LEARNING WHAT ACTIONS CHANGE: ACTION EFFECT MATCHING FOR DYNAMICS-FAITHFUL WORLD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

World models aim to predict environment dynamics under actions, enabling the prediction of action consequences without executing them in the real world. Yet despite producing increasingly realistic videos, current world models often fail to capture how small action changes affect future observations. This limits their use in tasks such as robotic policy evaluation and model-based planning, where nearly identical actions can lead to qualitatively different outcomes. To address this, we propose AFFECT, an auxiliary training objective for learning the distribution of visual changes induced by action perturbations. Specifically, we inject temporally correlated noise into the actions of a policy trained on task-agnostic human teleoperation. We then train the world model so that its generated videos respond to this noise as the recorded videos do, without requiring repeated executions from the same initial state. Across OGBench simulation and real-world DROID and ALOHA-2 setups, AFFECT makes world-model rollouts more faithful to real outcomes. On DROID, the Pearson correlation between policy success rates in the world model and on the real robot rises from 0.47 to 0.79, alongside up to 23% lower FVD. These gains carry over to planning, where selecting actions with AFFECT-trained world models improves task success by up to 8.3%.

1 INTRODUCTION

World models (Ha & Schmidhuber, 2018; Hafner et al., 2019; 2025) predict future states conditioned on action sequences without execution in the environment. Recently, robotics world models (RWMs) (Gao et al., 2026; Agarwal et al., 2026; Guo et al., 2026) realize this by building on video diffusion foundation models (Wan et al., 2025; Ali et al., 2025; Kong et al., 2024), generating photorealistic, temporally coherent videos conditioned on robot actions. This makes them a promising direction for diverse usage in evaluating policies (Guo et al., 2026; Jeon et al., 2026), planning manipulation trajectories (Assran et al., 2025; Gao et al., 2026), or generating training data (Jang et al., 2025).

However, serving these roles requires robotics world models to faithfully reflect how small action changes alter future outcomes. For example, slight changes in reaching motion can determine whether a gripper grasps an object, misses it, or pushes it away (Figure 1). Yet existing models tend to produce visually plausible videos that ignore such differences (Yang et al., 2026; Jiang et al., 2026). Since such models predict similar outcomes for actions with different real consequences, they are unreliable as policy evaluators and mislead planners into selecting the wrong actions.

Addressing this requires world models to acquire action sensitivity, and by extension, learn the relationship between action variations and resulting dynamics. We therefore treat action sensitivity as a distributional problem. Rather than simply reproducing recorded futures with high visual fidelity, we argue that an RWM should capture how the distribution of future observations shifts as actions vary under a given observation context.

To learn such distributions, we introduce action variation directly into the training data. Specifically, we collect robot-environment interactions by running a policy trained on task-agnostic human teleoperation, in which an operator interacts freely with objects without a predefined task. Following Lynch et al. (2020), we refer to such data as play data and to this policy as the play policy.

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.

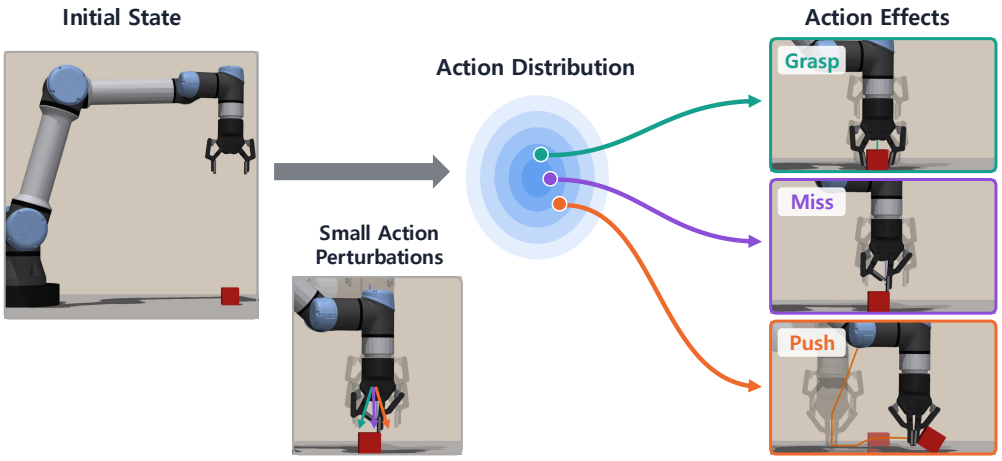


Figure 1: **Small action perturbations lead to distinct action effects.** From the same initial state, actions drawn from a narrow distribution change the gripper's approach only slightly, yet produce qualitatively different outcomes, where the cube is grasped, missed, or pushed away. A faithful world model must reproduce how these outcomes shift as the action is perturbed, rather than only rendering a plausible video for generic actions.

During data collection, we inject temporally correlated noise into the action chunks produced by the play policy before rollout and log the injected perturbations. These rollouts expose how visual outcomes vary with action changes without requiring repeated executions from the same initial state or explicit outcome labels.

Building on this data, we introduce AFFECT, an auxiliary training objective that teaches the world model the relationship between action changes and their visual consequences. AFFECT explicitly uses the noise injected into the training actions to supervise how generated visual outcomes should change with those action perturbations. In this way, the model is directly encouraged to respond to action variation in the same way as the real environment, rather than merely generating visually plausible futures. Since AFFECT operates on features from frozen encoders, it can be added to existing RWMs without architectural changes.

We evaluate AFFECT on a simulated UR5e environment and two real-robot settings following the DROID (Khazatsky et al., 2024) and ALOHA-2 (Aldaco et al., 2024) configurations, and apply it to two distinct backbones, DreamDojo (Gao et al., 2026) and Cosmos3 (Agarwal et al., 2026). To test whether the world model reflects action differences, we measure how well its predicted policy success rates correlate with real ones. On this measure, AFFECT improves every one of the six backbone–embodiment combinations, for instance raising the correlation of DreamDojo on DROID from 0.468 to 0.790 and that of Cosmos3 on ALOHA-2 from 0.298 to 0.694. AFFECT also improves most visual metrics, including FVD reductions of up to 23% on DROID and gains on every metric at both short and long horizons in simulation, suggesting that learning how actions shape outcomes also helps the model predict more realistic futures. Finally, this improved sensitivity translates into better model-based planning, where AFFECT raises task success rates by around 8% on both UR5e pick-and-place and ALOHA-2 long-horizon tasks.

2 Preliminaries

World model training. Our aim is to develop a world model W_θ capable of visually simulating the outcomes of an arbitrary robot policy π. Formally, let $\mathcal{D}$ denote a dataset of robot trajectories, each a sequence $\{(o_t, a_t)\}_{t\geq 0}$ of video observations o_t and corresponding actions a_t where t denotes the environment timestep. We optimize $W_\theta(O_t|o_t, A_t)$ during training to predict future states over an H-frame horizon. Specifically, it generates a future video sequence $O_t := (o_{t+1}, \ldots, o_{t+H})$ conditioned on the current observation o_t and action sequence $A_t := (a_t, \ldots, a_{t+H-1})$.

Following recent robotics world models (Gao et al., 2026; Agarwal et al., 2026), we build on a conditional rectified-flow video model (Lipman et al., 2023; Liu et al., 2022). We parameterize a

velocity field $v_\theta(O_t^\tau, \tau;\, o_t, A_t)$ and train it on $O_t^\tau := (1-\tau)O_t + \tau\varepsilon$, with flow timestep $\tau \in [0,1]$ and $\varepsilon \sim \mathcal{N}(0, I)$:

$$\mathcal{L}_{\text{flow}} = \mathbb{E}\Big[\big\|v_\theta(O_t^\tau, \tau;\, o_t, A_t) - (\varepsilon - O_t)\big\|^2\Big]. \tag{1}$$

During inference, we initialize $O_t^1 \sim \mathcal{N}(0, I)$ and integrate v_θ from $\tau = 1$ to $\tau = 0$ with the multistep UniPC solver (Zhao et al., 2023b) to obtain the predicted future video $\hat{O}_t$, whose distribution defines the world model $W_\theta(\cdot \mid o_t, A_t)$.

World model evaluation. For policy evaluation, given an initial observation o_0 and a language instruction l, the generalist policy π instead selects actions from the generated observations. Starting from $\hat{o}_0 = o_0$, the policy outputs $A_t = \pi(\hat{o}_t, l)$, and the world model predicts the resulting future $\hat{O}_t$ conditioned on $\hat{o}_t$ and A_t. The final predicted observation $\hat{o}_{t+H}$ is then fed back into π to select the next action sequence. This forms a fully closed-loop system of interaction between the policy and the world model (Guo et al., 2026), allowing long-horizon policy rollout without executing each candidate policy on a real robot. The usefulness of these rollouts depends on whether task progress and success in the generated videos agree with those in the real environment, so that the model supports reliable comparisons between policies.

Visual quality alone is insufficient for this purpose. A model may preserve the scene and generate plausible arm motion while missing whether the gripper makes contact with an object or whether the object moves as commanded. Such errors can change both the predicted task outcome and the actions selected by the policy at the next step. Evaluation therefore also needs to measure action fidelity, whether changes in the supplied actions produce the corresponding changes in robot-object interaction and task outcome. To this end, we utilize the Pearson coefficient of success rate between real rollouts of policies and their closed-loop rollouts within world models.

3 METHOD

We introduce AFFECT, an auxiliary training objective for RWMs that trains world models to capture how action perturbations change future observations. Before world model training, we collect robot-environment interactions by rolling out a play policy, where we inject temporally correlated noise into the action chunks produced by the play policy and log the injected perturbations. During training, we use frozen feature encoders to extract action effects from generated video features and logged perturbations. We then match action effects in recorded and generated videos, forcing the world model to learn the action effect relationship.

3.1 DATA GENERATION VIA PERTURBATION PLAY

To learn how small changes in action affect future observations, we require diverse robot-environment interaction data with variation in both action distribution and visual outcomes. Play data (Lynch et al., 2020; Yin et al., 2026), in which the robot interacts freely with the environments without specific task instructions, covers diverse interactions.

