ANYGOAL: ON WORLD MODEL GENERALIZATION BEYOND EXPERT ROLLOUTS

Anonymous authors
Paper under double-blind review

ABSTRACT

The goal of a world model is to capture the dynamics of an environment. This has the great appeal that it allows agents to plan and simulate a course of actions with the help of a world model. For these uses to be reliable, however, the learned dynamics must behave well beyond the specific state–action combinations observed in expert rollouts. In this work, we show that current world models broadly struggle to generalize to new goals in known environments. We explain this from the theoretical perspective of compositional generalization, which allows us to formulate desiderata and conditions of successful generalization. Based on these insights, we construct a new benchmark, AnyGoal. We carefully check that each tasks remain within the factor-wise support of the training data, so that they can be reasonably expected to be solvable by a good world model. By disentangling encoder–predictor failures, generative versus latent planning models, interaction-free versus interaction-rich regimes, and other axes of variation, our results reveal both where current world models can generalize and where major challenges remain for goal-shifted dynamics prediction and planning. This framework and benchmark will provide direction and a progress measure for the goal of building world models that generalize to new planning tasks.

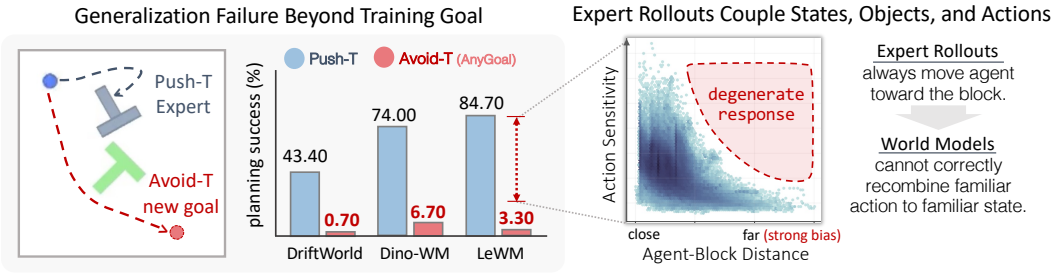

Figure 1: **Example of generalization failure under goal shift.** A world model trained on Push-T performs poorly on Avoid-T, despite Avoid-T requiring only familiar 2D navigation and having individually familiar states and actions. In Push-T, action interventions reveal reduced model sensitivity in biased regions, suggesting that the model learns shortcuts rather than reusable dynamics.

1 INTRODUCTION

A key goal of physical intelligence is the ability to plan and perform new tasks. Routine tasks, such as driving to work, can be automatized and executed without much cognitive effort. Yet novel tasks requires us to plan and simulate the consequences of our actions. World models (Ha & Schmidhuber, 2018; Maes et al., 2026b; Ivashkov et al., 2026; Lu et al., 2026; Zhu et al., 2024; Zhou et al., 2025) aim to learn action-conditioned dynamics of environment by predicting how the world evolves under actions rather than task-specific behavior (Sutton & Barto, 2018). Therefore, a good world model should not merely memorize behavioral biases present in the training distribution. However, in-distribution evaluation offers limited evidence of reusable dynamics. Figure 1 illustrates a simple but strong example of this caveat. In Push-T, the agent must push a block to a target pose, so expert rollouts strongly couple far-from-block states with actions that move the agent toward the block. A model trained on Push-T may therefore learn this rollout regularity rather than the action-conditioned

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.

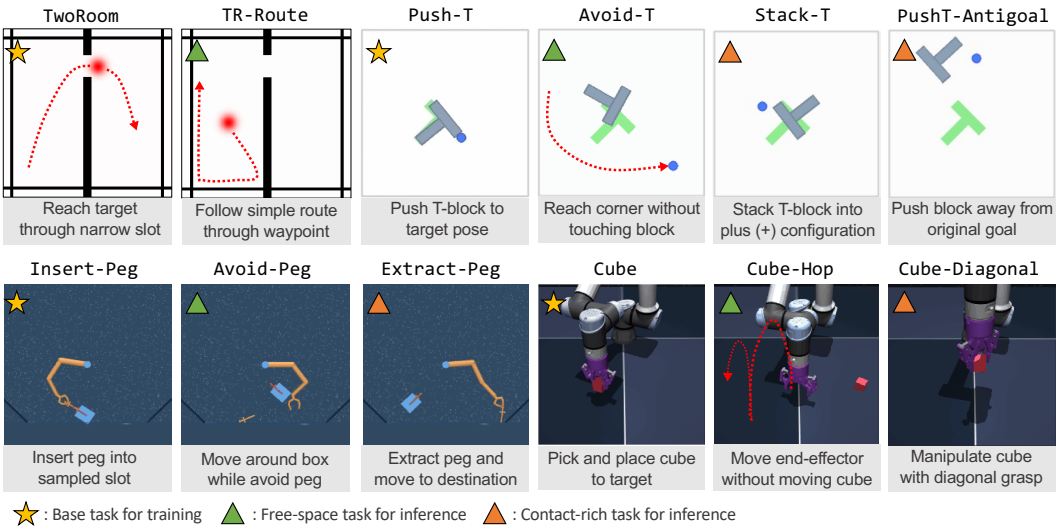

Figure 2: **Overview of AnyGoal.** Each tasks preserves the environment dynamics of the corresponding base tasks and keep states and actions individually within training coverage. Throughout the paper, AnyGoal is used solely as a test set for evaluating generalization.

effect of the supplied action, as reflected by the action-sensitivity plot in Fig. 1.When evaluated on Avoid-T, which requires avoiding the block using familiar 2D navigation, the model fails.

If the training data fully covered the relevant joint state–action space, then a model could in principle predict correctly on that support. However, such coverage is unrealistic in many control problems, especially when the relevant space is high-dimensional. First, offline datasets for world-model training are often collected from goal-directed policies that reach task-relevant states (Burda et al., 2019), which can introduce strong state–action correlations. One possible remedy is to mix data from multiple policies (Zhao et al., 2026). However, even with such mixtures, it can be difficult to determine whether the resulting dataset sufficiently covers the relevant state–action space, especially when the true reachable set is unknown. Even when a coverage gap is identified, collecting targeted rollouts to fill it can be difficult in offline settings. This makes it necessary to study generalization under incomplete coverage, rather than assuming that data scale alone will resolve the issue.

Generalization of world models has been studied to some extent. Closest to our setting, Hansen & Wang (2026) study hallucination detection and mitigation in world models, while we evaluate generalization in a statistically controlled space with explicit state-action support checks. Other OOD evaluations introduce states or factors outside the training support (Maes et al., 2026a; Wang et al., 2026; Li et al., 2025). Such settings are valuable for testing robustness to novel factors or extrapolation, but they may go beyond the generalization expected from a reusable dynamics model.

What remains missing is a systematic evaluation of the generalization that a good world model should support. First, we formalize this criterion as compositional generalization (Wiedemer et al., 2024b) and link it to goal shifts and behavior bias in the training data. We then propose AnyGoal (Fig. 2), a benchmark of goal-shifted task variants built from widely used world-model environments. To remain in the compositional-generalization regime, AnyGoal preserves the observed marginal support of states, actions, task-relevant factors, and physical rules, while changing the joint combinations induced by the training behavior distribution and targeting under-covered state–action regions. With AnyGoal, we test whether current world models can generalize to new tasks without requiring extrapolation to unseen states, actions, or visual factors. Across this controlled setting, we observe broad failures across modeling paradigms, including generative world models (Zhu et al., 2024; Lu et al., 2026) and JEPA-style architectures (Maes et al., 2026b; Ivashkov et al., 2026; Zhou et al., 2025). Our results then characterize this degradation across several axes, showing that failures can arise from both dynamics prediction and representation learning, and that their severity depends on task complexity, evaluation metric, and model family. Finally, we outline a theoretical direction toward compositional generalization for world models, framing what future models may need in order to move beyond behavior-biased rollouts.

2 Related Works

Compositional generalization diagnostics and benchmarks. Compositional generalization tests whether a model can recombine familiar components into unseen configurations, a key challenge when possible compositions exceed finite data coverage. Early sequence-to-sequence benchmarks such as SCAN (Lake & Baroni, 2018) and CFQ (Keysers et al., 2020) exposed this failure mode through controlled compositional splits. The same diagnostic was subsequently adopted in vision and multimodal settings, with benchmarks such as CLEVR (Johnson et al., 2017), Winoground (Thrush et al., 2022), and CVR (Zerroug et al., 2022) testing whether familiar objects, attributes, and relations can be composed correctly. These failures remain prominent in modern large-scale models: ConceptMix (Wu et al., 2024) shows that text-to-image models degrade as prompts require more familiar concepts to be composed jointly, while recent vision–language work continues to reveal failures in object–attribute and relational composition (Assouel et al., 2026). At the same time, compositional evaluation is becoming broader and more systematic, with scalable protocols for probing generalization across controlled compositions (Camposampiero et al., 2026), and similar shifts being used as diagnostics in mechanistic interpretability (Pacela et al., 2026). AnyGoal brings this perspective to world models, where goal-directed trajectories naturally induce restricted joint coverage over otherwise familiar physical states and actions.

World-model generalization and support mismatch. Recent world-model work has studied generalization through object-centric compositional dynamics (Zhao et al., 2022), composable causal components for unseen environments (Wang & Huang, 2025), multi-task world models (Georgiev et al., 2025), environment-level evaluation queries (Warrier et al., 2026), action-conditioned planning (Zhang & Du, 2026), and coverage-related hallucination failures (Hansen & Wang, 2026). However, less is known about whether expert-trained world models generalize to new goal-induced recombinations of familiar states and actions. This question is closely connected to support mismatch in offline reinforcement learning, where fixed behavior datasets can lead methods to fail when evaluating poorly covered state–action regions (Fujimoto et al., 2019; Kumar et al., 2019; Levine et al., 2020). Offline RL addresses support mismatch by constraining or regularizing policies toward covered regions (Gao et al., 2025b;a; Laroche et al., 2019; Chen et al., 2025; Kumar et al., 2020; Tang et al., 2025). Model-based offline RL extends this issue to learned dynamics, motivating uncertainty- or pessimism-based objectives for poorly covered regions (Yu et al., 2020; Kidambi et al., 2020; Yu et al., 2021; Rigter et al., 2022). D4RL further shows that dataset composition strongly affects offline policy improvement (Fu et al., 2021). AnyGoal is complementary to these approaches: it evaluates whether the world model learns reusable dynamics under goal shifts.

3 Compositional Generalization for World Models

When should we say that a world model has learned reusable dynamics, rather than only the training behavior? In this section, we formalize this criterion as compositional generalization.

Notation. We use uppercase letters for random variables and lowercase letters for their realizations. Let $z_t = (z_{1,t}, \ldots, z_{K,t})$ denote true underlying state of the environment, where each component corresponds to a semantic factor of variation[1] (Schölkopf et al., 2021), such as object position or end-effector pose. We model observations as $s_t = f(z_t)$, where f maps z_t to the RGB observation s_t presented to the world model. An action-paired trajectory $\tau = (s_0, a_0, s_1, a_1, \ldots, s_H)$ can therefore equivalently be represented through the corresponding sequence of physical factors and actions, $(z_0, a_0, z_1, a_1, \ldots, z_H)$, where H denotes horizon.

3.1 Action-Conditioned Prediction Can Degenerate into Behavior Prediction

An action-conditioned world model should learn a transition predictor $\hat{T}$ that receives s_t[2] and an action a_t, and predicts how the observation changes under the supplied action, $\hat{T}(s_t, a_t) \approx s_{t+1}$. We consider the common offline setting in which data are collected by a behavior policy π, so that

[1] We include only factors that vary over the training distribution. For example, if a block remains gray throughout training, its color is not included in z_t.

[2] For simplicity, s_t may represent a finite history of states or observations.

the observed action satisfies $a_t \sim \pi(\cdot \mid s_t)$. Thus, the observed transitions reflect the state–action pairings induced by π, rather than arbitrary interventions at each state:

$$T^{\pi}(s_{t+1} \mid s_t) = \int T(s_{t+1} \mid s_t, a)\, \pi(a \mid s_t)\, da. \tag{1}$$

A predictor trained only on such data may degenerate into an action-ignorant model for state regions with restricted conditional action support, fitting the observed transitions by approximating

$$\hat{T}(s_{t+1} \mid s_t, a_t) \approx T^{\pi}(s_{t+1} \mid s_t), \tag{2}$$

where the dependence on the supplied action a_t is effectively lost. We empirically show that the bias is empirically realized through targeted action interventions on familiar but under-covered state–action combinations, eliciting degenerate model responses in Appendix E.

3.2 Compositional Distribution Shifts in World Models

Now, let g_{tr} and g_{te} denote the training and test goals. Each goal induces a behavior policy, π_{tr} or π_{te}, which generates actions along the same environment dynamics T. We denote the resulting trajectory distributions by p_{tr} and p_{te}, respectively. We define the supports of physical factors and actions observed across trajectories, writing the training joint state–action support as

$$\mathcal{C}^{\text{tr}} := \bigcup_{t=0}^{H-1} \operatorname{supp}_{p_{\text{tr}}}(Z_{1,t}, \ldots, Z_{K,t}, A_t). \tag{3}$$

We also define the marginal support of each factor as $\mathcal{Z}_k^{\text{tr}} := \bigcup_{t=0}^{H-1} \operatorname{supp}_{p_{\text{tr}}}(Z_{k,t})$, and the action support as $\mathcal{A}^{\text{tr}} := \bigcup_{t=0}^{H-1} \operatorname{supp}_{p_{\text{tr}}}(A_t)$. These supports pool over temporal position: a physical factor is considered observed during training if it occurs at any point along a trajectory drawn from p_{tr}. The test supports $\mathcal{C}^{\text{te}}$, $\mathcal{Z}_k^{\text{te}}$, and $\mathcal{A}^{\text{te}}$ are defined analogously under p_{te}. We define the *combinatorial space* as the Cartesian product of the marginal training supports:

$$\mathcal{C}^{\times} := \left(\prod_{k=1}^{K} \mathcal{Z}_k^{\text{tr}} \right) \times \mathcal{A}^{\text{tr}}. \tag{4}$$

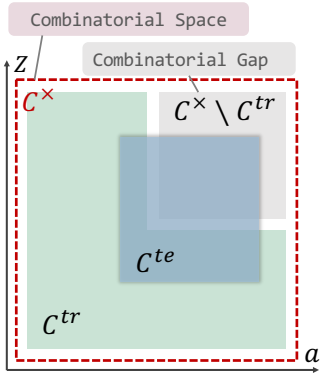

Figure 3: Compositional support structure. $\mathcal{C}^{\text{te}} \subseteq \mathcal{C}^{\times}$ and $\mathcal{C}^{te} \not\subset \mathcal{C}^{\text{tr}}$is necessary.

