ROLLOUT DEGENERATION UNDER UNSUPPORTED ACTIONS IN WORLD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Autoregressive world models often degenerate over long rollouts, a problem typically attributed to the train–test mismatch between clean training contexts and imperfect model-generated visual contexts. We show that world models face an additional source of instability: externally supplied actions can induce state–action pairs outside the regime covered during training. Through controlled action-space manipulations, we find that such *unsupported actions* substantially destabilize ongoing rollouts. Beyond the conventional degeneration pattern of progressive off-support drift, this instability can manifest as *conservative collapse*, where severely degraded visual states abruptly transition to visually plausible but dynamically unreliable states. Based on this analysis, we propose *validity-aware action gating* to mitigate degeneration under unsupported actions by estimating whether the state–action pair lies within the training support and adapting the influence of action conditioning accordingly. Experiments in visual navigation and robotic manipulation show that our method consistently reduces late-horizon degeneration under unsupported actions, lowering final 16-second FID by 45.3% and 13.3% over the original models in the two domains, respectively, while preserving competitive performance under supported actions.

1 INTRODUCTION

World models predict future observations from past context and future actions, approximating transitions of the form $(s_t, a_t) \mapsto s_{t+1}$, where s_t and a_t denote the visual state and externally supplied action at time t, respectively (Ha & Schmidhuber, 2018; Hafner et al., 2019; 2020). By rolling out candidate action sequences in imagination, world models provide a scalable surrogate for planning, policy evaluation, and robot learning without executing every candidate trajectory in the real world (Bar et al., 2025; Quevedo et al., 2026; Wu et al., 2022; Sharma et al., 2026).

Despite these benefits, a key challenge remains: autoregressive rollouts often degrade as the horizon grows. A model may produce plausible short-horizon predictions, yet gradually lose object identity, temporal coherence, or visual quality over extended rollouts (Huang et al., 2026; Bardhan et al., 2026; Chen et al., 2025). A common explanation for this phenomenon is exposure bias. During training, autoregressive models are typically conditioned on clean ground-truth data, whereas at inference time they must condition on their own imperfect predictions. As generation proceeds, these erroneous predictions are fed back into the model as contexts, allowing small local errors to propagate and compound over time. This train–test mismatch problem has been studied extensively in autoregressive language generation, diffusion sampling, and video generation (Bengio et al., 2015; Ranzato et al., 2016; Goyal et al., 2016; Holtzman et al., 2020; Ning et al., 2023; 2024; Li et al., 2026; Huang et al., 2025; Liu et al., 2026; Bardhan et al., 2026). As one geometric view in diffusion sampling interprets degeneration as deviation from the trained data manifold (Yao et al., 2025), these accounts are state-centric, locating the source of degeneration in model-generated context.

However, this interpretation is incomplete for action-conditioned world models, which additionally depend on external actions supplied by a user, planner, policy, or robot controller. These actions are generally not generated by the world model itself and need not remain within the state–action support observed during training. We call an action *unsupported* when the current state–action pair lies outside the region covered by the training distribution. That is, even when the current visual state appears plausible, its pairing with the commanded action may be unsupported. We argue that

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.

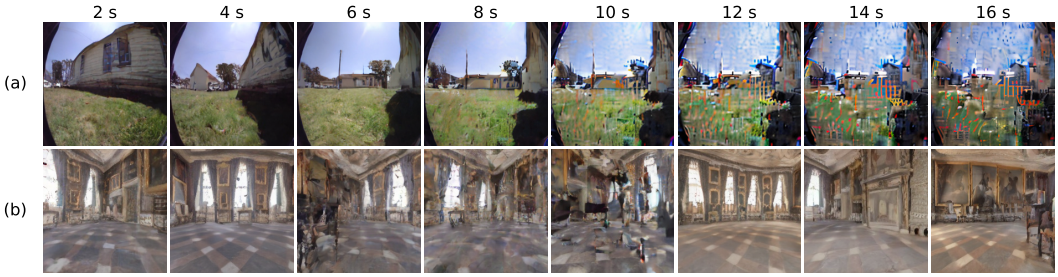

Figure 1: Examples of the two rollout degeneration patterns identified in our analysis. **(a)** Off-support drift under an ID action rollout on RECON, where visual quality progressively deteriorates over time. **(b)** Conservative collapse under an OOD-All action rollout on Habitat-Sim, where severely degraded states abruptly return to visually plausible but dynamically unreliable states.

this *state–action support mismatch* constitutes an additional source of rollout degeneration in world models. An unsupported action can bias the transition toward poorly covered regions of the state space, creating an off-support prediction that is subsequently amplified by autoregressive rollout.

We empirically study this mechanism through controlled action-space manipulations during autoregressive rollouts. To probe unsupported actions, we specifically construct out-of-distribution (OOD) action conditions that extend beyond the action regimes observed during training. Since our hypothesis concerns whether these unsupported actions drive generated states away from the training-state support, a behavior not directly captured by standard visual metrics, we also introduce a *support distance* metric, defined as the average distance from a generated visual feature to its nearest neighbors in a bank of training-state features. Across real-world and simulated navigation settings (Shah et al., 2021; Savva et al., 2019), these OOD conditions reveal that unsupported actions can substantially destabilize generation even when initialized from an in-distribution (ID) visual context. Interestingly, action-conditioned degeneration is not limited to the standard explanation of monotonic error accumulation. In some cases, rollouts exhibit the conventional failure mode of *off-support drift*, where visual quality metrics and support distance increase progressively over time. In other cases, however, we observe a distinct temporal pattern in which rollouts conditioned on unsupported actions initially perform worse than those conditioned on ID actions, but their visual quality metrics and support distance grow more slowly, plateau, or even decrease at later horizons, often crossing the corresponding ID curves. We refer to this characteristic behavior as *conservative collapse*. Although these late-horizon states may look visually plausible, they arise after the rollout has already entered a degraded regime and therefore remain unreliable as action-conditioned predictions.

Based on this analysis, we seek a method that can reduce the destabilizing effect of unsupported actions. Rather than forcing the model to follow an action through a poorly supported transition that may destabilize the rollout, we prioritize rollout reliability by reducing the influence of action conditioning when state–action compatibility is low. Motivated by the time-driven gating mechanism of RAE-NWM (Zhang et al., 2026b), we propose *validity-aware action gating*, which adjusts the strength of action conditioning according to the estimated validity of the current state–action pair. A pretrained verifier estimates this validity from the current visual context and commanded action, with the resulting score modulating the action signal during generation.

Experiments with NWM (Bar et al., 2025) on Habitat-Sim (Savva et al., 2019) show that validity-aware action gating substantially improves long-horizon robustness under unsupported actions while maintaining competitive ID performance. Ablations further show complementary benefits from state–action validity estimation, dual gating schedules, and OOD-aware post-tuning. We also evaluate the method in the robotic manipulation domain using WorldGym (Quevedo et al., 2026) on BridgeData V2 (Walke et al., 2023), where it improves robustness under perturbed actions. Together, these results suggest that unsupported actions are not a domain-specific failure case, but a structural challenge that should be explicitly considered in action-conditioned world models.

Our contributions are threefold. **First**, we formulate degeneration induced by unsupported actions as a state–action support mismatch problem in autoregressive world models, complementary to the standard view of generated-state drift. **Second**, we introduce support distance and characterize two failure patterns, off-support drift and conservative collapse, showing that visual quality alone is insufficient for fully understanding action-conditioned degeneration. **Third**, we propose validity-

aware action gating, a lightweight conditioning mechanism that modulates action influence based on state–action validity to reduce the destabilizing effect of unsupported actions.

2 Related Work

Exposure Bias in Autoregressive and Iterative Generation. Exposure bias was first widely studied in autoregressive text generation, where models are trained on ground-truth histories but condition on their own predictions at inference time, allowing prediction errors to accumulate over the generated sequence (Bengio et al., 2015; Ranzato et al., 2016). Classical mitigation strategies reduce this train–test mismatch by exposing models to generated histories, optimizing sequence-level objectives, or regularizing consistency between ground-truth and generated histories (Bengio et al., 2015; Ranzato et al., 2016; Goyal et al., 2016). However, later work has questioned whether such distortions are always incremental, arguing that language models can exhibit self-recovery behavior when conditioned on corrupted prefixes (He et al., 2021). Recent work has extended the concept of exposure bias to diffusion models, where iterative sampling creates its own form of train–test mismatch. The denoising network is trained on noisy samples derived from ground-truth data but, at inference time, operates on intermediate samples produced by its own preceding denoising steps (Ho et al., 2020; Song et al., 2021; Ning et al., 2023; 2024; Li et al., 2024). This mismatch can propagate prediction errors through the iterative sampling chain, causing sampling drift, degraded quality, or discrepancies between the nominal noise level and the actual intermediate state (Li & van der Schaar, 2024; Ning et al., 2024; Zhong et al., 2026). A related geometric view interprets this drift as a deviation from the learned manifolds, where off-manifold intermediate states amplify errors in subsequent denoising steps (Yao et al., 2025). Existing approaches mitigate this mismatch through perturbed denoising inputs (Ning et al., 2023), cumulative-error regularization (Li & van der Schaar, 2024), modified sampling trajectories (Li et al., 2024), scaled denoising updates (Ning et al., 2024), or geometric and noise-aware corrections (Yao et al., 2025; Zhong et al., 2026).

Long-horizon autoregressive video generation exhibits a similar exposure-bias problem, as generated frames become inputs to subsequent predictions, allowing small visual errors to compound into temporal inconsistency and quality degradation (Huang et al., 2025; Liu et al., 2026; Yang et al., 2026a). Recent methods reduce this train–test mismatch by training on degraded or error-corrupted visual histories (Li et al., 2026; Guo et al., 2025) or by using self-generated rollouts and related rollout-aware objectives (Huang et al., 2025; Liu et al., 2026; Yang et al., 2026a; Cui et al., 2026). Together with the preceding exposure bias literature, these works share a state-centric view, attributing degeneration to model-generated states fed back into subsequent generation steps. In contrast, our work studies action-conditioned world models, where degeneration can also be triggered by externally supplied actions unsupported under the current visual state.

Action-Conditioned World Models and Rollout Reliability. World models learn dynamics for imagining future trajectories and have been widely used for model-based reinforcement learning (RL) and planning (Ha & Schmidhuber, 2018; Hafner et al., 2019; 2020). Recent visual world models extend this idea to high-dimensional observations by predicting future frames conditioned on action sequences, enabling applications in navigation, robotic manipulation, interactive simulation, and policy evaluation (Bar et al., 2025; Quevedo et al., 2026; Jeon et al., 2026; Huang et al., 2026). RAE-NWM further improves navigation world model quality and action controllability using time-driven dynamics gating to regulate action injection (Zhang et al., 2026b). However, world models inherit the same exposure-bias problem as autoregressive video models, since generated frames are reused as future context, causing errors to accumulate over long-horizon rollouts (Huang et al., 2026; Bardhan et al., 2026; Yang et al., 2026b; Zhang et al., 2026a). Recent work mitigates this problem through cycle-consistent rollout recovery, prediction-corrupted histories, bootstrapped latents, or self-forwarded contexts (Huang et al., 2026; Yang et al., 2026b; Zhang et al., 2026a; Gao et al., 2026; Jeon et al., 2026), while other approaches use RL post-training to improve long-horizon rollout fidelity and interaction quality (Bardhan et al., 2026; Wang et al., 2026). While these approaches remain state-centric, focusing on robustness to model-generated histories, Hansen & Wang (2026) take a more action-aware perspective by relating world-model hallucinations to insufficient state–action coverage and mitigating them through coverage-aware training or targeted data collection. However, such data-centric solutions are difficult to apply when the dataset is fixed or additional environment interaction is costly, motivating a complementary model-side approach. We therefore

propose validity-aware action gating to estimate state–action validity and reduce the destabilizing influence of unsupported actions.

3 Empirical Analysis of Action-Conditioned Degeneration

To test our hypothesis that actions can affect rollout stability beyond the state-centric exposure-bias mechanism, we study action-conditioned degeneration using controlled OOD actions that induce unsupported state–action pairs. First, we compare rollout degeneration under ID and OOD actions, examining how OOD actions alter its severity and temporal structure relative to the ID baseline. We then perform an action switch experiment in which ID actions are replaced by OOD actions at specified times to test whether unsupported actions abruptly destabilize an ongoing rollout.

3.1 Experimental Setup

Model. We use Navigation World Model (NWM) (Bar et al., 2025), an action-conditioned diffusion world model for visual navigation. Given a context of past egocentric observations and future navigation actions, NWM autoregressively predicts future observations. Each action is represented as a three-dimensional command $a_t = (\Delta x_t, \Delta y_t, \Delta \psi_t)$, where Δx_t and Δy_t denote planar translation and $\Delta \psi_t$ denotes yaw change. Following the NWM protocol, both training and evaluation use videos sampled at 4 fps, with a context size of 4 and autoregressive rollouts of up to 16 seconds. We train a CDiT-S/2 backbone with approximately 50M parameters for 100k optimization steps.

Datasets and action protocols. To ensure that rollout degeneration is not specific to a particular environment, we evaluate NWM on both a real-world navigation dataset, RECON (Shah et al., 2021; Bar et al., 2025), and a simulated environment, Habitat-Sim (Savva et al., 2019). We refer to rollouts using ID and OOD action sequences as ID and OOD rollouts, respectively. We first consider RECON, where ID rollouts use trajectories from the test split with their recorded actions. For OOD rollouts, we independently add zero-mean Gaussian noise to each action dimension with standard deviation $\sigma \in \{0.5, 1, 2, 5, 10\}$. The resulting action distributions progressively depart from the empirical training-action support with increasing σ, as shown in Appendix A.1.

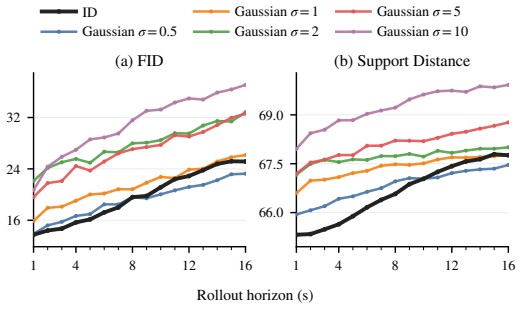

Figure 2: Long-horizon rollout degeneration under ID and Gaussian-perturbed OOD actions on RECON. We report FID and support distance for Gaussian perturbations with $\sigma \in \{0.5, 1, 2, 5, 10\}$.

Since these perturbed actions are synthetically constructed, RECON does not provide corresponding ground-truth observations. We therefore complement the real-world evaluation by constructing a simulation dataset using Habitat-Sim. Unlike RECON, Habitat-Sim allows us to execute OOD action sequences and render the resulting future observations. It also enables strictly controlled OOD evaluation by sampling actions explicitly outside the ID action range. We render RGB observations in `skokloster-castle`, one of the test scenes provided with Habitat-Sim, and generate goal-directed trajectories with mild stochastic perturbations. The dataset contains 5,000 ID trajectories, divided into 4,000 training and 1,000 test trajectories. We consider three OOD settings of varying axes of perturbation: along forward motion, yaw, or all action dimensions jointly, denoted as OOD-Δx, OOD-$\Delta \psi$, and OOD-ALL, respectively. The exact action ranges and sampling procedures are provided in Appendix A.2.

Metrics. We evaluate rollouts using LPIPS (Zhang et al., 2018), DreamSim (Fu et al., 2023), FID (Heusel et al., 2017), and support distance. All metrics are reported as distances, with lower values indicating better rollout quality. LPIPS and DreamSim measure pairwise perceptual distance to ground-truth frames, while FID measures distribution-level distance between generated and reference image sets. However, our hypothesis concerns unsupported actions driving predictions

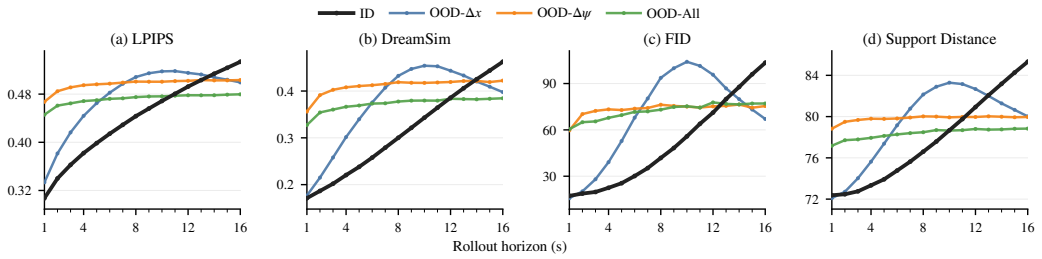

Figure 3: Long-horizon rollout degeneration under ID and structured OOD actions in Habitat-Sim. We compare ID actions with actions sampled outside the training support along forward translation (OOD-Δx), yaw (OOD-$\Delta\psi$), and all action dimensions jointly (OOD-ALL).

into poorly covered regions of the state space. These metrics capture perceptual and distributional degradation, but do not directly measure whether generated states drift away from the empirical training-state support. To measure this off-support aspect of degeneration, we define support distance as a nearest-neighbor proxy for distance to the training-data manifold. Given a generated state $\hat{s}_t$, a visual encoder ϕ, and a training-state feature bank $\mathcal{B}$, support distance is defined as:

$$d_{\text{support}}(\hat{s}_t) = \frac{1}{K} \sum_{x \in \text{kNN}_K(\phi(\hat{s}_t), \mathcal{B})} \|\phi(\hat{s}_t) - x\|_2 . \tag{1}$$

Throughout all experiments, we use $K = 5$ and compute ϕ using the frozen VAE of the corresponding world model. A higher value indicates that generated observations lie farther from the empirical support of training states. Since ground-truth frames are unavailable for RECON OOD rollouts, we report only FID and support distance for both ID and OOD conditions on RECON. For readability, the main figures emphasize temporal trends, while exact metric values are reported in Appendix D.

3.2 Rollout Degeneration under ID and OOD Actions

We first examine how rollout metrics evolve under ID and OOD actions, as shown in Figures 2 and 3. Under ID actions, all visual metrics deteriorate nearly monotonically with rollout horizon, consistent with the standard failure mode of autoregressive video models. Support distance also increases as the rollout proceeds, suggesting that degeneration involves not only visual degradation but also drift away from the empirical training-state support. Relative to these ID baselines, OOD actions alter both the severity and temporal structure of degeneration. We observe two characteristic patterns across these settings. The first is *off-support drift*, where visual error and support distance increase progressively over time, as in ID rollouts. Here, however, the initial deviation can be induced by an unsupported transition, and the resulting errors are subsequently amplified by autoregressive rollout.

The second pattern is what we refer to as *conservative collapse*. In this case, OOD rollouts can initially be worse than ID rollouts, but their metrics later grow more slowly, plateau, or even decrease, sometimes crossing the ID curves. We also observe this behavior qualitatively. Some rollouts pass through severely degraded states and then abruptly produce frames that appear visually plausible, as illustrated in Figures 1, 11, and 12. However, once a rollout has entered a degraded regime, the emergence of plausible frames cannot be taken as reliable evidence that the model has recovered the correct dynamics. Instead, the model may have fallen back to visual states similar to those seen during training, regardless of whether they are consistent with the preceding actions and trajectory. In this sense, conservative collapse resembles degeneration in language models, where likelihood-maximizing decoding often yields bland, generic, or repetitive outputs (Holtzman et al., 2020).

3.3 Switching to OOD Actions Abruptly Destabilizes Rollout

To isolate the temporal effect of unsupported actions, we additionally perform an action switch experiment in Habitat-Sim. For each rollout, we follow an ID trajectory until 1, 2, 4, or 8 seconds, after which the remaining actions are resampled from the OOD-ALL distribution. This yields a common subset of 968 trajectories from 1,000 test trajectories, with invalid trajectories excluded under simulator constraints.