A policy trained on a dataset of human-teleoperated play trajectories is capable of running autonomously over extended periods of time, accruing large amounts of action trajectories. This means that task-specific collection procedures such as scene resets are not required, allowing scalable generation of data. We further diversify the data by injecting noise into play rollouts as $A_t = \bar{A}_t + u_t$, where $\bar{A}_t$ is the nominal actions generated by the play policy and u_t is the injected noise. A_t is executed by the robot after clipping to ensure safety during autonomous rollout.

To relate action perturbations to their visual outcomes during training, we record the nominal and executed commands, the sampled perturbation u_t, and its noise-distribution parameters for each window. We retain u_t before clipping so that the logged samples follow the specified noise distribution. The world model takes the observation o_t and executed actions A_t as inputs, while the AFFECT loss below uses u_t to match how recorded and generated futures vary with the perturbation. This matching is performed across rollouts, so each window requires only one recorded future rather than repeated executions from the same starting state. Appendix A.1 details the collection protocol.

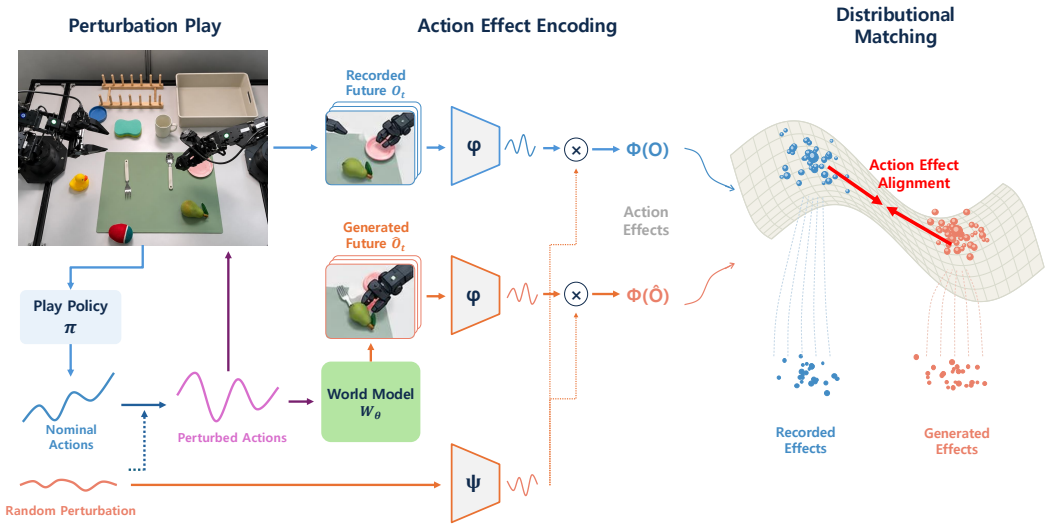


Figure 2: **Overview of AFFECT.** Frozen encoders ϕ, ψ extract video features from both the recorded and generated futures, and perturbation features from the logged action perturbations. Pairing the centered perturbations with each set of video features gives the recorded and predicted action effects $(\Phi(O), \Phi(\hat{O}))$, which measure how visual change moves with the perturbation across rollouts. The AFFECT loss optimizes over the distance between the two distributions, so that generated futures respond to perturbations in the same direction and with the same strength as real ones.

3.2 Matching action effects with frozen encoders

A world model should capture how changing an action changes the resulting video. For example, if a stronger push tends to move an object farther in recorded interactions, generated videos should reflect that relationship. We call these relationships *action effects* and measure them through perturbation and video features across rollouts. Training the model to match these measurements encourages it to reproduce the observed responses to actions.

During world model training, once every n steps, we generate $\hat{O}_t \sim W_\theta(\cdot \mid o_t, A_t)$ from fresh Gaussian noise using a short differentiable Euler sampler. The model receives the same initial observation and executed actions as in the recorded interaction, without access to the recorded future O_t. We then extract features from the perturbation and both videos, center the perturbation features, and compare action effects across recorded and generated windows.

Frozen feature encoders. We first represent the action perturbation and its video outcome as feature vectors with encoders ψ and ϕ, respectively. Each encoder applies an elementwise nonlinearity η followed by a fixed matrix P:

$$\psi(u_t) := P_\psi \, \eta_\psi(u_t) \in \mathbb{R}^{K_\psi}, \qquad \phi(O_t) := P_\phi \, \eta_\phi(O_t) \in \mathbb{R}^{K_\phi}, \tag{2}$$

where K_ϕ, K_ψ each denote dimensions of the extracted feature vector. The matrices P and nonlinearities η are pre-chosen to compute hand-specified features, such as a signed or squared sum of the noise over the window, a change in a channel mean, or a projection of the latent change. The encoders are fixed and remain frozen throughout training, providing a stable way to measure action effects without representational collapse (Li et al., 2015; Dziugaite et al., 2015). Appendix A.5 gives the encoder details and full feature bank.

Centering the perturbation features. To isolate variation associated with the perturbation, we first remove its expected baseline, due to non-zero means under η_ψ. For example, a squared feature with positive mean causes static video features such as background to be supervised. Centering makes these unchanged contributions cancel on average. We center the perturbation features as

$$g_t := \psi(u_t) - \mathbb{E}[\psi(u_t) \mid \mathcal{H}_t] \in \mathbb{R}^{K_\psi}, \tag{3}$$

where $\mathcal{H}_t$ contains o_t and pre-window information that determines the perturbation distribution.

Matching action effects. We now propose the AFFECT loss to match the relationship of action perturbations and video features between O_t and $\hat{O}_t$. For each perturbation feature i and video feature j, we multiply the centered perturbation feature $g_{t,i}$ by the video feature $\phi_j(O_t)$ and average across windows: $\mathbb{E}[g_{t,i}\,\phi_j(O_t)]$. Perturbation features above their expected value receive positive weight, and those below it receive negative weight. The average product is therefore positive when larger perturbation features tend to accompany larger video features, and negative when they tend to accompany smaller ones. Averaging the features separately would lose this relationship.

We use all $K_\psi K_\phi$ pairs and train the model to match their action effects between recorded and generated videos:

$$d_{ij} := \frac{g_{t,i}}{s_{ij}}\left[\phi_j(O_t) - \phi_j(\hat{O}_t)\right], \tag{4}$$

$$\mathcal{L}_{\text{AFFECT}} := \sum_{i=1}^{K_\psi}\sum_{j=1}^{K_\phi}\left(\mathbb{E}[d_{ij}]\right)^2, \qquad \mathcal{L} := \mathcal{L}_{\text{flow}} + \lambda\mathcal{L}_{\text{AFFECT}}. \tag{5}$$

Here $\mathbb{E}[d_{ij}]$ is the average scaled difference between a recorded and a generated action effect. Each scale $s_{ij} > 0$ is the standard deviation of $g_{t,i}\,\phi_j(O_t)$ on recorded training windows, fixed before AFFECT training. The auxiliary loss therefore matches measured action effects across windows, without requiring each generated video to reproduce its particular recorded counterpart. We estimate the action-effect loss from minibatches using an unbiased estimator (Appendix A.6). In addition, to avoid a large computational overhead, we apply this loss periodically to reduce training cost.

3.3 Statistical interpretation of AFFECT

The two central design choices of Section 3.2, centering the perturbation features and multiplying them with video features, both have standard statistical interpretations. We develop them below to clarify what the objective measures and why it takes this form.

Action effects as covariances. We begin with centering. When g_t has zero conditional mean given $\mathcal{H}_t$, as equation 3 intends, each action effect is an averaged conditional covariance:

$$\mathbb{E}[g_{t,i}\,\phi_j(O_t)] = \mathbb{E}[\mathrm{Cov}(\psi_i(u_t), \phi_j(O_t) \mid \mathcal{H}_t)]\,. \tag{6}$$

Each term therefore measures how strongly video feature j moves together with perturbation feature i within the same scene. For instance, it is large and positive if stronger pushes move the object farther. The same property explains why centering suppresses static appearance, since any fixed video feature r whose value $r(o_t)$ is determined by the context alone satisfies $\mathbb{E}[g_t\,r(o_t)] = 0$ (Appendix B.3.1). Consequently, $\mathbb{E}[d_{ij}]$ compares the direction and strength of the response to actions between recorded and generated videos, while largely ignoring what the scene looks like. In a linear response model, for example, each term of the loss is proportional to the squared gap between the recorded and generated response coefficients, so it penalizes a model that ignores the action, overreacts, or responds in the wrong direction, and it is minimized only by the recorded response (Appendix B.2).

Feature matching on action effects. We now turn to the products, which connect AFFECT to feature matching. Standard feature matching compares the average video features of recorded and generated futures, whereas AFFECT compares how these averages change with the perturbation. This distinction matters in practice. Consider a model that moves the object left whenever the real object moves right. Since the perturbations and the object positions have the same distributions in both cases, matching average features cannot detect this error, whereas the product with the perturbation flips its sign. Formally, collecting the scaled products into a feature map $\Phi_t(O) \in \mathbb{R}^{K_\psi K_\phi}$ with entries $g_{t,i}\,\phi_j(O)/s_{ij}$, we can rewrite the loss as

$$\mathcal{L}_{\text{AFFECT}} = \left\|\mathbb{E}[\Phi_t(O_t)] - \mathbb{E}[\Phi_t(\hat{O}_t)]\right\|_2^2, \tag{7}$$

which is a finite-feature maximum mean discrepancy (MMD) (Gretton et al., 2012), the objective behind moment-matching generative models (Li et al., 2015; Dziugaite et al., 2015; Ren et al., 2016). This view motivates the unbiased estimator of Section 3.2, and it bounds the gap of any normalized combination of the selected tests (Appendix B.3.2).