We study compositional distribution shifts (Liu et al., 2023; Mahajan et al., 2025) where the underlying transition dynamics are invariant across the train and test distribution, i.e., $T_{\text{tr}}(s_{t+1} \mid s_t, a_t) = T_{\text{te}}(s_{t+1} \mid s_t, a_t)$, but the test support introduces state, action pairs not present in the training support ($\mathcal{C}^{\text{te}} \not\subset \mathcal{C}^{\text{tr}}$). AnyGoal targets the *combinatorial gap*, regions of the combinatorial space $\mathcal{C}^{\times}$ that are absent by the training joint support $\mathcal{C}^{\text{tr}}$. Since test trajectories need not cover the entire gap or lie only within it, we require $\mathcal{C}^{\text{te}} \subseteq \mathcal{C}^{\times}$ and choose goals with substantial mass in under-covered regions of $\mathcal{C}^{\times}$. This means that the test distribution does not introduces novel physical factors or actions, only novel combinations of observed physical factors and actions. Importantly, training-distribution accuracy does not identify the extension of the learned transition function to $\mathcal{C}^{\times} \setminus \mathcal{C}^{\text{tr}}$: predictors that agree on the training support may disagree on novel compositions (Wiedemer et al., 2024b). This motivates evaluating the combinatorial gap directly. We discuss more about why it is an essential test in Appendix A.

4 AnyGoal: Controlled Goal-Shift Evaluation

We introduce AnyGoal, a benchmark for testing whether world models generalize beyond the goal-directed trajectories on which they are trained. AnyGoal uses the controlled goal shifts discussed in Sec. 3.2 and builds on four widely used world-model tasks: TwoRoom (Sobal et al., 2025), Push-T (Chi et al., 2025), DMC Manipulator Insert-Peg (Tassa et al., 2018), and OGBench Cube (Park et al., 2025). AnyGoal contains eight variants: one free-space variant per base task and one or two contact-rich variants except TwoRoom. We release 400 episodes per variant, with data-collection tools for generating additional datasets. Figure 2 summarizes the tasks, with details in Appendix B.

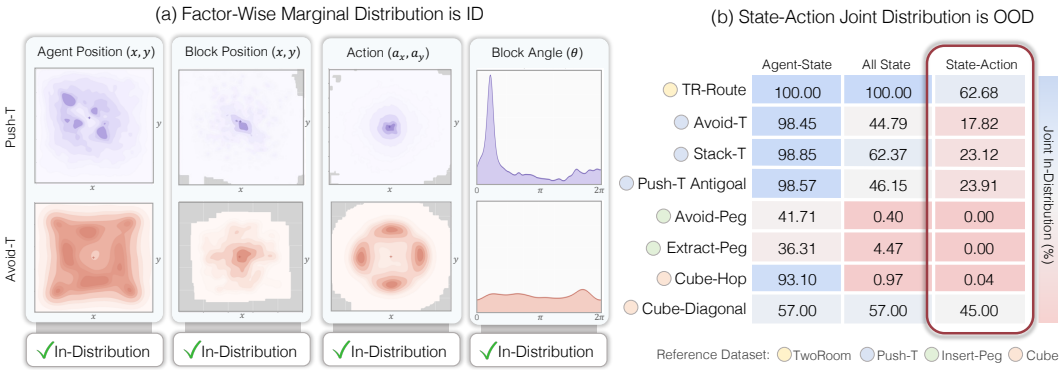

Figure 4: **AnyGoal targets the combinatorial gap.** (a) Example factor-wise support check for Avoid-T relative to Push-T using KDE. Darker regions indicate higher density, and gray regions mark areas whose density is below 1% of the peak reference density. (b) Estimated overlap between each target variant and the base-task training distribution via KDE.

4.1 AnyGoal Targets the Combinatorial Gap

Validating $\mathcal{C}^{\text{te}} \subseteq \mathcal{C}^{\times}$. For each base–target task pair, we apply three factor-wise support checks to verify that the target distribution remains marginally in-distribution relative to its reference distribution. The range check requires all query samples to lie within the coordinate-wise reference range. The Mahalanobis check uses the reference mean and covariance, thresholds distances at the reference 0.99-quantile, and requires at least 95% of query samples to fall below it. The KDE check thresholds reference densities at the 0.01-quantile and requires at least 95% of query samples to lie above it. The final support verdict is the conjunction of all three checks. All AnyGoal variants are designed to pass these factor-wise checks, which indicates that all tasks are in combinatorial space of training data. Appendix C provides the full details of pass-criterion definitions, the verified factor groups for each base task, and the complete score of range, Mahalanobis, and KDE results.

Measuring combinatorial-gap coverage of $\mathcal{C}^{\text{te}}$. We then estimate how much each target distribution overlaps with the base-task joint support. Unlike the factor-wise checks above, this joint test is meant to diagnose the combinatorial gap: individual factors may be marginally familiar while their full state–action combination is rare under the training distribution. For each base–target pair, we concatenate the action, agent state, and task-object or target state, fit a full-dimensional Gaussian KDE to the reference vectors, and set the density threshold to the empirical 0.01-quantile of log-densities. We then report the fraction of target vectors whose log-density exceeds this threshold. Lower joint scores indicate less overlap with the base-task joint distribution and greater coverage of the combinatorial gap. Full vector definitions and KDE implementation details are in Appendix D.

4.2 Relative Evaluation Against Base Tasks

We evaluate rollout prediction error and downstream MPC planning. Models are trained only on the base dataset and evaluated on both base and goal-shifted trajectories to measure relative changes. Rollout-error changes more directly diagnose whether action-conditioned dynamics generalize to unseen state–action combinations. However, planning success remains important. For JEPA-style models, it provides complementary evidence that the representation retains task-relevant information. For generative models, low rollout error can miss small but task-critical observation changes. We therefore report both metrics over seeds $0, 1, 2$. Evaluation details are in Appendix H

Prediction Error. We measure MSE in each model's native prediction space, which may be a learned latent space, oracle state space, or pixel space depending on the model class. We evaluate autoregressive prediction over horizons of 5 and 25 environment steps, with frame skip of 5. For each seed, we sample 100 episodes and average over all valid starting points within each episode.

MPC Planning Success Rate. We evaluate downstream control using the same CEM-based MPC implementation from Stable-WorldModel Maes et al. (2026a) for all models. We use identical planning configurations, with a planning horizon of 25 steps and 5 actions executed before replanning.

Table 1: End-to-End World models and oracle state predictors evaluated in our experiments.

Model	Family	Representation	Action conditioning
End-to-End World Models			
LeWorldModel	JEPA	Learned latent	AdaLN
Sensorimotor WM	JEPA	Learned latent	AdaLN
DINO-WM	JEPA	Frozen DINO features	Feature concatenation
DriftWorld	Generative	Pixels	Conditional generator
IRASim	Generative	Frozen SDXL VAE	DiT conditioning
Predictor-Only Models			
AdaLN	Transformer	Oracle state	AdaLN
Action-Concatenated	Transformer	Oracle state	Feature concatenation
Toy-DriftWorld	Generative	Oracle state	Conditional generator

5 Experiment and Results

We evaluate a diverse model suite spanning JEPA-style and generative objectives, learned and frozen visual encoders, and action-conditioning mechanisms (Tab. 1). To separate dynamics learning from visual representation failure, we also include oracle-state predictors which replace visual encoders with oracle states. For consistency, all models use frame skip 5 and one-step prediction. Model details and oracle-state variables for each environment are provided in Appendix F, and the full numerical results are summarized in Appendix I.1.

End-to-End World Models. LeWorldModel (LeWM) (Maes et al., 2026b) is an end-to-end JEPA that jointly learns a visual encoder and an action-conditioned transformer predictor from pixels. It uses SIGReg (Balestriero & LeCun, 2025) for anti-collapse regularization and conditions the predictor through AdaLN (Peebles & Xie, 2023). Sensorimotor World Model (SMWM) (Ivashkov et al., 2026) uses the same encoder–predictor family as LeWM with an inverse-dynamics objective that encourages representations to retain controllable changes. DINO-WM (Zhou et al., 2025) decouples perception from dynamics by using a frozen DINOv2 (Oquab et al., 2024) patch encoder and predicting future patch representations, with actions concatenated to the patch tokens. DriftWorld (Lu et al., 2026) is a generative model that uses an action-conditioned spatial–temporal U-Net to predict future observations directly from observation history, actions, and noise. IRASim (Zhu et al., 2024) is a latent generative model that encodes observations with a frozen pretrained VAE and uses a Diffusion Transformer (Peebles & Xie, 2023) to predict future latents conditioned on action trajectories.

Oracle-State Predictor. AdaLN predictor follows the architecture used by LeWM and SMWM. Action-Concatenated predictor follows the DINO-WM predictor, where an embedded action is concatenated to the patch representation before being processed by a causal transformer. Toy-DriftWorld is adapted from DriftWorld's action-conditioned generative backbone. We use a reduced DriftWorld-style model and replace pixel inputs and outputs with true state inputs and next-state targets.

5.1 Dimensions of Goal-Shift Performance

Goal-shift performance cannot be summarized by a single number. The same degradation can reflect different sources depending on whether it appears in prediction or planning, in latent or pixel space, in free-space or contact-rich tasks, and with or without oracle state access. We therefore organize the results around five recurring dimensions for interpreting the results:

1. Failure source: Predictor failures can be isolated from encoder failures via oracle-state predictors.

2. Evaluation metric: Rollout prediction and planning are showing different aspects.

3. Task complexity: Free-space and contact-rich tasks impose different dynamics challenges.

4. Representation quality: Learned representations sometimes obscure task-relevant dynamics, although they sometimes can improve planning geometry.

5. Prediction space: JEPA-style and generative models show distinct trends, but comparisons should be interpreted cautiously because prediction errors are measured in different spaces.

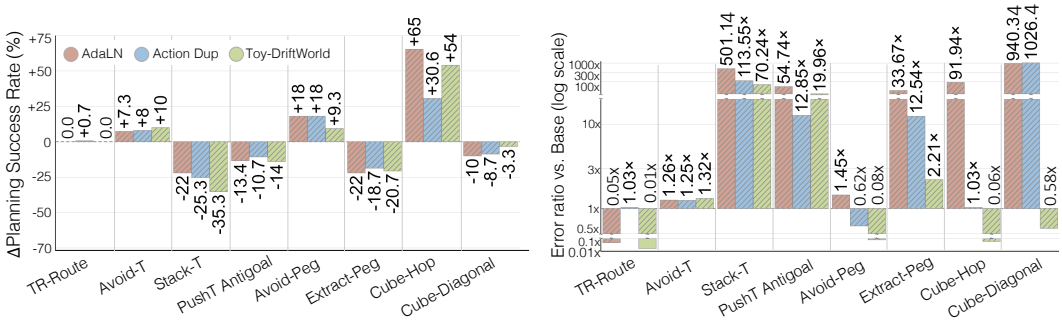

Figure 5: **Oracle-state predictors reveal dynamics-level failures.** Even with task-relevant state factors provided directly, goal shifts still degrade prediction and planning, showing that the AnyGoal gap cannot be reduced to visual representation failure. See Appendix I.1for absolute metrics.

5.2 Dynamics Generalization Beyond Representation Failure

To examine dynamics generalization, we use the oracle states provided by AnyGoal, which expose task-relevant factors for action-conditioned next-step prediction and remove the need to infer them from observations under goal shifts. Overall, oracle-state predictors show that the AnyGoal gap cannot be reduced to visual representation failure.

Oracle-state predictors reveal an interaction-dependent dynamics gap. Figure 5 shows that removing visual abstraction does not yield a uniform outcome across AnyGoal variants. In free-space variants such as TR-Route, Avoid-T, Avoid-Peg, and Cube-Hop, MPC performance is preserved or improves relative to the corresponding base task. In contrast, contact-rich variants consistently degrade, despite using the same privileged state inputs. This suggests that the difficulty is not merely access to the relevant state factors (Klindt et al., 2026; Shi et al., 2026), but whether the goal shift requires precise object interaction. This asymmetry is consistent with prior work showing that compositional generalization requires constraints on the learned function. Additive or interaction-free settings are especially favorable, since factor-specific mechanisms can be learned from observed compositions and reused under novel recombinations (Dong & Ma, 2022; Lachapelle et al., 2023; Wiedemer et al., 2024a; Mahajan et al., 2025). This intuition aligns with the independent causal mechanisms principle, under which reusable mechanisms remain autonomous as their joint occurrence changes (Parascandolo et al., 2018). Contact-rich variants depart from this favorable regime: action effects may no longer decompose into reusable factor-specific mechanisms. Compositional generalization may still be possible, but only with additional structure on these interactions (Brady et al., 2025). The same asymmetry appears in rollout prediction. Contact-rich variants show much larger error increases, often by one to three orders of magnitude, even with oracle states. Free-space variants degrade far less, and in some cases match or improve over the base-task error. Thus, removing representation learning does not make goal-shifted dynamics prediction uniformly reliable.

Metric alignment depends on task complexity. We additionally observe that rollout error and planning performance sometimes diverges in free-space variants where planning can tolerate imperfect prediction. The closer agreement in contact-rich settings may reflect the structure of the planning problem. Rollout error measures whether the model predicts the full trajectory accurately, whereas MPC only needs to find some action sequence that reaches the goal, regardless of the exact intermediate path. Free-space tasks often admit many such trajectories, while contact-rich tasks allow fewer viable sequences because success also depends on the correct object interaction.

5.3 Representation Effects in End-to-End World Models

End-to-end world models can degrade in both metric under the goal shift even when oracle-state predictors remain stable. Such cases show that the AnyGoal gap is not solely a failure of action-conditioned dynamics, but can also reflect encoder failure under unseen factor combinations. Due to space constraints, we discuss two main observations, and more are in Appendix I.3.