As shown in Figure 4, the metric curves for LPIPS and DreamSim increase sharply after the switch point, showing that unsupported actions can destabilize an ongoing rollout at the moment they are introduced. This supports our interpretation that state–action support mismatch can destabilize rollouts beyond the effects of accumulated context errors. Complete metric values for the action switch experiment, including FID and support distance, are provided in Appendix D.3.

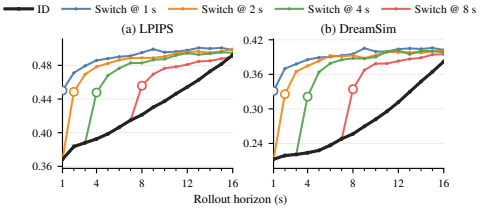

Figure 4: Rollout degeneration when switching from ID to OOD-ALL actions at 1, 2, 4, or 8 seconds in Habitat-Sim.

4 VALIDITY-AWARE ACTION GATING

The preceding analysis shows that unsupported actions can destabilize autoregressive rollouts, inducing both off-support drift and conservative collapse. Such actions are difficult to avoid in open-ended deployment because actions are supplied externally, while exhaustively covering all possible state–action combinations during training is impractical. This vulnerability can pose problems for world models used in planning, policy learning, or interactive applications. Conservative collapse further complicates practical use, as visually plausible frames can mask an already degraded rollout, making such failures difficult to diagnose.

This motivates detecting unsupported actions before they strongly influence subsequent generation. When faithful dynamics are essential, as in planning or policy learning, a low-validity action may warrant abstaining from further simulation. In applications where visual continuity is more important, however, continuing the rollout may still be useful if the destabilizing effect of an unsupported action can be attenuated. We therefore propose a *validity-aware action gating* framework with three components. First, a pretrained state–action verifier estimates whether the commanded action is supported under the current visual state. Second, drawing inspiration from the time-driven dynamics gating of RAE-NWM (Zhang et al., 2026b), validity-aware action gating modulates action conditioning across denoising timesteps and uses the validity score to interpolate between separate schedules for supported and unsupported transitions. Finally, OOD-aware post-tuning drives predictions under low-validity actions toward a frozen null-action reference while retaining the standard diffusion objective for supported transitions.

4.1 STATE–ACTION VERIFIER

We pretrain a state–action verifier V_ϕ to estimate whether a commanded action is supported under the current visual context. Given the latent $\mathbf{z}_i$ of the latest context observation and the action $\mathbf{a}_{i,k}$, it produces a calibrated validity score $q_{i,k} = V_\phi(\mathbf{z}_i, \mathbf{a}_{i,k}) \in [0, 1]$, where i indexes the current rollout step and k indexes the prediction horizon. Higher values indicate greater confidence that the state–action pair lies within the empirical ID support. During inference, the verifier computes $q_{i,k}$ once for each target frame and keeps it fixed throughout the corresponding denoising process.

The verifier is a lightweight cross-attention classifier that encodes the action into a query token and uses spatial tokens from the latest visual latent as keys and values to predict a scalar validity score. It is trained with binary cross-entropy and remains frozen throughout world model training and autoregressive inference. The training data consist of positive state–action pairs from ID training trajectories and a mixture of synthetic OOD and mismatched state–action negative pairs. Full details of the verifier training data construction and calibration are provided in Appendix C.2.

4.2 DUAL VALIDITY-AWARE ACTION GATING

Since supported and unsupported transitions may require different action-conditioning strengths across denoising timesteps, we explicitly separate their timestep-dependent gating schedules:

$$\alpha_\tau^{\mathrm{ID}} = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})\sigma\left(G_{\mathrm{ID}}(\mathbf{e}_\tau(\tau))\right), \tag{2a}$$

$$\alpha_\tau^{\mathrm{OOD}} = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})\sigma\left(G_{\mathrm{OOD}}(\mathbf{e}_\tau(\tau))\right), \tag{2b}$$

where τ denotes the denoising timestep, $\mathbf{e}_\tau(\tau)$ is the corresponding timestep embedding, σ is the sigmoid function, and G_{ID} and G_{OOD} are independently parameterized lightweight MLPs, each

with a single hidden layer. The hyperparameters $\alpha_{\min}$ and $\alpha_{\max}$ determine the lower and upper bounds of the action-conditioning strength. The validity score softly interpolates between the two schedules, and the resulting scalar gate rescales the action embedding $\mathbf{e}_a(\mathbf{a}_{i,k})$ before it is combined with the denoising timestep embedding $\mathbf{e}_\tau(\tau)$ and prediction-horizon embedding $\mathbf{e}_k(k)$:

$$\alpha_{\tau,i,k} = q_{i,k}\alpha_\tau^{\mathrm{ID}} + (1 - q_{i,k})\alpha_\tau^{\mathrm{OOD}}. \tag{3}$$

$$\mathbf{c}_{\tau,i,k} = \mathbf{e}_\tau(\tau) + \mathbf{e}_k(k) + \alpha_{\tau,i,k}\mathbf{e}_a(\mathbf{a}_{i,k}). \tag{4}$$

Here, the resulting conditioning vector $\mathbf{c}_{\tau,i,k}$ is fed to the AdaLN blocks of the backbone world model to condition the network throughout the denoising process (Bar et al., 2025). Setting $\alpha_{\tau,i,k} = 1$ for all diffusion timesteps and state–action pairs recovers the original NWM conditioning. RAE-NWM applies an element-wise, time-dependent gate to a fused dynamics feature that jointly encodes the action and prediction horizon (Zhang et al., 2026b). Our formulation instead applies a scalar gate explicitly to the action embedding, while using the estimated state–action validity to interpolate between separate ID and OOD schedules. The validity score $q_{i,k}$ remains fixed for a target frame, while the resulting action gate $\alpha_{\tau,i,k}$ varies across denoising timesteps, acting as a learned schedule for action-conditioning strength. High-validity transitions are therefore routed primarily through the ID schedule, whereas low-validity transitions are routed through the OOD schedule.

4.3 OOD-Aware World-Model Training

We train the gated world model in two stages. We first pretrain the gated model on ID trajectories using the same standard diffusion objective as NWM (Bar et al., 2025). In addition, we apply action dropout with probability p_{drop} by replacing the action embedding $\mathbf{e}_a(\mathbf{a}_{i,k})$ with a zero vector. This allows the model to produce valid predictions without action conditioning. Both the ID and OOD gating networks are initialized to output $\alpha_\tau^{\mathrm{ID}} = \alpha_\tau^{\mathrm{OOD}} = 1$ for all τ, such that training starts from the original NWM conditioning.

We then perform OOD-aware post-tuning. A trainable student D_θ and a frozen EMA reference model $D_{\bar{\theta}}$ are initialized from the pretrained checkpoint. Each post-tuning step contains two branches: an ID branch that retains the diffusion objective used during pretraining, and an OOD branch that applies a consistency objective using synthetic unsupported actions.

For the OOD branch, we draw an unsupported action $\mathbf{a}_{i,k}^-$ from the same OOD action range used to construct the synthetic negatives for the verifier and compute its validity score $q_{i,k}^- = V_\phi(\mathbf{z}_i, \mathbf{a}_{i,k}^-)$. Using the same noisy target, visual context, denoising timestep, and prediction horizon, the student is conditioned on the unsupported action, whereas the reference model receives no action conditioning:

$$\begin{aligned} \widehat{\mathbf{u}}_{\mathrm{OOD}} &= D_\theta\left(\mathbf{x}_{i+k}^{(\tau)}, \tau, \mathbf{z}_{\mathrm{cond},i}, \mathbf{a}_{i,k}^-, k, q_{i,k}^-\right), \\ \widehat{\mathbf{u}}_{\varnothing} &= D_{\bar{\theta}}\left(\mathbf{x}_{i+k}^{(\tau)}, \tau, \mathbf{z}_{\mathrm{cond},i}, \mathbf{a}_{\varnothing}, k\right). \end{aligned} \tag{5}$$

Here, $\mathbf{a}_\varnothing$ denotes that the action embedding is replaced with a zero vector in the frozen reference model. Because the training data contain no paired ground-truth future for the sampled unsupported action, we encourage the student prediction to approach the null-action reference:

$$\mathcal{L}_{\mathrm{OOD}} = \mathbb{E}\left[\|\widehat{\mathbf{u}}_{\mathrm{OOD}} - \mathrm{sg}\left[\widehat{\mathbf{u}}_\varnothing\right]\|_2^2\right], \tag{6}$$

where sg denotes the stop-gradient operator. The complete post-tuning objective is

$$\mathcal{L}_{\mathrm{post}} = \mathcal{L}_{\mathrm{ID}} + \lambda_{\mathrm{OOD}}\mathcal{L}_{\mathrm{OOD}}, \tag{7}$$

where $\mathcal{L}_{\mathrm{ID}}$ is the standard diffusion objective for the ID branch. We update the full student world model while keeping the verifier and reference model frozen. Action dropout is disabled in the OOD branch to avoid trivial cases in which both models receive null action conditioning.

We note that our OOD-aware post-tuning can be viewed as an action-space analogue of prior approaches that train on corrupted visual inputs to approximate the imperfect states encountered during autoregressive inference (Ning et al., 2023; Li et al., 2026; Guo et al., 2025; Zhang et al., 2026a). While these works improve robustness to imperfect generated states, we instead train on synthetic actions outside the training support, reducing the destabilizing influence of unsupported actions.

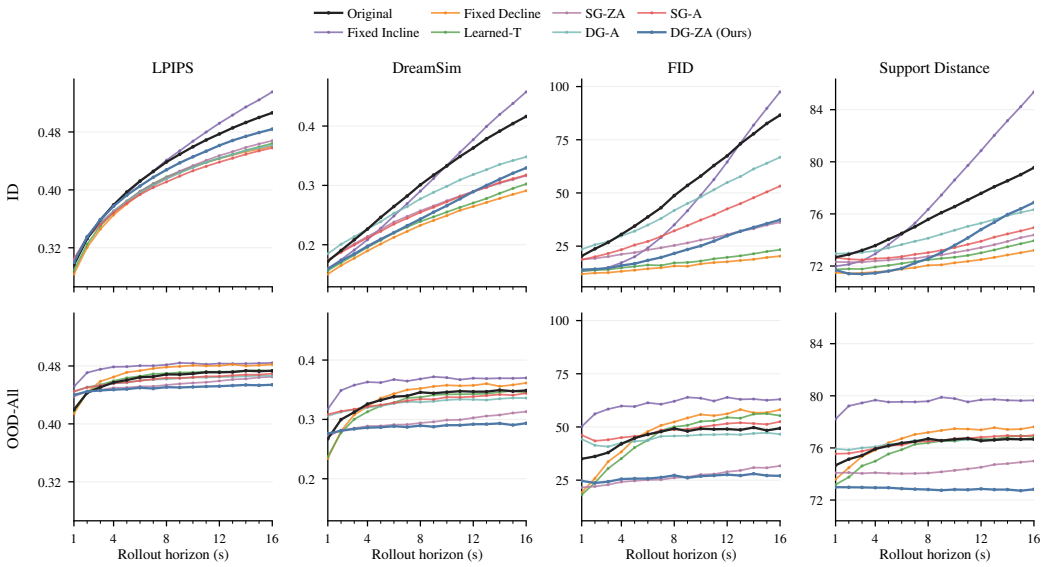

Figure 5: Long-horizon rollout performance of the final 120k models on Habitat-Sim. The rows report ID and OOD-All rollouts, and the columns show LPIPS, DreamSim, FID, and support distance.

5 EXPERIMENTS

We evaluate validity-aware action gating with three questions. First, does the proposed method improve rollout robustness under unsupported actions while preserving generation quality under ID actions? Second, does each component, including the state–action verifier, dual gating schedules, and OOD-aware post-tuning, contribute to the final performance? Third, does the method transfer beyond visual navigation to a larger world model for robotic manipulation?

5.1 EXPERIMENTAL SETUP

We follow the Habitat-Sim experimental setup described in Section 3. We compare the original NWM with fixed incline and decline schedules, a learned timestep-only gate, and several variants of validity-aware gating. The fixed schedules linearly increase or decrease the action-conditioning strength across denoising timesteps, while the learned timestep-only gate predicts a scalar action gate solely from the denoising-timestep embedding. For ablation, we construct a 2×2 set of validity-aware gating variants by independently varying the gating architecture and the verifier input. We denote the single- and dual-gate architectures by SG and DG, and the action-only and state–action verifiers by A and ZA, respectively. This yields SG-A, SG-ZA, DG-A, and DG-ZA, with DG-ZA corresponding to our full method. The action-only verifier estimates validity from the commanded action alone, whereas the state–action verifier additionally uses the current visual context. SG directly predicts a single action gate from the denoising timestep and validity score, whereas DG interpolates between separate ID and OOD gating schedules. We additionally evaluate DG-ZA before OOD-aware post-tuning to assess the effect of the subsequent post-tuning stage.

All models use the same compact CDiT-S/2 backbone and are initially trained for 100k steps on the training split of ID trajectories. The validity-aware variants apply action dropout with $p_{\text{drop}} = 0.1$ during this stage and are subsequently post-tuned for 20k steps using Equation 7. For a fair comparison, the baselines are also trained for an additional 20k steps using their respective ID objectives, resulting in the same total number of optimization steps. We set $\lambda_{\text{OOD}} = 0.1$ and linearly warm it up over the first 2k post-tuning steps. For all gating methods, we set the lower and upper bounds of the action-conditioning strength to $\alpha_{\min} = 0.1$ and $\alpha_{\max} = 2.0$, respectively. All models are evaluated under the same ID and OOD-All rollout protocols. Additional architectural and training details are provided in Appendices B and C, with complete numerical results in Appendix D.4.

5.2 RESULTS

Long-horizon robustness. Figure 5 compares the final 120k models under ID and OOD-All actions. Under ID actions, Fixed Decline and Learned-T, which are trained exclusively with the ID ob-

Table 1: Ablation results on Habitat-Sim OOD-All rollouts at 16 seconds. All methods use the 120k checkpoints except DG-ZA w/o post-tuning, which uses the corresponding 100k checkpoint.

Variant	LPIPS ↓	DreamSim ↓	FID ↓	Support ↓
Original	0.473	0.349	49.360	76.679
SG-A	0.469	0.344	52.479	76.973
DG-A	0.466	0.336	46.595	76.631
SG-ZA	0.465	0.313	31.707	74.994
DG-ZA w/o post-tuning	0.467	0.326	42.373	75.751
DG-ZA (Ours)	**0.454**	**0.294**	**27.022**	**72.823**

Table 2: FID comparison between the released WorldGym model and DG-ZA on BridgeData V2 under ID and Gaussian-perturbed OOD actions with $\sigma = 1.0$.

Condition	Model	4s	8s	12s	16s
ID	Original	57.291	**76.307**	85.378	96.017
	DG-ZA	**57.181**	76.643	**85.240**	**90.912**
OOD	Original	102.398	123.654	139.259	150.144
	DG-ZA	**97.558**	**115.042**	**121.620**	**130.226**

jective, provide the strongest long-horizon stability. Although DG-ZA is less competitive with these ID-only baselines, it remains substantially more robust than Original, reducing LPIPS, DreamSim, FID, and support distance by 4.3%, 20.9%, 56.9%, and 3.4%, respectively, at 16 seconds. Under OOD-All actions, DG-ZA achieves the strongest late-horizon performance across all metrics, reducing the same metrics by 4.0%, 15.8%, 45.3%, and 5.0%, respectively, relative to Original at 16 seconds. These results show that validity-aware action gating substantially mitigates degeneration under unsupported actions while preserving strong ID performance relative to the Original model.

Ablation analysis. Table 1 disentangles the effects of verifier input, gating design, and OOD-aware post-tuning on OOD-All rollouts. Replacing the action-only verifier with the state–action verifier improves both single- and dual-gate variants, while dual gating likewise improves both verifier variants. These consistent gains show that the two design choices are complementary, with DG-ZA providing the strongest OOD robustness. Comparing DG-ZA with its 100k checkpoint before post-tuning further shows that the additional 20k-step post-tuning stage improves all four metrics.

5.3 Generalization to WorldGym and BridgeData V2

We further evaluate our method on WorldGym (Quevedo et al., 2026), a 609M-parameter world model for robotic manipulation, using BridgeData V2 (Walke et al., 2023). We compare the released 580k-step pretrained model with DG-ZA after a brief OOD-aware post-tuning stage of 20k steps. For the OOD condition, we add Gaussian noise with $\sigma = 1.0$ to the six continuous end-effector dimensions of the WorldGym action. Full architectural, training, and evaluation details, together with complete quantitative results, are provided in Appendix E.

Table 2 compares FID between the two models under both ID and OOD rollouts. DG-ZA preserves ID performance while consistently reducing FID under OOD actions, with the improvement increasing at longer horizons. At 16 seconds, DG-ZA reduces FID by 5.3% under ID actions and 13.3% under OOD actions relative to the original model. These results show that the benefit of validity-aware action gating extends beyond navigation to the robotic manipulation domain.

6 Conclusion

We studied rollout degeneration in action-conditioned world models from the perspective of state–action support mismatch. Our analysis shows that unsupported actions can destabilize autoregressive rollouts and induce both off-support drift and conservative collapse. We proposed validity-aware action gating to reduce the influence of unsupported actions, consistently improving OOD rollout robustness across navigation and robotic manipulation domains.

AI USE STATEMENT

Generative AI tools were used to assist with method implementation and debugging, the preparation and formatting of tables and figures, and editing the manuscript for clarity, readability, and phrasing. The research questions, methodological ideas, and experimental design were developed by the authors. All AI-assisted code and manuscript content were reviewed and verified by the authors, who take responsibility for the final content of this work.

REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the datasets, OOD action construction, model architectures, verifier and gating implementations, training protocols, and complete quantitative results in the appendix to facilitate reproduction of our experiments.

REFERENCES

Amir Bar, Gaoyue Zhou, Danny Tran, Trevor Darrell, and Yann LeCun. Navigation world models. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15791–15801. IEEE, 2025.

Jai Bardhan, Patrik Drozdik, Josef Sivic, and Vladimir Petrik. Persistent robot world models: Stabilizing multi-step rollouts via reinforcement learning. *arXiv preprint arXiv:2603.25685*, 2026.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. *Advances in neural information processing systems*, 28, 2015.

Junyi Chen, Haoyi Zhu, Xianglong He, Yifan Wang, Jianjun Zhou, Wenzheng Chang, Yang Zhou, Zizun Li, Zhoujie Fu, Jiangmiao Pang, et al. Deepverse: 4d autoregressive video generation as a world model. *arXiv preprint arXiv:2506.01103*, 2025.

Justin Cui, Jie Wu, Ming Li, Tao Yang, Xiaojie Li, Rui Wang, Andrew Bai, Yuanhao Ban, and Cho-Jui Hsieh. Self-Forcing++: Towards Minute-Scale High-Quality Video Generation. In *International Conference on Learning Representations*, volume 2026, pp. 85802–85822, 2026. URL https://openreview.net/forum?id=DzvPiqh23f.

Stephanie Fu, Netanel Tamir, Shobhita Sundaram, Lucy Chai, Richard Zhang, Tali Dekel, and Phillip Isola. DreamSim: Learning new dimensions of human visual similarity using synthetic data. In *Advances in Neural Information Processing Systems*, 2023.

Jiaxuan Gao, Yongjian Guo, Zhong Guan, Wen Huang, Wanlun Ma, Xi Xiao, Junwu Xiong, and Sheng Wen. Sword: Style-robust world models as simulators via dynamic latent bootstrapping for vla policy post-training. *arXiv preprint arXiv:2605.07288*, 2026.

Anirudh Goyal, Alex Lamb, Ying Zhang, Saizheng Zhang, Aaron Courville, and Yoshua Bengio. Professor Forcing: A new algorithm for training recurrent networks. In *Advances in Neural Information Processing Systems*, volume 29, 2016.

Yuwei Guo, Ceyuan Yang, Hao He, Yang Zhao, Meng Wei, Zhenheng Yang, Weilin Huang, and Dahua Lin. End-to-end training for autoregressive video diffusion via self-resampling. *arXiv preprint arXiv:2512.15702*, 2025.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 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, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020.