Table 1: **Simulation evaluation results** Video fidelity metrics (PSNR, LPIPS, FVD) for single-window short horizon video generation and autoregressive long horizon video generation in the simulated UR5e setting. We also report the Pearson correlation of task success rates on pick-and-place tasks in OGBench between real and generated rollouts.

		Short Horizon			Long Horizon		
Method	Pearson ↑	PSNR ↑	LPIPS ↓	FVD ↓	PSNR ↑	LPIPS ↓	FVD ↓
iVideoGPT	0.334	21.53	0.49	1134	20.20	0.53	1314
CoCo	0.130	21.69	0.49	1094	20.29	0.52	1331
Ctrl-World	0.310	24.99	0.10	156	24.27	0.11	196
Cosmos3	0.091	19.07	0.19	470	18.72	0.22	562
+ Ours	0.334	19.95	0.18	463	19.21	0.20	540
DreamDojo	0.451	26.03	0.06	85	24.87	0.08	108
+ Ours	**0.686**	**26.78**	**0.05**	**73**	**25.58**	**0.07**	**93**

Table 2: **Real-world robot evaluation results** Video fidelity metrics for autoregressive long horizon video generation in the domain of real-world DROID and ALOHA-2 hardware configurations. We also report the Pearson correlation of task success rates between real and generated rollouts.

	DROID (Real)				ALOHA2 (Real)			
Method	PSNR ↑	LPIPS ↓	FVD ↓	Pearson ↑	PSNR ↑	LPIPS ↓	FVD ↓	Pearson ↑
iVideoGPT	16.16	0.52	1946	0.151	16.77	0.52	3808	0.155
CoCo	18.13	0.50	1676	0.212	19.09	0.49	3959	0.225
Ctrl-World	17.48	0.36	1141	0.236	20.32	0.11	396	0.399
Cosmos3	18.69	0.24	910	0.425	22.94	0.08	333	0.298
+ Ours	**19.67**	0.22	744	0.583	24.05	**0.05**	257	**0.694**
DreamDojo	18.53	0.16	807	0.468	24.20	0.06	**183**	0.325
+ Ours	19.04	**0.14**	**622**	**0.790**	**24.91**	**0.05**	188	0.490

4 EXPERIMENTS

We validate AFFECT on simulated and real-robot perturbation play. We design our experiments to investigate the following questions:

- Does AFFECT improve the visual/action fidelity of generated futures? (Tables 1 and 2)
- Does a more action-faithful world model help select better actions at test time? (Figure 3)
- Do the frozen features actually see the action effect? (Figure 4)

4.1 ROBOTICS WORLD MODELS FOR POLICY EVALUATION

Embodiments and Datasets. We consider three embodiments, each collected with the perturbation play protocol of Section 3.1. In simulation, we use a UR5e manipulation environment from OGBench (Park et al., 2025). On real robots, we use a Franka Research 3 in the hardware configuration proposed by DROID (Khazatsky et al., 2024) and a bimanual ALOHA-2 (Aldaco et al., 2024) setup. Every episode is a single unpaired rollout, since AFFECT never uses repeated executions from the same start. Further details are in Appendix A.1.

Implementation details. We apply AFFECT to two action-conditioned rectified-flow backbones, DreamDojo (Gao et al., 2026) and Cosmos3 (Agarwal et al., 2026), which represent the state-of-the-art among robotics world models. Both are post-trained from released weights on the same windows for each embodiment. For all baselines and AFFECT variants, they share the same training dataset, schedule, and initializations. We fire the AFFECT term every 8 iterations and set λ so that it is 15% of the flow loss on a short calibration run. Other details are in Appendices A.2–A.9.

Baselines. We compare against competing robot world models as baselines for AFFECT. First, iVideoGPT (Wu et al., 2024), is an autoregressive token-based model. CoCo (Shi et al., 2026), which adds counterfactual-consistency constraints to iVideoGPT, was reproduced by us as close as possible while following their paper. As noted in their paper, the ASC2 component of their paper is not suited for joint-space actions like in our settings, so we only implement the MSC2 component. Ctrl-World (Guo et al., 2026) is a video-diffusion world model for policy evaluation. We also report DreamDojo and Cosmos3 trained with the flow loss alone, which isolates the effect of AFFECT. All baselines are trained on the same perturbation-play corpora, and have a fixed budget, where they see the same amount of samples during training.

Evaluation. Following Ctrl-World (Guo et al., 2026), we evaluate the models in a closed loop, where policies generate actions, world models predict videos conditioned on those actions, and the generated observations are fed back into the policy. We report PSNR, LPIPS (Zhang et al., 2018), and FVD (Unterthiner et al., 2018) for a single prediction window (short horizon) and for autoregressive rollouts chained over multiple windows (long horizon). To measure action fidelity, we compute the Pearson correlation between real and generated rollout success rates across policy checkpoints and tasks. Each generated video receives a binary success label from human evaluators, with success labels only given when the task is clearly succeeded in the output video and the target object and environment remains visible throught the rollout. For the DROID and ALOHA-2 setup, we utilize GR00T N1.5 (NVIDIA GEAR, 2025), FLARE (Zheng et al., 2025), and DUST (Won et al., 2026) as policy models for action generation. For UR5e on the OGBench setup, we utilize GCIQL and GCIVL policies, which are both defined in OGBench (Park et al., 2025). For each embodiment, we utilize 12 episodes per 4 checkpoints across 2 tasks.

Simulation results. Table 1 reports results on UR5e. AFFECT improves both backbones on every reported metric, including video quality at both horizons and the correlation between real and imagined policy success rates. For DreamDojo, Pearson correlation increases from 0.451 to 0.686, while FVD decreases from 85 to 73 at the short horizon and from 108 to 93 at the long horizon. DreamDojo + AFFECT achieves the best score on every metric among the evaluated methods. For Cosmos3, correlation increases from 0.091 to 0.334, alongside smaller FVD reductions from 470 to 463 and from 562 to 540. These results show that the gains extend beyond single-window visual quality to policy evaluation and autoregressive rollout.

Real robot results. Table 2 reports results on the DROID and ALOHA-2 configurations. On DROID, AFFECT raises Pearson correlation from 0.468 to 0.790 for DreamDojo and from 0.425 to 0.583 for Cosmos3, while reducing FVD from 807 to 622 (23%) and from 910 to 744 (18%), respectively. Both backbones also improve in PSNR and LPIPS. On ALOHA-2, Cosmos3's correlation increases from 0.298 to 0.694, the highest among the evaluated methods, while its FVD decreases from 333 to 257. DreamDojo's correlation improves from 0.325 to 0.490, with PSNR increasing from 24.20 to 24.91 and LPIPS decreasing from 0.06 to 0.05, although FVD rises slightly from 183 to 188. Thus, AFFECT consistently improves agreement with real policy success rates, while generally improving visual metrics.

4.2 Robotics world models for planning

Evaluation setup. We test whether improved action fidelity helps select actions at test time using the proposal-selection scheme of DreamDojo (Gao et al., 2026). At the start of each execution horizon, 5 policy checkpoints trained on the target task are each queried for candidate action trajectories. Using the current camera frame and the candidate actions as input, the world model imagines future video for each actions, and a trained success estimator scores each video. The candidate action chunk with the highest score is executed open-loop before replanning, at which we repeat the process. We compare DreamDojo and DreamDojo + AFFECT with the same policy ensemble and success estimator. An additional baseline, Uniform, selects a candidate action at random rather than utilizing a world model. The score estimator uses frozen DINOv2 features with a light transformer head to predict task-success probability given video input. Further details are in Appendix A.10.

Tasks and Results. For the UR5e setup, we evaluate on success rate over 5 select pick-and-place tasks from OGBench, while we use a long-horizon task for ALOHA-2, consisting of 3 consecutive tasks where the robot must open a trash bin, and throw away two items on the table. Figure 3 shows that average UR5e success increases from 46.7% with Uniform to 60.0% with DreamDojo

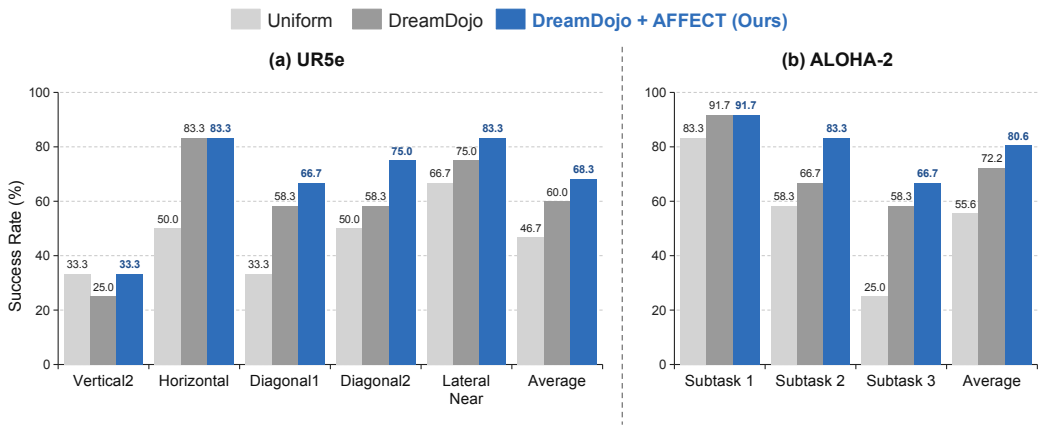


Figure 3: **Planning with action-faithful world models.** Success rates (%) of plans selected with each world model on (a) UR5e and (b) ALOHA-2, where one long-horizon task consists of three consecutive subtasks. Candidate action chunks are proposed by an ensemble of five policy checkpoints. Uniform picks one at random, while DreamDojo and DreamDojo + AFFECT imagine each chunk and select the one whose imagined video scores highest under a learned success estimator.

and 68.3% with DreamDojo + AFFECT, an additional gain of 8.3 percentage points. ALOHA-2 results also show that AFFECT provides a better environment for planning over the baseline world model, with an average gain of 8.4 percentage points. With the proposal ensemble and success estimator held fixed, the improvements show that while the usefulness of action-effect matching for improving model-based planning.