Factor recombination can induce observation-level shifts. AnyGoal controls generalization at the level of physical factors, but end-to-end models observe rendered inputs $s = f(z)$. Recombining

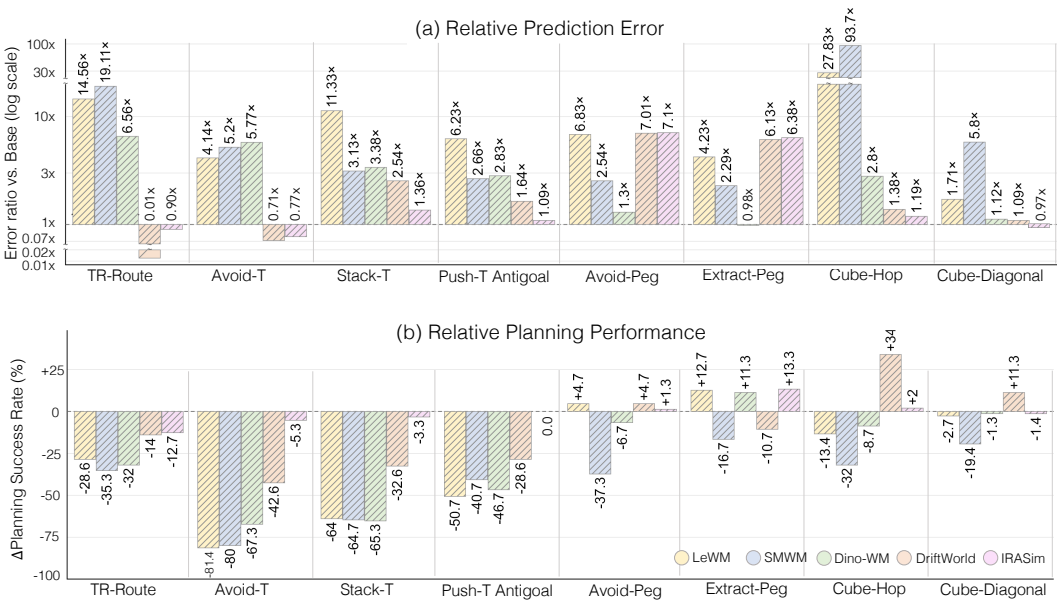

Figure 6: **End-to-end degradation reflects representation effects.** Compared with oracle-state predictors, end-to-end models degrade more under goal shifts, suggesting additional failures from observation-level shift and learned representation geometry. See Appendix I.1for absolute metrics.

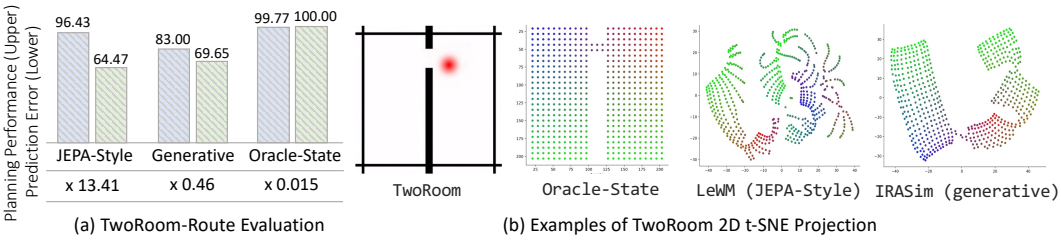

Figure 7: **Representation geometry matters under goal shifts.** (a) The observations remain visually in support, but end-to-end models degrade while oracle-state predictors remain stable. (b) The 2D t-SNE projections illustrate that learned representations can organize the same underlying agent positions with substantially different geometry, which can affect prediction and planning.

marginally familiar factors can therefore create observations outside the joint visual support of expert rollouts, adding an encoder-inference challenge beyond dynamics prediction. (See Appendix I.2 for details.) Crucially, correct factor inference on training observations need not extend to unseen compositions. Even when factors are identifiable in distribution, an encoder may fail to invert novel recombined observations without compositional-consistency constraints (Wiedemer et al., 2024a). Thus, end-to-end world models face a representation-level generalization problem before dynamics prediction is considered. Hence, end-to-end models must therefore infer task-relevant factors from observations generated by unseen factor combinations, in addition to learning dynamics.

Latent geometry shapes degradation under goal shifts. TwoRoom provides a useful case where end-to-end degradation cannot be attributed to an observation-level shift. Because agent position is the only changing factor and TR-Route positions remain marginally within the TwoRoom training support, the corresponding observations stay visually in support, enabling a controlled comparison between end-to-end models and oracle-state predictors. Figure 7 (a) summarizes results from TwoRoom and TR-Route across different family of models. Even without an apparent visual shift, end-to-end models degrade whereas oracle-state predictors remain stable. This indicates that representation quality still matters: the learned latent space may fail to preserve the agent location precisely, introduce unnecessary dimensions or nuisance variation, or organize states in a geometry

that is unfavorable for prediction and planning. Thus, even visually in-support goal shifts can fail if the learned latent geometry is not aligned with the task-relevant state geometry.

6 TOWARDS WORLD MODELS THAT GENERALIZE COMPOSITIONALLY

Our oracle-state experiments show that compositional failures cannot be attributed only to visual representation learning. Even when task-relevant physical state is provided directly, goal shifts remain difficult, particularly when prediction involves contact-rich interactions. This raises a broader question for future work: what assumptions on the dynamics, data, and learned representation are sufficient for a world model to generalize compositionally? We see two distinct challenges.

1. Identifying mechanisms. Fitting training trajectories does not ensure that a predictor has learned reusable action-conditioned mechanisms. This becomes harder in interaction-rich dynamics where action effects can vary with state and regime. Characterizing when such mechanisms can be identified from offline trajectories remains an open problem.

2. Extrapolation beyond the training support. Even if the correct mechanisms are identified, their behavior on novel state–action compositions is generally not determined without additional structural assumptions. Interaction-free dynamics provide a favorable case for compositional generalization because factor effects can often be reused independently, but world models must also handle interaction-rich dynamics where action effects vary with the joint physical configuration and regime. In this setting, extrapolation is fundamentally harder because the relevant interactions may change across compositions, and is a fruitful direction for future work.

A simple illustrative case. One favorable case for extrapolation in interaction-rich dynamics arises when, within each interaction regime, each transition mechanism depends only on a small subset of the state and action variables. Consider regime-dependent dynamics of the following form $z_{t+1} = T_{r_t}(z_t, a_t)$, where r_t denotes an interaction regime, such as free motion, contact, or grasping. Both the autonomous state evolution and the effect of the action may therefore change across regimes. Suppose that, within a fixed regime r, the transition of each next-state factor $z_{k,t+1}$ depends only on a subset of the coordinates of (z_t, a_t). Writing $x_t = (z_t, a_t)$, this sparse dependence can be expressed as follows, $\|\nabla_x T_{k,r_t}(x)\|_0 \ll \dim(x)$.

Let u_k denote the relevant state and action coordinates for k-th transition mechanism T_{k,r_t}, so that we may write $z_{k,t+1} = T_{k,r_t}(u_{k,t})$. This allows a simple case where compositional shift in the full state–action space need not require extrapolation for each transition mechanism.

$$\mathcal{C}^{\mathrm{te}} \not\subseteq \mathcal{C}^{\mathrm{tr}}, \qquad \text{while} \qquad \mathcal{U}_k^{\mathrm{te}} \subseteq \mathcal{U}_k^{\mathrm{tr}} \quad \forall\, k$$

where $\mathcal{U}_k^{\mathrm{tr}}$ and $\mathcal{U}_k^{\mathrm{te}}$ denote the train and test supports of u_k for the fixed regime r. Thus, the complete state–action composition may be unseen, while every input relevant to predicting each next-state factor remains within its training support. Hence, if the corresponding mechanisms have been correctly learned, we can generalize to novel compositions as we do not need to extrapolate any individual mechanism beyond its training support. Appendix J (Proposition 1) gives a formal statement.

7 CONCLUSION

We introduced AnyGoal, a benchmark for evaluating whether world models generalize to new behavioral goals by recombining familiar states and actions. Across a diverse set of models, goal-shift generalization remains difficult even when the relevant state and action factors are marginally covered by the training distribution. Together, these analyses show that goal-shift failures are heterogeneous, reflecting interactions among representation, dynamics prediction, task structure, and evaluation metric. More broadly, AnyGoal connects world-model evaluation with compositional generalization. It asks for prediction on novel joint state–action combinations within the product of marginal training supports, a reasonable but unresolved requirement for reusable dynamics. Our theoretical perspective suggests that global coverage of all state–action combinations is neither realistic nor necessary. Instead, world models need coverage of local mechanism inputs, together with objectives or architectures that identify action-conditioned mechanisms rather than behavior-policy shortcuts. By isolating goal-induced recombination as a failure mode, AnyGoal provides a controlled basis for evaluating progress toward world models that learn reusable dynamics beyond expert rollouts.

AI USE STATEMENT

In this work, we used generative AI tools for implementing methods, cleaning and reformating dataset, and assist in the writing of proofs, propose or refine hypotheses. We have not used generative AI tools for generating synthetic AnyGoal tasks, developing conceptual frameworks, formulating mathematical claims, providing critical ingredients for proving claims, providing feedback on research methodology or experiments, and interpreting results. Additionally, we used generative AI tools for summarizing existing literature and editing a research paper to improve readability. We have reviewed all AI-assisted work. We used these tools only for literature summarization, wording, and clarity edits; all technical claims, citations, experiments, and conclusions were manually checked by the authors. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work studies controlled evaluation of world models in simulated environments. It does not involve human subjects, personal data, or deployment in safety-critical settings. The benchmark is intended to diagnose limitations of learned dynamics models under goal shifts, rather than to enable harmful applications. We do not identify additional ethical concerns beyond standard considerations for releasing simulation datasets, model checkpoints, and evaluation code.

REPRODUCIBILITY STATEMENT

We provide detailed information needed to reproduce the benchmark and experiments in the main paper and appendix. The benchmark construction, task variants, support-verification criteria, model families, oracle-state predictors, planning setup, and evaluation metrics are described. We will also release the dataset-generation code, evaluation scripts, model configurations, and checkpoints to support reproduction of the reported results.

REFERENCES

Rim Assouel, Pietro Astolfi, Florian Bordes, Michal Drozdzal, and Adriana Romero-Soriano. Object-centric binding in contrastive language-image pretraining. *Advances in Neural Information Processing Systems*, 38:131418–131458, 2026.

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

Jack Brady, Julius von Kügelgen, Sébastien Lachapelle, Simon Buchholz, Thomas Kipf, and Wieland Brendel. Interaction asymmetry: A general principle for learning composable abstractions. In *International Conference on Learning Representations*, volume 2025, pp. 46606–46660, 2025.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=H1lJJnR5Ym.

Giacomo Camposampiero, Pietro Barbiero, Michael Hersche, Roger Wattenhofer, and Abbas Rahimi. Scalable evaluation and neural models for compositional generalization. *Advances in Neural Information Processing Systems*, 38:7341–7401, 2026.

Tianyuan Chen, Ronglong Cai, Faguo Wu, and Xiao Zhang. Active: Offline reinforcement learning via adaptive imitation and in-sample v-ensemble. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 77269–77287, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/c06f788963f0ce069f5b2dbf83fe7822-Paper-Conference.pdf.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *Int. J. Rob. Res.*, 44(10–11):1684–1704, September 2025. ISSN 0278-3649. doi: 10.1177/02783649241273668. URL https://doi.org/10.1177/02783649241273668.

Kefan Dong and Tengyu Ma. First steps toward understanding the extrapolation of nonlinear models to unseen domains. *arXiv preprint arXiv:2211.11719*, 2022.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning, 2021. URL https://arxiv.org/abs/2004.07219.

Scott Fujimoto, David Meger, and Doina Precup. Off-policy deep reinforcement learning without exploration. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2052–2062. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/fujimoto19a.html.

Chen-Xiao Gao, Chenyang Wu, Mingjun Cao, Chenjun Xiao, Yang Yu, and Zongzhang Zhang. Behavior-regularized diffusion policy optimization for offline reinforcement learning. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 18630–18657. PMLR, 13–19 Jul 2025a. URL https://proceedings.mlr.press/v267/gao25q.html.

Yunkai Gao, Jiaming Guo, Fan Wu, and Rui Zhang. Policy constraint by only support constraint for offline reinforcement learning, 2025b. URL https://arxiv.org/abs/2503.05207.

Ignat Georgiev, Varun Giridhar, Nick Hansen, and Animesh Garg. Pwm: Policy learning with multi-task world models. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 13737–13757, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/240ea1741b205ea295721d55184ac43b-Paper-Conference.pdf.

David Ha and Jürgen Schmidhuber. Recurrent world models facilitate policy evolution. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper_files/paper/2018/file/2de5d16682c3c35007e4e92982f1a2ba-Paper.pdf.

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

Petr Ivashkov, Randall Balestriero, and Bernhard Schölkopf. Sensorimotor world models: Perception for action via inverse dynamics, 2026. URL https://arxiv.org/abs/2606.20104.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

Daniel Keysers, Nathanael Schärli, Nathan Scales, Hylke Buisman, Daniel Furrer, Sergii Kashubin, Nikola Momchev, Danila Sinopalnikov, Lukasz Stafiniak, Tibor Tihon, Dmitry Tsarkov, Xiao Wang, Marc van Zee, and Olivier Bousquet. Measuring compositional generalization: A comprehensive method on realistic data. In *ICLR*, 2020. URL https://arxiv.org/pdf/1912.09713.pdf.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 21810–21823. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/f7efa4f864ae9b88d43527f4b14f750f-Paper.pdf.

David Klindt, Yann LeCun, and Randall Balestriero. When does lejepa learn a world model?, 2026. URL https://arxiv.org/abs/2605.26379.

Aviral Kumar, Justin Fu, Matthew Soh, George Tucker, and Sergey Levine. Stabilizing off-policy q-learning via bootstrapping error reduction. In H. Wallach,

H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/c2073ffa77b5357a498057413bb09d3a-Paper.pdf.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 1179–1191. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/0d2b2061826a5df3221116a5085a6052-Paper.pdf.

Sébastien Lachapelle, Divyat Mahajan, Ioannis Mitliagkas, and Simon Lacoste-Julien. Additive decoders for latent variables identification and cartesian-product extrapolation. *Advances in Neural Information Processing Systems*, 36:25112–25150, 2023.

Brenden Lake and Marco Baroni. Generalization without systematicity: On the compositional skills of sequence-to-sequence recurrent networks. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 2873–2882. PMLR, 10–15 Jul 2018. URL https://proceedings.mlr.press/v80/lake18a.html.

Romain Laroche, Paul Trichelair, and Remi Tachet Des Combes. Safe policy improvement with baseline bootstrapping. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3652–3661. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/laroche19a.html.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems, 2020. URL https://arxiv.org/abs/2005.01643.

Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified Video Action Model. In *Proceedings of Robotics: Science and Systems*, LosAngeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.074.

Yuejiang Liu, Alexandre Alahi, Chris Russell, Max Horn, Dominik Zietlow, Bernhard Schölkopf, and Francesco Locatello. Causal triplet: An open challenge for intervention-centric causal representation learning. In *Conference on Causal Learning and Reasoning*, pp. 553–573. PMLR, 2023.

Susie Lu, Haonan Chen, Weirui Ye, and Yilun Du. Driftworld: Fast world modeling through drifting, 2026. URL https://arxiv.org/abs/2607.15065.