Nicklas Hansen and Xiaolong Wang. Hallucination in world models is predictable and preventable. *arXiv preprint arXiv:2606.27326*, 2026.

Tianxing He, Jingzhao Zhang, Zhiming Zhou, and James Glass. Exposure bias versus self-recovery: Are distortions really incremental for autoregressive text generation? In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pp. 5087–5102, 2021.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. GANs trained by a two time-scale update rule converge to a local nash equilibrium. In *Advances in Neural Information Processing Systems*, 2017.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rygGQyrFvH.

Junchao Huang, Ziyang Ye, Xinting Hu, Tianyu He, Guiyu Zhang, Shaoshuai Shi, Jiang Bian, and Li Jiang. LIVE: Long-horizon interactive video world modeling. In *International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=iKKZmgxkbG.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *Advances in Neural Information Processing Systems*, 38:167283–167308, 2025. URL https://openreview.net/forum?id=mSiN7i0BYH.

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.

Mingxiao Li, Tingyu Qu, Ruicong Yao, Wei Sun, and Marie-Francine Moens. Alleviating exposure bias in diffusion models through sampling with shifted time steps. In *International Conference on Learning Representations*, volume 2024, pp. 16816–16838, 2024.

Wuyang Li, Wentao Pan, Po-Chien Luan, Yang Gao, and Alexandre Alahi. Stable video infinity: Infinite-length video generation with error recycling. In *International Conference on Learning Representations*, volume 2026, pp. 23406–23432, 2026.

Yangming Li and Mihaela van der Schaar. On error propagation of diffusion models. In *International Conference on Learning Representations*, volume 2024, pp. 32791–32807, 2024.

Kunhao Liu, Wenbo Hu, Jiale Xu, Ying Shan, and Shijian Lu. Rolling forcing: Autoregressive long video diffusion in real time. In *International Conference on Learning Representations*, volume 2026, pp. 91177–91196, 2026.

Mang Ning, Enver Sangineto, Angelo Porrello, Simone Calderara, and Rita Cucchiara. Input perturbation reduces exposure bias in diffusion models. In *International Conference on Machine Learning*, volume 202, pp. 26245–26265. PMLR, 2023.

Mang Ning, Mingxiao Li, Jianlin Su, Albert Ali Salah, and Itir Onal Ertugrul. Elucidating the exposure bias in diffusion models. In *International Conference on Learning Representations*, volume 2024, pp. 15167–15189, 2024.

Julian Quevedo, Ansh Kumar Sharma, Yixiang Sun, Varad Suryavanshi, Percy Liang, and Sherry Yang. Worldgym: World model as an environment for policy evaluation. In *International Conference on Learning Representations*, volume 2026, pp. 78932–78957, 2026.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *International Conference on Learning Representations*, 2016.

Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9339–9347, 2019.

Dhruv Shah, Benjamin Eysenbach, Gregory Kahn, Nicholas Rhinehart, and Sergey Levine. Rapid exploration for open-world navigation with latent goal models. *arXiv preprint arXiv:2104.05859*, 2021.

Ansh Kumar Sharma, Yixiang Sun, Ninghao Lu, Yunzhe Zhang, Jiarao Liu, and Sherry Yang. World-gymnast: Training robots with reinforcement learning in a world model. *arXiv preprint arXiv:2602.02454*, 2026.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=St1giarCHLP.

Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on robot learning*, pp. 1723–1736. PMLR, 2023.

Zehan Wang, Tengfei Wang, Haiyu Zhang, Xuhui Zuo, Junta Wu, Haoyuan Wang, Wenqiang Sun, Zhenwei Wang, Chenjie Cao, Hengshuang Zhao, Chunchao Guo, and Zhou Zhao. WorldCompass: Reinforcement learning for long-horizon world models. In *International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=zN8QDhuOHN.

Philipp Wu, Alejandro Escontrela, Danijar Hafner, Ken Goldberg, and Pieter Abbeel. Daydreamer: World models for physical robot learning. In *Conference on Robot Learning*, 2022.

Shuai Yang, Wei Huang, Ruihang Chu, Yicheng Xiao, Yuyang Zhao, Xianbang Wang, Muyang Li, Enze Xie, Ying-Cong Chen, Yao Lu, Song Han, and Yukang Chen. LongLive: Real-time interactive long video generation. In *International Conference on Learning Representations*, 2026a. URL https://openreview.net/forum?id=nCAODkpsPJ.

Yifei Yang, Zehua Fan, Huan Li, Aoqi Wang, Lida Huang, Haibao Yu, Haiyan Liu, Xuanyao Mao, Jason Bao, Liang Xu, et al. Ar forcing: Towards long-horizon robot navigation world model. *arXiv preprint arXiv:2605.31314*, 2026b.

Yuzhe Yao, Jun Chen, Zeyi Huang, Haonan Lin, Mengmeng Wang, Guang Dai, and Jingdong Wang. Manifold constraint reduces exposure bias in accelerated diffusion sampling. In *International Conference on Learning Representations*, volume 2025, pp. 96580–96616, 2025.

Conglang Zhang, Yifan Zhan, Qingjie Wang, Zhanpeng Ouyang, Yu Li, Zihao Yang, Xiaoyang Guo, Weiqiang Ren, Qian Zhang, Zhen Dong, et al. Horizondrive: Self-corrective autoregressive world model for long-horizon driving simulation. *arXiv preprint arXiv:2605.11596*, 2026a.

Mingkun Zhang, Wangtian Shen, Fan Zhang, Haijian Qin, Zihao Pei, and Ziyang Meng. Rae-nwm: Navigation world model in dense visual representation space. *arXiv preprint arXiv:2603.09241*, 2026b.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 586–595. IEEE, 2018.

Jincheng Zhong, Boyuan Jiang, Xin Tao, Pengfei Wan, Kun Gai, and Mingsheng Long. Mitigating noise shift in denoising generative models with noise awareness guidance. In *International Conference on Learning Representations*, volume 2026, pp. 34193–34211, 2026.

Table 3: Mahalanobis distances of the RECON ID evaluation actions and Gaussian-perturbed actions with $\sigma \in \{0.5, 1, 2, 5, 10\}$ from the training-action distribution at 4 fps. The final column reports the percentage of actions whose distance exceeds the 95th percentile of the training-action distances.

Condition	Mean d_{M}	P95 d_{M}	Outside train 95%
ID	1.450	2.096	5.0%
$\sigma = 0.5$	3.952	7.597	81.9%
$\sigma = 1$	7.503	14.308	96.3%
$\sigma = 2$	14.751	28.080	99.4%
$\sigma = 5$	36.731	70.010	100.0%
$\sigma = 10$	73.392	139.776	100.0%

A OOD Action Construction and Distribution Analysis

We provide additional details on the construction and analysis of the OOD actions used in the RECON and Habitat-Sim experiments.

A.1 RECON Action Distribution Analysis

Each RECON action is represented as $\mathbf{a}_t = (\Delta x_t, \Delta y_t, \Delta \psi_t)$, where Δx_t and Δy_t denote translational changes and $\Delta \psi_t$ denotes yaw change. Starting from the recorded actions in the RECON test split, we construct OOD action sequences as

$$\widetilde{\mathbf{a}}_t = \mathbf{a}_t + \boldsymbol{\epsilon}_t, \qquad \boldsymbol{\epsilon}_t \sim \mathcal{N}\left(\mathbf{0}, \sigma^2 \mathbf{I}\right). \tag{8}$$

The noise is sampled independently for each action dimension and is not clipped. We evaluate five perturbation scales, $\sigma \in \{0.5, 1, 2, 5, 10\}$.

To verify that the Gaussian perturbations produce action distributions outside the training regime, Figure 6 compares the actions from the RECON training split with the ID actions from the test split and their corresponding Gaussian-perturbed actions. Because the action space is three-dimensional, we report all three pairwise projections $(\Delta x, \Delta y)$, $(\Delta y, \Delta \psi)$, and $(\Delta \psi, \Delta x)$. The ID evaluation actions remain concentrated near the training-action distribution, whereas the Gaussian-perturbed actions extend progressively beyond its empirical support as σ increases.

We additionally quantify this shift using the Mahalanobis distance of each action to the training-action distribution. Let $\boldsymbol{\mu}_{\mathrm{train}}$ and $\boldsymbol{\Sigma}_{\mathrm{train}}$ denote the mean and covariance estimated from the training actions. We compute

$$d_{\mathrm{M}}(\mathbf{a}) = \sqrt{(\mathbf{a} - \boldsymbol{\mu}_{\mathrm{train}})^{\top} \boldsymbol{\Sigma}_{\mathrm{train}}^{-1} (\mathbf{a} - \boldsymbol{\mu}_{\mathrm{train}})}. \tag{9}$$

Table 3 reports the mean and 95th-percentile distances, together with the proportion of actions exceeding the 95th percentile of the training-action distances. The ID evaluation actions closely follow the training distribution, with 5.0% lying outside this reference boundary. Even the mildest perturbation with $\sigma = 0.5$ increases this proportion to 81.9%, and it rises to 96.3% for $\sigma = 1$. Both the mean and 95th-percentile Mahalanobis distances increase consistently with σ, confirming that the Gaussian perturbations produce progressively stronger departures from the empirical training-action distribution. Together, these results establish the Gaussian-perturbed actions as controlled OOD conditions.

A.2 Habitat-Sim Action Sampling

We construct both ID and OOD navigation trajectories in Habitat-Sim by executing locally defined actions on the simulator navigation mesh. Each action is represented as $\mathbf{a}_t = (\Delta x_t, \Delta y_t, \Delta \psi_t)$, where Δx_t and Δy_t denote forward and lateral translation in the agent coordinate frame and $\Delta \psi_t$ denotes the yaw change.

Among the three lightweight test scenes provided by Habitat-Sim, `apartment_1`, `skokloster-castle`, and `van-gogh-room`, we use `skokloster-castle` for all

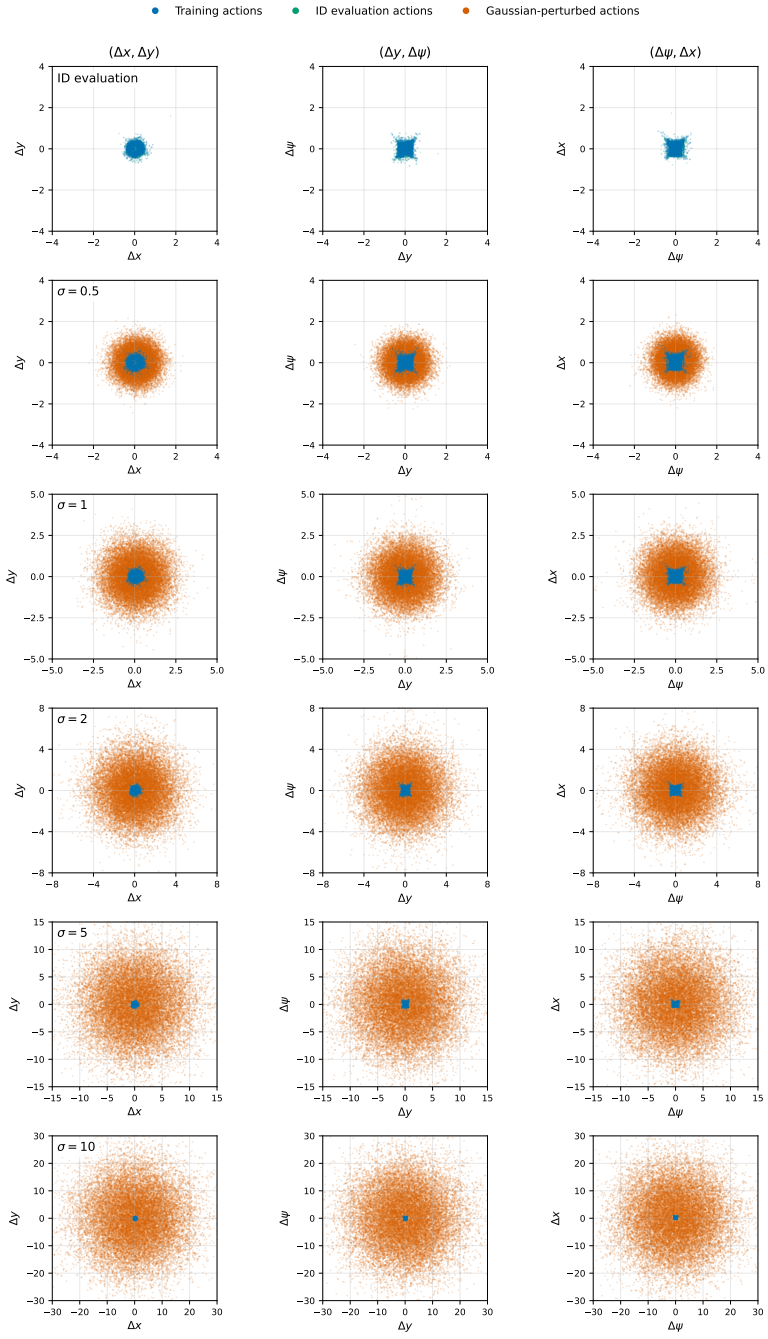

Figure 6: Pairwise projections of the RECON action distributions at 4 fps. Columns show $(\Delta x, \Delta y)$, $(\Delta y, \Delta \psi)$, and $(\Delta \psi, \Delta x)$. The first row compares the training actions with the ID evaluation actions, while the remaining rows show Gaussian-perturbed actions with $\sigma \in \{0.5, 1, 2, 5, 10\}$. Axis ranges are adjusted for each row to improve visibility.

Table 4: Action sampling rules used for the Habitat-Sim ID and structured OOD conditions. For distributions preceded by $\pm$, the sign is sampled independently with equal probability at each step. Here, e_t denotes the heading error to the next waypoint and $\eta_t \sim \mathcal{N}(0, 0.05^2)$.

Setting	Δx	Δy	$\Delta\psi$
ID	$\mathcal{U}(0.05, 0.25)$	$\mathcal{N}(0, 0.03^2)$	$\mathrm{clip}(0.35e_t + \eta_t, -1, 1)$
OOD-Δx	$\mathcal{U}(0.35, 0.50)$	$\mathcal{N}(0, 0.03^2)$	$\mathrm{clip}(0.35e_t + \eta_t, -1, 1)$
OOD-$\Delta\psi$	$\mathcal{U}(0.05, 0.25)$	$\mathcal{N}(0, 0.03^2)$	$\pm\mathcal{U}(1.0, 1.5)$
OOD-All	$\mathcal{U}(0.35, 0.50)$	$\pm\mathcal{U}(0.15, 0.30)$	$\pm\mathcal{U}(1.0, 1.5)$

experiments because its relatively open layout and sparse obstacles allow long trajectories with fewer navigation constraints. In contrast, `van-gogh-room` is spatially confined and cluttered, while `apartment_1` contains multiple connected rooms with narrow passages and obstacles, resulting in frequent projection or collision effects. Such effects can confound our analysis because a commanded action may result in substantially reduced executed motion when constrained by the environment, making it difficult to distinguish world-model failure from action truncation induced by the simulator. We therefore use only `skokloster-castle` to isolate the effect of OOD actions on rollout degeneration.

ID trajectory generation. We first sample navigable start and goal positions whose shortest-path distance lies between 3 and 20 meters. The initial heading is sampled uniformly from $[-\pi, \pi]$, and the agent follows the resulting shortest path through a sequence of waypoints. At each step, we construct a stochastic goal-directed action proposal as

$$\Delta x_t^{\mathrm{ID}} \sim \mathcal{U}(0.05, 0.25), \tag{10a}$$

$$\Delta y_t^{\mathrm{ID}} \sim \mathcal{N}(0, 0.03^2), \tag{10b}$$

$$\Delta\psi_t^{\mathrm{ID}} = \mathrm{clip}\left(0.35e_t + \eta_t, -1, 1\right), \qquad \eta_t \sim \mathcal{N}(0, 0.05^2), \tag{10c}$$

where e_t is the wrapped heading error between the current orientation and the direction of the next shortest-path waypoint. The forward and lateral components introduce variation in local motion, while the yaw component retains goal-directed behavior with stochastic rotational perturbations.

OOD trajectory generation. For structured OOD trajectories, the first four actions are sampled according to the ID action proposal rule in Equation 10, providing an ID initial context. Beginning at $t_{\mathrm{ood}} = 4$, actions are sampled according to the corresponding OOD condition in Table 4. For signed lateral and yaw intervals, the sign is selected independently and uniformly at each step.

Algorithm 1 summarizes the trajectory generation procedure. Each sampled action specifies a requested target pose whose translational component is projected onto the navigation mesh by Habitat-Sim. Since simulator constraints may prevent the requested action from being fully executed, we recompute the executed action from the resulting consecutive poses. We then reject trajectories whose executed actions violate the corresponding acceptance ranges. ID trajectories are retained only if every executed action satisfies $|\Delta x_t^{\mathrm{exec}}| \le 0.35$ m, $|\Delta y_t^{\mathrm{exec}}| \le 0.15$ m, and $|\Delta\psi_t^{\mathrm{exec}}| \le 1.0$ rad. For OOD trajectories, at every step with $t \ge t_{\mathrm{ood}}$, the acceptance range shifts to $\Delta x_t^{\mathrm{exec}} \in (0.35, 0.50]$ m for OOD-Δx and $|\Delta\psi_t^{\mathrm{exec}}| \in (1.0, 1.5]$ rad for OOD-$\Delta\psi$. For OOD-All, both constraints apply simultaneously, together with $|\Delta y_t^{\mathrm{exec}}| \in (0.15, 0.30]$ m.

We additionally reject trajectories that fail our quality criteria, including those with excessive stationary motion or without a valid navigable continuation. Specifically, we reject a trajectory if more than 10% of its steps have a horizontal displacement below 0.02 m, or if the fifth percentile of its per-step horizontal displacements is below 0.025 m. A trajectory is also discarded if, after reaching the current goal, no valid path to a new goal can be found within the allowed sampling attempts. Consequently, the retained trajectories satisfy both the prescribed action-range constraints and the quality criteria, with action labels reflecting the motion actually executed rather than the initially requested commands. Using this procedure, we generate 5,000 ID trajectories and split them into training and test sets containing 4,000 and 1,000 trajectories, respectively. For each structured OOD condition, we construct a test set of 1,000 trajectories.

Algorithm 1 Habitat-Sim trajectory generation

Require: Action mode m, trajectory length T, OOD onset t_{ood}
Sample a navigable start p_0 and goal g whose geodesic distance lies in $[3, 20]$
Compute the shortest path $\mathcal{P} = (w_0, \ldots, w_J)$ from p_0 to g
Set waypoint index $j \leftarrow 1$
Sample initial yaw $\psi_0 \sim \mathcal{U}(-\pi, \pi)$
for $t = 0, \ldots, T-2$ **do**
    Advance j past waypoints already reached by the agent
    if the final waypoint has been reached **then**
        Resample a navigable goal g and its shortest path $\mathcal{P} = (w_0, \ldots, w_J)$ within the allowed sampling attempts
        if no valid path is found **then**
            Reject the trajectory and restart
        end if
        Set $j \leftarrow 1$
    end if
    Set the current target waypoint to w_j
    Compute the wrapped heading error e_t to w_j
    Set the active mode $m_t \leftarrow$ ID if $t < t_{\mathrm{ood}}$, otherwise m
    Sample $\mathbf{a}_t^{\mathrm{req}}$ according to mode m_t in Table 4
    Compute the requested target pose from $\mathbf{a}_t^{\mathrm{req}}$
    Project the target position onto the navigation mesh
    Update the agent pose and render the next observation
end for
Recompute $\mathbf{a}_{0:T-2}^{\mathrm{exec}}$ from the executed poses
if the executed actions violate the action ranges for the selected condition or the trajectory fails the quality criteria **then**
    Reject the trajectory and restart
end if
return observations, poses, and executed actions

Action switch protocol. For the action switch experiment, we use trajectories from the ID test set rather than independently generating new source trajectories. For each source trajectory, we preserve its recorded observations and poses up to a designated switch time t_s. Beginning at t_s, every remaining action is replaced by an independently sampled OOD-All action and executed through the same navigation-mesh procedure. We use switch times of 1, 2, 4, and 8 seconds, corresponding to action indices $t_s \in \{4, 8, 16, 32\}$ at 4 fps.