4.3 Analysis and Ablations

Do the frozen features see the action effect? The frozen video features detect action effects primarily where the robot changes the scene (Figure 4). We establish this on recorded UR5e videos by measuring the consistency of perturbation–feature associations at each latent cell. We check whether products of perturbation and video features have a consistent sign across episodes, then average over feature pairs. We also evaluate on a baseline setup that randomly pairs perturbations with videos from other windows, breaking the link between action perturbations and outcomes. Responses are 1.6–2.2 times stronger than this shuffled baseline in regions traversed by the arm, cube, or shadow, which is where action-induced visual change is concentrated, but remain at chance on the static table and backdrop. This confirms that the features provide a localized action-effect signal for AFFECT.

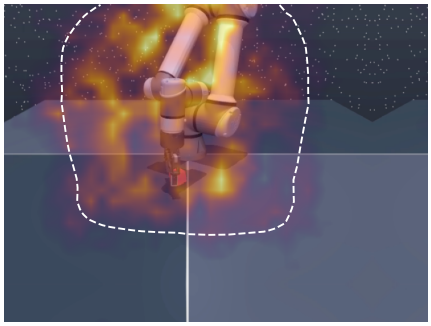


Figure 4: **Action-effect localization.** Frozen-feature responses on UR5e relative to shuffled pairs, where brighter cells indicate stronger responses.

Ablation studies. We compare six Cosmos3 configurations after 10k training iterations on perturbation play data, evaluated on video fidelity metrics (PSNR, LPIPS, FVD) to examine the role of action–outcome correspondence, perturbation weighting, and training-data quantity (Table 3). Alongside the flow-only baseline and Cosmos3 + AFFECT, we consider two controls. *Placebo matching* shuffles perturbation features g_t across episodes, isolating the importance of correct action–outcome correspondence. *Unconditional feature matching* sets $g_t \equiv \mathbf{1}$ to match mean video features, isolating the contribution of perturbation weighting beyond generic feature matching. Finally, we train the flow-only baseline and Cosmos3 + AFFECT on the same 20% subset of the training data to assess the method under limited data.

Results in Table 3 show that both controls show degraded video fidelity, supporting the importance of both preserving action-outcome correspondence and suing perturbation-weighted features. Meanwhile, AFFECT trained under data constraints shows a lower performance gain on its base-

Table 3: **Ablations.** We report long-horizon video metrics (PSNR, LPIPS, FVD) for Cosmos3 using the same closed-loop policy evaluation system as Table 2 after 10,000 training iterations, varying the auxiliary objective and training-data quantity.

Method	PSNR ↑	LPIPS ↓	FVD ↓
Cosmos3	23.43	0.07	284
w/ Placebo matching	21.80	0.09	460
w/ Unconditional matching	21.72	0.09	451
w/ AFFECT (Ours)	**24.12**	**0.06**	**256**
20% training data			
Cosmos3	22.58	0.08	405
w/ AFFECT (Ours)	22.97	0.08	393

line compared to the full data run, showing that its performance gain increases with further action distribution coverage as expected, while still helping at lower data scales.

5 Related works

Robotics world models (RWMs). Learned world models began as compact latent dynamics models used for planning or actor-critic learning in imagination (Ha & Schmidhuber, 2018; Hafner et al., 2019; 2025; Hansen et al., 2024), and have since become large video generators conditioned on actions or text (Yang et al., 2024; Du et al., 2023; Bruce et al., 2024; Parker-Holder & Fruchter, 2025; Valevski et al., 2025; Alonso et al., 2024). For manipulation, the dominant recipe is to post-train a general pretrained video diffusion or flow model with action conditioning (Zhu et al., 2024; Wu et al., 2024; Huang et al., 2026; Guo et al., 2026). These models are used to generate data (Jang et al., 2025), to plan (Du et al., 2024; Bar et al., 2025), and increasingly to evaluate policies without actual real-world rollout (Li et al., 2025; Jeon et al., 2026; Team et al., 2026).

Action fidelity in world models. A growing body of mostly concurrent work finds that RWMs often render plausible videos that ignore their action condition. Benchmarks pair successful and failure-inducing actions from a fixed scene (Yang et al., 2026; Shi et al., 2026) or score robot-object interaction directly (Jiang et al., 2026), and position papers argue that counterfactual fidelity rather than visual quality should be the deciding criterion (Wang et al., 2026c; Yu et al., 2026). Inverse-dynamics probes (Li et al., 2025; Wang et al., 2026b; Yeom et al., 2026) test action recoverability from generated video, and similar ideas have been turned into training objectives that reconstruct the action or separate predictions made under different actions (Ye et al., 2025; Han & Yilmaz, 2026; Gan et al., 2026). Because these objectives can be satisfied by distinguishable robot motion alone, they do not guarantee correct robot-object interaction. Other methods supervise outcomes directly, through replayed off-expert actions in simulation (Chen et al., 2026) or through additional outcome labels and rewards (Zhou & Zhang, 2026; Wang et al., 2026a), which limits how far they scale on real robots. AFFECT needs neither paired counterfactual rollouts nor outcome labels.

6 Conclusion

We presented AFFECT, an auxiliary training objective that trains robot world models to learn how action perturbations change future observations. Using perturbation play rollouts with logged perturbations and action effect encoders to extract video/perturbation features, it adds action effect supervision without outcome labels, repeated executions from the same state, or architectural changes. Across DreamDojo and Cosmos3 backbones on UR5e, DROID, and ALOHA-2 hardware, AFFECT improves the correlation between real and imagined policy success rates, with improvements in most visual metrics. The additional model-based planning results further suggest that these gains can translate into better action selection.

AI USE STATEMENT

In this work, we used generative AI tools for editing and proofreading the manuscript text, including fixing typographical errors and checking the LaTeX source against the formatting instructions, and upscaling image resolutions for showcasing task settings. We have reviewed all AI-assisted work, and every suggested edit was checked against the source and results by the authors. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work trains video world models for robot manipulation and uses them to evaluate robot policies. All real-robot data were collected by the authors on a Franka Research 3 and a bimanual ALOHA-2 in a laboratory setting, using the perturbation-play protocol of Section 3.1. Human involvement was limited to teleoperated play episodes used to fine-tune the data-collection policy and to resetting objects that left the workspace (Appendix A.1). We are not aware of other ethical concerns specific to this work.

REPRODUCIBILITY STATEMENT

Section 3 and Appendix A specify AFFECT in full. The data-collection protocol and per-robot configurations are in Appendix A.1. The backbones, their action inputs, and warm starts are in Appendix A.2. The feature bank, estimator, scales, sampler, and training schedule are in Appendices A.5–A.9. The planning setup is in Appendix A.10. Appendix B states the assumptions behind the objective. Baselines and AFFECT variants share data, initialization, and schedule, differing only in the auxiliary loss (Section 4.1).

REFERENCES

Niket Agarwal, Arslan Ali, Jon Allen, Martin Antolini, Adeline Aubame, Alisson Azzolini, Junjie Bai, Maciej Bala, Yogesh Balaji, Josh Bapst, et al. Cosmos 3: Omnimodal world models for physical ai. *arXiv preprint arXiv:2606.02800*, 2026.

Jorge Aldaco, Travis Armstrong, Robert Baruch, Jeff Bingham, Sanky Chan, Kenneth Draper, Debidatta Dwibedi, Chelsea Finn, Pete Florence, Spencer Goodrich, et al. Aloha 2: An enhanced low-cost hardware for bimanual teleoperation. *arXiv preprint arXiv:2405.02292*, 2024.

Arslan Ali, Junjie Bai, Maciej Bala, Yogesh Balaji, Aaron Blakeman, Tiffany Cai, Jiaxin Cao, Tianshi Cao, Elizabeth Cha, Yu-Wei Chao, et al. World simulation with video foundation models for physical ai. *arXiv preprint arXiv:2511.00062*, 2025.

Eloi Alonso, Adam Jelley, Vincent Micheli, Anssi Kanervisto, Amos Storkey, Tim Pearce, and François Fleuret. Diffusion for world modeling: Visual details matter in atari. In *Advances in Neural Information Processing Systems*, 2024.

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

Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2025.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *International Conference on Machine Learning*, 2024.

Sixiang Chen, Jiaming Liu, Jixian Wu, Yichen Guo, Tinghao Wang, Siyuan Qian, Hao Chen, Jiajun Cao, Jian Tang, and Shanghang Zhang. Do robotic world models really follow actions? diagnosing and aligning action-conditioned generation for policy learning. *arXiv preprint arXiv:2608.24885*, 2026.

Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. In *Advances in Neural Information Processing Systems*, 2023.

Yilun Du, Sherry Yang, Pete Florence, Fei Xia, Ayzaan Wahid, Pierre Sermanet, Tianhe Yu, Pieter Abbeel, Joshua B Tenenbaum, Leslie Kaelbling, et al. Video language planning. In *International Conference on Learning Representations*, 2024.

Gintare Karolina Dziugaite, Daniel M. Roy, and Zoubin Ghahramani. Training generative neural networks via maximum mean discrepancy optimization. In *Uncertainty in Artificial Intelligence (UAI)*, 2015.

Zhenfeng Gan, ZiTong Zeng, Jiajun Cheng, Yeke Song, Yongyi Tang, and Xueqian Wang. Actswm: Action-sensitive world models for long-horizon planning in open-world games. *arXiv preprint arXiv:2607.26712*, 2026.

Shenyuan Gao, William Liang, Kaiyuan Zheng, Ayaan Malik, Seonghyeon Ye, Sihyun Yu, Wei-Cheng Tseng, Yuzhu Dong, Kaichun Mo, Chen-Hsuan Lin, et al. Dreamdojo: A generalist robot world model from large-scale human videos. In *International Conference on Machine Learning*, 2026.

Arthur Gretton, Karsten M Borgwardt, Malte J Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *The journal of machine learning research*, 13:723–773, 2012.