Lucas Maes, Quentin Le Lidec, Dan Haramati, Nassim Massaudi, Damien Scieur, Yann LeCun, and Randall Balestriero. stable-worldmodel-v1: Reproducible world modeling research and evaluation, 2026a. URL https://arxiv.org/abs/2602.08968.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworldmodel: Stable end-to-end joint-embedding predictive architecture from pixels, 2026b. URL https://arxiv.org/abs/2603.19312.

Divyat Mahajan, Mohammad Pezeshki, Charles Arnal, Ioannis Mitliagkas, Kartik Ahuja, and Pascal Vincent. Compositional risk minimization. In *Proceedings of the 42nd International Conference on Machine Learning*, pp. 42503–42553. PMLR, 2025. URL https://proceedings.mlr.press/v267/mahajan25a.html.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel HAZIZA, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=a68SUt6zFt. Featured Certification.

Vitória Barin Pacela, Shruti Joshi, Isabela Camacho, Simon Lacoste-Julien, and David Klindt. Stop probing, start coding: Why linear probes and sparse autoencoders fail at compositional generalisation. *arXiv preprint arXiv:2603.28744*, 2026.

Giambattista Parascandolo, Niki Kilbertus, Mateo Rojas-Carulla, and Bernhard Schölkopf. Learning independent causal mechanisms. In *International Conference on Machine Learning*, pp. 4036–4044. PMLR, 2018.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 94937–94982, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/ecd92623ac899357312aaa8915853699-Paper-Conference.pdf.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4195–4205, October 2023.

Marc Rigter, Bruno Lacerda, and Nick Hawes. Rambo-rl: Robust adversarial model-based offline reinforcement learning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 16082–16097. Curran Associates, Inc., 2022. doi: 10.52202/068431-1170. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/6691c5e4a199b72dffd9c90acb63bcd6-Paper-Conference.pdf.

B. Schölkopf, F. Locatello, S. Bauer, N. R. Ke, N. Kalchbrenner, A. Goyal, and Y. Bengio. Toward causal representation learning. *Proceedings of the IEEE*, 109(5):612–634, 2021. doi: 10.1109/JPROC.2021.3058954. URL https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9363924. *equal contribution.

Yuhong Shi, Zhenhao Chu, Jie Wei, Jun Hao, Jianyi Liu, and Jingwen Fu. Overcoming statistical bias in action-controllable world models, 2026. URL https://arxiv.org/abs/2608.04653.

Uladzislau Sobal, Wancong Zhang, Kyunghyun Cho, Randall Balestriero, Tim G. J. Rudner, and Yann LeCun. Learning from reward-free offline data: A case for planning with latent dynamics models. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 43905–43941. Curran Associates, Inc., 2025. doi: 10.52202/085713-1465. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/3e7cf447f21cd11c846463affefce665-Paper-Conference.pdf.

Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. The MIT Press, Cambridge, MA, 2nd edition, 2018.

Cheng Tang, Zhishuai Liu, and Pan Xu. Robust offline reinforcement learning with linearly structured f-divergence regularization. In Aarti Singh, Maryam Fazel, Daniel Hsu, Simon Lacoste-Julien, Felix Berkenkamp, Tegan Maharaj, Kiri Wagstaff, and Jerry Zhu (eds.), *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 58842–58882. PMLR, 13–19 Jul 2025. URL https://proceedings.mlr.press/v267/tang25f.html.

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, Timothy Lillicrap, and Martin Riedmiller. Deepmind control suite, 2018. URL https://arxiv.org/abs/1801.00690.

Tristan Thrush, Ryan Jiang, Max Bartolo, Amanpreet Singh, Adina Williams, Douwe Kiela, and Candace Ross. Winoground: Probing vision and language models for visio-linguistic compositionality. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5238–5248, June 2022.

Xinyue Wang and Biwei Huang. Modeling unseen environments with language-guided composable causal components in reinforcement learning. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu

(eds.), *International Conference on Learning Representations*, volume 2025, pp. 48784–48812, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/79d86433c2acd12b6fa98553435d226e-Paper-Conference.pdf.

Ying Wang, Oumayma Bounou, Yann LeCun, and Mengye Ren. Adajepa: An adaptive latent world model, 2026. URL https://arxiv.org/abs/2606.32026.

Archana Warrier, Dat Nguyen, Michelangelo Naim, Moksh Jain, Yichao Liang, Karen Schroeder, Cambridge Yang, Joshua B. Tenenbaum, Sebastian Josef Vollmer, Kevin Ellis, and Zenna Tavares. Benchmarking world-model learning with environment-level queries. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=Ny6UYZ8ysN.

Thaddäus Wiedemer, Jack Brady, Alexander Panfilov, Attila Juhos, Matthias Bethge, and Wieland Brendel. Provable compositional generalization for object-centric learning. In *International Conference on Learning Representations*, volume 2024, pp. 20137–20164, 2024a.

Thaddäus Wiedemer, Prasanna Mayilvahanan, Matthias Bethge, and Wieland Brendel. Compositional generalization from first principles. *Advances in Neural Information Processing Systems*, 36, 2024b.

Xindi Wu, Dingli Yu, Yangsibo Huang, Olga Russakovsky, and Sanjeev Arora. Conceptmix: A compositional image generation benchmark with controllable difficulty. *Advances in Neural Information Processing Systems*, 37:86004–86047, 2024.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 14129–14142. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/a322852ce0df73e204b7e67cbbef0d0a-Paper.pdf.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 28954–28967. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/f29a179746902e331572c483c45e5086-Paper.pdf.

Aimen Zerroug, Mohit Vaishnav, Julien Colin, Sebastian Musslick, and Thomas Serre. A benchmark for compositional visual reasoning. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 29776–29788. Curran Associates, Inc., 2022. doi: 10.52202/068431-2159. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/c08ee8fe3d19521f3bfa4102898329fd-Paper-Datasets_and_Benchmarks.pdf.

Xiangcheng Zhang and Yilun Du. World action planner: Generalizable decision-making with action-conditioned world models, 2026. URL https://arxiv.org/abs/2607.27599.

Linfeng Zhao, Lingzhi Kong, Robin Walters, and Lawson L.S. Wong. Toward compositional generalization in object-oriented world modeling. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 26841–26864. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/zhao22b.html.

Yi Zhao, Aidan Scannell, Wenshuai Zhao, Yuxin Hou, Tianyu Cui, Le Chen, Dieter Büchler, Arno Solin, Juho Kannala, and Joni Pajarinen. Efficient reinforcement learning by guiding world models with non-curated data. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 79246–79280, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/7ff013b7e372ba5b790352ccd6908f03-Paper-Conference.pdf.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=D5RNACOZEI.

Fangqi Zhu, Hongtao Wu, Song Guo, Yuxiao Liu, Chi-Hou Cheang, and Tao Kong. Irasim: A fine-grained world model for robot manipulation. *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9834–9844, 2024. URL https://api.semanticscholar.org/CorpusID:270620594.

Supplementary Material

AnyGoal: On World Model Generalization Beyond Expert Rollouts

A. Compositional Generalization: Essential World-Model Test . . . 17
B. AnyGoal Task Details . . . 17
B.1 Two Room and Its Variants . . . 17
B.2 PushT and Its Variants . . . 18
B.3 Insert-Peg and Its Variants . . . 18
B.4 OGBench Cube and Its Variants . . . 18
C. Statistical Test 1: Marginal Distribution Support . . . 19
C.1 Distribution Support Criteria . . . 19
C.2 Verified Factor Groups . . . 20
C.3 Support-Verification Results . . . 20
D. Statistical Test 2: Full-Dimensional Joint Distribution Support . . . 22
E. Example of Existing Bias . . . 23
E.1 Interventional Action Set . . . 23
E.2 Comp Gen Failed and show degeneracy due to Goal Bias . . . 25
F. Model Explanation . . . 25
F.1 End-to-End World Models . . . 26
F.2 Predictor-Only Models from True States . . . 27
G. Training Details . . . 28
H. Detailed Planning and Evaluation Configuration . . . 28
H.1 Planning Evaluation Criteria . . . 28
H.2 Configuration . . . 30
I. Results . . . 30
I.1 Full Results . . . 30
I.2 Factor Support versus Observation Support . . . 30
I.3 Additional Observations . . . 32
J. Toward Compositional Generalization in World Models . . . 32

A Compositional Generalization: Essential World-Model Test

We formalize why our compositional evaluation is an essential test for world models. The key distinction is between extrapolating to unseen factors and recombining observed ones. Good action-conditioned world model should predict the effect of an observed action on familiar physical factors, even when that particular combination was not selected by the behavior policy.

For clarity, consider a deterministic oracle-state setting with true controlled dynamics $z_{t+1} = T^\star(z_t, a_t)$. Suppose the offline dataset is collected by a behavior policy $a = \pi_E(z)$. Then the observed state–action support lies on the policy graph,

$$\Gamma_{\pi_E} = \{(z, a) : a = \pi_E(z)\}. \tag{5}$$

Thus, expert rollouts cover only the state–action pairings induced by the behavior policy, not arbitrary interventions with observed actions.

Zero Training Error Does Not Imply Action-Conditioned Dynamics. A predictive world model trained on $(z, a, z') \sim p_{\mathrm{tr}}$ is constrained only on the state–action pairs generated by the behavior policy. Since observed actions satisfy $a = \pi_E(z)$, the training loss constrains $\hat{T}(z, \pi_E(z))$, but not $\hat{T}(z, a)$ for alternative observed actions at the same state. Consequently, the action-ignorant predictor

$$\hat{T}_{\mathrm{deg}}(z, a) = T^\star(z, \pi_E(z)) \tag{6}$$

achieves zero prediction error on the behavior dataset while ignoring the queried action a. Thus, expert-rollout prediction cannot distinguish a controlled dynamics model from a predictor that only models behavior-policy transitions.

Compositional Evaluation Requires Action-Conditioned Dynamics Compositional evaluation rules out this degenerate solution. Assume actions have nonzero effect within the observed marginal supports: for a set of oracle states with positive measure, there exist observed actions $a, a' \in \mathcal{A}^{\mathrm{tr}}$ such that

$$T^\star(z, a) \neq T^\star(z, a').$$

Now consider any action-ignorant predictor $\hat{T}(z, a) = h(z)$. If it were correct on the relevant compositional state–action space, then for every observed action $a \in \mathcal{A}^{\mathrm{tr}}$, $h(z) = T^\star(z, a)$. Therefore, for any two observed actions a, a',

$$T^\star(z, a) = h(z) = T^\star(z, a'), \tag{7}$$

which contradicts the action-relevance assumption. Hence, no action-ignorant model can be correct on recombinations of observed state and action factors when actions causally affect the next state.

> **Interpretation.** Expert rollouts test whether a model predicts what happens under the behavior policy. Compositional state–action evaluation instead tests whether the model predicts what happens when observed actions are applied to familiar physical factors, including combinations not generated by the behavior policy. This is an essential requirement for model-based control, since planning must evaluate candidate actions other than the behavior action.

B AnyGoal Task Details

B.1 Two Room and Its Variants

TwoRoom is a simple navigation environment with two rooms connected by a narrow slot. The agent must move through the slot and reach a target location, with little to no meaningful interaction with the environment beyond collision with walls. Because the dynamics are essentially planar motion, this environment isolates whether models can preserve action-conditioned movement under goal-biased trajectory data.

TwoRoom Triangles is an interaction-light navigation variant. The agent must follow a piecewise-linear triangular route defined by a start point, an intermediate waypoint, and an invisible endpoint, rather than moving directly through the slot toward the target. Episodes mix clean, noisy, and variable-speed actions within the observed action support. This variant tests whether the model preserves planar action-conditioned dynamics when familiar positions and actions are recombined outside the dominant slot-crossing behavior of the base dataset.

B.2 PushT and Its Variants

PushT is a planar manipulation environment where the agent must push a T-shaped block into a fixed T-shaped goal pose. Unlike TwoRoom, the base task already requires meaningful contact, object translation, and object rotation. The expert trajectories are strongly goal-directed, coupling agent motion with the block pose and the target configuration. This environment tests whether world models learn action-conditioned contact dynamics rather than only the dominant expert pushing manifold.

AvoidT is a non-interactive variant. The agent must avoid touching the physical T-block and move to one of four corner regions. This variant keeps the agent and action factors familiar, but changes the required behavior from goal-directed pushing to free-space navigation around the object. It primarily tests whether the model preserves the action effect when the expert pushing objective is no longer appropriate.

StackT is an interaction-rich variant. The physical T-block must be placed so that its head overlaps with the fixed goal T, while its rotation is opposite to the original PushT goal orientation, creating a plus-shaped target configuration. This variant requires contact and rotation dynamics under a goal that differs from the original training objective.

PushT Anti-goal is also interaction-rich and directly reverses the dominant goal bias in the base dataset. The physical T-block is initialized near the original PushT goal pose, but the agent must push it away from the fixed goal region and move it toward one of four corner regions, with rotation opposite to the original goal orientation. This tests whether the model can predict familiar pushing dynamics when the required behavior contradicts the original expert trajectory distribution.

B.3 Insert-Peg and Its Variants

Insert-Peg is a planar manipulation environment where an arm must move a peg toward a sampled slot and insert it into the target opening. The task involves articulated robot motion, object contact, and alignment between the peg and the slot. Expert trajectories are strongly structured around successful insertion behavior. This environment tests whether models can learn action-conditioned manipulation dynamics under goal-biased contact-rich data.

Avoid-Peg is an interaction-light variant. The arm must move around the target box in a specified hugging direction while avoiding contact with the peg. This changes the goal-conditioned arm behavior without requiring peg manipulation, testing whether the model can predict familiar arm motion under a non-insertion objective.

Extract-Peg is an interaction-rich variant. The arm must extract the peg from the target box, move it through a waypoint, and place it at a destination, with episodes balanced between left- and right-side box placements. This reverses the dominant insertion behavior while keeping the peg, box, workspace, and action space familiar.

B.4 OGBench Cube and Its Variants

Cube is a robot manipulation environment where the end-effector must interact with a cube and move it toward a target configuration. The base behavior follows a structured pick-and-place pattern, including approach, grasp, lift, transport, and release. This creates a goal-biased trajectory distribution in which robot motion, gripper state, cube pose, and target location are tightly coupled. The environment tests whether models learn manipulation dynamics beyond this dominant expert sequence.

CubeHop is a non-interactive variant. The robot does not intentionally move the cube. Instead, the end-effector moves between waypoint and target positions while gradually changing its yaw. This variant changes the goal-conditioned end-effector behavior while avoiding object interaction. It tests whether the model can predict familiar robot motion and yaw changes outside the dominant cube-manipulation trajectory distribution.

CubeDiagonal is an interaction-rich variant. The robot manipulates the cube using a diagonal grasp configuration, where the gripper contacts opposite edges of the cube instead of the standard parallel faces. This introduces a novel contact regime with different grasp geometry while keeping the ob-

ject, task objective, and action space unchanged. It tests whether the model can compose familiar manipulation factors under unseen interaction configurations.