We evaluate all switch conditions using the common subset of source trajectories that admit valid continuations for every switch time. This produces 968 switch-paired trajectories from the original 1,000 ID test trajectories.

B Gating Baselines and Ablation Variants

We provide architectural details for the gating baselines and ablation variants used in the NWM experiments. All variants retain the original NWM backbone and differ only in the verifier and gating modules introduced to estimate action validity and modulate the strength of action conditioning.

B.1 Action Validity Verifiers

We consider two verifier architectures to distinguish marginal action validity from state-dependent action compatibility. We denote the action-only and state–action verifier variants by V_ϕ^{A} and V_ϕ^{ZA}, respectively. The latter corresponds to the verifier V_ϕ used in the main method. The underlying classifiers produce scalar logits that are temperature-scaled to obtain the calibrated validity scores used by the gating module. Throughout the paper, V_ϕ denotes the resulting calibrated verifier mapping. Details of verifier training and temperature calibration are provided in Appendix C.

Action-only verifier. The action-only verifier estimates the validity of a commanded action without observing the current visual state,

$$q_{i,k}^{\mathrm{A}} = V_{\phi}^{\mathrm{A}}\left(\mathbf{a}_{i,k}\right). \tag{11}$$

In our Habitat-Sim setting, its input is the normalized three-dimensional navigation action $\mathbf{a}_{i,k} = (\Delta x, \Delta y, \Delta\psi)$. The verifier is implemented as an MLP with two 128-dimensional hidden layers and a scalar output, with SiLU activations after both hidden layers. Because it receives only the commanded action, this verifier captures validity with respect to the marginal training-action distribution but cannot assess whether an otherwise plausible action is compatible with the current visual state.

State–action verifier. The state–action verifier additionally conditions validity estimation on the latest visual observation,

$$q_{i,k}^{\mathrm{ZA}} = V_{\phi}^{\mathrm{ZA}}\left(\mathbf{z}_i, \mathbf{a}_{i,k}\right). \tag{12}$$

The latest context frame is encoded by the frozen VAE into a spatial latent $\mathbf{z}_i \in \mathbb{R}^{4\times 28\times 28}$. We preserve its spatial resolution and reshape it into 784 tokens. Each four-dimensional latent token is linearly projected to a 128-dimensional representation, to which we add a deterministic two-dimensional sinusoidal positional embedding followed by LayerNorm.

The three-dimensional action is independently encoded into a single 128-dimensional query token using a two-layer MLP with a 128-dimensional hidden representation and a SiLU activation. LayerNorm is applied to the resulting action token. We then use a four-head cross-attention layer with the action token as the query and the spatial state tokens as keys and values. The attention model dimension is 128.

The attended 128-dimensional state representation is concatenated with the original 128-dimensional action token, producing a 256-dimensional joint representation. A classifier with a 128-dimensional hidden layer, SiLU activation, and scalar output maps this representation to the state–action validity logit.

The resulting action-only and state–action verifiers contain 17.2K and 117.2K parameters, respectively. The verifier training protocol and construction of positive and negative examples are described in Appendix C.

B.2 Action-Gating Baselines and Variants

All gating variants produce a scalar action-conditioning strength $\alpha_{\tau,i,k}$ bounded to $[\alpha_{\min}, \alpha_{\max}]$. The resulting scalar modulates only the action embedding as described in Equation 4.

Original. The original NWM introduces no additional gating module and corresponds to $\alpha_{\tau,i,k} = 1$ for every denoising timestep and state–action pair.

Fixed Incline and Fixed Decline. We first consider two parameter-free linear schedules. Let $\rho_\tau \in [0, 1]$ denote normalized progress through the denoising process, where $\rho_\tau = 0$ and $\rho_\tau = 1$ correspond to the beginning and end of denoising, respectively. The two schedules are

$$\alpha_{\tau}^{\mathrm{inc}} = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})\rho_\tau, \tag{13}$$

$$\alpha_{\tau}^{\mathrm{dec}} = \alpha_{\max} - (\alpha_{\max} - \alpha_{\min})\rho_\tau. \tag{14}$$

These baselines test whether predefined denoising-time modulation alone is sufficient to improve robustness.

Learned-T. Learned-T is inspired by the time-driven dynamics gating of RAE-NWM (Zhang et al., 2026b), which applies an element-wise timestep-dependent gate to a fused dynamics feature jointly encoding the action and prediction horizon. However, for direct comparison with the other gating baselines, we adapt this idea to the same scalar action-gating interface used throughout our NWM experiments, such that the gate depends only on the denoising timestep. Specifically, the 384-dimensional timestep embedding is passed through an MLP with one 256-dimensional hidden

Table 5: Summary of the action-gating baselines and ablation variants evaluated with NWM. A and ZA denote the action-only and state–action verifiers, respectively.

Method	Time-varying	Learned	Verifier	Dual schedule
Original	–	–	–	–
Fixed Incline	✓	–	–	–
Fixed Decline	✓	–	–	–
Learned-T	✓	✓	–	–
SG-A	✓	✓	A	–
SG-ZA	✓	✓	ZA	–
DG-A	✓	✓	A	✓
DG-ZA (Ours)	✓	✓	ZA	✓

layer, SiLU activation, and a scalar output. The output is mapped to the allowed action-conditioning range as

$$\alpha_\tau^{\mathrm{T}} = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})\sigma\left(G_{\mathrm{T}}(\mathbf{e}_\tau(\tau))\right). \tag{15}$$

Learned-T can therefore learn a flexible denoising-time schedule, but the resulting gate is agnostic to the commanded action and current visual state.

SG-A and SG-ZA. Like Learned-T, the single-gate variants use a single learned gating function to modulate action-conditioning strength over the denoising process. Unlike Learned-T, however, the gate additionally conditions on the estimated validity of the commanded action. Specifically, the 384-dimensional diffusion-timestep embedding is concatenated with the scalar validity score, producing a 385-dimensional input. This representation is processed by an MLP with one 256-dimensional hidden layer, SiLU activation, and a scalar output:

$$\alpha_{\tau,i,k}^{\mathrm{SG}} = \alpha_{\min} + (\alpha_{\max} - \alpha_{\min})\sigma\left(G_{\mathrm{SG}}\left([\mathbf{e}_\tau(\tau); q_{i,k}]\right)\right). \tag{16}$$

Thus, a single gating network directly learns the interaction between denoising time and action validity, rather than maintaining separate schedules for supported and unsupported transitions. SG-A uses the action-only validity score $q_{i,k} = q_{i,k}^{\mathrm{A}}$, whereas SG-ZA uses the state–action validity score $q_{i,k} = q_{i,k}^{\mathrm{ZA}}$.

DG-A and DG-ZA. Unlike the single-gate variants, the dual-gate variants explicitly maintain separate learned denoising-time schedules for supported and unsupported transitions. The ID and OOD schedules are parameterized by two independent MLPs, each taking the 384-dimensional timestep embedding as input and using one 256-dimensional hidden layer with SiLU activation followed by a scalar output. They correspond to the schedules $\alpha_\tau^{\mathrm{ID}}$ and $\alpha_\tau^{\mathrm{OOD}}$ defined in Equation 2. The validity score then interpolates between the two schedules according to Equation 3. DG-A uses the action-only validity score $q_{i,k} = q_{i,k}^{\mathrm{A}}$, whereas DG-ZA uses the state–action validity score $q_{i,k} = q_{i,k}^{\mathrm{ZA}}$. DG-ZA corresponds to our complete validity-aware action-gating architecture.

The gating modules introduce only a small number of additional parameters relative to the backbone. Fixed Incline and Fixed Decline are parameter-free, while Learned-T contains 98.8K gating parameters. The single-gate variants SG-A and SG-ZA use 99.1K gating parameters, and the dual-gate variants DG-A and DG-ZA use 197.6K gating parameters. Table 5 summarizes the architectural differences among the evaluated variants.

C NWM Training Details

C.1 World Models for Empirical Analysis

For the empirical analysis in Section 3, we train separate NWM models with a CDiT-S/2 backbone on the RECON training split and the Habitat-Sim ID training split. The CDiT-S/2 backbone consists of 12 transformer blocks with a hidden dimension of 384, 6 attention heads, and a patch size of 2,

resulting in approximately 50M parameters. Both models use four context frames at a resolution of 224×224 and 1,000 diffusion steps.

Both models are trained for 100k optimization steps using the original NWM diffusion objective. We use AdamW with a learning rate of 10^{-4}, a per-GPU batch size of 8 on two NVIDIA RTX A5000 GPUs, and gradient clipping at 10. The analysis models are trained without action dropout, unlike the validity-aware variants used in the mitigation experiments described below.

C.2 Verifier Training

For the mitigation experiments in Section 5, we train the action-only and state–action verifiers before training the validity-aware world models. Verifier training uses only the 4,000 trajectories in the Habitat-Sim ID training split, which are further partitioned at the trajectory level into 80% training, 10% validation, and 10% test sets. The 1,000 Habitat-Sim test trajectories used for world model rollout evaluation are not used for verifier training or calibration. For both verifier variants, we construct balanced binary classification datasets with equal numbers of positive and negative examples. Positive examples are derived from valid transitions in the ID training trajectories, while the construction of negative examples depends on the verifier architecture, as detailed below.

Action-only verifier. Positive examples consist of actions from the Habitat-Sim ID training trajectories. We do not use actions from the structured OOD test trajectories as negative training examples. Instead, we independently sample synthetic actions from the same ranges used to construct the OOD-All condition in Table 4. Because the action-only verifier does not observe the visual state, all of its negative examples are constructed from these synthetic OOD actions.

State–action verifier. Positive examples pair a ground-truth observation with the executed action at the corresponding timestep. For negative examples, we use a mixture of synthetic OOD actions and state–action mismatches. Synthetic OOD negatives account for 80% of the negative examples and pair the current ground-truth state with an independently sampled OOD-All action. Cross-trajectory negatives account for 10% and pair the current state with an ID range action drawn from a different trajectory. The remaining 10% are temporal mismatch negatives, which pair the current state with an action from a different timestep of the same trajectory. The latter two negative types expose the verifier to actions that can be marginally plausible under the training action distribution but incompatible with the current visual state.

All state inputs used to train the state–action verifier are ground-truth observations from the dataset. The latest observation is encoded using the same frozen VAE used by the world model. We sample from the VAE posterior during training and use the posterior mean for validation and testing.

Optimization and calibration. Both verifiers are trained for 20 epochs using binary cross-entropy and AdamW with a learning rate of 3×10^{-4}, weight decay of 10^{-4}, and gradient clipping at 5. We use bfloat16 mixed precision. The action-only verifier uses a batch size of 512, while the state–action verifier uses a batch size of 64. For each verifier, we select the checkpoint with the highest validation AUROC. We then fit a scalar temperature T_v to the validation logits using L-BFGS. Given a classifier logit v, the calibrated validity score used by the gating module is

$$q = \sigma\left(\frac{v}{T_v}\right), \tag{17}$$

where $\sigma(\cdot)$ denotes the sigmoid function. The calibrated verifier is frozen during subsequent world model training and rollout inference.

Verifier performance. On the verifier test set, the action-only verifier achieves an accuracy of 99.98% and an AUROC of 1.000, while the state–action verifier achieves an accuracy of 89.70% and an AUROC of 0.898 under the mixed negative construction described above. The near-perfect performance of the action-only verifier reflects the strong marginal separation between ID actions and the synthetic OOD-All action range, whereas the state–action verifier addresses the more challenging problem of identifying state-dependent action compatibility.

C.3 World Model Training for Mitigation Experiments

ID pretraining. For the mitigation experiments, the Original baseline uses the same 100k Habitat-Sim checkpoint as the empirical analysis, while the remaining variants are trained for 100k steps on the same Habitat-Sim ID training split using the original NWM diffusion objective. The Original, Fixed Incline, Fixed Decline, and Learned-T baselines are trained without action dropout.

For validity-aware variants, the corresponding pretrained verifier remains frozen and provides the validity score for each training transition. We apply action dropout with probability $p_{\text{drop}} = 0.1$ by replacing the action embedding with a zero vector. The learned single- and dual-gate modules are initialized such that the action-conditioning strength is $\alpha = 1$, corresponding to the original NWM conditioning at initialization. For the dual-gate variants, the ID and OOD gates therefore begin from the same constant schedule and are subsequently allowed to diverge through training. All gating methods use $\alpha_{\min} = 0.1$ and $\alpha_{\max} = 2.0$.

OOD-aware post-tuning. Starting from the 100k checkpoint, we further train each validity-aware model for 20k steps using the OOD-aware objective in Equation 7. The full student world model, including the CDiT-S/2 backbone and action gate, remains trainable during this stage, while the VAE and verifier remain frozen. We additionally construct a frozen null-action reference model from the corresponding 100k EMA checkpoint.

Each post-tuning step combines an ID branch using the standard diffusion objective with an OOD branch using the consistency objective. For the ID branch, we retain the same diffusion objective used during the initial 100k steps, including action dropout with $p_{\text{drop}} = 0.1$. For the OOD branch, we independently sample a synthetic unsupported action from the OOD-All ranges in Table 4 and compute its validity score using the frozen verifier. The student is conditioned on this synthetic action, whereas the frozen reference receives null action conditioning. The two predictions otherwise use the same target frame, sampled diffusion noise, noisy target latent, denoising timestep, visual context, and prediction horizon, and therefore differ only in their action conditioning and model parameters. Action dropout is disabled for the OOD branch to avoid trivial consistency cases in which the unsupported action is removed and both models receive null action conditioning.

During training, we use AdamW with a learning rate of 10^{-4}, a per-GPU batch size of 8 on two NVIDIA RTX A5000 GPUs, gradient clipping at 10, bfloat16 mixed precision, and an EMA decay of 0.9999. The OOD consistency weight is linearly increased from zero to $\lambda_{\text{OOD}} = 0.1$ over the first 2k post-tuning steps and remains fixed thereafter. The 20k-step continuation resumes the model, EMA, optimizer, and mixed-precision scaler states from the corresponding 100k checkpoint.

Baseline continuation. For a fair comparison, the Original, Fixed Incline, Fixed Decline, and Learned-T baselines are also trained for an additional 20k steps from their respective 100k checkpoints using the same optimization settings. These baselines continue to optimize only their ID training objectives and do not use the verifier, synthetic OOD samples, null-action reference, or OOD consistency objective. Consequently, all methods are compared after the same total number of 120k optimization steps.

D Quantitative Results

The following tables report exact numerical results at every second of the 16-second rollout horizon for the empirical analysis in Section 3 and the mitigation experiments in Section 5. All reported metrics are distances, and lower values indicate better performance. For RECON, we report FID and support distance because ground-truth observations are unavailable for the Gaussian-perturbed OOD action sequences. For Habitat-Sim, where corresponding ground-truth frames can be rendered, we additionally report LPIPS and DreamSim. All values are reported to three decimal places.

For support distance evaluation, the training-state feature bank is constructed from all frames in the corresponding training split. Each frame is encoded with the frozen VAE, and its latent representation is flattened and standardized per dimension using the mean and standard deviation of the training-state bank. The same normalization is applied to generated-frame features before computing the mean Euclidean distance to the $K = 5$ nearest training-state features.

D.1 RECON LONG-HORIZON ROLLOUTS

Table 6 reports the exact values corresponding to Figure 2. The Gaussian action perturbations are applied independently to each action dimension without clipping.

The Gaussian perturbations generally shift both FID and support distance upward as the perturbation magnitude increases, although this severity ordering is not perfectly monotonic at every horizon. Even the mildest perturbation, $\sigma = 0.5$, departs from the ID rollout from the early stages, with OOD conditions generally exhibiting worse initial metrics but slower subsequent growth. As a result, some milder OOD curves, most notably $\sigma = 0.5$, approach or cross the ID curve at later horizons despite their earlier degradation.

D.2 HABITAT-SIM STRUCTURED OOD ROLLOUTS

Table 7 reports all four metrics for the ID and structured OOD conditions corresponding to Figure 3.

The ID rollout exhibits the same gradual degeneration observed on RECON, with all metrics worsening steadily over the rollout horizon. In contrast, OOD-$\Delta\psi$ and OOD-All begin with substantially worse metrics from the early stages but grow much more slowly thereafter, approaching a broad plateau in the later stages. This difference in temporal slope causes the OOD and ID curves to eventually cross despite the initial metric gap. The OOD-Δx condition exhibits an even more extreme behavior. Its metrics initially worsen, reach a peak around 10–11 seconds, and then begin to improve, producing an inverted-U-shaped trajectory. As discussed in Section 3.2, this apparent improvement reflects conservative collapse rather than genuine recovery, as further illustrated by the qualitative examples in Appendix F.

D.3 ACTION SWITCH EXPERIMENT

Table 8 reports the complete metric values for the switch experiment, including FID and support distance in addition to the LPIPS and DreamSim results shown in Figure 4. Each condition is evaluated on the same 968 switch-paired trajectories. We note that each switch trajectory contributes a single rollout since the switch time is fixed for each condition, whereas the preceding Habitat-Sim evaluation extracts eight rollout windows from different temporal offsets of each trajectory. The switch results may therefore differ in absolute value and exhibit slightly noisier trends.

For each switch time, the metrics closely track the paired ID rollout before the transition and change abruptly at the first OOD-controlled second. After the switch, all metrics rapidly approach a plateau, closely resembling the behavior of the persistent OOD-All rollout reported in Appendix D.2.

D.4 ACTION-GATING BASELINES AND ABLATIONS

We provide the complete temporal results for the action-gating baselines and validity-aware variants evaluated in Section 5. Tables 9, 10, 11, and 12 report all four metrics at every second of the 16-second rollout horizon under ID and OOD-All actions. We report both the 100k checkpoints obtained after the initial training stage and the corresponding 120k checkpoints after an additional 20k optimization steps. For Original, Fixed Incline, Fixed Decline, and Learned-T, the additional steps continue the ID training objective, whereas SG-A, SG-ZA, DG-A, and DG-ZA undergo the OOD-aware post-tuning stage described in Appendix C.3.

Figure 7 reports the corresponding 100k performance before OOD-aware post-tuning, while Figure 5 presents the final 120k results.

The 100k results in Tables 9 and 10 already reveal substantial performance differences among the methods before OOD-aware post-tuning. Under ID actions, DG-ZA achieves the lowest 16-second values across all four metrics, reducing LPIPS, DreamSim, FID, and support distance by 14.6%, 38.0%, 82.0%, and 14.4%, respectively, relative to the 100k Original model. Fixed Decline and Learned-T exhibit similarly stable long-horizon behavior. Under OOD-All actions, DG-ZA reduces the same metrics by 2.7%, 15.3%, 45.0%, and 3.9%, respectively, relative to the 100k Original model at 16 seconds. It achieves the lowest DreamSim, FID, and support distance at this horizon, with LPIPS nearly tied with SG-ZA.

Table 6: Long-horizon rollout metrics under ID and Gaussian-perturbed OOD actions on RECON. Values are reported at every second of the 16-second rollout.

