is nonzero (minimum 7%), but ratios can still amplify changes for weak checkpoints; we therefore keep the percentage-point difference as the primary outcome and treat the ratio only as a sensitivity check. These are descriptive factorial associations. The 384 rows are not independent replicates, and the four evaluation-severity rows for a checkpoint share the same clean-performance estimate; therefore we do not attach row-level p-values or confidence intervals.

E Prediction-Error Scope and Controls

We checked Proposition 1 for every logged eight-step pair and every candidate-specific five-step pair across all tasks, training runs, and training conditions. No numerical violations occurred. This check confirms that the implementation follows the inequality. It is not an independent statistical success count or evidence of model quality, because the inequality holds for every correctly computed pair.

The prediction-error and planning experiments answer different questions. The prediction-error experiment measures how a visual perturbation changes rollout error relative to the observed future under the recorded actions. The planning experiment instead computes ACPC for the action candidates evaluated by CEM and relates it to selection regret. We therefore treat the planning analysis as separate evidence rather than infer it from the prediction-error result (Section 4.5).

The prediction-error analysis uses only the unaugmented checkpoints. Each history pair contributes rows at Gaussian-noise standard deviations $\sigma \in \{0.02, 0.05, 0.08\}$ with two perturbation draws; zero-severity probes serve only as identity checks. Base+Control$_8$ reports the lowest test MAE among the three action controls for each task and training run. Table 7 reports the relative reduction in cross-validated MAE for each training run. Table 8 reports the corresponding MAE values and the selected destroyed-action controls. Base+ACPC$_8$ achieves lower MAE than both comparison models on all four tasks in every training run.

Table 7: Relative reduction in out-of-group MAE from Base+ACPC$_8$ when predicting the error drift d on the unaugmented checkpoints (protocol and model definitions in Section 4.4). Base+Control$_8$ uses the per-cell oracle control; higher is better. The last column counts task–run cells in which Base+ACPC$_8$ is best.

Training run (seed)	vs. Base	vs. Base+Control$_8$	Task–run cells improved
3072	55.5%	51.4%	4/4
3073	60.8%	54.8%	4/4
3074	51.4%	47.7%	4/4
mean $\pm$ SD	55.9 $\pm$ 4.7%	51.3 $\pm$ 3.5%	12/12

Table 8: Out-of-group MAE (16-fold leave-one-trajectory-group-out) for predicting the error drift d on the unaugmented checkpoints; lower is better, model definitions in Section 4.4. "Selected control" is the destroyed-action control chosen by the per-cell oracle; "paired wins" counts the folds (of 16) in which Base+ACPC$_8$ beats Base+Control$_8$.

Task	run (seed)	Base	Base +Control$_8$	Base +ACPC$_8$	selected control	paired wins
TwoRoom	3072	2.535	2.099	0.755	shuffled actions	15/16
TwoRoom	3073	2.336	2.015	0.866	shuffled actions	15/16
TwoRoom	3074	2.075	1.911	1.006	swapped actions	13/16
PushT	3072	2.068	2.068	1.762	shuffled actions	11/16
PushT	3073	1.969	2.008	1.315	zeroed actions	13/16
PushT	3074	1.909	1.833	1.439	zeroed actions	13/16
Reacher	3072	1.358	1.097	0.288	shuffled actions	16/16
Reacher	3073	1.399	0.793	0.247	shuffled actions	16/16
Reacher	3074	1.409	1.136	0.263	shuffled actions	16/16
Cube	3072	1.171	1.037	0.489	zeroed actions	14/16
Cube	3073	1.416	1.212	0.500	zeroed actions	14/16
Cube	3074	1.061	1.008	0.552	shuffled actions	14/16

F Cross-Task Threshold Partitions

With four tasks, there are 14 nonempty choices of tasks for threshold selection: four single-task choices, six two-task choices, and four three-task choices. The selected thresholds are applied to the remaining task or tasks. Each evaluation includes all three training runs and all nine checkpoints for every remaining task. Checkpoints are not counted as independent samples: metrics first average the training runs within each evaluation task and then weight the evaluation tasks equally.

When thresholds are selected using only Reacher, the procedure chooses $(t_{\mathrm{IR}}, t_{\mathrm{SR}}) = (0.1, 0.95)$ instead of the predominant $(0.3, 0.95)$. Both threshold pairs locate Reacher recovery within one augmentation-grid step, but $t_{\mathrm{IR}} = 0.1$ matches the Reacher onset exactly and therefore wins the tie-break. Recovered Reacher checkpoints have relative IR well below 0.1, whereas the earliest recovered checkpoints on the other tasks remain above 0.1. The Reacher-only threshold therefore accepts recovery on those tasks too late, by as many as five augmentation levels.