C Statistical Test 1: Marginal Distribution Support

C.1 Distribution Support Criteria

We describe the statistical criteria used to decide whether a query distribution remains inside the support of a reference distribution. Let $P = \{p_n\}_{n=1}^N$ denote the reference or training point set, and let $Q = \{q_m\}_{m=1}^M$ denote the query point set.

For each factor, we compute the empirical mean $\mu_P = \frac{1}{N}\sum_{n=1}^N p_n$ and regularized covariance $\Sigma_P = \mathrm{Cov}(P) + \lambda I$, where $\lambda > 0$ is a small diagonal regularizer used to ensure that the covariance matrix is invertible. This is particularly important for low-rank or manifold-valued factors, such as circular variables represented by sine and cosine coordinates.

Now we measure three metric respectively. First, the Mahalanobis criterion checks whether query points remain within the reference distribution's main covariance structure.

Empirical Mahalanobis criterion. The empirical squared Mahalanobis distance of a point x from the reference distribution is

$$d_P^2(x) = (x - \mu_P)^\top \Sigma_P^{-1}(x - \mu_P). \tag{8}$$

We compute the reference threshold as the 99th percentile of the empirical Mahalanobis distances of the reference points, $\tau_{0.99} = \mathrm{Quantile}_{0.99}(\{d_P^2(p_n)\}_{n=1}^N)$. The query set passes the Mahalanobis support criterion if at least 95% of its points lie below this reference threshold:

$$\mathrm{Mahalanobis}_{0.99}(Q; P) = \left[\frac{1}{M}\sum_{m=1}^M \mathbf{1}\left[d_P^2(q_m) \le \tau_{0.99}\right] \ge 0.95\right]. \tag{9}$$

For a one-dimensional factor, this criterion reduces to the regularized squared z-score $d_P^2(x) = (x - \mu_P)^2/(\sigma_P^2 + \lambda)$.

Second, the KDE criterion checks whether query points lie in regions with sufficient empirical reference density, beyond an ellipsoidal approximation.

Histogram-binned KDE criterion. We estimate a reference density ρ_P with a histogram-binned Gaussian KDE fit only on the reference set P. Query points are then evaluated under this fixed density estimate. The histogram grid is constructed from the pooled reference–query range with a small padding, and Gaussian smoothing uses Scott's rule with a lower bandwidth clip to avoid degenerate bins.
After fitting the KDE, we evaluate the density of all reference samples and set the threshold to the empirical first percentile,

$$\delta_{0.01} = \mathrm{Quantile}_{0.01}\left(\{\rho_P(p_n)\}_{n=1}^N\right).$$

The query set passes the KDE criterion if at least 95% of query points have density no smaller than this reference-sample threshold:

$$\mathrm{KDE}_{0.01}(Q; P) = \left[\frac{1}{M}\sum_{m=1}^M \mathbf{1}[\rho_P(q_m) \ge \delta_{0.01}] \ge 0.95\right]. \tag{10}$$

For periodic yaw variables, KDE smoothing is applied with wrapped boundary conditions.

Lastly, the range criterion checks whether query points avoid direct coordinate-wise extrapolation beyond the observed marginal range.

Observed-range criterion. As a complementary support check, we compute an axis-wise observed-range test. For each query point q_m, we check whether all of its coordinates lie inside

the coordinate-wise minimum and maximum of the reference set, i.e., $q_m \in [\min(P), \max(P)]$ iff $\min_n p_{n,j} \leq q_{m,j} \leq \max_n p_{n,j}$ for all j. The observed-range criterion is

$$\text{range}(Q; P) = \left[\frac{1}{M} \sum_{m=1}^{M} \mathbf{1}[q_m \in [\min(P), \max(P)]] \geq 0.99 \right]. \tag{11}$$

For circular variables, such as yaw, the observed-range criterion is ignored because the interval boundary is representation-dependent. In that case, the Mahalanobis and KDE criteria are used without the range check.

The final verdict is the conjunction of all non-ignored criteria:

$$\text{pass}(Q; P) = \text{range}(Q; P) \wedge \text{Mahalanobis}_{0.99}(Q; P) \wedge \text{KDE}_{0.01}(Q; P). \tag{12}$$

If the observed-range criterion is ignored for a circular factor, the verdict is computed over the remaining criteria:

$$\text{pass}(Q; P) = \text{Mahalanobis}_{0.99}(Q; P) \wedge \text{KDE}_{0.01}(Q; P). \tag{13}$$

This design separates marginal factor support from full trajectory-distribution typicality. The support checks are applied at the level of semantic factor groups to verify that each physical factor remains within the reference support. Combinatorial novelty can still arise when individually supported factors appear in new joint configurations that are rare or absent under the expert trajectory distribution.

C.2 Verified Factor Groups

We apply the three support criteria above to task-relevant factor groups rather than to the full joint state–action vector. This verifies marginal coverage of the physical factors and actions used in each goal-shifted variant, while allowing the full combinations to remain novel. Table 2 lists the factor groups checked for each reference–query pair.

Table 2: **Every independent factor is in-distribution with respect to the reference dataset.** Factor groups used for support verification. For each reference dataset P, we compare the corresponding goal-shifted variant datasets Q against the listed task-relevant factors..

Reference dataset P	Query dataset Q	Factor groups checked
TwoRoom	TR-Route	Agent position $(2D)$; action position delta $(2D)$
PushT	Avoid-T; Stack-T; Push-T Antigoal	Agent position $(2D)$; block position $(2D)$; action $(2D)$; block angle.
Insert-Peg	Avoid-Peg; Extract-Peg	Arm root angle; Arm shoulder angle; Arm elbow angle; Arm wrist angle; Finger angle; Fingertip angle; Thumb angle; Thumbtip angle; hand, peg, and target position (2D); hand, peg, and target orientation (2D) ; 5 motor commands (1D) independently.
Cube	Cube-Hop; Cube-Diagonal	End-effector position $(3D)$; block position $(3D)$; block quaternion $(4D)$; action position delta $(3D)$; gripper opening; gripper contact; end-effector yaw; action yaw delta; action gripper delta. ; 6 joint pos independently

C.3 Support-Verification Results

Table 3 reports the full support-verification results for the factor groups in Table 2. For each reference–query pair, we report the pass rate under the Mahalanobis, KDE, and observed-range criteria, together with the final conjunction verdict. All checked factors pass, confirming that the AnyGoal variants remain marginally in distribution with respect to the corresponding base datasets.

Table 3: Distribution-support verification between each reference dataset (P) and query dataset (Q). A dash indicates that the range criterion is not applied to circular variables.(**All Passed**)

P	Q	Factor of variants	Mahalanobis 0.99	KDE	Range	Final verdict
TwoRoom	TR-Route	agent pos (2D)	0.970	1.000	1.000	pass
		action (2D)	1.000	0.957	1.000	pass
PushT	Avoid-T	block angle	0.970	0.963	–	pass
		agent pos (2D)	0.968	0.984	1.000	pass
		block pos (2D)	1.000	0.990	1.000	pass
		action (2D)	1.000	0.995	1.000	pass
PushT	Stack-T	block angle	0.998	0.989	–	pass
		agent pos (2D)	0.992	0.989	1.000	pass
		block pos (2D)	1.000	0.999	1.000	pass
		action (2D)	1.000	0.954	1.000	pass
PushT	Push-T Antigoal	block angle	1.000	0.988	–	pass
		agent pos (2D)	0.988	0.986	1.000	pass
		block pos (2D)	1.000	1.000	1.000	pass
		action (2D)	1.000	0.989	1.000	pass
Insert-Peg	Avoid-Peg	arm root angle	0.980	0.963	1.000	pass
		arm shoulder angle	1.000	0.999	1.000	pass
		arm elbow angle	0.999	0.998	1.000	pass
		arm wrist angle	1.000	1.000	1.000	pass
		finger angle	1.000	1.000	1.000	pass
		fingertip angle	0.996	0.998	1.000	pass
		thumb angle	1.000	1.000	1.000	pass
		thumbtip angle	0.994	0.997	1.000	pass
		hand position (x,z)	0.978	1.000	1.000	pass
		hand orientation (qw,qy)	1.000	1.000	1.000	pass
		peg position (x,z)	1.000	1.000	1.000	pass
		peg orientation (qw,qy)	0.999	0.999	1.000	pass
		target position (x,z)	1.000	1.000	1.000	pass
		target orientation (qw,qy)	1.000	1.000	1.000	pass
		motor command 0	1.000	0.982	1.000	pass
		motor command 1	1.000	1.000	1.000	pass
		motor command 2	1.000	0.991	1.000	pass
		motor command 3	1.000	1.000	1.000	pass
		motor command 4	1.000	1.000	1.000	pass
Insert-Peg	Extract-Peg	arm root angle	0.998	0.985	1.000	pass
		arm shoulder angle	0.993	0.990	1.000	pass
		arm elbow angle	0.993	0.992	1.000	pass
		arm wrist angle	1.000	0.999	1.000	pass
		finger angle	1.000	1.000	1.000	pass
		fingertip angle	0.993	0.994	1.000	pass
		thumb angle	0.999	0.998	1.000	pass
		thumbtip angle	0.979	0.976	1.000	pass
		hand position (x,z)	1.000	0.999	1.000	pass
		hand orientation (qw,qy)	1.000	1.000	1.000	pass
		peg position (x,z)	0.999	0.998	1.000	pass
		peg orientation (qw,qy)	0.998	0.998	1.000	pass
		target position (x,z)	0.983	0.978	1.000	pass
		target orientation (qw,qy)	1.000	1.000	1.000	pass
		motor command 0	1.000	1.000	1.000	pass
		motor command 1	1.000	1.000	1.000	pass
		motor command 2	1.000	0.989	1.000	pass
		motor command 3	1.000	0.999	1.000	pass
		motor command 4	1.000	1.000	1.000	pass
Cube	Cube-Hop	joint pos 0	0.995	0.995	1.000	pass
		joint pos 1	0.997	0.998	1.000	pass
		joint pos 2	0.974	0.974	1.000	pass
		joint pos 3	0.990	0.991	1.000	pass
		joint pos 4	1.000	1.000	1.000	pass
		joint pos 5	0.957	0.960	1.000	pass
		gripper opening	0.980	1.000	1.000	pass
		gripper contact	1.000	1.000	1.000	pass
		EE yaw	0.958	0.961	–	pass
		delta yaw	1.000	1.000	1.000	pass

Continued on next page

Table 3 continued.

P	Q	Factor of variants	Mahalanobis 0.99	KDE	Range	Final verdict
		delta gripper	1.000	1.000	1.000	pass
		EE pos (3D)	0.969	0.970	1.000	pass
		block pos (3D)	1.000	1.000	1.000	pass
		block quat (4D)	1.000	1.000	1.000	pass
		delta pos (3D)	1.000	1.000	1.000	pass
Cube	Cube-Diagonal	joint pos 0	0.989	0.990	1.000	pass
		joint pos 1	0.992	0.991	1.000	pass
		joint pos 2	0.990	0.990	1.000	pass
		joint pos 3	0.991	0.992	1.000	pass
		joint pos 4	0.978	0.979	1.000	pass
		joint pos 5	1.000	1.000	1.000	pass
		gripper opening	0.989	0.984	1.000	pass
		gripper contact	1.000	0.985	1.000	pass
		EE yaw	1.000	1.000	–	pass
		delta yaw	1.000	0.992	1.000	pass
		delta gripper	0.995	0.995	1.000	pass
		EE pos (3D)	0.989	0.990	1.000	pass
		block pos (3D)	0.991	0.990	1.000	pass
		block quat (4D)	0.997	0.987	1.000	pass
		delta pos (3D)	0.989	0.990	1.000	pass

D Statistical Test 2: Full-Dimensional Joint Distribution Support

We complement the factor-wise support tests in Table 3 with a full-dimensional joint test. The factor-wise tests verify whether each task-relevant factor or factor group is marginally supported by the reference data. In contrast, the joint test asks whether the complete state–action combination lies in a high-density region of the reference distribution. This test uses the concatenated vectors directly, without coordinate marginalization or worst-pair reduction.

For each transition, let a denote the action, s_{agent} the agent-observed state, and s_{other} the task-object and target state. We evaluate three joint vectors:

$$z_{\text{agent}} = s_{\text{agent}}, \qquad z_{\text{all}} = [s_{\text{agent}}, s_{\text{other}}], \qquad z_{a,\text{all}} = [a, s_{\text{agent}}, s_{\text{other}}].$$

Only rows that are finite in every coordinate of the corresponding vector are retained. Thus, terminal rows with undefined actions contribute to the state-only vectors but not to the action–state vector.

Table 4: Components and dimensionalities of the full-dimensional joint KDE.

Family	s_{agent}	s_{all}	$[a, s_{\text{all}}]$
TwoRoom	position (2D)	position (2D)	4D
PushT	agent position (2D)	+ block pose (5D total)	7D
Cube	EE/gripper state (6D)	+ block pose (13D total)	18D
Manipulator	joints + hand pose (20D)	+ peg/target poses (28D total)	33D

For each reference vector set, we quantize each coordinate using reference-based limits with small padding, and assign zero density to target values outside these limits. We then whiten the quantized vectors using the full reference covariance with a small diagonal regularizer. This preserves correlations between action and state coordinates when computing kernel distances.

We fit a full-dimensional Gaussian KDE to the whitened reference vectors using Scott's bandwidth rule. The density is evaluated without reference subsampling, dimension dropping, nearest-neighbor truncation, or pairwise marginalization. The density threshold is calibrated only from the reference samples:

$$\tau = Q_{0.01}\big(\{\log \widehat{p}(x_i)\}_{i=1}^{N}\big)\,. \tag{14}$$

For target vectors $Y = \{y_j\}_{j=1}^{M}$, we report the fraction whose log-density exceeds this reference-sample threshold:

$$\mathrm{ID}(Y; X) = \frac{1}{M} \sum_{j=1}^{M} \mathbf{1}[\log \widehat{p}(y_j) \geq \tau] \,. \tag{15}$$

The joint score is therefore a full-vector KDE support percentage. Individual factors can pass their marginal support tests while the concatenated vector has a low joint score, indicating that the corresponding combination of familiar values is rare under the reference distribution.

E Example of Existing Bias

In this section, we show that the bias is empirically realized through targeted action interventions on familiar but under-covered state–action combinations, eliciting degenerate model responses. For each state s, we replace the original action with a fixed set of interventional actions $\mathcal{A}_{\mathrm{cf}}$ and measure the resulting change in the model prediction:

$$S_\theta(s) = \frac{1}{|\mathcal{A}_{\mathrm{cf}}|} \sum_{a' \in \mathcal{A}_{\mathrm{cf}}} \|\phi_\theta(s, a) - \phi_\theta(s, a')\|_2 \,,$$

where ϕ_θ denotes the predicted representation used for the sensitivity measurement. With LeWM, we compute this score over training states and plot it against an environment-specific bias coordinate, such as distance to the slot, block, target, or end-effector height.