Setting	Metric	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
ID	FID	13.746	14.397	14.697	15.664	16.134	17.217	18.000	19.603	19.783	21.134	22.405	22.892	23.797	24.768	25.182	25.146
	Support	65.321	65.341	65.488	65.642	65.890	66.161	66.390	66.574	66.872	67.037	67.250	67.423	67.570	67.638	67.791	67.761
$\sigma = 0.5$	FID	13.891	15.201	15.757	16.679	16.958	18.480	18.462	19.559	19.425	20.038	20.654	21.187	21.508	22.234	23.158	23.250
	Support	65.942	66.072	66.197	66.424	66.500	66.637	66.750	66.959	67.057	67.043	67.078	67.212	67.283	67.327	67.349	67.462
$\sigma = 1$	FID	15.889	17.937	18.118	19.045	20.010	20.162	20.827	20.828	21.845	22.746	22.563	23.878	24.030	25.150	25.817	26.151
	Support	66.597	66.981	67.012	67.090	67.213	67.282	67.437	67.482	67.466	67.511	67.629	67.696	67.686	67.706	67.732	67.762
$\sigma = 2$	FID	22.129	24.145	25.033	25.557	24.956	26.670	26.579	27.982	28.080	28.478	29.566	29.498	30.751	31.448	31.379	32.825
	Support	67.162	67.493	67.619	67.551	67.632	67.616	67.736	67.732	67.802	67.719	67.897	67.833	67.903	67.961	67.961	68.004
$\sigma = 5$	FID	19.612	21.793	22.109	24.454	23.731	25.143	26.409	27.076	27.392	27.726	29.243	29.032	29.747	30.820	31.962	32.594
	Support	67.197	67.536	67.627	67.767	67.764	68.051	68.054	68.210	68.204	68.192	68.295	68.417	68.481	68.578	68.664	68.766
$\sigma = 10$	FID	20.692	24.337	25.877	26.968	28.592	28.895	29.512	31.568	33.048	33.231	34.337	34.969	34.793	35.880	36.369	37.077
	Support	67.957	68.436	68.545	68.830	68.837	69.030	69.133	69.221	69.474	69.624	69.725	69.744	69.705	69.875	69.844	69.919

Table 7: Long-horizon rollout metrics under ID and structured OOD actions in Habitat-Sim. Values are reported at every second of the 16-second rollout.

Setting	Metric	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
ID	LPIPS	0.307	0.340	0.363	0.382	0.399	0.414	0.429	0.443	0.456	0.469	0.481	0.492	0.503	0.514	0.524	0.534
	DreamSim	0.171	0.187	0.202	0.220	0.238	0.258	0.279	0.300	0.321	0.343	0.364	0.386	0.406	0.425	0.444	0.463
	FID	17.302	18.766	19.796	22.503	25.485	30.130	35.267	41.823	48.273	55.796	63.914	71.181	79.922	87.778	96.005	103.439
	Support	72.374	72.472	72.756	73.334	73.920	74.767	75.632	76.597	77.579	78.671	79.761	80.926	82.070	83.176	84.270	85.315
OOD-Δx	LPIPS	0.334	0.381	0.416	0.444	0.465	0.482	0.497	0.508	0.515	0.518	0.518	0.515	0.513	0.508	0.504	0.499
	DreamSim	0.177	0.215	0.258	0.302	0.340	0.375	0.407	0.432	0.447	0.454	0.453	0.443	0.433	0.420	0.409	0.398
	FID	15.795	20.203	28.040	39.106	52.780	67.913	80.317	93.668	99.911	104.087	101.426	95.710	86.876	79.880	73.169	67.067
	Support	72.093	72.714	74.026	75.630	77.376	79.155	80.769	82.143	82.896	83.287	83.154	82.672	82.001	81.282	80.653	80.020
OOD-$\Delta\psi$	LPIPS	0.467	0.485	0.491	0.495	0.496	0.497	0.499	0.501	0.501	0.501	0.502	0.502	0.503	0.503	0.503	0.503
	DreamSim	0.356	0.391	0.403	0.408	0.411	0.412	0.415	0.419	0.418	0.417	0.418	0.419	0.421	0.421	0.419	0.422
	FID	59.696	70.238	72.293	73.293	72.833	73.727	74.163	76.275	75.515	75.063	74.459	75.224	75.547	76.409	74.359	75.435
	Support	78.829	79.495	79.674	79.794	79.776	79.817	79.911	80.023	80.000	79.928	79.968	79.963	80.027	79.986	79.938	79.963
OOD-All	LPIPS	0.446	0.461	0.465	0.468	0.470	0.472	0.473	0.475	0.476	0.476	0.477	0.478	0.478	0.478	0.479	0.480
	DreamSim	0.327	0.354	0.360	0.366	0.369	0.373	0.374	0.378	0.379	0.380	0.379	0.383	0.383	0.382	0.383	0.385
	FID	60.463	64.975	65.488	67.805	69.547	71.642	71.962	73.091	74.807	75.297	74.418	77.782	76.821	76.488	77.057	77.076
	Support	77.174	77.712	77.788	77.939	78.129	78.272	78.398	78.486	78.694	78.648	78.684	78.801	78.737	78.757	78.819	78.835

Table 8: Rollout metrics when switching from ID actions to OOD-All actions after 1, 2, 4, or 8 seconds in Habitat-Sim. The ID condition is evaluated on the same 968 source ID trajectories used to construct the switch conditions. Values are reported at every second of the 16-second rollout.

Condition	Metric	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
ID	LPIPS	0.368	0.383	0.388	0.393	0.399	0.407	0.415	0.421	0.430	0.437	0.446	0.454	0.463	0.473	0.481	0.492
	DreamSim	0.213	0.219	0.221	0.224	0.228	0.237	0.248	0.257	0.270	0.282	0.296	0.312	0.330	0.348	0.364	0.382
	FID	27.293	27.292	26.832	26.841	27.832	28.570	29.771	33.043	36.097	39.585	42.296	49.351	56.967	63.578	69.378	76.913
	Support	73.959	73.452	73.072	72.796	72.704	72.949	73.326	73.735	74.304	74.921	75.514	76.443	77.358	78.319	79.237	80.352
Switch @ 1 s	LPIPS	0.450	0.471	0.479	0.486	0.488	0.490	0.491	0.495	0.499	0.495	0.496	0.498	0.501	0.500	0.501	0.498
	DreamSim	0.331	0.370	0.378	0.386	0.389	0.390	0.393	0.395	0.405	0.400	0.400	0.404	0.405	0.404	0.406	0.402
	FID	59.289	70.151	70.700	66.743	66.558	66.638	68.558	66.872	68.125	68.065	68.715	69.505	66.785	70.945	71.362	68.573
	Support	77.776	78.541	78.461	78.469	78.450	78.479	78.637	78.745	78.844	78.891	78.915	78.913	78.950	78.918	78.982	78.817
Switch @ 2 s	LPIPS	0.368	0.449	0.469	0.478	0.482	0.486	0.489	0.489	0.489	0.491	0.494	0.496	0.497	0.495	0.497	0.499
	DreamSim	0.212	0.326	0.365	0.375	0.383	0.392	0.391	0.393	0.388	0.393	0.399	0.398	0.398	0.398	0.400	0.401
	FID	27.516	59.841	67.524	66.070	70.969	71.752	70.530	69.422	66.157	68.601	70.438	71.162	69.347	69.342	69.482	70.648
	Support	73.952	77.164	78.094	78.094	78.505	78.744	78.636	78.654	78.521	78.655	78.854	78.834	78.857	78.787	78.896	78.814
Switch @ 4 s	LPIPS	0.368	0.385	0.388	0.448	0.468	0.476	0.483	0.483	0.486	0.487	0.492	0.494	0.493	0.494	0.495	0.495
	DreamSim	0.213	0.219	0.221	0.321	0.364	0.379	0.385	0.388	0.387	0.390	0.400	0.401	0.395	0.401	0.400	0.398
	FID	27.359	27.248	26.905	60.235	71.087	71.752	73.353	70.367	67.506	69.468	74.900	74.427	69.683	72.110	71.690	72.492
	Support	74.042	73.500	73.091	76.707	78.112	78.261	78.559	78.572	78.421	78.570	78.986	78.950	78.750	78.948	78.885	78.998
Switch @ 8 s	LPIPS	0.369	0.384	0.389	0.393	0.399	0.406	0.415	0.456	0.469	0.476	0.478	0.481	0.485	0.485	0.488	0.490
	DreamSim	0.213	0.218	0.221	0.223	0.228	0.236	0.247	0.334	0.367	0.379	0.379	0.383	0.387	0.389	0.394	0.395
	FID	27.120	27.200	26.730	26.619	27.719	28.041	29.826	60.289	74.772	71.583	70.150	70.814	69.847	72.742	74.035	71.581
	Support	73.998	73.437	72.979	72.721	72.634	72.915	73.319	76.889	78.119	78.223	78.342	78.440	78.468	78.613	78.839	78.847

Table 9: ID rollout metrics for all methods after 100k optimization steps on Habitat-Sim. Values are reported at every second of the 16-second rollout.

Method	Metric	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Original	LPIPS	0.307	0.340	0.363	0.382	0.399	0.414	0.429	0.443	0.456	0.469	0.481	0.492	0.503	0.514	0.524	0.534
	DreamSim	0.171	0.187	0.202	0.220	0.238	0.258	0.279	0.300	0.321	0.343	0.364	0.386	0.406	0.425	0.444	0.463
	FID	17.302	18.766	19.796	22.503	25.485	30.130	35.267	41.823	48.273	55.796	63.914	71.181	79.922	87.778	96.005	103.439
	Support	72.374	72.472	72.756	73.334	73.920	74.767	75.632	76.597	77.579	78.671	79.761	80.926	82.070	83.176	84.270	85.315
Fixed Incline	LPIPS	0.315	0.344	0.365	0.384	0.400	0.415	0.429	0.442	0.454	0.465	0.476	0.487	0.497	0.508	0.517	0.526
	DreamSim	0.172	0.188	0.203	0.221	0.240	0.261	0.281	0.302	0.323	0.343	0.364	0.384	0.404	0.423	0.440	0.458
	FID	16.453	17.066	18.355	20.960	24.350	28.763	33.977	40.946	48.507	56.072	64.080	73.120	82.349	90.976	99.863	108.659
	Support	72.714	72.754	73.041	73.553	74.081	74.829	75.676	76.580	77.527	78.536	79.458	80.579	81.628	82.601	83.608	84.589
Fixed Decline	LPIPS	0.294	0.328	0.352	0.370	0.384	0.397	0.407	0.416	0.424	0.430	0.436	0.442	0.446	0.451	0.454	0.459
	DreamSim	0.158	0.171	0.184	0.196	0.206	0.217	0.228	0.236	0.244	0.251	0.258	0.266	0.272	0.278	0.285	0.290
	FID	13.489	13.527	13.956	14.645	14.908	15.762	15.985	16.370	16.876	17.407	17.964	18.446	19.239	19.578	20.532	20.854
	Support	71.882	71.855	71.775	71.854	71.915	72.024	72.133	72.226	72.314	72.369	72.563	72.723	72.904	73.048	73.268	73.387
Learned-T	LPIPS	0.301	0.334	0.357	0.374	0.387	0.399	0.410	0.418	0.425	0.431	0.437	0.442	0.448	0.453	0.456	0.460
	DreamSim	0.164	0.177	0.190	0.202	0.212	0.222	0.233	0.241	0.249	0.256	0.264	0.269	0.277	0.284	0.289	0.296
	FID	14.837	15.151	15.189	15.822	16.193	16.504	17.226	17.924	18.256	18.892	19.894	20.128	21.005	21.645	22.399	23.130
	Support	72.067	72.023	71.968	72.065	72.085	72.193	72.318	72.369	72.464	72.594	72.782	72.891	73.013	73.256	73.389	73.532
SG-A	LPIPS	0.312	0.343	0.365	0.382	0.397	0.412	0.425	0.437	0.449	0.460	0.470	0.480	0.490	0.499	0.508	0.516
	DreamSim	0.172	0.187	0.203	0.221	0.239	0.259	0.278	0.299	0.319	0.340	0.360	0.380	0.401	0.420	0.439	0.456
	FID	16.660	17.685	19.122	22.017	24.897	29.440	34.035	40.578	47.157	54.543	62.996	71.104	79.785	87.971	97.493	105.977
	Support	72.871	73.046	73.392	73.921	74.470	75.285	76.009	76.878	77.749	78.744	79.664	80.631	81.620	82.587	83.503	84.426
SG-ZA	LPIPS	0.317	0.349	0.370	0.389	0.404	0.418	0.430	0.440	0.450	0.459	0.467	0.475	0.482	0.487	0.492	0.496
	DreamSim	0.181	0.198	0.215	0.232	0.249	0.267	0.284	0.299	0.315	0.332	0.347	0.362	0.376	0.388	0.399	0.409
	FID	20.012	22.538	25.237	29.352	33.425	38.721	43.445	49.280	54.796	61.096	67.212	73.728	79.496	84.725	88.781	93.076
	Support	73.165	73.219	73.449	73.770	74.258	74.710	75.193	75.695	76.289	76.962	77.532	78.136	78.692	79.170	79.651	80.053
DG-A	LPIPS	0.306	0.338	0.359	0.378	0.394	0.410	0.424	0.437	0.448	0.459	0.469	0.479	0.488	0.496	0.504	0.511
	DreamSim	0.170	0.187	0.204	0.222	0.240	0.260	0.280	0.301	0.319	0.340	0.359	0.377	0.395	0.413	0.430	0.445
	FID	16.565	17.719	19.317	21.960	24.784	29.259	33.980	39.448	44.986	52.039	58.670	66.014	73.744	80.719	88.467	95.547
	Support	72.614	72.709	73.145	73.612	74.166	74.902	75.627	76.429	77.238	78.112	78.934	79.790	80.593	81.421	82.240	82.992
DG-ZA (Ours)	LPIPS	0.296	0.329	0.351	0.369	0.383	0.396	0.406	0.415	0.422	0.429	0.436	0.440	0.445	0.449	0.453	0.456
	DreamSim	0.162	0.175	0.188	0.199	0.210	0.221	0.229	0.238	0.245	0.253	0.260	0.265	0.271	0.276	0.283	0.287
	FID	14.803	14.786	14.930	15.008	15.138	15.829	15.746	16.222	16.469	16.943	17.188	17.400	17.891	17.832	18.328	18.653
	Support	72.268	72.208	72.146	72.142	72.181	72.173	72.227	72.345	72.406	72.476	72.593	72.614	72.696	72.820	72.880	73.008

Table 10: OOD-All rollout metrics for all methods after 100k optimization steps on Habitat-Sim. Values are reported at every second of the 16-second rollout.

Method	Metric	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Original	LPIPS	0.446	0.461	0.465	0.468	0.470	0.472	0.473	0.475	0.476	0.476	0.477	0.478	0.478	0.478	0.479	0.480
	DreamSim	0.327	0.354	0.360	0.366	0.369	0.373	0.374	0.378	0.379	0.380	0.379	0.383	0.383	0.382	0.383	0.385
	FID	60.463	64.975	65.488	67.805	69.547	71.642	71.962	73.091	74.807	75.297	74.418	77.782	76.821	76.488	77.057	77.076
	Support	77.174	77.712	77.788	77.939	78.129	78.272	78.398	78.486	78.694	78.648	78.684	78.801	78.737	78.757	78.819	78.835
Fixed Incline	LPIPS	0.447	0.460	0.464	0.467	0.469	0.471	0.472	0.474	0.473	0.474	0.475	0.474	0.474	0.474	0.475	0.474
	DreamSim	0.324	0.349	0.354	0.361	0.365	0.367	0.369	0.371	0.371	0.373	0.371	0.372	0.372	0.371	0.372	0.372
	FID	58.221	63.201	64.257	66.868	69.207	69.776	71.668	72.215	72.001	73.576	72.723	73.002	72.489	72.492	73.151	72.051
	Support	77.359	77.892	78.007	78.164	78.413	78.373	78.592	78.662	78.550	78.732	78.628	78.701	78.526	78.608	78.656	78.621
Fixed Decline	LPIPS	0.419	0.446	0.458	0.462	0.468	0.471	0.472	0.474	0.475	0.475	0.476	0.477	0.478	0.478	0.478	0.478
	DreamSim	0.242	0.285	0.306	0.318	0.329	0.336	0.339	0.343	0.344	0.346	0.350	0.349	0.351	0.351	0.352	0.350
	FID	20.577	27.254	32.400	35.976	39.875	43.679	45.927	47.994	48.766	50.299	51.826	52.301	52.850	53.117	52.915	51.976
	Support	73.895	74.791	75.401	75.659	76.149	76.465	76.660	76.704	76.886	76.915	77.127	77.009	77.239	77.157	77.266	77.062
Learned-T	LPIPS	0.423	0.443	0.451	0.458	0.460	0.463	0.464	0.467	0.467	0.468	0.468	0.469	0.469	0.469	0.469	0.470
	DreamSim	0.246	0.279	0.297	0.310	0.317	0.323	0.327	0.333	0.332	0.336	0.335	0.337	0.337	0.338	0.339	0.340
	FID	19.526	23.810	28.513	33.840	37.760	40.447	43.784	45.820	46.737	49.673	49.465	51.259	50.940	51.630	51.319	52.894
	Support	73.409	73.843	74.507	75.109	75.337	75.610	75.921	76.104	76.264	76.437	76.393	76.526	76.540	76.578	76.576	76.671
SG-A	LPIPS	0.449	0.465	0.470	0.475	0.476	0.480	0.480	0.482	0.484	0.485	0.486	0.487	0.488	0.487	0.488	0.489
	DreamSim	0.333	0.366	0.378	0.384	0.385	0.392	0.393	0.397	0.398	0.399	0.401	0.403	0.404	0.402	0.404	0.404
	FID	65.482	77.905	82.232	83.880	84.501	87.973	88.056	90.229	91.083	92.799	92.424	94.392	95.187	94.182	94.083	94.863
	Support	77.450	78.547	78.986	79.216	79.316	79.632	79.692	79.835	79.915	80.008	80.046	80.193	80.272	80.099	80.150	80.179
SG-ZA	LPIPS	0.438	0.449	0.454	0.458	0.460	0.463	0.462	0.465	0.466	0.466	0.465	0.467	0.466	0.466	0.466	0.466
	DreamSim	0.293	0.311	0.321	0.329	0.335	0.340	0.339	0.343	0.346	0.345	0.343	0.346	0.345	0.344	0.345	0.344
	FID	39.134	40.745	43.333	47.776	49.249	51.096	51.760	52.756	53.866	54.286	53.302	55.024	54.727	54.352	53.759	53.221
	Support	75.066	75.368	75.834	76.235	76.483	76.624	76.653	76.759	76.906	76.784	76.747	76.903	76.807	76.716	76.740	76.697
DG-A	LPIPS	0.450	0.461	0.465	0.468	0.469	0.472	0.472	0.474	0.474	0.475	0.475	0.475	0.476	0.477	0.477	0.478
	DreamSim	0.330	0.352	0.358	0.362	0.364	0.369	0.368	0.372	0.371	0.372	0.371	0.373	0.372	0.375	0.374	0.376
	FID	60.677	66.005	67.178	68.333	68.676	70.044	70.221	71.742	71.509	71.898	71.261	72.700	72.454	73.570	72.760	72.973
	Support	77.368	77.924	78.002	78.092	78.262	78.391	78.406	78.520	78.402	78.607	78.503	78.505	78.405	78.616	78.599	78.557
DG-ZA (Ours)	LPIPS	0.428	0.448	0.454	0.459	0.461	0.463	0.462	0.464	0.466	0.467	0.466	0.468	0.467	0.467	0.468	0.467
	DreamSim	0.255	0.287	0.302	0.311	0.317	0.320	0.320	0.322	0.325	0.325	0.325	0.328	0.326	0.325	0.329	0.326
	FID	21.113	26.275	30.860	34.859	37.311	39.646	40.463	41.161	42.607	42.911	42.085	44.038	43.595	43.071	43.527	42.373
	Support	74.038	74.559	75.038	75.306	75.549	75.576	75.607	75.637	75.809	75.779	75.792	75.888	75.909	75.729	75.927	75.751

Table 11: ID rollout metrics for all methods after 120k optimization steps on Habitat-Sim. The Original, Fixed Incline, Fixed Decline, and Learned-T baselines receive 20k additional ID training steps, whereas the validity-aware variants receive 20k OOD-aware post-tuning steps from their respective 100k-step checkpoints.

