ACWM-PHYS: INVESTIGATING GENERALIZED PHYSICAL INTERACTION IN ACTION-CONDITIONED VIDEO WORLD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Action-conditioned world models (ACWMs) have shown strong promise for video prediction and decision-making, yet existing benchmarks are largely restricted to egocentric navigation or narrow robot manipulation, covering only a small part of the physical interactions a general world model must capture. We introduce **ACWM-Phys**, a simulated benchmark spanning four physical regimes (rigid-body, deformable, particle, and kinematic dynamics) across eight environments, each paired with a controlled out-of-distribution (OoD) split that shifts a single physical factor so failures are attributable. Alongside it we release ACWM-DiT, a latent video diffusion transformer baseline. Because every environment is simulated, we score rollouts not only with perceptual metrics but directly against physics: point-track error, conservation of simulator-conserved quantities, and a zero-action counterfactual. Both families of metrics agree on the same trend: OoD degradation is governed by task complexity rather than physics category, with low-dimensional geometric tasks nearly unaffected and contact-rich deformation and high-DoF kinematics degrading sharply. A second, architecturally distinct model post-trained from a pretrained video foundation model shows the same degradation, indicating the gap is not an artifact of our backbone. The physics metrics additionally expose failures that perceptual metrics miss, such as mass that is not conserved under distribution shift. We release ACWM-Phys as a diagnostic instrument for physically grounded world modeling.

1 INTRODUCTION

Action-conditioned world models (ACWMs) have emerged as a promising paradigm for learning predictive models of the physical world from raw visual observations and control signals [6, 11, 13, 25, 4]. By directly forecasting future observation sequences conditioned on agent actions, ACWMs hold the potential to serve as general-purpose simulators for robot planning, policy learning, and data augmentation without requiring hand-crafted dynamics models. Recent advances in diffusion-based video generation [7, 16, 31] have substantially improved visual fidelity and temporal consistency of generated sequences, further fueling interest in pixel-space world models that operate directly on high-dimensional observations.

Despite this progress, existing ACWMs and their accompanying benchmarks suffer from a critical blind spot: *physical diversity*. The vast majority of current work (Appendix Table 4) is either confined to egocentric navigation [25, 9, 29], where actions correspond primarily to camera motion and objects rarely deform or interact, or to narrow robot manipulation [11, 3, 13, 4] involving mostly rigid-body pick-and-place. Yet the physical world encompasses a far richer spectrum of interaction regimes: deformable objects such as ropes and cloth, granular and fluid particle systems, and complex kinematic chains, each governed by fundamentally different dynamics. It remains unclear whether current diffusion-based ACWMs can learn and generalize across these diverse interaction modes, or whether they silently fail when the underlying physics departs from the training distribution.

We address this gap with **ACWM-Phys**, a new benchmark designed to systematically evaluate ACWMs across four categories of physical interaction: rigid-body dynamics, deformable-object