E.1 Interventional Action Set

For each environment, we choose a fixed interventional action set that remains within the empirical action support while covering the task-relevant intervention directions as uniformly as possible. For environments with direct 2D controls, this is a regular grid in action space. For higher-dimensional manipulation actions, we vary only the dimensions needed for the intended spatial intervention and keep the remaining dimensions fixed.

TwoRoom. For TwoRoom, actions are two-dimensional position deltas, so interventional interventions are defined directly in the action space. We use a fixed 3×3 grid that is uniformly spread over the two action dimensions and includes the zero action as the center intervention.

Selected TwoRoom interventional actions.

$$\mathcal{A}_{\mathrm{cf}}^{\mathrm{TwoRoom}} = \left\{ \begin{array}{rr} -1 & -1 \\ -1 & 0 \\ -1 & 1 \\ 0 & -1 \\ 0 & 0 \\ 0 & 1 \\ 1 & -1 \\ 1 & 0 \\ 1 & 1 \end{array} \right\} = \{-1, 0, 1\} \times \{-1, 0, 1\}.$$

Each row is a 2D action.

Push-T. For Push-T, actions are also two-dimensional planar controls. We therefore use a fixed 3×3 grid in the planar action space. The grid is chosen to provide evenly distributed in-distribution interventions at the empirical action scale of the dataset.

Selected Push-T interventional actions.

$$\mathcal{A}_{\mathrm{cf}}^{\mathrm{PushT}} = \left\{ \begin{array}{rr} 0 & 0 \\ 0.25 & 0 \\ -0.25 & 0 \\ 0.25 & -0.25 \\ 0.25 & 0.25 \\ -0.25 & -0.25 \\ -0.25 & 0.25 \\ 0 & -0.25 \\ 0 & 0.25 \end{array} \right\}.$$

Each row is a 2D planar action.

Insert-Peg. Unlike the 2D navigation and planar pushing tasks, Insert-Peg actions are 5-dimensional motor commands rather than direct task-space displacements. Therefore, interventional actions cannot be chosen by simply placing actions on a Euclidean grid in task space. Instead, we select actions by their realized effect on the hand endpoint.

We first choose a representative state near the center of the expert-state distribution. Each candidate action is sampled from the dataset and repeated for five environment control steps from this same state. We then measure the resulting hand endpoint in the x-z plane. To ensure that interventions remain in distribution, we retain only actions whose 20-nearest-neighbor distance in the raw 5D action space is below the 95th-percentile threshold estimated from held-out dataset actions.

Among the retained actions, we select a fixed set of nine interventional actions whose realized endpoints form the widest approximately uniform 3×3 grid. The zero action is assigned to the center, and the remaining actions are chosen to maximize grid spacing while keeping each endpoint within 10% of its corresponding ideal grid point. The resulting grid spacing is 0.0358, with a maximum endpoint mismatch of 9.95% of the grid spacing.

Selected Insert-Peg actions.

$$\mathcal{A}_{\mathrm{cf}}^{\mathrm{Insert}} = \left\{ \begin{array}{rrrrr} 0.6728 & -0.9814 & -0.9932 & -1.0000 & 0.9900 \\ 0.1353 & -1.0000 & 0.6557 & -0.9972 & 1.0000 \\ -0.6023 & -0.1204 & 0.3142 & 0.9943 & -1.0000 \\ 1.0000 & -0.8002 & -1.0000 & -1.0000 & 0.9739 \\ 0.0000 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ -0.7797 & 1.0000 & -0.6672 & 0.6553 & -1.0000 \\ 0.9130 & -0.0082 & -1.0000 & -0.9820 & 1.0000 \\ -0.0513 & 1.0000 & -1.0000 & 0.5704 & 0.1936 \\ -0.5833 & 0.8958 & 0.8234 & 0.0582 & 0.1618 \end{array} \right\}.$$

Each row is a 5D motor command. The zero action is the center intervention.

Cube. For Cube, the raw action is five-dimensional, but the interventional protocols are designed to probe spatial sensitivity of end-effector motion. We therefore vary only translational action dimensions and keep $\Delta\mathrm{yaw}$ and $\Delta\mathrm{gripper}$ fixed at zero. The horizontal protocol uses an x-y grid.

Selected Cube horizontal interventional actions.

$$\mathcal{A}_{\mathrm{cf}}^{\mathrm{Cube\text{-}xy}} = \left\{ \begin{array}{rrrrr} -0.5427 & -0.9504 & 0 & 0 & 0 \\ -0.5427 & 0 & 0 & 0 & 0 \\ -0.5427 & 0.9504 & 0 & 0 & 0 \\ 0 & -0.9504 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0.9504 & 0 & 0 & 0 \\ 0.5427 & -0.9504 & 0 & 0 & 0 \\ 0.5427 & 0 & 0 & 0 & 0 \\ 0.5427 & 0.9504 & 0 & 0 & 0 \end{array} \right\}.$$

Each row is a normalized 5D Cube action $(a_x, a_y, a_z, \Delta\mathrm{yaw}, \Delta\mathrm{gripper})$.

E.2 Comp Gen Failed and show degeneracy due to Goal Bias

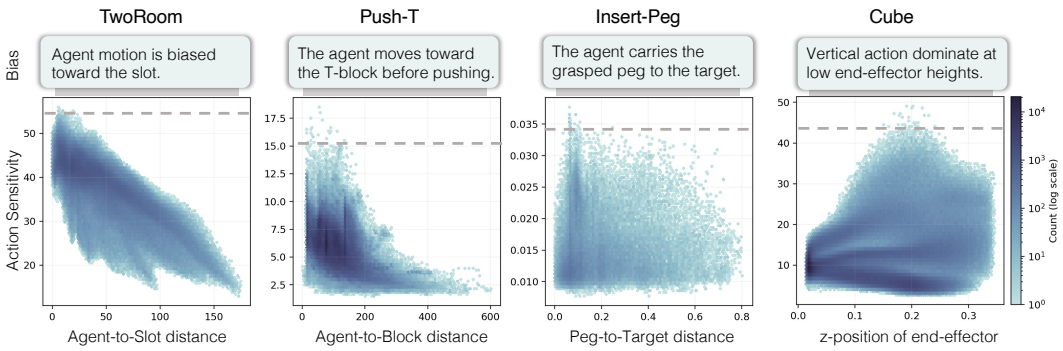

Figure 8: **Goal-biased training support is associated with reduced action sensitivity.** Each panel plots model action sensitivity against an environment-specific bias coordinate induced by the expert policy. The analysis suggest that the model can partially rely on a goal-directed prior rather than the supplied action.

Figure 8 shows a consistent pattern across environments. Expert rollouts induce strong correlations between state regions and preferred actions: agents move toward the slot in TwoRoom, approach the block before pushing in Push-T, carry the peg toward the target in Insert-Peg, and favor vertical actions at low end-effector heights in Cube. In these biased regions, the model often exhibits lower sensitivity to interventional actions. This suggests that the predictor can partially internalize the goal-directed regularities of the behavior policy, rather than remaining uniformly responsive to the supplied action. This analysis does not by itself prove a causal mechanism for every AnyGoal failure, but it provides diagnostic evidence for a central failure mode: goal-biased training support can make a learned world model behave more like a behavior predictor than a reusable action-conditioned dynamics model.

Implementation Detail Within orignal LeWM architecture, we measure action sensitivity using hidden representations from the predictor blocks. Let f_l denote the last-token hidden representation after the l-th ConditionalBlock,

$$f_l = \texttt{output[:, -1, :]}$$

obtained from a forward hook on `model.predictor.transformer.layers[layer_idx]`. Thus, f_l is not the representation immediately after AdaLN modulation; it is the output of the full AdaLN-conditioned transformer block, including the attention and MLP residual updates. In all reported layer-wise analyses, we average the sensitivity scores across these block outputs. For the final prediction representation, we use `pred[:, -1, :]`, the output of `model.predict(...)` after the final `pred_proj` layer.

F Model Explanation

Our model selection is designed to span several substantially different approaches to visual world modeling rather than to concentrate on a single architectural family. In particular, we evaluate both joint-embedding predictive architectures (JEPAs) and generative visual world models. Within the JEPA family, we include methods that jointly learn perception and dynamics from the offline dataset as well as a method that keeps a large pretrained visual encoder frozen. Within the generative family, we include both a direct action-conditioned generator and a latent video diffusion model with an explicit visual encoder–decoder. This diversity is important for our central question: whether goal-induced combinatorial bias is specific to a particular representation or generative objective, or can arise more broadly when action-conditioned prediction is trained on a biased offline distribution. For consistency, all models use frame skip 5, history length 3, and one-step prediction, except Sensorimotor-WM, where we keep the original history length 1 to preserve its inverse-dynamics setup.

F.1 End-to-End World Models

LeWorldModel (LeWM). LeWorldModel (Maes et al., 2026b) is an end-to-end joint-embedding predictive architecture trained directly from pixels and actions. An image encoder maps observations to compact latent representations, and an autoregressive transformer predictor models the next latent representation conditioned on the action. Actions are injected into each transformer layer through adaptive layer normalization (AdaLN-zero). Unlike approaches that rely on a frozen encoder, stop-gradient target network, or exponential-moving-average encoder, LeWM jointly optimizes its encoder and predictor.

A central design choice in LeWM is its anti-collapse regularization. In addition to the next-embedding mean-squared prediction loss, LeWM applies the Sketched-Isotropic-Gaussian Regularizer (SIGReg), which encourages the empirical latent distribution to match an isotropic Gaussian. The resulting model is particularly useful for our study because it represents a recent example of a fully end-to-end JEPA in which both the visual representation and dynamics predictor are shaped by the same offline trajectories.

Sensorimotor World Model (SMWM). The Sensorimotor World Model (Ivashkov et al., 2026) is also an end-to-end JEPA, but uses a different principle to prevent representation collapse. Rather than prescribing the marginal geometry of the latent representation, SMWM introduces an inverse dynamics model that predicts the executed action from consecutive latent states. The encoder, forward dynamics model, and inverse model are optimized jointly using forward-prediction and inverse-dynamics losses.

The inverse objective requires consecutive representations to preserve information that is useful for recovering the action that produced the transition. Consequently, the learned representation is explicitly biased toward controllable, action-relevant structure. Architecturally, SMWM adopts the LeWM encoder and transformer forward predictor, including AdaLN-zero action conditioning, while replacing Gaussian latent regularization with inverse dynamics as its anti-collapse mechanism. This makes SMWM particularly relevant to our benchmark: it provides a strong test of whether explicitly encouraging action-aware representations is sufficient to prevent failures induced by goal-biased state–action coverage.

DINO World Model (DINO-WM). DINO-WM (Zhou et al., 2025) separates representation learning from dynamics learning. Instead of jointly learning an image encoder from the offline control dataset, it uses pretrained DINOv2 patch features as the observation representation and keeps the DINOv2 encoder frozen during both world-model training and evaluation. A causal ViT-based transition model then predicts future patch representations from a history of latent observations and actions.

DINO-WM also differs in how actions enter the predictor. The action is first mapped through an MLP and the resulting action representation is concatenated to each spatial patch representation before the transformer processes the latent state. Thus, compared with LeWM and SMWM, DINO-WM provides a useful separation between perception and dynamics: the visual representation is inherited from large-scale pretraining, while only the action-conditioned transition model is learned from the potentially goal-biased offline trajectories. This lets us test whether combinatorial failures persist even when the visual encoder itself is not learned from the biased task dataset.

DriftWorld. DriftWorld (Lu et al., 2026) represents a substantially different, reconstruction-based family of world models. It is an action-conditioned generative model based on drifting rather than a JEPA or iterative diffusion model. Its core generator is a spatial–temporal U-Net that receives observation history, Gaussian noise, and a future action sequence and generates the corresponding future visual observations. Actions condition the U-Net frame-by-frame through FiLM, such that each future frame is explicitly conditioned on its corresponding action.

For relatively simple simulated domains such as PushT, DriftWorld operates directly in pixel space and does not require a learned visual state encoder between the observations and the action-conditioned generator. Its drifting objective moves generated samples toward ground-truth future observations and away from generated negatives, enabling future video generation in a single forward pass at inference time. DriftWorld therefore provides an important contrast to the latent JEPA

models: prediction and visual generation are implemented by a single action-conditioned generative backbone rather than an explicit encoder followed by a latent dynamics predictor.

IRASim. IRASim (Zhu et al., 2024) represents the latent generative end of our model suite. It formulates world modeling for robot manipulation as trajectory-conditioned video generation. Frames are first compressed using a pretrained VAE, after which a Diffusion Transformer (DiT) models future video latents conditioned on historical observations and a robot action trajectory. The generated latents are subsequently decoded back to RGB observations.

A defining component of IRASim is its fine-grained action conditioning. Rather than conditioning the complete video only on a global trajectory embedding, IRASim can associate individual actions with their corresponding generated frames through frame-level adaptive normalization. This architecture gives us a generative model that is structurally different from DriftWorld: visual prediction occurs in an encoded latent space, the dynamics generator is a diffusion transformer, and an explicit encoder–decoder surrounds the dynamics model.

F.2 Predictor-Only Models from True States

End-to-end evaluation mixes perception and dynamics-prediction errors. To isolate the latter, we use predictor-only models that replace the visual encoder with oracle states from simulator metadata and proprioceptive signals. These oracle states contain task-relevant physical variables, such as positions, orientations, agent or gripper configurations, and other environment-specific state factors needed for one-step prediction.

We refer to these inputs as oracle state representations rather than unique or minimal Markov states. Their role is operational: they provide the predictor with relevant physical information directly, removing visual representation learning as the explanation for prediction failure. We instantiate three predictor families derived from the end-to-end models above. All predictor-only models are trained on the same base datasets using a common next-state mean-squared-error objective,

$$\mathcal{L}_{\text{pred}} = \|\hat{s}_{t+1} - s_{t+1}\|_2^2 . \tag{16}$$

AdaLN Transformer Predictor. Our first predictor follows the forward-model architecture shared by LeWM and SMWM. Ground-truth state factors replace the learned visual embedding, while the action is injected into the transformer through AdaLN-zero. The transformer predicts the next true state directly. Because LeWM and SMWM use the same forward-predictor family, a single predictor-only architecture represents both models in this controlled setting. Their end-to-end distinction—SIGReg versus inverse-dynamics representation regularization—is intentionally removed once the encoder is replaced by the oracle state.