Yanjiang Guo, Lucy Xiaoyang Shi, Jianyu Chen, and Chelsea Finn. Ctrl-world: A controllable generative world model for robot manipulation. In *International Conference on Learning Representations*, 2026.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In *Advances in Neural Information Processing Systems*, 2018.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, 2019.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, 2025.

Yuci Han and Alper Yilmaz. Enhancing policy learning with world-action model. *arXiv preprint arXiv:2603.28955*, 2026.

Nick Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024.

Siqiao Huang, Jialong Wu, Qixing Zhou, Shangchen Miao, and Mingsheng Long. Vid2world: Crafting video diffusion models to interactive world models. In *International Conference on Learning Representations*, 2026.

Joel Jang, Seonghyeon Ye, Zongyu Lin, Jiannan Xiang, Johan Bjorck, Yu Fang, Fengyuan Hu, Spencer Huang, Kaushil Kundalia, Yen-Chen Lin, et al. Dreamgen: Unlocking generalization in robot learning through video world models. In *Conference on Robot Learning*, 2025.

Byeongguk Jeon, Seonghyeon Ye, JaeHyeok Doo, Sungdong Kim, Minjoon Seo, Hyungmok Son, and Kimin Lee. Roboworld: Fast and reliable neural simulators for generalist robot policy evaluation. *arXiv preprint arXiv:2607.01060*, 2026.

Feng Jiang, Yang Chen, Kyle Xu, Yuchen Liu, Haifeng Wang, Zhenhao Shen, Jasper Lu, Shengze Huang, Yuanfei Wang, Chen Xie, et al. Robowm-bench: A benchmark for evaluating world models in robotic manipulation. *arXiv preprint arXiv:2604.19092*, 2026.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

Weijie Kong, Qi Tian, Zijian Zhang, Rox Min, Zuozhuo Dai, Jin Zhou, Jiangfeng Xiong, Xin Li, Bo Wu, Jianwei Zhang, et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

Yaxuan Li, Yichen Zhu, Junjie Wen, Chaomin Shen, and Yi Xu. Worldeval: World model as real-world robot policies evaluator. *arXiv preprint arXiv:2505.19017*, 2025.

Yujia Li, Kevin Swersky, and Richard Zemel. Generative moment matching networks. In *International Conference on Machine Learning*, 2015.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on Robot Learning*, 2020.

NVIDIA GEAR. GR00T N1.5: An improved open foundation model for generalist humanoid robots. https://research.nvidia.com/labs/gear/gr00t-n1_5, 2025.

Maxime Oquab, Timothée Darcet, Theo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Russell Howes, Po-Yao Huang, Hu Xu, Vasu Sharma, Shang-Wen Li, Wojciech Galuba, Mike Rabbat, Mido Assran, Nicolas Ballas, Gabriel Synnaeve, Ishan Misra, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. Dinov2: Learning robust visual features without supervision, 2023.

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

Jack Parker-Holder and Shlomi Fruchter. Genie 3: A new frontier for world models. Google DeepMind blog, https://deepmind.google/blog/genie-3-a-new-frontier-for-world-models/, 2025.

Yong Ren, Jun Zhu, Jialian Li, and Yucen Luo. Conditional generative moment-matching networks. In *Advances in Neural Information Processing Systems*, 2016.

Yuhong Shi, Zhenhao Chu, Jie Wei, Jun Hao, Jianyi Liu, and Jingwen Fu. Overcoming statistical bias in action-controllable world models. *arXiv preprint arXiv:2608.04653*, 2026.

GigaWorld Team, Angyuan Ma, Boyuan Wang, Bohan Li, Chaojun Ni, Guo Li, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, et al. Gigaworld-1: A roadmap to build world models for robot policy evaluation. *arXiv preprint arXiv:2607.02642*, 2026.

Thomas Unterthiner, Sjoerd Van Steenkiste, Karol Kurach, Raphael Marinier, Marcin Michalski, and Sylvain Gelly. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.

Dani Valevski, Yaniv Leviathan, Moab Arar, and Shlomi Fruchter. Diffusion models are real-time game engines. In *International Conference on Learning Representations*, 2025.

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

Han Wang, Zijun Wang, Shuoshuo Xue, Rui Cao, Fengjiao Cheng, Xiaodan Liang, and Roy Ka-Wei Lee. Pavxplorerl: Physical-action-visual world model reinforcement learning with action exploration. *arXiv preprint arXiv:2607.16602*, 2026a.

Ruixiang Wang, Qingming Liu, Yueci Deng, Guiliang Liu, Zhen Liu, and Kui Jia. Eva: Aligning video world models with executable robot actions via inverse dynamics rewards. *arXiv preprint arXiv:2603.17808*, 2026b.

Xin Wang, Wenxuan Liu, Tongtong Feng, and Wenwu Zhu. Autonomous video generation with counterfactual controllability for self-evolving world models. *arXiv preprint arXiv:2606.24152*, 2026c.

John Won, Kyungmin Lee, Huiwon Jang, Dongyoung Kim, and Jinwoo Shin. Dual-stream diffusion for world-model augmented vision-language-action model. In *International Conference on Machine Learning*, 2026.

Jialong Wu, Shaofeng Yin, Ningya Feng, Xu He, Dong Li, Jianye Hao, and Mingsheng Long. ivideogpt: Interactive videogpts are scalable world models. In *Advances in Neural Information Processing Systems*, 2024.

Mengjiao Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. In *International Conference on Learning Representations*, 2024.

Tianzhuo Yang, Zihan Shen, Zirui Mi, Zhaoyi Zhang, Jiayi Zhou, Jiaming Ji, Juntao Dai, Jiawei Chen, Boyuan Chen, and Yaodong Yang. Mirabench: Evaluating action-conditioned reliability in robotic world models. *arXiv preprint arXiv:2605.29360*, 2026.

Yang Ye, Tianyu He, Shuo Yang, and Jiang Bian. Reinforcement learning with inverse rewards for world model post-training. *arXiv preprint arXiv:2509.23958*, 2025.

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.

Tenny Yin, Zhiting Mei, Zhonghe Zheng, Miyu Yamane, David Wang, Jade Sceats, Samuel M Bateman, Lihan Zha, Apurva Badithela, Ola Shorinwa, et al. Playworld: Learning robot world models from autonomous play. *arXiv preprint arXiv:2603.09030*, 2026.

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.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

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

Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. In *Advances in Neural Information Processing Systems*, 2023b.

Ruijie Zheng, Jing Wang, Scott Reed, Johan Bjorck, Yu Fang, Fengyuan Hu, Joel Jang, Kaushil Kundalia, Zongyu Lin, Loic Magne, et al. Flare: Robot learning with implicit world modeling. *arXiv preprint arXiv:2505.15659*, 2025.

Todd Y Zhou and Daniel Zhang. Learning counterfactual world models for embodied reasoning under partial observability. *arXiv preprint arXiv:2609.05834*, 2026.

Fangqi Zhu, Hongtao Wu, Song Guo, Yuxiao Liu, Chilam Cheang, and Tao Kong. Irasim: Learning interactive real-robot action simulators. *arXiv preprint arXiv:2406.14540*, 2024.

A AFFECT IMPLEMENTATION DETAILS

This appendix specifies more implementation details on AFFECT. Section A.1 describes how the training corpus is collected. Section A.2 describes the two video backbones, and Section A.3 names the quantities used by the loss. Sections A.4–A.9 give the objective, the feature banks, the estimator, the per-test scales, the sampler and the training schedule.

A.1 DATA COLLECTION BY PERTURBATION PLAY ON REAL ROBOTS

AFFECT is trained on *perturbation play*. A policy is run closed-loop on the real robot, and every command it issues is perturbed by noise drawn from a process we control and log. Neither demonstrations of the target task nor paired rollouts are needed.

Play policy and rollouts. The nominal behaviour comes from a generalist imitation policy fine-tuned on a small set of human *play* teleoperation episodes, in which an operator interacts freely with a handful of tabletop objects instead of repeating one task. We use GR00T-N1.5 with the FLARE objective (NVIDIA GEAR, 2025; Zheng et al., 2025) on both robots. At each replanning point the policy observes the cameras and the proprioceptive state, emits a chunk of absolute joint-position targets, and the controller executes the first H steps before the policy is queried again. Episodes are rolled to a fixed step budget and never gated on success. Objects are left where the previous rollout put them, and a human intervenes only to return objects that leave the workspace, so coverage grows by drift, in the spirit of autonomous play collection (Yin et al., 2026). Failed grasps, collisions and knocked-over objects are kept, since they are action-dependent outcomes the model has to learn.

Noising. Let $\bar{A}_t \in \mathbb{R}^D$ be the nominal target at step t. We draw a perturbation u_t and execute

$$A_t \;=\; \mathrm{clip}\big(\bar{A}_t + u_t\big), \tag{8}$$

where clip enforces the joint limits and a per-step velocity limit. The noise is injected after the policy, in the absolute joint-target coordinates the robot is commanded in, so its law is specified independently of the policy. Within an episode the recipe is as follows.

- **Declared scale.** Each episode declares an amplitude ρ drawn from a small ladder (five rungs on FR3) and a fixed per-dimension shape $c \in \mathbb{R}^D$, so that the per-dimension scale is $s = \rho\, c$. The shape is the per-dimension standard deviation of $\bar{A}_t - \bar{A}_{t-1}$ on the play corpus, normalised to mean one, and is frozen once for the corpus. It is not re-fitted to the realized noise, since a scale fitted to the same u_t would turn the magnitude tests of Section A.5 into constants.
- **Temporal correlation.** Per-step white noise averages out within a chunk and only makes the arm jitter in place. We therefore draw white Gaussian noise for the whole episode, smooth it along time with a Gaussian filter of width 4 steps, re-standardise each dimension, and multiply by s.
- **Grippers are not perturbed.** The gripper slots of u_t are zero, because gripper noise produces failure modes that no policy produces.
- **Independent streams.** Each episode draws from its own generator, seeded from the arm and episode indices. The $\rho = 0$ arm is the unmodified policy.