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

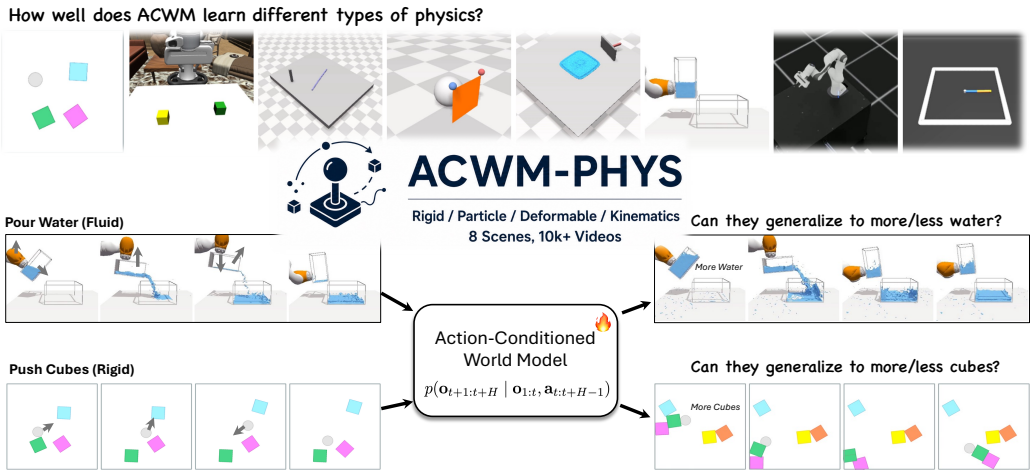

Figure 1: ACWM-Phys provides diverse physical scenes to help answer two questions: **how well can ACWMs learn different types of physics, and can they generalize beyond the training distribution?** We evaluate both in-distribution prediction and out-of-distribution generalization, such as more/fewer water particles or cubes.

dynamics, particle dynamics, and kinematics. ACWM-Phys comprises eight robotic simulation environments, each with carefully curated in-distribution (InD) training and test splits and a physically motivated out-of-distribution (OoD) test split that targets the specific generalization challenge most relevant to that environment such as unseen cloth sizes, doubled particle counts, or workspace regions excluded during training. Because every environment is fully simulated, distribution shifts are exactly reproducible and free from sensor noise, enabling clean measurement of the generalization gap.

Alongside the benchmark, we introduce **ACWM-DiT**, a latent video diffusion transformer baseline. ACWM-DiT builds on a pretrained causal video VAE [31] for compact spatiotemporal encoding and couples a bidirectional DiT backbone [26] with an action-embedding module that injects action-conditioning signals into pixel rendering. Because conclusions drawn from a single backbone are easy to dismiss as architectural artifacts, we additionally post-train Cosmos-Predict2.5-2B [1], a large pretrained video foundation model, with action conditioning on ACWM-Phys. It shows the same out-of-distribution degradation as ACWM-DiT (Section 5.3), indicating that the gap this benchmark measures is a property of the task rather than of our model, and that large-scale video pretraining alone does not confer physical generalization.

Through systematic experiments on ACWM-Phys, we make the following contributions and findings:

- We introduce ACWM-Phys, the first benchmark spanning four distinct physical interaction regimes with controlled InD/OoD evaluation protocols across eight environments.
- We design ACWM-DiT as a strong diffusion-based baseline, which achieves strong performance across all environments and establishes a solid starting point for future work.
- We introduce three physics-grounded metrics that the simulator makes available (track error, conservation error, and an action-fidelity counterfactual), and show that they *misrank relative to perceptual metrics*: PSNR can improve out-of-distribution on Cloth Move while pose error grows by 47% (Section 5.2).
- We find that OoD degradation tracks task complexity more closely than physics category: environments with low-dimensional geometric constraints (e.g., Push Cube, Reacher) degrade little, while high-DoF kinematics (Robot Arm) and contact-rich deformation (Cloth Move) degrade sharply on both pixel and physical measures.

2 Related Works

Action-conditioned World Models The idea of learning a model of the environment [5] for planning and decision-making has a long history in reinforcement learning. Recently, driven by rapid advances in diffusion-based image and video generation [7, 16, 31, 32, 12, 36], pixel-space world models have regained significant attention for generating high-quality visual predictions conditioned on actions [6, 11, 9, 25, 37]. However, most existing works focus on egocentric settings, where actions primarily correspond to navigation, such as Genie-3 [25], RELIC [9], and WorldPlay [29]. These settings involve limited direct interaction with the environment. Other works instead concentrate on narrow domains, such as robot manipulation, including Vid2World [11], BridgeV2W [3], WoVR [13], and Ctrl-World [4], or on Minecraft gameplay [27, 6]. A key limitation of these approaches is their limited investigation of complex physical interactions, as most mainly focus on simple navigation, or rigid-body dynamics such as picking, pushing, and grasping.