Method	Metric	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Original	LPIPS	0.296	0.333	0.359	0.379	0.397	0.412	0.425	0.438	0.449	0.459	0.469	0.477	0.486	0.493	0.500	0.506
	DreamSim	0.172	0.190	0.208	0.226	0.246	0.264	0.282	0.301	0.317	0.333	0.349	0.363	0.379	0.391	0.404	0.416
	FID	20.367	23.716	26.746	30.531	34.428	38.736	43.056	48.756	53.537	57.923	62.896	67.431	73.082	77.804	82.651	86.551
	Support	72.674	72.888	73.204	73.567	74.050	74.504	75.014	75.578	76.097	76.558	77.073	77.576	78.086	78.541	79.013	79.550
Fixed Incline	LPIPS	0.302	0.336	0.358	0.378	0.395	0.412	0.426	0.441	0.454	0.467	0.480	0.492	0.503	0.514	0.524	0.535
	DreamSim	0.159	0.175	0.191	0.208	0.228	0.249	0.269	0.290	0.312	0.334	0.356	0.377	0.399	0.420	0.438	0.458
	FID	13.223	13.850	14.958	17.183	19.996	24.129	29.082	35.025	41.613	48.898	56.305	64.539	73.356	81.880	89.766	97.496
	Support	71.984	72.128	72.392	72.937	73.636	74.431	75.292	76.347	77.450	78.602	79.717	80.854	82.026	83.154	84.231	85.365
Fixed Decline	LPIPS	0.284	0.320	0.346	0.365	0.381	0.395	0.406	0.416	0.424	0.431	0.438	0.443	0.448	0.453	0.456	0.460
	DreamSim	0.151	0.165	0.177	0.190	0.201	0.212	0.223	0.233	0.241	0.249	0.258	0.264	0.271	0.278	0.285	0.291
	FID	11.884	12.354	12.554	13.156	13.698	14.360	14.829	15.665	15.547	16.632	17.282	17.662	18.258	18.802	19.664	20.304
	Support	71.492	71.431	71.470	71.539	71.610	71.762	71.878	72.056	72.090	72.246	72.360	72.496	72.663	72.847	73.023	73.199
Learned-T	LPIPS	0.289	0.325	0.350	0.370	0.385	0.398	0.409	0.418	0.425	0.432	0.438	0.444	0.449	0.455	0.459	0.464
	DreamSim	0.156	0.170	0.182	0.196	0.208	0.220	0.230	0.239	0.247	0.255	0.263	0.270	0.278	0.287	0.295	0.302
	FID	13.423	13.669	13.975	14.811	15.440	16.139	15.939	17.130	17.325	17.989	19.044	19.723	20.449	21.408	22.460	23.313
	Support	71.744	71.782	71.769	71.928	72.047	72.196	72.353	72.470	72.584	72.675	72.813	73.010	73.233	73.476	73.710	73.940
SG-A	LPIPS	0.305	0.333	0.352	0.369	0.382	0.393	0.403	0.411	0.419	0.426	0.432	0.438	0.444	0.449	0.454	0.458
	DreamSim	0.174	0.187	0.200	0.212	0.222	0.235	0.245	0.254	0.263	0.272	0.281	0.289	0.297	0.305	0.311	0.317
	FID	18.626	19.988	21.549	23.320	25.532	27.206	29.541	32.193	34.635	37.279	39.712	42.499	44.961	47.759	50.486	53.215
	Support	72.642	72.530	72.475	72.569	72.613	72.725	72.894	73.030	73.205	73.418	73.661	73.902	74.218	74.463	74.694	74.949
SG-ZA	LPIPS	0.304	0.334	0.354	0.371	0.384	0.397	0.408	0.417	0.426	0.433	0.441	0.447	0.453	0.459	0.463	0.468
	DreamSim	0.174	0.189	0.201	0.214	0.226	0.238	0.248	0.257	0.265	0.274	0.282	0.290	0.297	0.304	0.310	0.316
	FID	18.784	19.222	20.086	21.152	21.977	23.225	24.224	25.336	26.508	27.881	28.967	30.467	32.030	33.195	34.927	36.182
	Support	72.315	72.298	72.293	72.387	72.451	72.567	72.581	72.741	72.844	73.042	73.213	73.417	73.635	73.891	74.178	74.383
DG-A	LPIPS	0.303	0.334	0.354	0.370	0.383	0.396	0.406	0.415	0.423	0.431	0.438	0.444	0.449	0.454	0.458	0.461
	DreamSim	0.185	0.201	0.214	0.226	0.239	0.253	0.264	0.277	0.288	0.298	0.309	0.318	0.326	0.335	0.342	0.348
	FID	23.526	25.649	27.162	29.815	32.026	34.914	37.995	41.781	45.052	48.090	51.505	54.916	57.708	61.274	63.958	66.772
	Support	72.945	72.977	73.042	73.166	73.392	73.649	73.898	74.133	74.446	74.748	75.048	75.289	75.578	75.874	76.112	76.327
DG-ZA (Ours)	LPIPS	0.300	0.335	0.359	0.377	0.393	0.405	0.417	0.428	0.437	0.446	0.453	0.461	0.468	0.474	0.479	0.484
	DreamSim	0.160	0.173	0.185	0.198	0.209	0.220	0.232	0.243	0.255	0.266	0.277	0.289	0.300	0.311	0.321	0.329
	FID	13.930	14.208	14.625	15.895	16.767	18.321	19.695	21.579	23.434	25.138	27.364	29.873	31.993	33.817	35.608	37.344
	Support	71.690	71.412	71.385	71.456	71.611	71.817	72.234	72.569	73.038	73.591	74.160	74.801	75.358	75.953	76.394	76.877

Table 12: OOD-All rollout metrics for all methods after 120k optimization steps on Habitat-Sim. The Original, Fixed Incline, Fixed Decline, and Learned-T baselines receive 20k additional ID training steps, whereas the validity-aware variants receive 20k OOD-aware post-tuning steps from their respective 100k-step checkpoints.

Method	Metric	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Original	LPIPS	0.420	0.443	0.451	0.457	0.461	0.464	0.465	0.468	0.468	0.469	0.471	0.471	0.472	0.473	0.473	0.473
	DreamSim	0.267	0.300	0.312	0.326	0.332	0.338	0.340	0.346	0.344	0.346	0.348	0.347	0.347	0.349	0.347	0.349
	FID	35.017	36.149	37.962	42.033	44.762	46.519	47.782	49.233	48.039	49.150	48.931	49.017	48.629	49.717	48.386	49.360
	Support	74.672	75.133	75.424	75.919	76.162	76.376	76.492	76.705	76.560	76.660	76.731	76.546	76.611	76.680	76.660	76.679
Fixed Incline	LPIPS	0.451	0.471	0.475	0.479	0.479	0.480	0.480	0.482	0.484	0.484	0.482	0.483	0.483	0.483	0.484	0.484
	DreamSim	0.317	0.349	0.358	0.363	0.362	0.367	0.365	0.368	0.372	0.370	0.367	0.369	0.369	0.369	0.369	0.370
	FID	50.195	56.140	58.392	59.814	59.604	61.353	60.693	62.082	63.928	63.517	62.197	63.904	62.969	63.119	62.554	63.003
	Support	78.218	79.222	79.454	79.670	79.521	79.539	79.532	79.582	79.887	79.787	79.531	79.694	79.727	79.659	79.629	79.654
Fixed Decline	LPIPS	0.414	0.444	0.459	0.465	0.471	0.474	0.476	0.478	0.479	0.481	0.480	0.480	0.482	0.480	0.481	0.482
	DreamSim	0.234	0.280	0.308	0.322	0.336	0.343	0.350	0.352	0.356	0.358	0.357	0.357	0.360	0.356	0.358	0.361
	FID	19.264	25.850	33.677	38.309	44.413	47.874	50.740	52.370	54.238	55.864	55.312	56.166	58.172	56.668	56.740	58.100
	Support	73.596	74.478	75.345	75.837	76.398	76.718	77.035	77.188	77.342	77.475	77.447	77.386	77.554	77.415	77.445	77.621
Learned-T	LPIPS	0.418	0.443	0.454	0.459	0.464	0.466	0.469	0.470	0.471	0.471	0.472	0.472	0.472	0.473	0.474	0.473
	DreamSim	0.237	0.278	0.300	0.313	0.323	0.328	0.336	0.337	0.341	0.342	0.342	0.343	0.343	0.346	0.348	0.345
	FID	18.093	23.790	30.487	35.141	40.331	43.758	48.325	50.063	50.789	52.588	52.942	54.484	54.116	56.040	56.254	55.311
	Support	73.187	73.757	74.619	74.984	75.535	75.866	76.279	76.385	76.536	76.521	76.679	76.685	76.659	76.839	76.946	76.847
SG-A	LPIPS	0.445	0.450	0.453	0.456	0.457	0.460	0.462	0.464	0.463	0.464	0.466	0.466	0.467	0.468	0.468	0.469
	DreamSim	0.309	0.314	0.317	0.322	0.324	0.328	0.331	0.334	0.334	0.337	0.337	0.338	0.340	0.342	0.341	0.344
	FID	46.168	43.441	43.991	44.991	45.618	46.253	48.505	48.747	48.947	49.946	50.771	51.581	51.977	51.617	51.232	52.479
	Support	75.548	75.579	75.751	75.963	76.176	76.215	76.469	76.506	76.566	76.718	76.740	76.825	76.879	76.951	76.898	76.973
SG-ZA	LPIPS	0.438	0.445	0.447	0.450	0.450	0.452	0.452	0.454	0.455	0.457	0.458	0.459	0.461	0.462	0.464	0.465
	DreamSim	0.271	0.282	0.285	0.289	0.289	0.291	0.291	0.294	0.296	0.299	0.299	0.303	0.306	0.307	0.311	0.313
	FID	21.608	22.029	22.864	24.176	24.661	25.139	25.228	26.098	26.477	27.595	27.814	28.929	29.560	30.895	30.800	31.707
	Support	74.082	74.104	74.046	74.096	74.050	74.029	74.038	74.071	74.164	74.276	74.388	74.517	74.722	74.796	74.903	74.994
DG-A	LPIPS	0.444	0.451	0.454	0.456	0.457	0.460	0.461	0.462	0.463	0.464	0.464	0.465	0.465	0.466	0.467	0.466
	DreamSim	0.306	0.313	0.317	0.320	0.323	0.327	0.330	0.329	0.330	0.333	0.334	0.333	0.333	0.335	0.336	0.336
	FID	44.314	41.361	40.740	42.515	43.262	43.697	45.612	45.769	45.876	46.329	46.392	46.545	46.366	46.946	47.248	46.595
	Support	75.932	75.850	76.006	76.086	76.297	76.337	76.579	76.525	76.568	76.601	76.640	76.572	76.610	76.686	76.714	76.631
DG-ZA (Ours)	LPIPS	0.440	0.445	0.446	0.447	0.448	0.450	0.449	0.451	0.450	0.451	0.452	0.452	0.453	0.454	0.453	0.454
	DreamSim	0.276	0.281	0.284	0.286	0.287	0.288	0.287	0.290	0.288	0.290	0.290	0.292	0.292	0.293	0.291	0.294
	FID	24.659	23.664	24.284	25.511	25.721	25.743	26.282	27.238	26.096	26.863	27.178	27.590	27.168	27.999	27.160	27.022
	Support	72.987	72.981	72.970	72.948	72.945	72.877	72.836	72.807	72.751	72.807	72.789	72.856	72.800	72.803	72.720	72.823

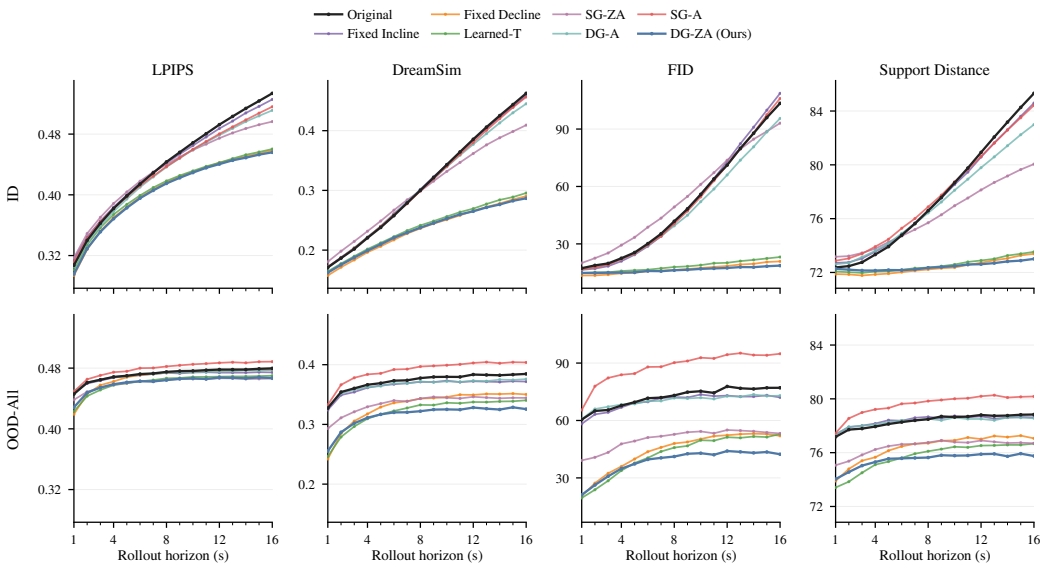

Figure 7: Long-horizon rollout performance after the initial 100k training stage on Habitat-Sim. The top and bottom rows report ID and OOD-All rollouts, respectively, and the columns show LPIPS, DreamSim, FID, and support distance.

Comparing Tables 10 and 12, the additional 20k-step post-tuning stage further improves robustness under unsupported actions. At 16 seconds under OOD-All actions, all four validity-aware variants improve across all four metrics relative to their corresponding 100k checkpoints. For DG-ZA, LPIPS decreases from 0.467 to 0.454, DreamSim from 0.326 to 0.294, FID from 42.373 to 27.022, and support distance from 75.751 to 72.823. The resulting 120k DG-ZA model achieves the strongest 16-second OOD-All performance across all four metrics, reducing them by 4.0%, 15.8%, 45.3%, and 5.0%, respectively, relative to the 120k Original model.

In contrast, the effect of post-tuning on ID rollouts is less consistent across models, as shown in Tables 9 and 11. SG-A, SG-ZA, and DG-A improve all four 16-second ID metrics after post-tuning, whereas DG-ZA trades some ID performance relative to its 100k checkpoint for substantially improved OOD robustness. Nevertheless, the final DG-ZA model still outperforms the 120k Original model under ID actions at 16 seconds, reducing LPIPS, DreamSim, FID, and support distance by 4.3%, 20.9%, 56.9%, and 3.4%, respectively, although Fixed Decline and Learned-T, which are trained exclusively with the ID objective, retain stronger ID performance. Together, the 100k and 120k results show that DG-ZA already provides strong robustness before post-tuning, and the OOD-aware training stage further improves performance under unsupported actions.

We note that these gains should not be interpreted as exploiting conservative collapse to improve visual metrics. Conservative collapse is characterized by apparent recovery after a rollout has entered a severely degraded regime. DG-ZA, by contrast, is designed to limit degeneration before the rollout reaches such a regime, and empirically maintains comparatively stable rollout metrics over the horizon. The improvements in LPIPS and DreamSim measured against the ground-truth futures further suggest that the gains are not simply an artifact of producing visually plausible states through conservative collapse.

E WorldGym and BridgeData V2 Experimental Details

We additionally apply our method in the robotic manipulation domain using WorldGym (Quevedo et al., 2026) and BridgeData V2 (Walke et al., 2023). We evaluate both ID rollouts using the BridgeData V2 test actions and OOD rollouts constructed by perturbing the continuous end-effector actions. This section provides details on the action construction and distribution, model architecture,

Table 13: Mahalanobis distances of the WorldGym evaluation actions from the BridgeData V2 training-action distribution. Distances are computed over the six continuous end-effector dimensions. The final column reports the percentage of actions exceeding the 95th percentile of the training-action distances.

Condition	Mean d_{M}	P95 d_{M}	Outside train 95%
ID	2.239	4.594	5.2%
Gaussian $\sigma = 1.0$	10.235	17.637	95.7%

verifier training, world model training, and quantitative results for the WorldGym experiments in Section 5.3.

E.1 Action Construction and Distribution Analysis

For evaluation, we select trajectories from the BridgeData V2 test set that support a 16-second rollout, corresponding to an 80-step action segment at 5 fps. The OOD evaluation set is constructed from these ID trajectories by perturbing their continuous end-effector actions.

To specify this perturbation, we first describe the action representation used by WorldGym. WorldGym represents each robot command as a 10-dimensional action vector

$$\mathbf{a}_t = \left[\mathbf{a}_t^{\mathrm{trans}}, \mathbf{a}_t^{\mathrm{rot}}, a_t^{\mathrm{grip}}, \mathbf{a}_t^{\mathrm{base}}\right], \tag{18}$$

where $\mathbf{a}_t^{\mathrm{trans}} \in \mathbb{R}^3$ and $\mathbf{a}_t^{\mathrm{rot}} \in \mathbb{R}^3$ denote end-effector translation and rotation, a_t^{grip} denotes the scalar gripper command, and $\mathbf{a}_t^{\mathrm{base}} \in \mathbb{R}^3$ denotes base motion. The base-motion dimensions remain zero for BridgeData V2, yielding an effective 7-dimensional control signal within the 10-dimensional WorldGym action representation.

We construct Gaussian OOD actions by perturbing the six continuous end-effector dimensions. Let $\mathbf{a}_t^{\mathrm{ee}}$ denote the six-dimensional end-effector action composed of translation and rotation:

$$\mathbf{a}_t^{\mathrm{ee}} = \left[\mathbf{a}_t^{\mathrm{trans}}, \mathbf{a}_t^{\mathrm{rot}}\right] \in \mathbb{R}^6. \tag{19}$$

The OOD action is generated as

$$\widetilde{\mathbf{a}}_t^{\mathrm{ee}} = \mathbf{a}_t^{\mathrm{ee}} + \sigma\boldsymbol{\epsilon}_t, \qquad \boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_6), \qquad \sigma = 1.0. \tag{20}$$

The Gaussian noise is sampled independently across timesteps and action dimensions. The gripper command and base-motion dimensions remain unchanged:

$$\widetilde{a}_t^{\mathrm{grip}} = a_t^{\mathrm{grip}}, \qquad \widetilde{\mathbf{a}}_t^{\mathrm{base}} = \mathbf{a}_t^{\mathrm{base}}. \tag{21}$$

The perturbed continuous values are not clipped, allowing the resulting commands to extend beyond the empirical range of the BridgeData V2 training actions.

We compare training actions from the BridgeData V2 train set with ID evaluation actions from the test set and their Gaussian-perturbed counterparts with $\sigma = 1.0$. Figure 8 visualizes the marginal distributions of the six continuous end-effector action dimensions. The distributions of the ID evaluation actions closely follow the corresponding training distributions, whereas the Gaussian-perturbed actions exhibit substantially broader marginals.

To quantify the resulting distribution shift beyond these marginal comparisons, we compute the Mahalanobis distance over the six continuous end-effector dimensions:

$$d_{\mathrm{M}}(\mathbf{a}^{\mathrm{ee}}) = \sqrt{(\mathbf{a}^{\mathrm{ee}} - \boldsymbol{\mu}_{\mathrm{train}})^\top \boldsymbol{\Sigma}_{\mathrm{train}}^{-1} (\mathbf{a}^{\mathrm{ee}} - \boldsymbol{\mu}_{\mathrm{train}})}, \tag{22}$$

where $\boldsymbol{\mu}_{\mathrm{train}}$ and $\boldsymbol{\Sigma}_{\mathrm{train}}$ are estimated from the BridgeData V2 training actions. We use the 95th percentile of the training-action distances, $d_{\mathrm{M}} = 4.567$, as a reference boundary.