Logged quantities. For every step we log the nominal command $\bar{A}_t$, the noise u_t *before clipping*, the executed command A_t, the number of clipped dimensions, the joint state and the position within the chunk. For every episode we log ρ, c, the smoothing width, the seed and one video per camera aligned with A_t. The noise is logged before clipping so that it follows the specified noise process. Clipping is rare at our scales (on FR3, at most 10^{-4} clipped dimensions per step on average). The scale is declared rather than estimated, because one batch mixes several amplitude rungs and a batch estimate would make the centering data-dependent. Because the declared parameters are part of $\mathcal{H}_t$, pooling rungs keeps each perturbation feature centered within every rung.

The two instantiations.

	FR3 (DROID rig)	ALOHA-2
Robot	Franka Research 3, 7 joints + parallel gripper (Khazatsky et al., 2024)	two 6-DoF arms + two grippers, $D = 14$ (Zhao et al., 2023a; Aldaco et al., 2024)
Perturbed dims	7 arm joints (gripper zero)	12 arm joints (both grippers zero)
Cameras	two exterior + wrist, 256×192	high, low, two wrist, 480×640
Control rate	10 Hz	25 Hz

A.2 ACTION-CONDITIONED RECTIFIED-FLOW BACKBONES

We train AFFECT on two backbones that differ in almost every architectural respect. Both are conditional video generators post-trained from released weights, and both consume the same windows through the same loader.

Window and latent. A training example is a window of $T+1$ consecutive frames $x_{0:T}$ from one camera and the T executed commands $a_{1:T}$ between them ($T = 12$ at 10 Hz on FR3, so a window is 1.2 s). A frozen causal video tokenizer $\mathcal{E}$ with temporal compression 4 maps the window to a latent $z = \mathcal{E}(x_{0:T}) \in \mathbb{R}^{c\times(1+T/4)\times h\times w}$. For $T = 12$ this gives four latent frames, the first of which encodes the current frame. All video features in Section A.5 are functions of this latent, and O_t and $\hat{O}_t$ there denote the recorded and generated latents.

Action input. The loader applies a fixed operator Δ to the executed commands and passes $\Delta(a_{1:T}) \in \mathbb{R}^{T\times D}$ to the model. The operator expresses each four-step chunk relative to the step before the chunk and standardises each dimension by a corpus-level statistic. The two backbones then ingest the chunk differently.

	DreamDojo (Cosmos-Predict2, 2B)	Cosmos3-Edge (4B)
Architecture	video DiT, flow matching (Gao et al., 2026; Ali et al., 2025)	omnimodal mixture-of-transformers with a frozen reasoning tower and a trained generation tower (Agarwal et al., 2026)
Latent for 480×640	$16 \times 4 \times 60 \times 80$	$48 \times 4 \times 30 \times 40$ (Wan2.2 VAE, Wan et al., 2025)
Frame conditioning	frame replacement, in which the first one or two latent frames are overwritten with the ground truth at every denoising step and excluded from the loss	causal sequence with the first latent frame clean and later frames generated autoregressively over the latent frames
Action conditioning	the whole chunk $\Delta(a_{1:T})$ is written into a fixed slot of a 384-wide vector and passed through two MLPs whose outputs are added to the timestep embedding	the actions driving the transition $z_i \to z_{i+1}$ are projected by an embodiment-indexed linear map and interleaved as clean tokens before the frame they drive
Text	empty caption	one fixed sentence per embodiment
Warm start	DreamDojo GR-1 post-train	released Cosmos3-Edge
Trainable	all	generation tower and action I/O (reasoner frozen)

Objective and sampler. Both are rectified-flow models (Liu et al., 2022; Lipman et al., 2023). With $\varepsilon \sim \mathcal{N}(0, I)$ and a noise level $\tau \in [0, 1]$, the network sees $\tilde{O}_t = \tau\varepsilon + (1-\tau)O_t$ and regresses the velocity

$$\mathcal{L}_{\text{flow}} \;=\; \mathbb{E}\left\|W_\theta(\tilde{O}_t, \tau, o_t, A_t) - (\varepsilon - O_t)\right\|^2, \tag{9}$$

where o_t contains the conditioning observations and text, and A_t is the executed command sequence, represented by $\Delta(a_{1:T})$. Training draws the noise level from a time-shifted schedule with shift $\kappa = 5$, and generation integrates the same probability-flow ODE with an Euler scheme on that schedule. The two implementations of AFFECT share all settings except the pooling grid of the full-random features (Section A.5), and each baseline differs from its AFFECT counterpart only in the loss.

A.3 SETTING AND DATA

AFFECT adds one auxiliary training term and changes nothing else. The architecture, the model's inputs, the deployed sampler and the conditioning data all stay the same, so a trained model is evaluated exactly like its baseline. The loss uses the quantities below.

Symbol	Name	Description
o_t	context	The conditioning frames available to the model at the start of the window.
A_t	action	The executed command sequence, represented by the loader as $\Delta(a_{1:T})$.
u_t	perturbation	The noise assigned to the commands in this window, an $H \times D$ block, logged before clipping.
O_t	recorded future	The recorded video for this window, encoded to the model's latent.
$\hat{O}_t$	generated future	A future generated from fresh noise under the same o_t and A_t.

The perturbation u_t is never a model input. The model receives the executed command, as it would at deployment, and u_t is used only to build the loss. A model that exactly reproduces $p(O_t \mid o_t, A_t)$ then attains zero AFFECT loss whenever o_t and A_t carry all the information about the outcome that u_t and $\mathcal{H}_t$ carry. This holds only approximately in our data, since the play policy observes all cameras and the proprioceptive state while the world model is conditioned on its own frames, so the auxiliary loss can shift the optimum away from that of the flow loss. Each example also carries the declared per-dimension scale s of its episode. No paired counterfactual futures are required, because the objective aggregates evidence across windows instead of differencing two futures from one state.

A.4 The objective

A single *test* is a pair (g, f) of a centered scalar perturbation feature $g(u_t)$ and a fixed scalar video feature $f(\cdot)$. Section 3.2 indexes all pairs by (i, j), with g and f the i-th entry of g_t and the j-th entry of ϕ. Here the same pairs are indexed $b \in \mathcal{B}$, with $|\mathcal{B}| = K_\psi K_\phi$. The per-example discrepancy and the objective are

$$d_b \;=\; \frac{g_b(u_t)}{s_b}\,\big[f_b(O_t) - f_b(\hat{O}_t)\big], \qquad \mathcal{L}_{\text{AFFECT}} \;=\; \sum_{b\in\mathcal{B}} \big(\mathbb{E}[d_b]\big)^2, \qquad \mathcal{L} \;=\; \mathcal{L}_{\text{flow}} + \lambda\,\mathcal{L}_{\text{AFFECT}}. \tag{10}$$

Two properties motivate this form. First, when g_b has zero conditional mean given the pre-window information, an additive term of f_b that is fixed before the window contributes nothing to $\mathbb{E}[d_b]$, so the statistic measures what the action changed rather than what the scene looked like (Appendix B.3.1). Second, the optimum is the observed action effect, not an extreme. For a nonzero recorded effect, an attenuated and an exaggerated generated effect produce discrepancies of opposite sign, so the loss increases on both sides. Objectives that reward action recoverability or the separation of futures under different actions lack this property, since a larger response always helps them. As in Section 3.3, the loss is a squared finite-feature MMD with feature map $\Phi_b = g_b f_b / s_b$ (Appendix B.3.2).

A.5 The feature banks

The bank is the full product $\{g\} \times \{f\}$ of the outputs of the two frozen encoders of equation 2, whose matrices P and nonlinearities η are chosen to compute the statistics below. Nothing in either encoder is learned, and the whole bank is reproduced from two integer seeds.

Perturbation features g. Let $u_{t,j}$ be the noise at step t in dimension j, and s_j the declared scale. For each perturbed dimension we use a *signed* and a *magnitude* feature,

$$g_j^{\text{sgn}} \;=\; \frac{1}{s_j\sqrt{H}} \sum_{t=1}^{H} u_{t,j}, \qquad g_j^{\text{mag}} \;=\; \frac{1}{s_j^2\sqrt{2H}} \Big(\sum_{t=1}^{H} u_{t,j}^2 - H s_j^2 \Big), \tag{11}$$

plus their counterparts pooled over all perturbed dimensions, $g^{\text{sgn}} = (HD)^{-1/2} \sum_{t,j} u_{t,j}/s_j$ and $g^{\text{mag}} = (2HD)^{-1/2} \big(\sum_{t,j} u_{t,j}^2/s_j^2 - HD \big)$. The signed features measure the directional response, and the magnitude features measure effects that depend on how large the perturbation was, which a signed feature cannot see. Dividing by s_j makes the features comparable across dimensions and amplitudes. The features are undefined for the $\rho = 0$ arm, whose windows do not enter the normalized tests.

Video features f. The video features are fixed reductions of the model's own latent, of shape (channels $\times$ time $\times$ height $\times$ width), and are applied identically to the recorded and generated futures.

Family	Definition
channel	For the first four channels, the change in the spatial mean from the first latent frame to the last, and to the second.
projection	The same two differences, and the squared end difference, along 8 fixed unit-norm random directions in the channel axis.
flow	Mean, max, path length, late-window mean and spatial spread of the absolute temporal difference of the latent.
energy	Mean squared latent value over the whole window.
full-random	The latent is average-pooled to a fixed grid (12×16 on DreamDojo, 15×20 on Cosmos3), the first-to-last difference is flattened and projected onto 8 fixed unit-norm random directions. Both the signed projection and its square are used.

Bank size. On FR3 there are 16 perturbation features (7 signed and 7 magnitude over the arm joints, plus the two pooled) and 54 video features (8 channel, 24 projection, 5 flow, 1 energy and 16 full-random), for 864 tests. The UR5e configuration has 192.