Physics in Video Diffusion Models Recent work has begun to investigate how well video diffusion models capture physical principles and whether they can serve as implicit world models [14, 34, 24, 39], and further align current video diffusion to certain physics scenes [35, 40, 38, 19]. These studies examine aspects such as physical law consistency [38, 19], intuitive physics [35, 20], and physical reasoning ability [40, 41] in generated videos, providing useful evidence on the current strengths and limitations of video generation models. However, most of this line of work remains centered on text-to-video (T2V) or image-to-video generation (T2V), where the model is asked to produce visually plausible dynamics from passive prompts or observations. As a result, these benchmarks primarily evaluate whether models can reflect physics in generated videos, rather than whether they can predict physically grounded futures under explicit action control. In contrast, our work focuses on the action-conditioned setting of ACWMs.

3 Background

Video Diffusion Models Video diffusion models [8, 36, 31, 32] generate videos by transforming noise into data, typically conditioned on text, images, or other context. Given a video $\mathbf{x} \in \mathbb{R}^{T\times C\times H\times W}$ and condition $\mathbf{c}$, modern flow-matching formulations [23, 22] learn a time-dependent vector field $\mathbf{v}_\theta$ that transports noise $\mathbf{x}_0 \sim p_0$ to data $\mathbf{x}_1 \sim p_{\text{data}}(\mathbf{x} \mid \mathbf{c})$ along a predefined path. In practice, this process is performed in a compressed latent space: a video encoder $\mathcal{E}$ maps $\mathbf{x}$ to $\mathbf{z} = \mathcal{E}(\mathbf{x})$, and a decoder $\mathcal{D}$ reconstructs $\mathbf{x} = \mathcal{D}(\mathbf{z})$. This reduces spatial-temporal cost and enables scalable Transformer-based denoisers such as DiT [26].

Action-Conditioned Video World Models Action-conditioned video world models extend video generation to controlled dynamics prediction. Given past observations $\mathbf{o}_{1:t}$ and future actions $\mathbf{a}_{t:t+H-1}$, the goal is to model

$$p(\mathbf{o}_{t+1:t+H} \mid \mathbf{o}_{1:t}, \mathbf{a}_{t:t+H-1}),$$

or equivalently in latent space,

$$p(\mathbf{z}_{t+1:t+H} \mid \mathbf{z}_{1:t}, \mathbf{a}_{t:t+H-1}), \qquad \mathbf{z}_t = \mathcal{E}(\mathbf{o}_t).$$

Following recent diffusion-based ACWMs [11, 2, 13], we train the model to generate future latent trajectories conditioned on observation history and actions. Let $\mathbf{z}^{\text{fut}}$ be the future latent video and $\mathbf{h}_t = \{\mathbf{z}_{1:t}, \mathbf{a}_{t:t+H-1}\}$ be the conditioning context. Under flow matching, we sample $\mathbf{z}_0 \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, set $\mathbf{z}_1 = \mathbf{z}^{\text{fut}}$, interpolate

$$\mathbf{z}_\tau = \alpha(\tau)\mathbf{z}_0 + \beta(\tau)\mathbf{z}_1,$$

and optimize

$$\mathcal{L}_{\text{ACWM}} = \mathbb{E}\left\|\mathbf{v}_\theta(\mathbf{z}_\tau, \tau, \mathbf{h}_t) - \dot{\alpha}(\tau)\mathbf{z}_0 - \dot{\beta}(\tau)\mathbf{z}_1\right\|_2^2.$$

In our settings, we only use current frame as condition. After training, future observations are generated by integrating the learned vector field from noise while conditioning on past observations and candidate actions. In this work, we focus on the end-to-end rollout setting, where the model predicts the next fixed H frames at once, while we can also do autoregressive generation like prior work [13, 4] by iteratively denoising and conditioning on clean frames (in Appendix Figure 5).