Table 13 shows that the ID evaluation actions closely follow the BridgeData V2 training-action distribution. Their 95th-percentile Mahalanobis distance is 4.594, compared with the training reference value of 4.567, and only 5.2% exceed the training 95% boundary. In contrast, the Gaussian-perturbed actions have a mean Mahalanobis distance of 10.235 and a 95th-percentile distance of

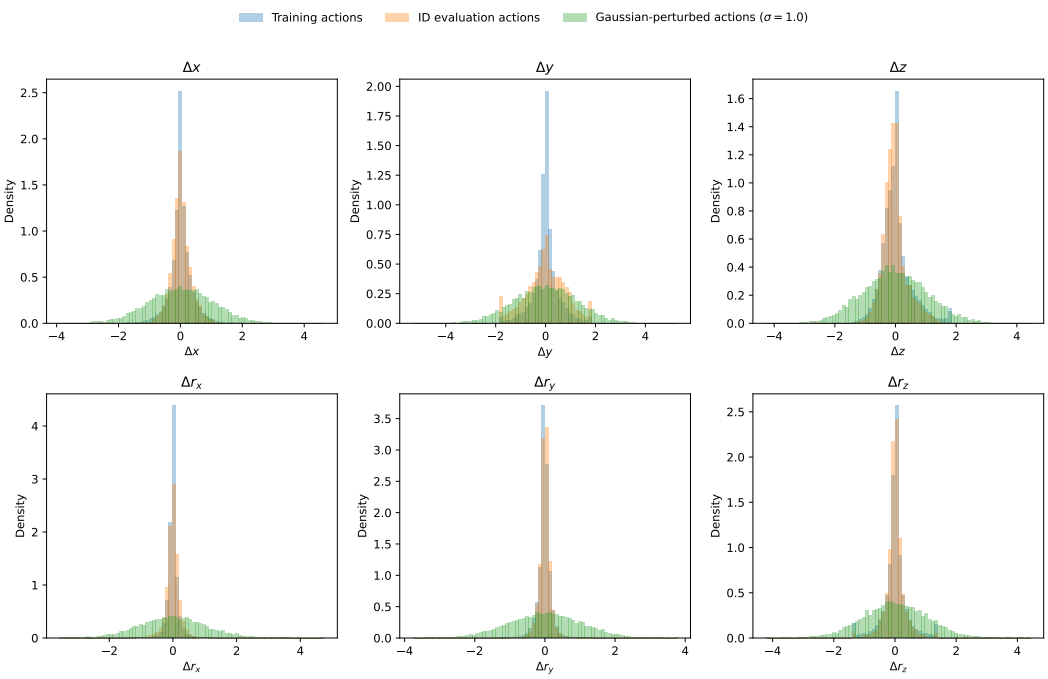

Figure 8: Marginal distributions of the six continuous WorldGym action dimensions on BridgeData V2. The panels correspond to the three end-effector translation dimensions and the three end-effector rotation dimensions. In the axis labels, $(\Delta x, \Delta y, \Delta z)$ denote the components of $\mathbf{a}_t^{\text{trans}}$, while $(\Delta r_x, \Delta r_y, \Delta r_z)$ denote the components of $\mathbf{a}_t^{\text{rot}}$. Each panel compares the training actions, ID evaluation actions, and Gaussian-perturbed actions with $\sigma = 1.0$. The distributions of the ID evaluation actions remain close to the training distributions, while the Gaussian perturbation produces substantially broader action distributions.

17.637, with 95.7% of actions exceeding the reference boundary. Together, these results confirm that the $\sigma = 1.0$ perturbation produces a substantial departure from the empirical BridgeData V2 training-action distribution.

E.2 WorldGym Architecture and Action Conditioning

We use the released 609M-parameter WorldGym model, pretrained on multiple datasets including BridgeData V2. The model operates on 256×256 images and uses a latent diffusion transformer with 16 layers, a hidden dimension of 1024, 16 attention heads, and a patch size of 2. WorldGym autoregressively generates each frame using a sliding context of 20 frames, and we use 10 denoising sampling steps per generated frame during evaluation.

To apply our gating formulation to WorldGym, the 10-dimensional action is linearly projected to the 1024-dimensional model space, and the projected action embedding is modulated by the scalar gate before being combined with the denoising-timestep embedding. The resulting conditioning vector is used by the AdaLN blocks throughout the diffusion transformer. We use the same conditioning bounds as in the NWM experiments, with $\alpha_{\min} = 0.1$ and $\alpha_{\max} = 2.0$.

For DG-ZA, the ID and OOD schedules are represented by two independently parameterized MLPs, each with one 256-dimensional hidden layer and a scalar output, for a total of 525.3K gating parameters. Both schedules are initialized to output $\alpha = 1$, corresponding to the original WorldGym action conditioning at the start of post-tuning. The calibrated state–action validity score interpolates between the two schedules following Equation 3.

E.3 State–Action Verifier Training

We train a separate state–action verifier for WorldGym because its visual representation and action space differ from those of NWM. Given the latent representation of the current observation and the corresponding 10-dimensional action, the verifier predicts a calibrated scalar validity score.

The current observation is encoded by the frozen WorldGym VAE into a $16 \times 32 \times 32$ spatial latent. A 1×1 projection maps each latent location to a 128-dimensional token, to which we add a two-dimensional sinusoidal positional embedding. The 10-dimensional action is encoded into a 128-dimensional query token using a two-layer MLP with a SiLU activation. We then apply four-head cross-attention using the action token as the query and the spatial visual tokens as keys and values. The attended representation is added to the original action query and normalized with LayerNorm. A classifier with one 128-dimensional hidden layer, SiLU activation, and a scalar output then predicts the state–action validity logit. The resulting state–action verifier contains 103K parameters and remains frozen during subsequent world model training.

The verifier dataset is constructed from BridgeData V2 training trajectories and is partitioned into 80% training, 10% validation, and 10% test sets. Each valid transition contributes one positive and one negative example, yielding a balanced binary classification dataset. Positive examples pair an observation with its corresponding recorded action. For negative examples, 80% use Gaussian-perturbed actions, 10% use actions sampled from different trajectories, and 10% use actions sampled from different timesteps of the same trajectory. For the Gaussian negatives, we independently add Gaussian noise with $\sigma = 1.0$ to the six continuous end-effector translation and rotation dimensions. The gripper command and the three zero-valued base-motion dimensions remain unchanged. The cross-trajectory and temporal-mismatch negatives provide harder mismatched pairs in which the action can remain plausible under the marginal action distribution while being less compatible with the current visual state.

We train the verifier for 10 epochs using binary cross-entropy and AdamW with a learning rate of 3×10^{-4}, weight decay of 10^{-4}, a batch size of 256, and gradient clipping at 5. Training uses bfloat16 mixed precision. We sample from the VAE posterior during training and use its posterior mean during validation and test. The checkpoint with the highest validation AUROC is selected, after which a scalar temperature is fitted to the validation logits. The resulting calibrated verifier remains frozen throughout subsequent WorldGym training and rollout inference. The final state–action verifier selected by this procedure achieves an accuracy of 89.25% and an AUROC of 0.938 on the verifier test set.

E.4 World Model Training

We initialize DG-ZA from the released 580k-step WorldGym checkpoint and perform 20k additional steps of OOD-aware post-tuning on BridgeData V2, yielding the 600k-step model used for evaluation. The released checkpoint was pretrained with action dropout $p_{\text{drop}} = 0.1$, providing the null-action behavior required by the frozen reference used in our OOD consistency objective. We therefore omit the separate ID-only pretraining stage used in the NWM experiments and begin OOD-aware post-tuning directly from the released checkpoint.

During post-tuning, the full WorldGym backbone and both gating networks are trainable, while the VAE and pretrained state–action verifier remain frozen. We construct the frozen reference model from the EMA weights of the released 580k-step checkpoint.

Each training step starts from an ID training clip and computes two loss branches, corresponding to the standard ID diffusion objective and the OOD consistency objective in Equation 7. For the ID branch, we use the original BridgeData V2 actions and retain action dropout with $p_{\text{drop}} = 0.1$. The frozen verifier computes validity scores for the corresponding state–action pairs, which are used by the dual gating mechanism.

For the OOD branch, we construct a perturbed counterpart from the same training clip by selecting one target position and adding independent Gaussian noise with $\sigma = 1.0$ to the six continuous end-effector translation and rotation dimensions of its action. The gripper command and the three base-motion dimensions remain unchanged. The frozen verifier evaluates the perturbed state–action pair to obtain its validity score. Using the same visual latents, diffusion noise, noisy target latents, and denoising timesteps, the student is conditioned on the perturbed action, while the frozen reference

Table 14: FID over the 16-second WorldGym rollout on BridgeData V2. OOD actions are generated using Gaussian perturbations with $\sigma = 1.0$.

	ID		OOD ($\sigma = 1.0$)	
Time	Original	DG-ZA	Original	DG-ZA
1s	35.452	34.660	63.945	61.645
2s	41.331	42.226	85.558	85.624
3s	49.792	51.299	102.091	98.176
4s	57.291	57.181	102.398	97.558
5s	66.809	68.594	114.498	105.843
6s	68.004	70.805	121.872	109.045
7s	77.742	78.272	136.486	124.061
8s	76.307	76.643	123.654	115.042
9s	74.691	78.526	121.303	117.278
10s	78.905	78.502	133.991	120.283
11s	82.076	82.344	131.281	122.029
12s	85.378	85.240	139.259	121.620
13s	85.219	86.715	144.005	125.504
14s	86.568	88.630	142.079	123.411
15s	90.229	89.609	141.134	122.836
16s	96.017	90.912	150.144	130.226

model receives null action conditioning. The OOD consistency loss is then applied to the selected target position. Action dropout is disabled for this branch to avoid trivial consistency cases in which the perturbed action is removed and both models receive null action conditioning.

We use 1,000 diffusion timesteps, a per-GPU batch size of 4 on two NVIDIA A100 GPUs, and AdamW with a learning rate of 10^{-5}, weight decay of 0.02, $\beta_1 = 0.9$, $\beta_2 = 0.99$, gradient clipping at 10, bfloat16 mixed precision, and an EMA decay of 0.999. The OOD consistency weight is linearly increased from zero to $\lambda_{\text{OOD}} = 0.1$ over the first 2k post-tuning steps and remains fixed thereafter.

E.5 Quantitative Results

We compare the Original 580k-step WorldGym model with the 600k-step DG-ZA model obtained after 20k steps of OOD-aware post-tuning. Since the Gaussian-perturbed OOD action sequences lack corresponding ground-truth observations, LPIPS and DreamSim are not applicable. We also omit support distance because the released WorldGym checkpoint was pretrained on multiple datasets, and a support bank constructed from BridgeData V2 alone would not represent its full training-state distribution. We therefore report FID as the common metric for both ID and OOD conditions.

Table 14 reports FID at every second of the 16-second rollout for the Original model and DG-ZA. Under ID actions, the two models perform similarly throughout the rollout. Under OOD actions, DG-ZA yields lower FID at nearly all horizons, with the difference becoming more pronounced at longer horizons. At the 16-second horizon, DG-ZA achieves FID scores that are 5.3% and 13.3% lower than those of the Original model under ID and OOD actions, respectively.

F Qualitative Results

F.1 Qualitative Analysis of Action-Conditioned Degeneration

Figures 9 and 10 show representative rollout degenerations of the original NWM under ID actions on RECON and Habitat-Sim, respectively. In both datasets, although the commanded actions remain within the training distribution, the visual quality of the generated observations gradually deteriorates as autoregressive prediction proceeds. These examples qualitatively illustrate the progressive off-support drift observed in Section 3.

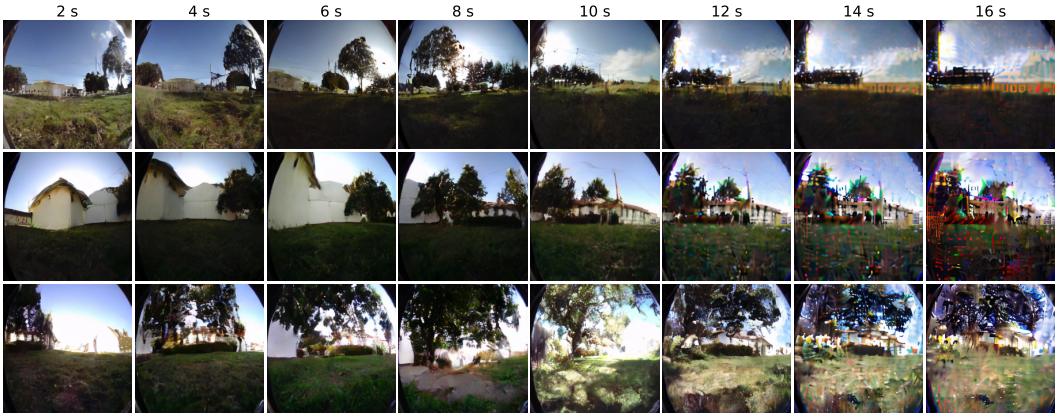

Figure 9: Representative examples of progressive rollout degeneration on RECON under ID actions. Each row shows a different autoregressive rollout over time. The generated observations remain relatively stable at shorter horizons but deteriorate substantially at longer horizons.

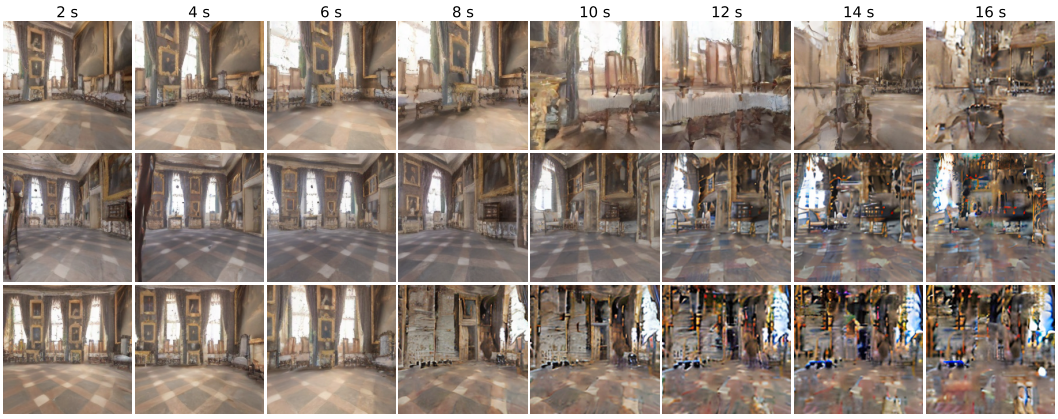

Figure 10: Representative examples of progressive rollout degeneration on Habitat-Sim under ID actions. Each row shows a different autoregressive rollout over time. The visual quality of the generated observations progressively deteriorates as the rollout proceeds.

Figures 11 and 12 show a qualitatively different failure pattern corresponding to conservative collapse on RECON and Habitat-Sim, respectively. In these examples, the generated observations enter severely degraded states but subsequently return to visually plausible states.

We also note that OOD rollouts do not follow a single degradation pattern. Some exhibit progressive degeneration similar to ID rollouts or remain severely degraded from early in the rollout, while others alternate between degraded and visually plausible states multiple times. The former can be regarded as cases without conservative collapse, whereas the latter can be viewed as cases involving multiple conservative collapses within a single rollout. Because a rollout can deteriorate again after an apparent visual recovery, such recovery should not be interpreted as restoration of the underlying action-conditioned dynamics.

F.2 Qualitative Comparison of NWM Gating Methods

Figures 13 and 14 compare the final 120k NWM models under ID actions in Habitat-Sim. We show the ground-truth observations together with rollouts from Original, Fixed Incline, Fixed Decline, Learned-T, SG-A, DG-A, SG-ZA, and DG-ZA.

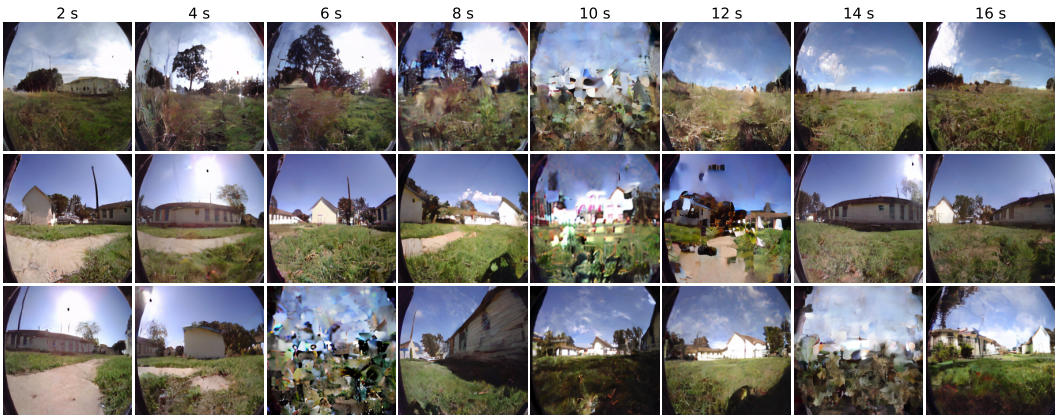

Figure 11: Representative examples of conservative collapse on RECON under Gaussian-perturbed OOD actions with $\sigma = 1.0$. Each row shows a different autoregressive rollout over time. These examples exhibit transitions from visually plausible frames to severely degraded states, followed by an abrupt reappearance of visually plausible frames.

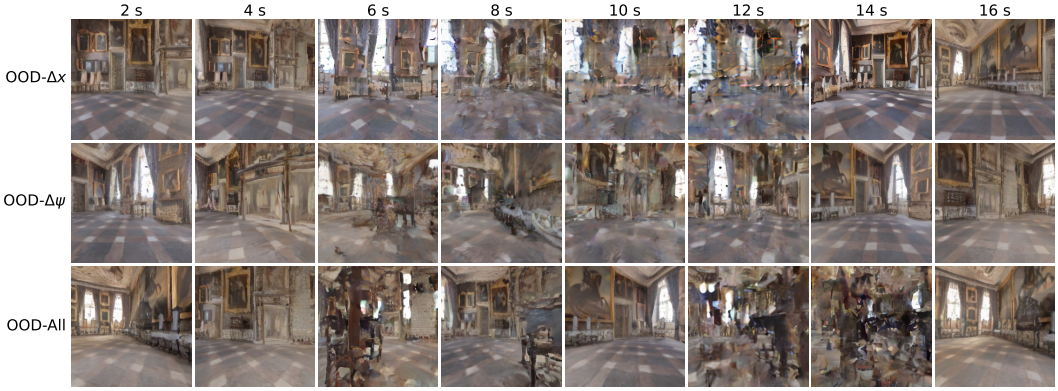

Figure 12: Representative examples of conservative collapse on Habitat-Sim under OOD actions. From top to bottom, the rows correspond to OOD-Δx, OOD-$\Delta\psi$, and OOD-All action conditions. In these examples, severely degraded observations later return to visually plausible states, although this apparent recovery does not imply consistency with the commanded trajectory.

Figures 15 and 16 present the corresponding comparison under OOD-All actions in Habitat-Sim. All models are initialized from the same context and executed under the same OOD-All action sequence. Because these actions lie outside the learned state–action support, the resulting dynamics are not reliably constrained by training, and the predicted rollouts can therefore diverge substantially across methods.