Sensitivity to the recovery criterion. CEM's random sampling introduces variation in measured success rates. We therefore use a recovery fraction of 0.8 to allow some deviation from the highest observed success rate. We also test fractions of 0.7 and 0.9, reselecting thresholds on source tasks

Table 9: Joint IR/SR checkpoint screening with thresholds chosen on other tasks. Thresholds are selected on the threshold-selection tasks and applied unchanged to all test tasks. Metrics first average training runs within each test task and then weight tasks equally. Recovery-onset mismatch is measured in training-augmentation grid levels (one level is 0.01 in $\sigma_{\max}$).

Selection tasks	Test tasks	t_{IR}	t_{SR}	BA	P / R	Onset mismatch mean / max
TwoRoom	PushT, Reacher, Cube	0.3	0.95	0.888	0.884 / 0.981	0.3 / 1.0
PushT	TwoRoom, Reacher, Cube	0.3	0.95	0.894	0.902 / 0.956	0.6 / 1.0
Reacher	TwoRoom, PushT, Cube	0.1	0.95	0.723	0.906 / 0.540	2.9 / 5.0
Cube	TwoRoom, PushT, Reacher	0.3	0.95	0.913	0.950 / 0.938	0.6 / 1.0
TwoRoom, PushT	Reacher, Cube	0.3	0.95	0.874	0.853 / 1.000	0.3 / 1.0
TwoRoom, Reacher	PushT, Cube	0.3	0.95	0.887	0.873 / 0.972	0.3 / 1.0
TwoRoom, Cube	PushT, Reacher	0.3	0.95	0.903	0.925 / 0.972	0.3 / 1.0
PushT, Reacher	TwoRoom, Cube	0.3	0.95	0.896	0.901 / 0.935	0.7 / 1.0
PushT, Cube	TwoRoom, Reacher	0.3	0.95	0.912	0.952 / 0.935	0.7 / 1.0
Reacher, Cube	TwoRoom, PushT	0.3	0.95	0.926	0.972 / 0.907	0.7 / 1.0
TwoRoom, PushT, Reacher	Cube	0.3	0.95	0.858	0.802 / 1.000	0.3 / 1.0
TwoRoom, PushT, Cube	Reacher	0.3	0.95	0.889	0.905 / 1.000	0.3 / 1.0
TwoRoom, Reacher, Cube	PushT	0.3	0.95	0.917	0.944 / 0.944	0.3 / 1.0
PushT, Reacher, Cube	TwoRoom	0.3	0.95	0.935	1.000 / 0.869	1.0 / 1.0

for each fraction while keeping the clean-performance constraint unchanged. Balanced accuracy is similar at 0.7 and 0.8 and lower at 0.9 (Table 10).

Table 10: Balanced accuracy under different recovery criteria. All test cases are included, including one case with no accepted checkpoint at fraction 0.9 when using a single source task.

Recovery fraction	1 source task	2 source tasks	3 source tasks
0.7	0.854	0.895	0.895
0.8	0.855	0.900	0.900
0.9	0.779	0.799	0.813

G Cross-Stressor Pairs and Boundary Cases

We test whether IR and SR remain informative under visual shifts not used for training. For each task and training run, we compare the unaugmented checkpoint with the checkpoint trained at $\sigma_{\max} = 0.08$ under blur ($k = 15$) and resize (scale 0.25), giving $4 \times 3 \times 2 = 24$ comparisons. IR and SR are recomputed under each shift.

For each pair, IR improvement is $1 - \mathrm{IR}_q^{\mathrm{rel}}$, and SR improvement is the SR of the augmented checkpoint minus that of the unaugmented checkpoint. Positive values therefore mean lower perturbation sensitivity and better state separation, respectively. We ask whether these diagnostic improvements agree with control performance: a pair is considered improved if planning success under the shift increases by at least five percentage points while clean success drops by no more than five.

IR and SR each agree with this criterion in 22 of the 24 comparisons. Their Spearman correlations with the change in planning success are 0.854 and 0.913, respectively, and remain positive when any

one task is omitted (Figure 8 and table 11). Thus, both diagnostics remain informative under the tested blur and resize shifts.

Component-wise reporting avoids assigning a continuous joint score to diagnostics with different roles. Figure 8 shows the corresponding per-task changes.

The two label disagreements are PushT run 3072 under resize and run 3074 under blur. Each gains four success points, one below the prespecified five-point cutoff, while both diagnostic components move in the favorable direction.

Table 12 compares recorded-action ACPC with encoder-only, one-step rollout, and action-destroying controls.