Conditioning frames. Under DreamDojo's frame-replacement conditioning, the generated conditioning frames equal the recorded latent exactly, so their contributions cancel in the linear video features.

A.6 The estimator

Squaring the batch mean of d has expectation $(\mathbb{E}[d])^2 + \mathrm{Var}(d)/n$. The extra term rewards reducing the variability of the generated futures, which pushes toward mode collapse. We instead use the unbiased U-statistic

$$\widehat{(\mathbb{E}[d])^2} = \frac{\left(\sum_i d_i\right)^2 - \sum_i d_i^2}{n(n-1)}, \tag{12}$$

the paired form of the unbiased squared-MMD estimator (Gretton et al., 2012). Its expectation is exactly $(\mathbb{E}[d])^2$ for independent d_i, so it removes the variance penalty, although it does not by itself guarantee sample diversity.

The index i runs over independent data units rather than rows. Windows from one episode are dependent, and in our paired corpora the amplitude rungs of one family share a noise seed. With dependent units, the expectation would additionally contain the average cross-covariance. The estimator therefore sums d within an independent unit of data, which is a single rollout episode in our case, before applying equation 12.

A.7 Per-test scales

Each test is divided by a constant s_b, the standard deviation of $g_b f_b(O_t)$ over recorded windows. This is the scale of the test statistic under the recorded data alone, and it can be computed before any model is trained. We compute it with a pass of the ordinary training loop at $\lambda = 0$, so the scales see exactly the windows, window length and weighting used in training, and we freeze them before AFFECT training. A per-batch scale would change the estimator and its expectation. Fixed scales change the relative weight of the tests but not the zero set of the loss. The video features are fixed for a similar reason: minimizing jointly over unrestricted features admits $f_b \equiv 0$, which makes the loss zero for any generator. Tests with zero spread on the recorded data are dropped rather than floored.

A.8 A differentiable sampler for $\hat{O}_t$

$\hat{O}_t$ must be a sample generated from fresh noise, and the gradient must reach the network through generation. We use an Euler discretization of the probability-flow ODE the model is trained on.

With time shift κ and N steps, the schedule is

$$\sigma(\tau) = \frac{\kappa\,\tau}{1 + (\kappa - 1)\tau}, \qquad \tau = 1, \tfrac{N-1}{N}, \ldots, 0, \tag{13}$$

and the update is $x \leftarrow x + (\sigma_{k+1} - \sigma_k)\, W_\theta(x, \sigma_k, o_t, A_t)$. We use $N = 2$ and $\kappa = 5$, matching the training schedule and the deployed sampler, and checkpoint the gradient of each network call. The recorded future is detached, so the gradient of the term flows only through the generated side.

Two cheaper alternatives are not valid substitutes. The denoiser's clean prediction from the training-noised latent is built from the recorded O_t, so its discrepancy vanishes as $\sigma \to 0$ whether or not the action is used. A single Euler step from pure noise returns, at the population optimum, the conditional mean $\mathbb{E}[O_t \mid o_t, A_t]$. It has no sampling variance, so for a squared projection Z^2 it omits the term $\mathrm{Var}(Z \mid o_t, A_t)$ of $\mathbb{E}[Z^2 \mid o_t, A_t]$, and it matches mean effects rather than distributions.

A.9 Integration into training

At a step where the term is applied, it takes the $m = 4$ examples of the micro-batch, generates $\hat{O}_t$ for each, computes both sets of features, and adds λ times the sum of equation 12 over tests to the flow loss, which is unchanged. The term is applied every 8 iterations, because each application costs N extra forward passes.

A.10 Planning protocol and success estimator

Proposal generation. Following the proposal-selection scheme of DreamDojo (Gao et al., 2026), we use five checkpoints from one policy training run to provide diverse action proposals. On UR5e, these are goal-conditioned, state-based GCIQL policies trained on a play dataset of OGBench (Park et al., 2025). For each candidate, the world model predicts 24 future frames; the highest-scoring chunk is executed open-loop, and selection repeats from the next observation.

Success estimator. DreamDojo uses a head on frozen DINOv2 features to regress the number of steps to the next subtask boundary. We retain the architecture but predict task-success probability, removing the need for subtask annotations. Frames are encoded by frozen DINOv2-base (Oquab et al., 2023). Each frame contributes 13 tokens: the CLS token and the latent patch grid average-pooled to 3×4. A clip contains four frames sampled every second frame from the 10 Hz feature stream. Features are projected to 256 dimensions with token and time embeddings, and a learned task embedding is added to a summary token. A two-layer transformer maps each clip to a success logit. The chunk score is the mean clip success probability over the 25-frame sequence comprising the current frame and 24 predicted frames.

Comparisons DreamDojo and DreamDojo + AFFECT share the proposal ensemble and success estimator, so the comparison changes only the world model used to imagine candidate futures. Uniform selects one proposal uniformly at random and serves as the ensemble reference without world-model selection. The results are consistent with imagined futures providing a more useful distinction among candidate actions when trained with action-effect matching.

B Rationale for perturbation–response matching

This appendix explains why AFFECT multiplies centered perturbation features by video features, and what the resulting loss does and does not constrain. Appendix B.1 shows that each product measures a covariance between the perturbation and the future. Appendix B.2 works through a simple response model to show what matching these covariances enforces. Appendix B.3 then states the formal results, with their assumptions and a counterexample. One caveat applies throughout: AFFECT only constrains responses that the chosen features can detect, not every physically relevant response.

B.1 Each product measures a covariance

We use the notation of Section 3. The conditioning set $\mathcal{H}_t$ collects everything fixed before the window: the context frames o_t and the parameters of the perturbation law, such as the declared noise

scale. It does not contain the perturbation u_t, the executed action A_t, or the future. Consider a single test, made of a scalar perturbation feature $g = g(u_t; \mathcal{H}_t)$, centered so that $\mathbb{E}[g \mid \mathcal{H}_t] = 0$, and a fixed scalar video feature f. Because g has zero conditional mean, $\mathbb{E}[gf(O_t) \mid \mathcal{H}_t] = \mathrm{Cov}(g, f(O_t) \mid \mathcal{H}_t)$. Subtracting the same quantity for the generated future and averaging over contexts gives

$$\mathbb{E}\Big[g\{f(O_t) - f(\hat{O}_t)\}\Big] = \mathbb{E}\Big[\mathrm{Cov}(g, f(O_t) \mid \mathcal{H}_t) - \mathrm{Cov}(g, f(\hat{O}_t) \mid \mathcal{H}_t)\Big]. \tag{14}$$

In words, each test asks whether the video feature moves with the perturbation equally strongly in recorded and generated futures, comparing within a context and then averaging over contexts.

Two consequences follow. First, the recorded and generated futures share the same windows and perturbations, so any error that depends only on the context cancels. For example, if the model renders every window of some scene slightly too dark, that offset does not enter the test (Proposition 1 in Appendix B.3.1). Second, the test is only as informative as its features. Finite-sample noise remains, and a video feature that mostly tracks the arm will mostly supervise arm motion rather than object motion.

B.2 A WORKED EXAMPLE: LINEAR RESPONSES

To see what the loss enforces, take a scalar perturbation with $\mathbb{E}[u_t \mid \mathcal{H}_t] = 0$ and $\mathbb{E}[u_t^2 \mid \mathcal{H}_t] = \sigma^2 > 0$, and suppose the recorded and generated features respond linearly:

$$f(O_t) = a(\mathcal{H}_t) + \beta u_t + \epsilon, \qquad \mathbb{E}[u_t \epsilon \mid \mathcal{H}_t] = 0, \tag{15}$$

$$f(\hat{O}_t) = \hat{a}(\mathcal{H}_t) + \hat{\beta} u_t + \hat{\epsilon}, \qquad \mathbb{E}[u_t \hat{\epsilon} \mid \mathcal{H}_t] = 0. \tag{16}$$

Here a and $\hat{a}$ absorb everything the context determines, the response coefficients β and $\hat{\beta}$ say how far the feature moves per unit of perturbation, and the noise terms are uncorrelated with the perturbation. For the signed test $g = u_t$ with constant coefficients, the context terms and the noise drop out, leaving

$$\mathbb{E}[u_t\{f(O_t) - f(\hat{O}_t)\}] = \sigma^2(\beta - \hat{\beta}), \qquad \Big(\mathbb{E}[u_t\{f(O_t) - f(\hat{O}_t)\}]\Big)^2 = \sigma^4(\beta - \hat{\beta})^2. \tag{17}$$

The squared term, which is this test's contribution to $\mathcal{L}_{\mathrm{AFFECT}}$, is zero exactly when $\hat{\beta} = \beta$. When $\beta \neq 0$, it penalizes a model that ignores the action ($\hat{\beta} = 0$), one that overreacts, and one that responds in the wrong direction, each in proportion to its squared distance from β. The target is the recorded response itself, not a larger one. Objectives that reward separating futures under different actions behave differently, since a stronger response always helps them.

If the coefficients and the perturbation variance depend on the context, the discrepancy becomes $\mathbb{E}[\sigma^2(\mathcal{H}_t)\{\beta(\mathcal{H}_t) - \hat{\beta}(\mathcal{H}_t)\}]$. The average over contexts is taken before squaring, so a model that overreacts in some contexts and underreacts in others can partly cancel. The loss therefore constrains the variance-weighted average response error, not the error in every context separately.