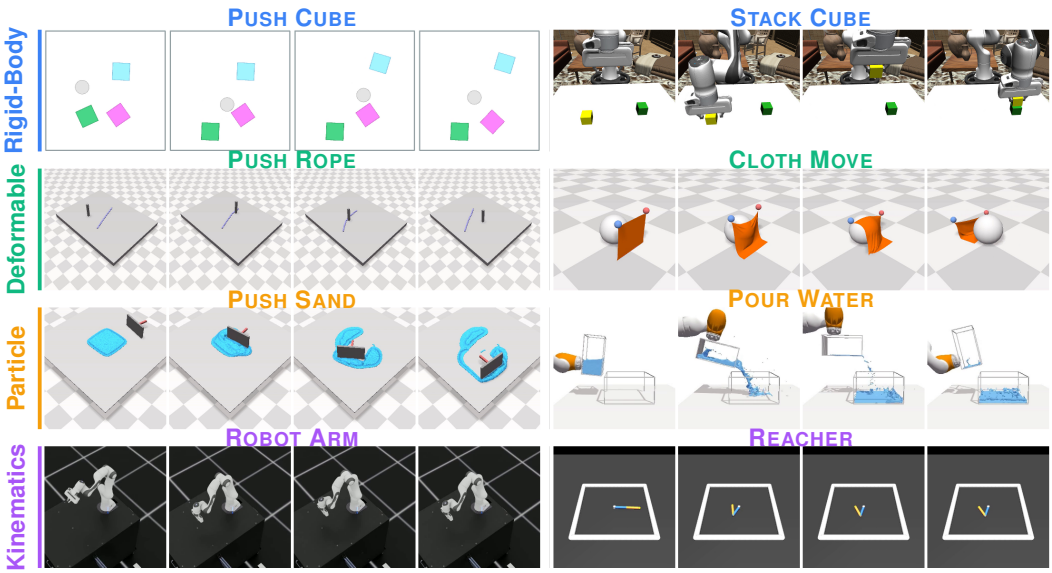

Figure 2: **ACWM-Phys dataset overview.** Four representative frames per environment across the eight tasks, grouped by physical interaction category. Each row shares a category color (left border and label): **rigid-body**, **deformable**, **particle**, and **kinematics**. Dataset statistics and action-space definitions are summarized in Appendix A.4.

4 Investigating ACWMs for Learning Generalized Physical Interactions

4.1 ACWM-Phys: A Benchmark Suite for Rich Physical Interactions

ACWM-Phys comprises eight robotic simulation environments grouped into four categories of physical interaction: rigid-body dynamics, deformable-object dynamics, particle dynamics, and kinematics (Figure 2). Each category contains two environments covering different object types, control spaces, and interaction patterns, ranging from low-dimensional pushing and reaching tasks to contact-rich deformable and particle-based dynamics. Each environment provides separate in-distribution (InD) training and test splits, as well as an out-of-distribution (OoD) test split with a controlled distribution shift along a physically meaningful axis, such as object count, workspace range, rope/cloth size, particle quantity, or goal region. In total, the benchmark contains more than 15k simulated trajectories with paired image observations, actions, and evaluation labels. Dataset sizes, action specifications, and split details are provided in Appendix A.4.

4.1.1 Categories of Physical Interactions

Rigid-Body Dynamics. `Push Cube` moves one to five colored cubes using a circular pusher, where $\mathbf{a} \in \mathbb{R}^2$ specifies the pusher's absolute 2D target position. `Stack Cube` uses a Franka Panda to place a red cube on a green cube, with $\mathbf{a} \in \mathbb{R}^7$ denoting delta 6-DoF end-effector pose plus gripper command.

Deformable-Object Dynamics. `Push Rope` uses a pole pusher to deform a flexible rope in PyFlex [21], with $\mathbf{a} \in \mathbb{R}^2$ as the pole's horizontal displacement. `Cloth Move` pushes a cloth over a fixed sphere using dual arms, with a shared 3D end-effector displacement $\mathbf{a} \in \mathbb{R}^3$; we study the full 8-D per-arm action space in the ablation.

Particle Dynamics. `Push Sand` rearranges granular material in PyFleX using a board pusher, with $\mathbf{a} \in \mathbb{R}^7$ encoding the board's 3D pose delta. `Pour Water` pours fluid by moving and tilting a cup, where $\mathbf{a} \in \mathbb{R}^4$ gives Cartesian and tilt-angle deltas from a spring-damper controller.