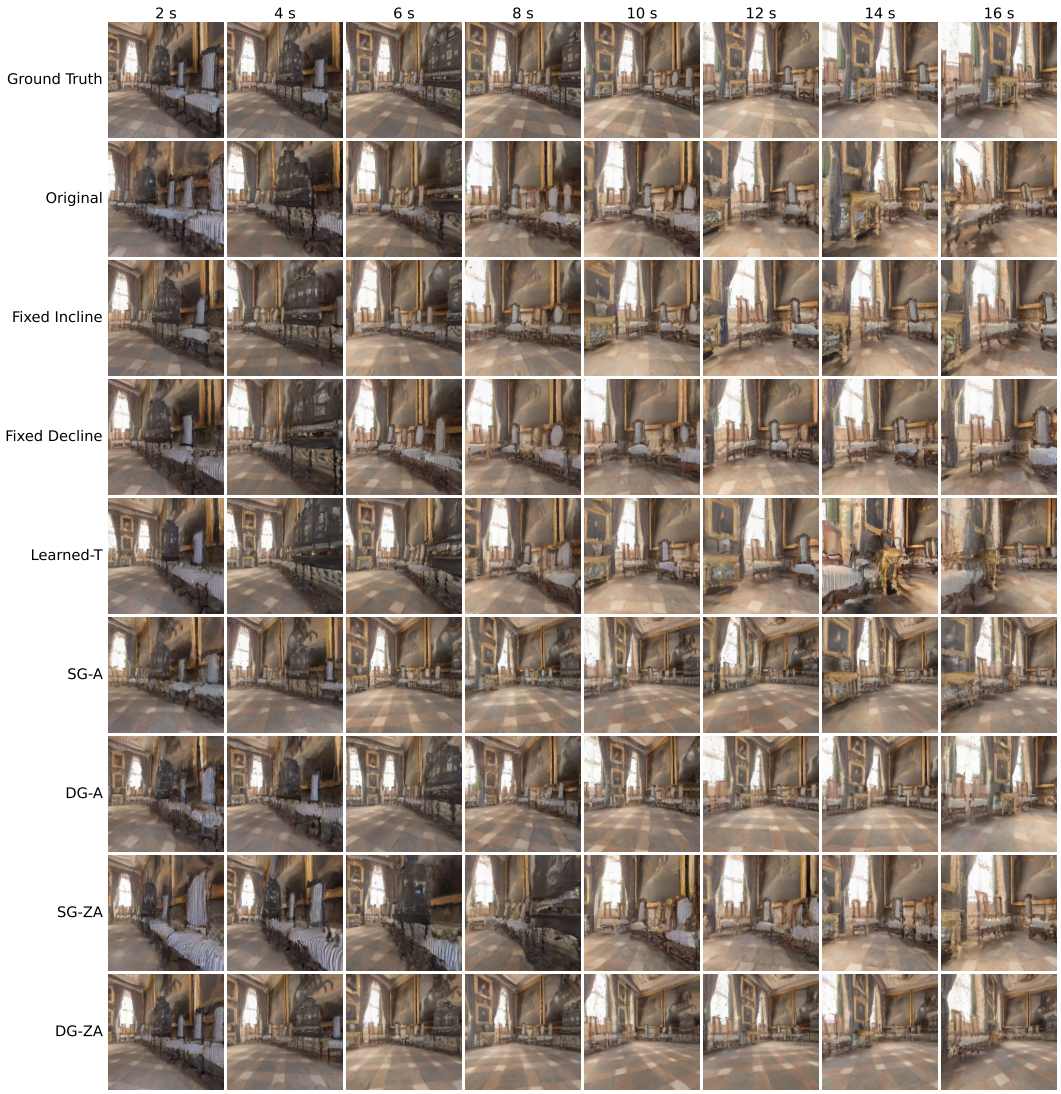

Figure 13: Qualitative comparison of the final 120k NWM models under ID actions in Habitat-Sim. Rows show Ground Truth, Original, Fixed Incline, Fixed Decline, Learned-T, SG-A, DG-A, SG-ZA, and DG-ZA, while columns show rollout times from 2 to 16 seconds at two-second intervals.

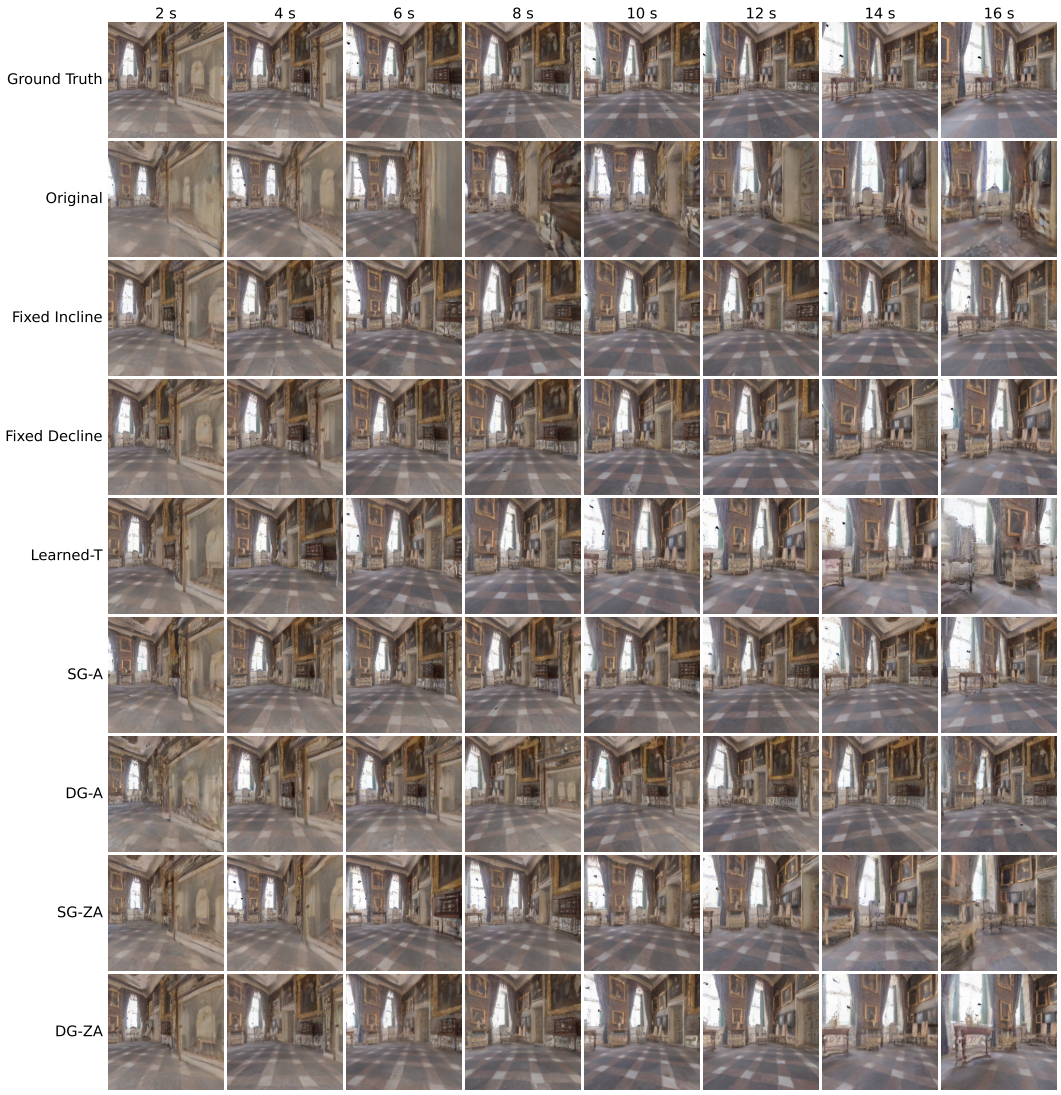

Figure 14: A second qualitative comparison of the final 120k NWM models under ID actions in Habitat-Sim. Rows show Ground Truth, Original, Fixed Incline, Fixed Decline, Learned-T, SG-A, DG-A, SG-ZA, and DG-ZA, while columns show rollout times from 2 to 16 seconds at two-second intervals.

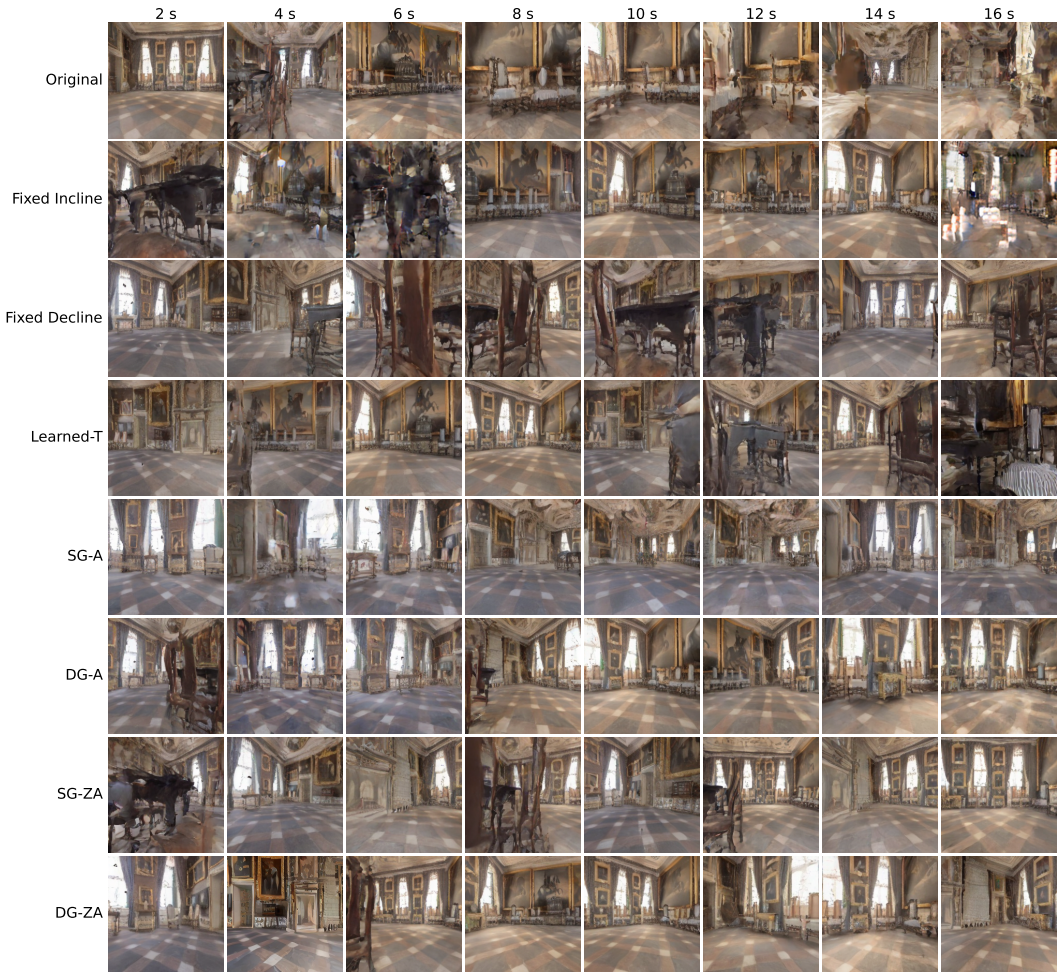

Figure 15: Qualitative comparison of the final 120k NWM models under identical OOD-All action sequences in Habitat-Sim. Rows show Original, Fixed Incline, Fixed Decline, Learned-T, SG-A, DG-A, SG-ZA, and DG-ZA, while columns show rollout times from 2 to 16 seconds at two-second intervals.

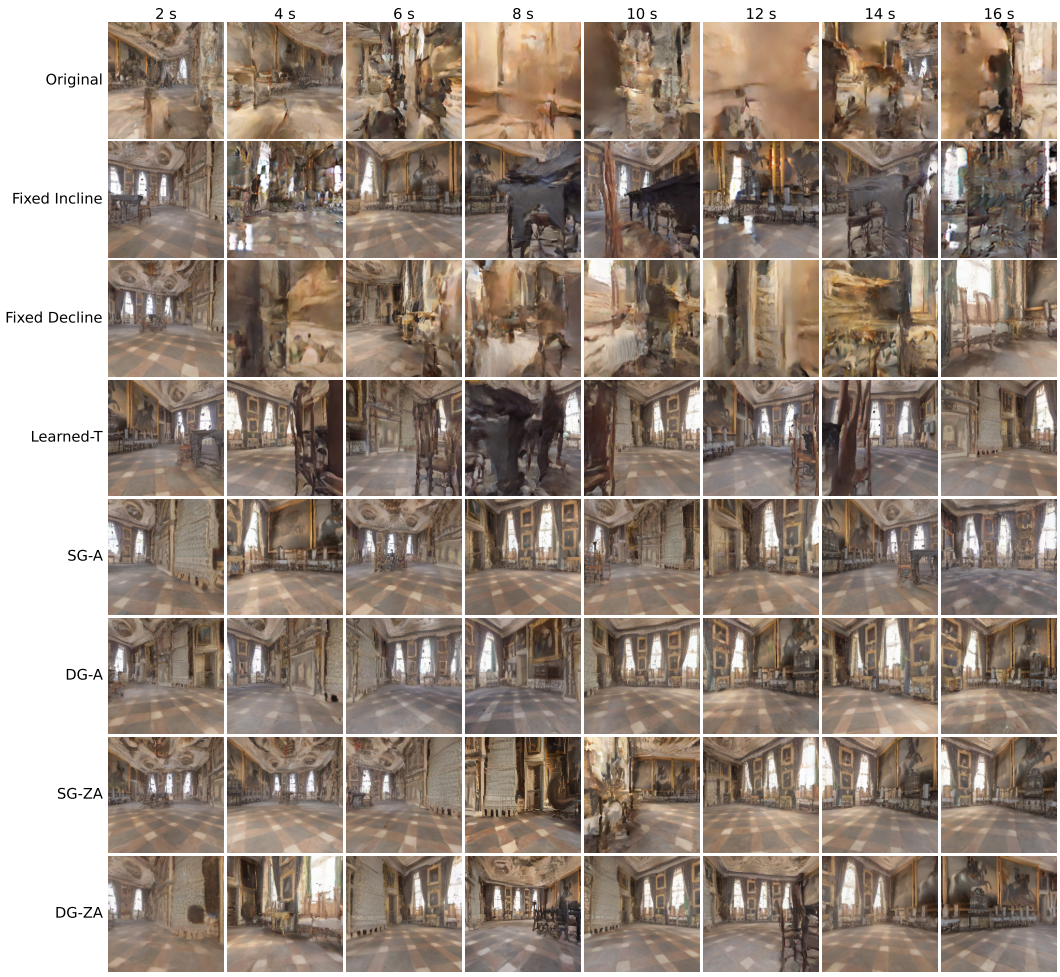

Figure 16: A second qualitative comparison of the final 120k NWM models under identical OOD-All action sequences in Habitat-Sim. Rows show Original, Fixed Incline, Fixed Decline, Learned-T, SG-A, DG-A, SG-ZA, and DG-ZA, while columns show rollout times from 2 to 16 seconds at two-second intervals.



Figure 17: Qualitative comparison of the pretrained WorldGym model and DG-ZA under ID evaluation actions on BridgeData V2. Rows show Ground Truth, Pretrained, and DG-ZA, while columns show rollout times from 2 to 16 seconds at two-second intervals.

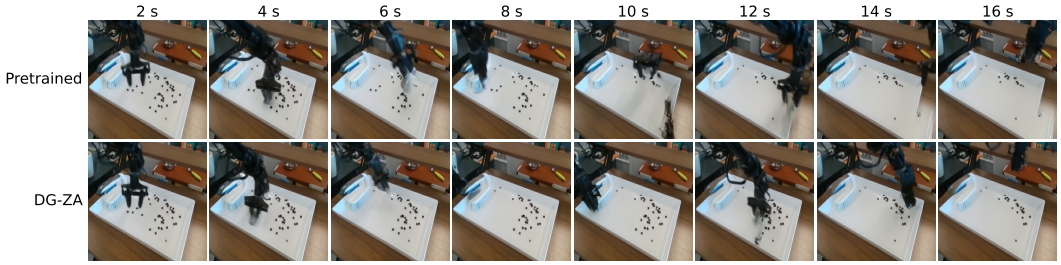

Figure 18: Qualitative comparison of the pretrained WorldGym model and DG-ZA under Gaussian-perturbed OOD actions with $\sigma = 1.0$ on BridgeData V2. Both models use the same initial context and identical perturbed action sequence. Rows show Pretrained and DG-ZA, while columns show rollout times from 2 to 16 seconds at two-second intervals.

F.3 WorldGym Qualitative Comparison

Figure 17 qualitatively compares the released 580k-step pretrained WorldGym model with DG-ZA after 20k steps of OOD-aware post-tuning under ID evaluation actions from the BridgeData V2 test split. Ground-truth observations are shown alongside the two model rollouts.

Figure 18 shows the corresponding comparison under Gaussian-perturbed OOD actions with $\sigma = 1.0$. Since these action sequences are counterfactual, no corresponding ground-truth observations are available. The two models are compared from the same initial context under the same perturbed action sequence.

We note that frame-level degeneration is less pronounced in WorldGym than in the NWM experiments. Unlike egocentric navigation, where camera motion induces large changes across the entire frame, the BridgeData V2 setting uses a fixed external camera, leaving much of the background visually stable over time. Consequently, degeneration in WorldGym tends to be localized around dynamically changing regions, such as the robot arm and manipulated objects, rather than degrading the entire frame. WorldGym also uses a substantially larger backbone and longer visual context than NWM, which may further contribute to this difference.

G LIMITATIONS AND FUTURE WORK

Scope of Unsupported Actions. Our evaluation focuses on a predefined family of OOD actions rather than the full range of possible unsupported actions. OOD-aware post-tuning exposes the model to synthetically sampled unsupported actions from this predefined family to reduce their destabilizing influence. As discussed in Section 4.3, this can be viewed as an action-space analogue of corruption-based training that anticipates imperfect states encountered at inference. Unlike imperfect visual states, which arise from the model's own predictions, unsupported actions are supplied externally, so the anticipated unsupported action family must be specified in advance. Constructing an unsupported-action dataset by exhaustively determining whether every candidate state–action pair lies outside the support of the entire training distribution is impractical at scale. We therefore construct controlled OOD conditions by placing the commanded action outside the marginal training-action range, providing an unambiguous probe of unsupported transitions. This design does not directly evaluate the state-conditional case in which an action is marginally plausible yet incompatible with the current visual state, although such mismatches are included as negatives when training the state–action verifier. We leave evaluation across a broader range of unsupported behaviors for future study.

Train–Test Mismatch of the State–Action Verifier. Our state–action verifier is trained exclusively on ground-truth observations, whereas at inference it receives model-generated contexts, inducing a train–test mismatch. Its validity estimates may therefore become less reliable as rollout degeneration progresses. However, unlike the autoregressive world model, the verifier maintains no recurrent state and estimates validity independently for each target frame, so its estimation errors are not recursively amplified within the verifier itself. Moreover, as long as gating succeeds in limiting degeneration, the verifier operates on contexts that remain close to its clean training distribution. Nevertheless, exposing the verifier to inference-time contexts during training is a natural next step.

Action Controllability under Unsupported Actions. Our method deliberately attenuates action conditioning when state–action validity is low, prioritizing rollout stability over strict action controllability under unsupported commands. Accordingly, the resulting robustness gains should not be interpreted as guaranteeing full controllability under such actions. Our evaluation also primarily measures the quality of generated observations rather than explicit adherence to the commanded action. The world models used in this work predict visual observations rather than explicit agent poses, making direct measurement of action adherence from their outputs nontrivial. Instead, we execute the OOD actions in the Habitat-Sim simulator to obtain ground-truth futures, allowing LPIPS and DreamSim to compare them directly with the action-conditioned model predictions. Although they are not direct measures of action controllability, the consistent improvements of DG-ZA on these pairwise metrics provide indirect evidence that our method better follows the intended dynamics under unsupported actions.

Scale, Domain Coverage, and Downstream Utility. Our experiments cover a relatively limited range of model scales and domains. This scope partly reflects computational constraints. As degeneration is less visually pronounced in our larger-model setting, extending the controlled analysis to larger models may require substantially longer rollouts. Such evaluations require longer-horizon data, which are more difficult to obtain, while simultaneously incurring higher computational cost. We therefore prioritize a compact setting in which degeneration can be clearly observed and systematically analyzed, allowing us to evaluate multiple gating baselines and factorial ablations within a feasible budget. We view this work as an initial step toward improving robustness to unsupported actions rather than a definitive solution to the problem. Extending the analysis to larger models and additional domains, and evaluating whether improved rollouts translate into better downstream planning or policy evaluation, remain important directions for future work.