Nonlinear responses and the magnitude test. The linear model illustrates what the statistic measures; it is not an assumption about real dynamics. Now suppose the perturbation is symmetric about zero given $\mathcal{H}_t$, let $v = \mathbb{E}[u_t^2 \mid \mathcal{H}_t]$, and add quadratic terms $\gamma(u_t^2 - v)$ and $\hat{\gamma}(u_t^2 - v)$ to the two responses above, with constant $\gamma, \hat{\gamma}$ and noise uncorrelated with $u_t^2 - v$. Symmetry makes all odd moments of u_t vanish, which has two effects. The centered squared test $g = u_t^2 - v$ ignores the linear terms and gives $(\gamma - \hat{\gamma})\,\mathbb{E}[\mathrm{Var}(u_t^2 \mid \mathcal{H}_t)]$. Conversely, the signed test $g = u_t$ ignores the quadratic terms, since $\mathbb{E}[u_t(u_t^2 - v) \mid \mathcal{H}_t] = \mathbb{E}[u_t^3 \mid \mathcal{H}_t] = 0$. The two tests thus measure different parts of the response. A purely even response, such as an object that is knocked over by a deviation in either direction, is invisible to the signed test, which is why the bank also contains magnitude features (Appendix A.5).

B.3 FORMAL RESULTS

This subsection states the results behind the discussion above, together with their assumptions. They concern the population objective, meaning expectations rather than minibatch estimates, and say nothing about whether neural-network training reaches its minimum. We make the following standing assumptions. The feature maps and the positive scales s_b are fixed. Since the scales are estimated

from recorded windows before AFFECT training, the results strictly hold conditional on those estimates, or exactly if the scales come from a separate calibration set. All displayed expectations are finite, and every feature is square-integrable wherever we use a variance or an L^2 projection. The conditioning set $\mathcal{H}_t$ is as defined in Appendix B.1. Following Appendix A.4, we index tests by $b \in \mathcal{B}$ with $|\mathcal{B}| = K_\psi K_\phi$, where each b corresponds to one pair (i, j) of the main text and consists of a centered perturbation feature g_b and a video feature f_b.

B.3.1 Centering removes context-only terms

We claim that centering the perturbation features makes static appearance drop out of the loss. Specifically, Proposition 1 proves centering is an orthogonal projection that removes exactly the part of a feature that the context can predict.

Proposition 1 (centering removes context-only components). Let $w = w(u_t; \mathcal{H}_t)$ be a square-integrable perturbation feature and let $g = w - \mathbb{E}[w \mid \mathcal{H}_t]$ be its centered version. Then g is orthogonal in L^2 to every square-integrable function of $\mathcal{H}_t$. Moreover, $\mathbb{E}[w \mid \mathcal{H}_t]$ is the unique (up to almost-sure equality) minimizer of $\mathbb{E}[(w - a(\mathcal{H}_t))^2]$ over functions a of $\mathcal{H}_t$.

Proof. Write $\bar{w} = \mathbb{E}[w \mid \mathcal{H}_t]$, so that $g = w - \bar{w}$. By conditional Jensen, $\mathbb{E}[\bar{w}^2] \le \mathbb{E}[w^2] < \infty$, so $\bar{w}$ and hence g are square-integrable. Since $\bar{w}$ is a function of $\mathcal{H}_t$, it can be taken out of the conditional expectation, which gives

$$\mathbb{E}[g \mid \mathcal{H}_t] = \mathbb{E}[w \mid \mathcal{H}_t] - \bar{w} = 0. \tag{18}$$

Now let $a = a(\mathcal{H}_t)$ be square-integrable. The product ga is integrable by Cauchy–Schwarz, so the tower property applies. Because a is also a function of $\mathcal{H}_t$, it too can be taken out of the inner expectation:

$$\mathbb{E}[ga] = \mathbb{E}\big[\mathbb{E}[ga \mid \mathcal{H}_t]\big] = \mathbb{E}\big[a\,\mathbb{E}[g \mid \mathcal{H}_t]\big] = \mathbb{E}[a \cdot 0] = 0. \tag{19}$$

This proves the first claim.

For the second claim, fix a square-integrable $a(\mathcal{H}_t)$ and split the residual as $w - a = g + (\bar{w} - a)$. Expanding the square gives

$$\mathbb{E}[(w - a)^2] = \mathbb{E}[g^2] + 2\,\mathbb{E}[g\,(\bar{w} - a)] + \mathbb{E}[(\bar{w} - a)^2]. \tag{20}$$

The function $\bar{w} - a$ depends only on $\mathcal{H}_t$ and is square-integrable, so the first claim makes the cross term zero. Hence

$$\mathbb{E}[(w - a)^2] = \mathbb{E}[g^2] + \mathbb{E}[(\bar{w} - a)^2] \;\ge\; \mathbb{E}[g^2] = \mathbb{E}[(w - \bar{w})^2]. \tag{21}$$

The first term does not depend on a, so the minimum is reached at $a = \bar{w}$. Equality holds if and only if $\mathbb{E}[(\bar{w} - a)^2] = 0$, that is, if and only if $a = \bar{w}$ almost surely, which gives uniqueness.

Finally, we apply this to the test. Suppose the generated feature contains a context-only error, so that $f(\hat{O}_t) = f(O_t) + r(\mathcal{H}_t)$ with r square-integrable, as when the model renders a scene o_t slightly too dark. Then

$$\mathbb{E}\big[g\{f(O_t) - f(\hat{O}_t)\}\big] = -\mathbb{E}[g\,r(\mathcal{H}_t)] = 0 \tag{22}$$

by the first claim, so the error contributes nothing to the test. By linearity the same holds for a context-only term added to any feature, recorded or generated. □

B.3.2 Why the loss uses products

Section 3.3 rewrites the loss as a finite-feature MMD. Proposition 2 spells out what this buys: a small loss bounds not only each individual test but every normalized linear combination of the tests.

Proposition 2 (the loss bounds all combinations of tests). Let P denote the joint law of $(\mathcal{H}_t, u_t, O_t)$ in the data, and let Q_θ denote the same law with O_t replaced by the generated $\hat{O}_t$. Define the feature map Φ with entries $\Phi_b(\mathcal{H}_t, u_t, O_t) = g_b(u_t; \mathcal{H}_t)\, f_b(O_t)/s_b$, and let $m = \mathbb{E}_P[\Phi] - \mathbb{E}_{Q_\theta}[\Phi]$. Then

$$\mathcal{L}_{\text{AFFECT}} = \|m\|_2^2 = \sup_{\|v\|_2 \le 1} \big(\mathbb{E}_P[v^\top \Phi] - \mathbb{E}_{Q_\theta}[v^\top \Phi]\big)^2. \tag{23}$$

Consequently, every linear combination of the tests satisfies $|v^\top m| \le \|v\|_2 \sqrt{\mathcal{L}_{\text{AFFECT}}}$.

Proof. First we show that $\mathcal{L}_{\text{AFFECT}} = \|m\|_2^2$. The laws P and Q_θ share the same $(\mathcal{H}_t, u_t)$ and differ only in the future. The b-th entry of m is therefore

$$m_b = \frac{\mathbb{E}[g_b f_b(O_t)] - \mathbb{E}[g_b f_b(\hat{O}_t)]}{s_b} = \mathbb{E}\left[\frac{g_b}{s_b}\{f_b(O_t) - f_b(\hat{O}_t)\}\right] = \mathbb{E}[d_b], \tag{24}$$

with d_b as in equation 10. Summing the squares gives $\|m\|_2^2 = \sum_{b\in\mathcal{B}}(\mathbb{E}[d_b])^2 = \mathcal{L}_{\text{AFFECT}}$.

Next, fix $v \in \mathbb{R}^{|\mathcal{B}|}$. By linearity of expectation,

$$\mathbb{E}_P[v^\top \Phi] - \mathbb{E}_{Q_\theta}[v^\top \Phi] = \sum_b v_b\big(\mathbb{E}_P[\Phi_b] - \mathbb{E}_{Q_\theta}[\Phi_b]\big) = v^\top m. \tag{25}$$

The supremum in equation 23 is thus $\sup_{\|v\|_2\le 1}(v^\top m)^2$. Cauchy–Schwarz gives the upper bound $(v^\top m)^2 \le \|v\|_2^2\|m\|_2^2 \le \|m\|_2^2$ whenever $\|v\|_2 \le 1$. The bound is attained. If $m \neq 0$, then $v^\star = m/\|m\|_2$ has unit norm and $(v^{\star\top} m)^2 = \|m\|_2^2$. If $m = 0$, both sides are zero. This proves equation 23. For the consequence, $v = 0$ is trivial, and otherwise we apply the bound to the unit vector $v/\|v\|_2$ and multiply by $\|v\|_2$. This gives $|v^\top m| \le \|v\|_2\|m\|_2 = \|v\|_2\sqrt{\mathcal{L}_{\text{AFFECT}}}$.

Finally, we connect this to the MMD. Let $z, z' \sim P$ and $\hat{z}, \hat{z}' \sim Q_\theta$ be independent. Under the linear kernel $k(z, z') = \Phi(z)^\top \Phi(z')$, the squared MMD (Gretton et al., 2012) factorizes as

$$\mathbb{E}[k(z, z')] - 2\,\mathbb{E}[k(z, \hat{z})] + \mathbb{E}[k(\hat{z}, \hat{z}')] = \big\|\mathbb{E}_P[\Phi]\big\|_2^2 - 2\,\mathbb{E}_P[\Phi]^\top \mathbb{E}_{Q_\theta}[\Phi] + \big\|\mathbb{E}_{Q_\theta}[\Phi]\big\|_2^2 = \|m\|_2^2, \tag{26}$$

where the first equality uses independence to factor each expectation of an inner product. □

Why products rather than concatenation. Before per-test scaling, Φ is the outer product of the perturbation and video feature vectors, so each entry tests an interaction between the two. A simpler alternative would be to match the means of the concatenation $[G(u_t), F(O_t)]$, but that only compares the marginal distributions of perturbations and videos separately. A small example shows that this is not enough. Let u_t be uniform on $\{-1, 1\}$, and suppose the recorded future is $O_t = u_t$ while the model generates $\hat{O}_t = -u_t$, the mirror-image response. Both O_t and $\hat{O}_t$ are uniform on $\{-1, 1\}$, so every additive test $a(u_t) + b(O_t)$ has the same expectation under both. The single product test, however, gives $\mathbb{E}[u_t O_t] = 1$ but $\mathbb{E}[u_t \hat{O}_t] = -1$, and so detects the reversal.