Kinematics. `Robot Arm` uses a 7-DoF Franka Panda in Isaac Sim with cuRobo planning, where $\mathbf{a} \in \mathbb{R}^7$ is the per-joint angle delta. `Reacher` controls a two-link MuJoCo [30] arm, with $\mathbf{a} \in \mathbb{R}^2$

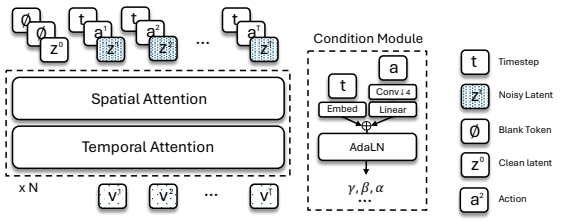

Figure 3: **ACWM-DiT architecture.** Noisy latent tokens $\mathbf{z}_{1:T_l}$ (conditioning frames at σ=0, predicted frames at diffusion step σ) are processed by N stacked DiT blocks with alternating spatial and temporal self-attention, modulated via AdaLN from a joint conditioning signal formed by summing the timestep embedding and the temporally compressed action embedding.

directly specifying joint torques. Please refer to Appendix Figure 8 for visualizations of scene rollouts.

4.1.2 In-Distribution and Out-of-Distribution Evaluation Protocols

A central design principle of ACWM-Phys is that *every* environment supports a controlled, physically motivated distribution shift between the InD and OoD splits. Rather than applying random perturbations, we shift the physical parameters or workspace regions that most directly challenge the generalization of a learned world model, detailed design of OoD scenes are in Appendx A.2:

- **Rigid**: Push Cube tests unseen cube counts; Stack Cube shifts target placement.
- **Deformable**: Push Rope changes rope length; Cloth Move varies cloth size.
- **Particle**: Push Sand increases particle count; Pour Water shifts water level.
- **Kinematics**: Robot Arm expands the goal workspace; Reacher tests unseen goal regions.

Because all environments are fully simulated, OoD shifts are exactly reproducible and free from sensor noise, enabling precise measurement of generalization gaps, unlike real-robot benchmarks [17]. Models are trained only on InD data and evaluated on both InD and OoD test trajectories. We report MSE, SSIM [10], and PSNR, and additionally use Masked-MSE (M-MSE), which computes MSE only on pixels with sufficient ground-truth temporal change, emphasizing motion-relevant regions while down-weighting static backgrounds; see Appendix A.3.

4.2 ACWM-DiT: An Action-Conditioned Video Diffusion Transformer in Latent Space

We use a DiT-based latent video diffusion model as a reproducible baseline for action-conditioned world modeling, named ACWM-DiT. Our goal is not to propose a new architecture, but to provide a strong and standardized diffusion-based baseline for diagnosing physical generalization on ACWM-Phys. As shown in Figure 3, ACWM-DiT encodes video observations with a frozen WanVAE [31] and denoises future latent tokens using a bidirectional DiT backbone with interleaved spatial and temporal self-attention and RoPE positional encoding. Actions are embedded by an MLP followed by a strided temporal convolution, which downsamples pixel-rate actions to the latent temporal resolution. The resulting action embeddings are injected into every transformer block through AdaLN as a joint action–timestep conditioning signal. The first frame is always kept clean and used as the history input, while the model predicts the remaining future frames in latent space. More architectural and training details are provided in Appendix A.1.

5 Experiments

Training Setup All models are trained seperately from scratch with the AdamW optimizer at a learning rate of 10^{-4} with gradient clipping at 1.0. We train for 100k steps with a batch size of 4 on 8 H100 GPUs for each task. The flow-matching scheduler uses 1000 noise levels with a shift parameter s=5.0 and a Gaussian weighting envelope centered at noise step 500 to focus supervision on intermediate denoising levels. Video observations are encoded by the frozen Wan 2.1 causal VAE into latent tokens of spatial resolution $H/8\times W/8$ with 16 channels and $4\times$ temporal compression. Input sequences are padded/trimmed to a fixed latent length of T_l=37 tokens. All environments are resized to 240×240 pixels prior to encoding, except Push Sand which uses 240×400 to preserve its