Action-Duplicated Transformer Predictor. Our second predictor follows DINO-WM's transition-model conditioning strategy. The true state replaces DINO patch features, while an embedded action is attached to the state representation before processing by a causal transformer. This preserves the central architectural distinction of DINO-WM's predictor: action information enters through the predictor input rather than modulating transformer layers through AdaLN. The model is again trained only to regress the next true state with MSE.

DriftWorld-Style Generative Predictor. Our third predictor is derived from the action-conditioned generative backbone of DriftWorld. Because DriftWorld does not require a separate visual encoder for the simple simulated setting, its U-Net constitutes the principal predictive computation. For the predictor-only experiment, we use a reduced-size version of this backbone and replace pixel observations and outputs with the corresponding true-state representation. To make the comparison focus on state-transition learning rather than the original drifting generative objective, this predictor is also trained with the same next-state MSE loss as the transformer predictors.

F.2.1 Oracle States

Oracle-state predictors replace RGB observations with task-relevant physical state variables from each base environment.

Table 5: Oracle state variables used by predictor-only models for each base environment.

Base environment	Dim.	Oracle state variables
TwoRoom	2D	Agent position (x, y).
Push-T	10D	Goal position (x, y), goal orientation $(\sin\theta, \cos\theta)$, block position (x, y), block orientation $(\sin\theta, \cos\theta)$, and agent position (x, y).
Cube	15D	End-effector position (x, y, z), end-effector yaw $(\cos\theta, \sin\theta)$, gripper opening, gripper contact, gripper velocity, block position (x, y, z), and block quaternion (w, x, y, z).
Manipulator Insert-Peg	33D	Arm joint positions, touch sensors, hand pose (x, z, q_w, q_y), peg pose (x, z, q_w, q_y), and target pose (x, z, q_w, q_y).

F.2.2 Model Architecture Details

For LeWM (Maes et al., 2026b) and Sensorimotor World-Model (Ivashkov et al., 2026), we use the authors' official implementations. For DINO-WM (Zhou et al., 2025), we use the Stable-WorldModel (Maes et al., 2026a) implementation without proprioceptive inputs. For DriftWorld (Lu et al., 2026), we use the authors' implementation with two minor modifications: we integrate frame skip into the action-conditioning interface and add a pixel-space cost function for MPC planning, without changing the main architecture. We use the Push-T model configuration for TwoRoom and Push-T, and the Robomimic model configuration for the Cube and Manipulator environments. For IRASim (Zhu et al., 2024), we use the authors' implementation with the same frame-skip integration and pixel-space planning cost, and use IRASim-XL/2 for all environments.

For the oracle-state experiments, we use three toy predictor families derived from the corresponding end-to-end modeling motifs. Toy-AdaLN keeps the LeWM transformer size unchanged, replacing only the pixel-latent input and output with true-state projections. Toy Action-Duplicate preserves the DINO-WM-style action duplication, causal masking, and shifted prediction design, but adapts it to scalar true-state tokens with a smaller 192D predictor instead of the original high-dimensional patch-token predictor. Toy-DriftWorld preserves the stochastic particle-generation objective and drifting loss, but replaces the image U-Net with a compact vector-valued residual FiLM MLP operating on oracle states.

G Training Details

AdaLN predictors and Action duplicated predictors use AdamW, batch size 256, weight decay 10^{-3}, linear warmup followed by cosine decay, gradient clipping at 1.0, and an MSE loss over all shifted state predictions. Toy-DriftWorld is trained with AdamW, batch size 128, learning rate 10^{-4}, weight decay 10^{-2}, a 500-step warmup followed by a constant schedule, gradient clipping at 2.0, and the drifting particle loss. For end-to-end world models, we follow the original settings.

H Detailed Planning and Evaluation Configuration

All planning experiments use the same MPC evaluation pipeline based on the Stable World Model library Maes et al. (2026a), with Cross-Entropy Method (CEM) solver.

H.1 Planning Evaluation Criteria

Table 8 summarizes the success criteria used for each environment. Some details are below.

TwoRoom. For TwoRoom, the goal is the dataset future agent position. A rollout is successful when $\|p_t - p_g\|_2 < 16$, where p_t is the current agent position and p_g is the goal agent position. Distances are measured in pixels in the 224×224 environment coordinate system. Target and door positions are not used.

Table 6: Training details and selected checkpoints for end-to-end world models.

Dataset	Model	LR	Configured duration	Selected checkpoint
Push-T	LeWM	-	-	official LeWM checkpoint
Push-T	Sensorimotor-WM	1×10^{-4}	10 epochs	epoch 10
Push-T	DINO-WM	-	-	official LeWM checkpoint
Push-T	DriftWorld	5×10^{-4}	max 500k steps	step 500k
Push-T	IRASim	5×10^{-5}	max 300k steps	step 300k
Cube	LeWM	-	-	official LeWM checkpoint
Cube	Sensorimotor-WM	1×10^{-4}	10 epochs	epoch 10
Cube	DINO-WM	-	-	official LeWM checkpoint
Cube	DriftWorld	5×10^{-5}	300-epoch cap	step 100k
Cube	IRASim	5×10^{-5}	max 300k steps	step 300k
TwoRoom	LeWM	-	-	official LeWM checkpoint
TwoRoom	Sensorimotor-WM	1×10^{-4}	10 epochs	epoch 10
TwoRoom	DINO-WM	-	-	official LeWM checkpoint
TwoRoom	DriftWorld	1×10^{-4}	300-epoch cap	step 500k
TwoRoom	IRASim	5×10^{-5}	max 300k steps	step 300k
Insert-Peg	LeWM	5×10^{-5}	10 epochs	epoch 10
Insert-Peg	Sensorimotor-WM	1×10^{-4}	10 epochs	epoch 10
Insert-Peg	PreJEPA	5×10^{-4}	10 epochs	epoch 10
Insert-Peg	DriftWorld	5×10^{-5}	max 100k steps	step 100k
Insert-Peg	IRASim	5×10^{-5}	max 300k steps	step 300k

Table 7: Training details for the selected oracle-state predictor checkpoints.

Dataset	Model	selected LR	Configured duration	Selected checkpoint
Push-T	Toy-AdaLN	3×10^{-4}	100 epochs	epoch 100
Push-T	Toy-Action-Duplicate	3×10^{-4}	100 epochs	epoch 100
Push-T	Toy-DriftWorld	1×10^{-4}	50 epochs, max 500k steps	step 500k
Cube	Toy-AdaLN	3×10^{-4}	100 epochs	epoch 100
Cube	Toy-Action-Duplicate	3×10^{-4}	100 epochs	epoch 100
Cube	Toy-DriftWorld	1×10^{-4}	50 epochs, max 500k steps	step 500k
TwoRoom	Toy-AdaLN	3×10^{-4}	100 epochs	epoch 100
TwoRoom	Toy-Action-Duplicate	3×10^{-4}	100 epochs	epoch 100
TwoRoom	Toy-DriftWorld	1×10^{-4}	50 epochs, max 500k steps	epoch 50
Insert-Peg	Toy-AdaLN	3×10^{-4}	100 epochs	epoch 100
Insert-Peg	Toy-Action-Duplicate	3×10^{-4}	100 epochs	epoch 100
Insert-Peg	Toy-DriftWorld	1×10^{-4}	50 epochs, max 500k steps	epoch 50

PushT. For PushT, the position error is computed over the 4D vector $[\text{agent}_x, \text{agent}_y, \text{block}_x, \text{block}_y]$. A rollout is successful when both $\|s_t^{\text{pos}} - s_g^{\text{pos}}\|_2 < 20$ and $d_{\text{angle}}(\theta_t, \theta_g) < \pi/9$ hold at the same timestep. The angle distance accounts for object symmetries: square and + shapes use 90° symmetry, `Z` and `I` shapes use 180° symmetry, and other shapes use 360°.

Cube. For Cube, the goal is defined by the dataset future end-effector pose and cube position. A rollout is successful when $\|p_t^{ee} - p_g^{ee}\|_2 \leq 0.04\,\text{m}$, $|\,\text{wrap}(\psi_g^{ee} - \psi_t^{ee})| \leq 0.10\,\text{rad}$, and $\|p_t^{cube} - p_g^{cube}\|_2 \leq 0.04\,\text{m}$ all hold at the same timestep. Cube orientation, gripper opening, contact, and velocity are not used. Once satisfied, episode-level success remains true. Cube rollouts are not terminated early by this criterion.

Table 8: Planning success criteria. Goals are defined by the dataset future state 25 raw steps after the initial state. An episode is successful if the criterion is satisfied at least once within 50 environment steps.

Environment	Planning success criterion
TwoRoom	The 2D Euclidean distance between the agent position and the goal agent position is less than 16 pixels. Target and door positions are not used.
PushT	The 4D Euclidean error over agent and block positions is less than 20 pixels, and the block angle error is less than $\pi/9$ radians. The angle error accounts for object symmetries.
Cube	The rollout must simultaneously satisfy: end-effector 3D position error ≤ 0.04 m, end-effector yaw error ≤ 0.10 rad, and cube 3D position error ≤ 0.04 m. Cube orientation, gripper opening, contact, and velocity are not used.
Insert Peg	The rollout must simultaneously satisfy: hand 2D position error ≤ 0.04 m, hand planar angle error ≤ 0.10 rad, peg 2D position error ≤ 0.04 m, and peg planar angle error ≤ 0.10 rad. Arm joints, touch, velocity, and the physical target or slot pose are not used.

Manipulator Insert Peg. For Manipulator Insert Peg, the goal is defined by the dataset future hand and peg poses. Positions are measured in the (x, z) plane, and planar angles are computed from quaternion components (q_w, q_y) as $2\,\mathrm{atan2}(q_y, q_w)$. A rollout is successful when the hand position, hand angle, peg position, and peg angle errors are all within their thresholds: $0.04\,\mathrm{m}$ for positions and $0.10\,\mathrm{rad}$ for angles. Arm joints, touch sensors, velocity, and the physical target or slot pose are not used. This dataset-goal success metric differs from the native task success based on the DeepMind Control shaped reward.

H.2 Configuration

We evaluated all true-state dynamics models using the same goal-conditioned receding-horizon controller. For each episode, the initial state and goal were sampled from the same dataset trajectory with a temporal offset of 25 environment steps. Planning was performed in the normalized action space using the cross-entropy method (CEM). Each candidate consisted of five action blocks, where each block contained five consecutive environment actions, yielding an effective planning horizon of 25 environment steps. At every planning round, CEM evaluated 300 candidate sequences for 30 iterations and refitted a factorized Gaussian distribution to the 30 lowest-cost elites. The objective was the squared Euclidean distance between the terminal predicted state and the goal in the model's normalized state space. The final mean sequence was executed open-loop for all five action blocks before replanning. Episodes were evaluated for at most 50 environment steps, corresponding to at most two planning rounds.

I Results

I.1 Full Results

This section reports the full numerical results used in the main analysis. We include absolute MPC planning success rates complementing the relative comparisons shown in the main text, as well as table of relative prediction errors with absolute base-environment rollout errors. For the five-step prediction error, corresponding to 25 environment steps, Avoid-T is omitted because its episodes are too short to support this horizon.

I.2 Factor Support versus Observation Support

Our compositional shifts are defined at the level of physical factors: test queries recombine factor values that are individually observed in the training data. However, this does not imply that the resulting observations are in-distribution for an end-to-end encoder.

Table 9: Absolute training MSE on base environments at 5- and 25-step prediction horizons.

Base-Env	Horizon	LeWM	SMWM	DINO-WM	DriftWorld	IRASim	AdaLN	Action Dup.	Toy-Drift
TwoRoom	5	0.065188	0.000058	0.033841	0.000154	0.003838	0.00000466	0.0000144	0.000051
	25	0.511843	0.000754	0.23277	0.002164	0.023646	0.000145	0.000636	0.00111671
PushT	5	0.00743	0.02933	0.05124	0.001331	0.005813	0.000005	0.000028	0.000066
	25	0.16739	0.23039	0.48649	0.018599	0.050163	0.000229	0.001068	0.00388751
Insert Peg	5	0.000717	0.002913	0.172737	0.001448	0.00939	0.003526	0.009356	0.054486
	25	0.010678	0.024729	1.410328	0.013113	0.0593	0.053553	0.149251	0.404549
Cube	5	0.00431	0.00203	0.15249	0.00095	0.01027	0.000056	0.000045	0.0521936
	25	0.07875	0.0193	0.89757	0.011023	0.072107	0.000486	0.000364	0.458214

Table 10: Base-relative one-step (5 environment step) prediction error. Values report the error ratio with respect to the corresponding base task.

		End-to-End World Models					Oracle State Predictors		
Env	**Variant**	**LeWM**	**SMWM**	**DINO-WM**	**DriftWorld**	**IRASim**	**AdaLN**	**Action Dup.**	**Toy-Drift**
TwoRoom	TR-Route	14.56	19.11	6.56	0.012	0.9	0.045	1.03	0.014
PushT	AvoidT	4.14	5.2	5.77	0.71	0.77	1.26	1.25	1.32
	StackT	11.33	3.13	3.38	2.54	1.36	501.14	113.55	70.24
	Anti-goal	6.23	2.66	2.83	1.64	1.09	54.74	12.85	19.96
Insert Peg	Avoid-Peg	6.83	2.54	1.3	7.01	7.1	1.45	0.62	0.08
	Extract-Peg	4.23	2.29	0.98	6.13	6.38	33.67	12.54	2.21
Cube	CubeHop	27.83	93.7	2.8	1.38	1.19	91.94	1.03	0.058
	Cube Diagonal	1.71	5.8	1.12	1.09	0.97	940.34	1026.48	0.58

Table 11: Base-relative five-step (25 environment step) prediction error. Values report the error ratio with respect to the corresponding base task.

		end-to-end world models					oracle state predictors		
Env	**Variant**	**LeWM**	**SMWM**	**DINO-WM**	**DriftWorld**	**IRASim**	**AdaLN**	**Action Dup.**	**Toy-Drift**
TwoRoom	TR-Route	11.2	41.6	9.66	0.01	0.82	0.062	0.816	0.01
PushT	StackT	9.83	4.56	3.8	2.34	1.44	158.5	47.57	22.09
	Anti-goal	8.49	4.27	3.94	1.72	1.28	29.97	12.68	8.96
Insert Peg	Avoid-Peg	9.73	16.69	2.8	1.14	1.12	0.67	0.42	0.09
	Extract-Peg	2.88	8.99	1.53	1.48	1.01	21.95	9.72	2.89
Cube	CubeHop	29.64	139.07	4.12	1.45	1.37	245.45	3.47	0.102
	Cube Diagonal	1.55	11.38	1.14	1.05	0.97	841.69	1202.22	0.46

Let $\mathcal{Z}^{\mathrm{tr}}$ denote the joint support of physical factors under the training trajectories, and let $\mathcal{Z}^{\times} = \prod_{k=1}^{K} \mathcal{Z}_k^{\mathrm{tr}}$ denote the Cartesian product of their marginal supports. A compositional query may satisfy $\mathcal{Z}^{\mathrm{te}} \subseteq \mathcal{Z}^{\times}$ while still containing factor combinations absent from $\mathcal{Z}^{\mathrm{tr}}$.