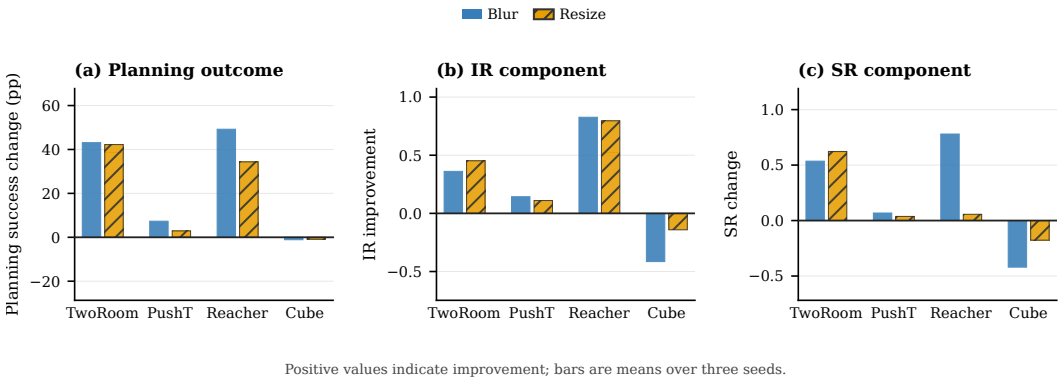


Figure 8: Changes from the unaugmented checkpoint to the checkpoint trained at $\sigma_{\max} = 0.08$ under blur and resize. Panels show (a) planning-success change in percentage points, (b) IR improvement $1 - \mathrm{IR}_q^{\mathrm{rel}}$, and (c) SR change. Bars are means across three training runs, and positive values indicate improvement.

Table 11: Descriptive cross-stressor component summary. IR improvement is $1 - \mathrm{IR}_{\mathrm{rel}}$ because the base value is 1; SR change is endpoint SR minus base SR. Spearman values correlate each component change with planning-success change. The leave-one-task-out (LOTO) range is obtained by omitting each task in turn and recomputing the pooled Spearman correlation. Binary agreement compares a positive component change with the prespecified positive behavior label. The final cell gives the 2.5th–97.5th percentiles from exact enumeration of all $4^4 = 256$ ordered task-cluster resamples, using linear interpolation. This is a sensitivity distribution, not a new-task population CI; the 24 paired rows are not IID.

Scope	n	ρ_{IR}	ρ_{SR}	binary agreement IR / SR	cluster-resampling sensitivity
Pooled	24	0.854	0.913	0.917 / 0.917	IR: [0.263, 0.950] SR: [0.352, 0.929]
Blur	12	0.874	0.953	0.917 / 0.917	–
Resize	12	0.748	0.846	0.917 / 0.917	–
TwoRoom	6	0.543	0.543	1.000 / 1.000	–
PushT	6	0.464	0.603	0.667 / 0.667	–
Reacher	6	0.886	0.886	1.000 / 1.000	–
Cube	6	-0.086	-0.543	1.000 / 1.000	–
LOTO range	18	[0.720, 0.911]	[0.846, 0.901]	–	–

Table 12: Diagnostic ablation on 24 paired blur/resize endpoints. Each metric predicts improvement when its endpoint-to-reference ratio is below one; no threshold is fitted or transferred. Spearman correlation relates diagnostic improvement, defined as one minus this ratio, to planning-success change. Correct/24 and BA compare the predicted sign with the prespecified behavioral label. The LOTO range recomputes pooled Spearman after omitting each task; the 24 pairs are clustered within four tasks.

Diagnostic	Spearman	Correct / 24	BA	LOTO Spearman range
Encoder only	0.859	20	0.778	[0.725, 0.901]
One-step rollout	0.738	20	0.778	[0.454, 0.877]
Zero actions	0.805	21	0.833	[0.626, 0.877]
Cross-anchor action shuffle	0.780	21	0.833	[0.586, 0.869]
Temporal action shuffle	0.844	22	0.889	[0.678, 0.923]
Recorded-action ACPC	0.854	22	0.889	[0.720, 0.911]

H Local Sensitivity under Gaussian Noise

To relate ACPC under Gaussian noise to local model sensitivity, consider a small isotropic perturbation $\delta x \sim \mathcal{N}(0, \sigma^2 I)$. Let J_E and J_G be the Jacobians of the encoder and rollout map, evaluated along the clean input. A first-order expansion gives

$$\Delta G = J_G J_E \, \delta x + o(\|\delta x\|_2).$$

Writing $A = J_G J_E$ and using $\mathbb{E}[\delta x \delta x^\top] = \sigma^2 I$, we obtain

$$\mathbb{E}\big[\|A\delta x\|_2^2\big] = \operatorname{tr}\big(A\,\mathbb{E}[\delta x \delta x^\top] A^\top\big) = \sigma^2 \operatorname{tr}(AA^\top) = \sigma^2 \|A\|_F^2, \tag{22}$$