End-to-end world models observe $s = f(z)$, where f is the rendering map. Thus the corresponding observation supports are $\mathcal{S}^{\mathrm{tr}} = f(\mathcal{Z}^{\mathrm{tr}})$ and $\mathcal{S}^{\times} = f(\mathcal{Z}^{\times})$. When f mixes factors, unseen factor combinations in $\mathcal{Z}^{\times} \setminus \mathcal{Z}^{\mathrm{tr}}$ can induce observations in $\mathcal{S}^{\times} \setminus \mathcal{S}^{\mathrm{tr}}$.

Therefore, a query can be factor-wise in support while still creating an observation-level shift for an end-to-end encoder. This is why we separately analyze oracle-state predictors: they remove the encoder-inference problem and isolate whether the learned dynamics model can generalize across recombined state-action factors.

Table 12: Absolute planning success rate (%) across different models and tasks.

Task	LeWM	SMWM	DINO-WM	DriftWorld	IRASim	AdaLN	Action Dup.	Drift Toy
TwoRoom	89.30	100.00	100.00	90.00	76.00	100.00	99.30	100.00
TR-Route	60.70	64.70	68.00	76.00	63.30	100.00	100.00	100.00
Push-T	84.70	84.70	74.00	43.30	5.30	92.70	92.00	89.30
AvoidT	3.30	4.70	6.70	0.70	0.00	100.00	100.00	99.30
StackT	20.70	20.00	8.70	10.70	2.00	70.70	66.70	54.00
Push-T Antigoal	34.00	44.00	27.30	14.70	5.30	79.30	81.30	75.30
Insert-Peg	26.67	70.00	35.33	63.33	2.43	74.00	71.33	71.33
Avoid-Peg	31.33	32.67	28.67	68.00	34.00	92.00	89.33	80.67
Extract-Peg	39.33	53.33	46.67	52.67	46.00	52.00	52.67	50.67
Cube	30.70	72.70	60.00	30.70	10.70	30.00	20.70	25.30
Cube-Hop	17.30	40.70	51.30	64.70	12.70	95.30	51.30	79.30
Cube-Diagonal	28.00	53.30	58.70	42.00	9.33	20.00	12.00	22.00

I.3 Additional Observations

Prediction Space Affects Metric Interpretation We do not use the rollout results to directly rank JEPA-style and generative world models. These models predict in different spaces: JEPA-style models are evaluated in latent space, whereas generative models are evaluated in pixel or reconstruction space. As a result, absolute prediction errors are not directly comparable across model families, and we primarily interpret rollout degradation relative to each model's own base-task performance.

Under this relative comparison, generative models often show smaller rollout degradation under some goal shifts, while several JEPA-style models show larger increases. However, this tendency does not translate consistently to planning: a model can show stable rollout error but poor MPC success, or larger rollout degradation while remaining competitive in control. We therefore treat prediction-space effects as an interpretation caveat rather than a model-family ranking. The results suggest that rollout robustness and downstream control utility are related but distinct properties.

Oracle state access does not guarantee better planning. Interestingly, access to the oracle state does not always lead to stronger planning performance. In Cube, several end-to-end models achieve higher planning success than their oracle-state counterparts, both on the base task and on CubeDiagonal. This should not be interpreted as evidence that learned latents are generally superior to physical state representations, since the end-to-end and oracle predictors differ in more than just their input space.

Rather, these results suggest that state observability and planning suitability are distinct properties. Oracle states preserve the underlying physical factors, but the resulting geometry may not be the most favorable one for the planner's cost and optimization procedure. Conversely, a learned representation may, in some cases, emphasize task-relevant variation or compress nuisance factors in a way that makes goal-directed optimization easier. We therefore treat planning performance as depending not only on how much state information is available, but also on how that information is organized in the representation space.

J Toward Compositional Generalization in World Models

Our experiments empirically show that removing visual abstraction is not sufficient for reliable goal-shift generalization: free-space dynamics often transfer across new state–action compositions, whereas contact-rich dynamics remain substantially harder even when the predictor receives task-relevant physical state directly. This raises a dynamics-level question: what structure must a world model learn for its dynamics to remain reusable under compositional shifts?

Ideally, we want to solve AnyGoal by learning reusable dynamics that extrapolate to the combinatorial space. That is, we want a learned transition function $\hat{T}$ that matches the observed training transitions on $\mathcal{C}^{\mathrm{tr}}$, while also predicting correctly on novel compositions in $\mathcal{C}^{\times} \setminus \mathcal{C}^{\mathrm{tr}}$. We separate this question into two parts: identifying reusable mechanisms on the training support, and extrapolating those mechanisms to novel compositions.

Note. The discussion below should be read as a conditional statement, not a claim that reusable mechanisms are identifiable from expert rollouts alone. Rather, we give one possible sufficient condition: if sparse regime-conditioned local mechanisms are identified on training support and test goals query only locally covered active inputs, then the model can generalize to globally novel state–action compositions.

J.1 Identifying Reusable Mechanisms on the Training Support

To discuss identifiability, we first need to specify what structure the true transition may have. One observation from our results is that world models often face multiple *regimes*, such free motion, contact, grasping, or collision.

In a single-regime setting with little interaction, such as free-space navigation of an agent, one may expect a transition of the form $z_{t+1} = f(z_t) + g(a_t)$. In this favorable case, identifiability and compositional generalization are easier to obtain, for example through an additive function class (Dong & Ma, 2022; Lachapelle et al., 2023; Wiedemer et al., 2024b;a; Mahajan et al., 2025).

However, world-model dynamics are often regime-dependent and more complex than a purely additive and separated form. The action effect is usually better written as $g_r(z_t, a_t)$ rather than $g(a_t)$, meaning that the transition cannot be decomposed into simple independent state-only and action-only terms. A simple additive form is therefore too limited for the setting we want to characterize.

We instead consider a factorized transition with regime-dependent local mechanisms. For each factor k, let

$$z_{k,t+1} = T_{k,r_t}(u_{k,t}), \qquad u_{k,t} = \big(z_{\mathrm{pa}(k),t}, a_{\mathrm{pa}(k),t}\big),$$

where $\mathrm{pa}(k)$ denotes the local parents of factor k, and $r_t \in \{1, \dots, R\}$ denotes a discrete local regime. The transition kernel is specified as follows

$$T_{r_t}(z_t, a_t) = \big(T_{1,r_t}(u_{1,t}), \dots, T_{K,r_t}(u_{K,t})\big).$$

Since multiple regime–mechanism decompositions can explain the same training transitions, we next state the favorable assumptions under which the discussion applies: a Markov, goal-invariant oracle state, and training-support identification of the relevant local mechanisms up to observational equivalence.

Assumption 1 (Markov and invariant oracle state) *The oracle state z_t is Markov, and the transition law is invariant across training and test goals:*

$$p(z_{t+1} \mid z_{\le t}, a_{\le t}) = p(z_{t+1} \mid z_t, a_t), \qquad T_{\mathrm{tr}}(z_{t+1} \mid z_t, a_t) = T_{\mathrm{te}}(z_{t+1} \mid z_t, a_t).$$

Thus, goal shifts change the behavior distribution, but not the environment dynamics.

Assumption 2 (Identification of local mechanisms) *For each state factor k, suppose that the interaction regime r_t and the local input $u_{k,t} = u_k(z_t, a_t)$ are observed or identifiable from the training transitions, and that the corresponding regime can be consistently inferred at prediction time.*

$$z_{k,t+1} = T_{k,r_t}(u_{k,t}), \qquad \hat{z}_{k,t+1} = \hat{T}_{k,r_t}(u_{k,t}).$$

For each factor and regime, define the training support

$$\mathcal{U}^{\mathrm{tr}}_{k,r} := \bigcup_{t=0}^{H-1} \mathrm{supp}_{p_{\mathrm{tr}}}(U_{k,t} \mid R_t = r).$$

Let $\mathcal{L}_{\mathrm{tr}}(\tilde{T})$ denote the population objective used to train the world model. We assume that this objective identifies the regime-conditioned mechanisms on their training support: any population minimizer $\tilde{T}$ satisfies

$$\tilde{T}_{k,r}(u) = T_{k,r}(u), \qquad \forall u \in \mathcal{U}^{\mathrm{tr}}_{k,r},$$

for every k, r, up to observational equivalence and regime relabeling.

For a model $\hat{T}$ learned from finite data, we allow estimation and optimization error

$$\epsilon_{k,r} := \sup_{u \in \mathcal{U}^{\mathrm{tr}}_{k,r}} \|\hat{T}_{k,r}(u) - T_{k,r}(u)\|_2,$$

and assume $\epsilon_{k,r} \to 0$ as the amount of relevant training data grows and the training objective is optimized consistently.

J.2 Extrapolating Identified Mechanisms to Novel Compositions

Training-support identification does not by itself imply compositional generalization. Two mechanisms can agree on all observed training transitions and still disagree on a novel state–action composition induced by a goal shift. Recombination therefore requires additional structure that separates coordinates that affect a mechanism from coordinates that are only part of the global state.

Additive or separable functions provide one favorable case, which allow us to reuse the learned mechanisms for each factor $z_{k,t+1}$, but contact-rich world-model dynamics are often regime-dependent rather than purely additive. We propose a simpler scenario where each transition mechanisms T_{k,r_t} depends sparsely on the inputs s_t, a_t. For a fixed regime r, let $x_t = (z_t, a_t)$, then

$$z_{k,t+1} = T_{k,r_t}(x_t).$$

Sparsity of $\nabla_x T_{k,r_t}(x)$ implies that only a subset of coordinates affects the k-th factor transition, while the remaining coordinates may change the global composition without changing this local mechanism.

Assumption 3 (Sparse local mechanisms) *For each factor k and regime r, there exists a fixed coordinate set $P_k \subseteq \{1, \dots, \dim(x)\}$, with $|P_k| \ll \dim(x)$, such that*

$$T_{k,r}(x) = T_{k,r}(x_{P_k}).$$

Equivalently,

$$\frac{\partial T_{k,r}(x)}{\partial x_j} = 0 \qquad \forall j \notin P_k.$$

We write $u_{k,t} = x_{P_k,t}$ for the inputs relevant to the k-th transition mechanism.

Sparse local mechanism can allow us to create favorable cases when a goal shift creates a novel *global* composition without creating novel *local* mechanism inputs.

Assumption 4 (Local support coverage) *Let*

$$\mathcal{U}^{\mathrm{te}}_k := \bigcup_{t=0}^{H-1} \mathrm{supp}_{p_{\mathrm{te}}}(U_{k,t})$$

denote the test support of the inputs of transition mechanism for factor k. For every factor k, assume

$$\mathcal{U}^{\mathrm{te}}_k \subseteq \mathcal{U}^{\mathrm{tr}}_k.$$

Thus, the global state–action composition may be novel, but every local mechanism is queried only on local inputs covered during training.

> **Intuition.** Within a fixed regime, sparse local support separates active & inactive coordinates. The active coordinates u_k determine the transition mechanism for the factor z_k, while coordinates outside u_k can be recombined to create globally novel state–action compositions without changing that local mechanism. Thus, the sparser the mechanism, the larger the set of globally novel compositions that can be covered by the same local transition rule.

Proposition 1 (Local active-coordinate coverage controls global error) *Let $\mathcal{L}_{\mathrm{tr}}(\tilde{T})$ denote the learning objective used to train the world model. Suppose that we perfectly optimize the world model on training support, then under assumption 1 and 2, we identify the learned model such that the true transition and learned transition share the same regime-conditioned local decomposition,*

$$T(z_t, a_t) = \big(T_{1,r_t}(u_{k,t}), \dots, T_{K,r_t}(u_{k,t})\big),$$

and

$$\hat{T}(z_t, a_t) = \big(\hat{T}_{1,r_t}(u_{k,t}), \ldots, \hat{T}_{K,r_t}(u_{k,t})\big).$$

Assume that, for every factor k and regime r, we satisfy the assumption of sparse local mechanisms (assumption 3) and local support coverage (assumption 4).

Then, for any test composition $(z_t, a_t) \in \mathcal{C}^{\text{te}}$,

$$\left\|\hat{T}(z_t, a_t) - T(z_t, a_t)\right\|_2 \leq \left(\sum_{k=1}^{K} \epsilon_{k,r_t}^2\right)^{1/2}.$$

where

$$\epsilon_{k,r} := \sup_{u \in \mathcal{U}_k^{\text{tr}}} \left\|\hat{T}_{k,r}(u) - T_{k,r}(u)\right\|_2.$$

Consequently, if $\epsilon_{k,r} \to 0$ for all k, r, then the world model generalizes to novel combinations present in the test support.

Proof. Take any novel $(z_t, a_t) \in \mathcal{C}^{\text{te}}$ such that $(z_t, a_t) \notin \mathcal{C}^{\text{tr}}$. For each factor k, using the local support coverage assumption (assumption 4) we have the following.

$$u_{k,t} \in U_k^{\text{te}} \implies u_{k,t} \in U_k^{\text{tr}} \quad \text{since} \quad \mathcal{U}_k^{\text{te}} \subseteq \mathcal{U}_k^{\text{tr}}.$$

Therefore, we are evaluating each mechanism on its training support. By Assumption 2,

$$\left\|\hat{T}_{k,r_t}(u_{k,t}) - T_{k,r_t}(u_{k,t})\right\|_2 \leq \epsilon_k.$$

Hence, the combined error in prediction of all the K components is given as follows.

$$\begin{aligned}\left\|\hat{T}(z_t, a_t) - T(z_t, a_t)\right\|_2^2 &= \sum_{k=1}^{K} \left\|\hat{T}_{k,r_t}(u_{k,t}) - T_{k,r_t}(u_{k,t})\right\|_2^2 \\ &\leq \sum_{k=1}^{K} \epsilon_k^2.\end{aligned}$$

Taking the square root gives the result.

Discussion. This construction should be read as an illustrative sufficient condition rather than a complete solution. Sparse local mechanisms reduce a high-dimensional global coverage problem to smaller local coverage problems, but they do not by themselves solve mechanism identification, regime discovery, or extrapolation to genuinely new local interactions. The broader challenge is to develop world-model architectures, objectives, and data-collection strategies that identify reusable action-conditioned mechanisms and impose enough structure on their interactions to support generalization beyond expert rollouts.