Therefore, for small σ,

$$\mathbb{E}[\|\Delta G\|_2^2] \approx \sigma^2 \|J_G J_E\|_F^2.$$

The squared Frobenius norm of the composed Jacobian thus measures how strongly the encoder–rollout path locally amplifies Gaussian observation noise. Jacobian Frobenius norms have previously been used to measure and regularize local representation sensitivity (Rifai et al., 2011). We estimate

$$\|J_G J_E\|_F^2 = \operatorname{tr}\big[(J_G J_E)^\top (J_G J_E)\big]$$

with Rademacher probes and the Hutchinson estimator (Hutchinson, 1989). Jacobian–vector products compute these estimates without constructing the full Jacobian. In every task, both the finite-difference and Jacobian-based estimates are lower for the noise-augmented checkpoint (Figure 9). This result is consistent with Gaussian-noise augmentation reducing ACPC by lowering the local sensitivity of the encoder–rollout path. The analysis applies only near the evaluated inputs and does not provide a global robustness guarantee.

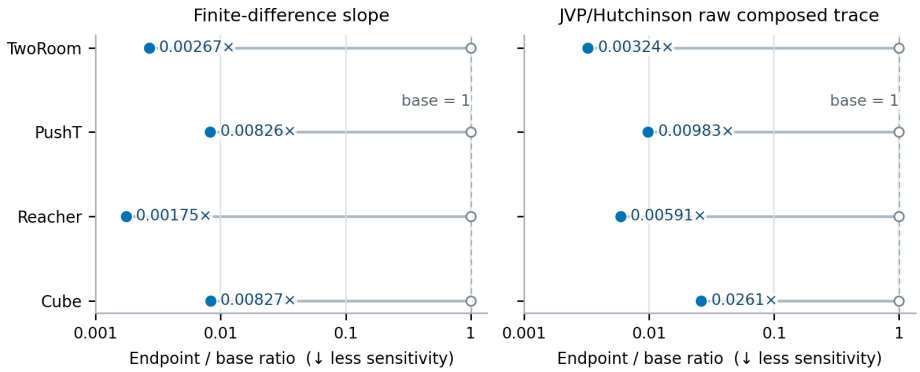


Figure 9: Ratio of local sensitivity for the Gaussian-noise-augmented checkpoint to that of the unaugmented checkpoint. We report a finite-difference estimate and a JVP-based Hutchinson estimate of the composed Jacobian's squared Frobenius norm. Both estimates are below one for every task, indicating lower local sensitivity after augmentation. These local measurements do not establish task performance or global robustness.

I Adaptive-CEM Selection Regret

The selection-regret analysis does not use task-success labels. We evaluate the unaugmented and $\sigma_{\max} = 0.08$ checkpoints from every task and training run. For each checkpoint, we use 100 histories and apply perturbations at severities 0.02, 0.05, and 0.08. Each clean or perturbed CEM run uses 64 candidate action sequences, eight elite candidates, eight iterations, and a five-step prediction horizon. The paired runs start from the same proposal and use the same random samples, but each run updates its proposal using its own elite candidates.

The planner uses the squared distance between the predicted endpoint and the goal embedding:

$$C_j = \|x_j - g\|_2^2.$$

For each CEM run, we compute ACPC for every candidate under that candidate's action sequence. We summarize the candidate pool by the q90 ACPC at horizons one and five. Let $\mathbf{a}$ be the plan selected from the clean history and $\tilde{\mathbf{a}}$ the plan selected from its perturbed counterpart. The regression target is the extra predicted cost of the perturbed-history plan when both plans are evaluated from the clean history:

$$\left[C_h(\tilde{\mathbf{a}}) - C_h(\mathbf{a})\right]_+.$$

Within each training run, we fit ridge regressions on three tasks and evaluate them on the remaining task, rotating through all four choices. Each regression feature is standardized using the mean and standard deviation computed from the three training tasks. The same values are used to standardize features from the remaining test task. The target is $\log(1 + \text{selection regret})$, without further standardization. MAE is computed on this scale and averaged equally across the four test tasks. The relative MAE reduction is computed separately for each training run before reporting the mean and sample standard deviation across the three runs. This experiment evaluates whether ACPC reflects the model-cost effect of a CEM selection change; it does not evaluate simulator return.

Table 13 summarizes the target values predicted in Sections 4.4 and 4.5.

Table 13: Summary statistics of the regression targets. The targets are prediction-error change d in Section 4.4 and $\log(1 + \text{selection regret})$ in Section 4.5. Each task includes 288 and 1,800 samples, respectively, pooled across the three training runs. SD denotes the standard deviation of target values, not uncertainty in the mean.

	Prediction-error change d		$\log(1 + \text{selection regret})$	
Task	Mean	SD	Mean	SD
TwoRoom	8.756	6.518	2.123	2.188
PushT	6.725	5.653	2.170	2.077
Reacher	8.335	5.666	2.080	2.312
Cube	9.772	6.900	2.189	2.243