Category	Environment	In-Distribution Test				Out-of-Distribution Test			
		MSE↓	M-MSE↓	SSIM↑	PSNR↑	MSE↓	M-MSE↓	SSIM↑	PSNR↑
Rigid	Push Cube	2.919	11.43	0.955	25.35	2.950↓	7.47	0.954↓	25.30↓
	Stack Cube	5.516	10.93	0.889	22.58	6.996↓	13.40↓	0.872↓	21.55↓
Deformable	Push Rope	0.214	2.61	0.988	36.70	0.329↓	12.86↓	0.985↓	34.83↓
	Cloth Move	10.667	63.68	0.920	19.72	23.820↓	93.67↓	0.864↓	16.23↓
Particle	Push Sand	0.519	6.22	0.975	32.85	1.526↓	15.54↓	0.941↓	28.16↓
	Pour Water	2.630	11.81	0.911	25.80	3.488↓	14.21↓	0.874↓	24.57↓
Kinematics	Robot Arm	1.434	13.45	0.969	28.43	6.559↓	53.80↓	0.902↓	21.83↓
	Reacher	0.260	5.63	0.992	35.85	0.272↓	8.89↓	0.992	35.65↓

Table 1: **ACWM-DiT-S evaluation on ACWM-Phys** (diffusion steps = 50, 100k training steps). MSE and Masked-MSE (M-MSE) are scaled by 10^{-3}. Environment names are shaded from green to red to qualitatively indicate overall prediction difficulty/performance, where greener backgrounds denote better model performance and redder backgrounds denote harder environments with larger prediction errors. ↓ denotes OoD performance worse than InD.

landscape aspect ratio. All environments are trained with the AdaLN action-conditioning variant; cross-attention conditioning is evaluated separately in the ablation studies (Section B).

5.1 Main Results

Table 1 reports ACWM-DiT-S on all eight ACWM-Phys environments at 100k training steps and 50 inference steps. We study the effect of sampling steps in Appendix A.5.

InD performance. ACWM-DiT-S fits all four physics categories in distribution (SSIM ≥ 0.89 everywhere), so the OoD gaps below reflect generalization failure rather than an inability to model the data. Environments with simple, repetitive dynamics are predicted most faithfully, such as Push Rope (M-MSE 2.61, SSIM 0.988) and Reacher (M-MSE 5.63, SSIM 0.992), while Stack Cube (SSIM 0.889) and Cloth Move (M-MSE 63.68) are hardest: the former from large foreground arm motion, the latter because large-scale deformation produces big errors in dynamic regions even InD.

Finding 1. Generalization tracks state dimensionality, not physics category. Low-dimensional geometric dynamics such as rigid-body translation and two-link joints transfer robustly. High-dimensional or stochastic dynamics such as cloth, sand, and a 7-DoF arm expose the model's reliance on appearance statistics. Physics category alone does not predict robustness.

OoD generalization. Under distribution shift, ACWM-DiT-S shows consistent degradation, especially in motion-sensitive regions. The largest drops occur for Robot Arm (ΔM-MSE $= +40.35$, ΔSSIM $= -0.067$) and Cloth Move (ΔM-MSE $= +29.99$, ΔSSIM $= -0.056$), where unseen articulated configurations and large-scale cloth deformation introduce complex motion beyond the training distribution. This suggests that the model still relies partly on learned visual regularities rather than fully internalizing general physical dynamics.

Overall, OoD robustness is shaped by both physical complexity and action/state dimensionality. Push Cube (ΔSSIM $= -0.001$) and Reacher (ΔSSIM $= 0.000$) remain nearly stable because their dynamics follow low-dimensional geometric constraints. Push Sand shows increased motion-region error (ΔM-MSE $= +9.32$) while retaining moderate structural similarity (OoD SSIM $= 0.941$, ΔSSIM $= -0.034$), indicating difficulty in fine-grained particle redistribution. Pour Water is more stable in M-MSE (ΔM-MSE $= +2.40$), likely because the pouring trajectory is repeatable, although SSIM still drops under unseen water volumes (ΔSSIM $= -0.037$).

More per-environment case studies for all remaining environments are provided in Appendix A.7.

5.2 Physics-Grounded Evaluation

The analysis above rests on perceptual metrics, which measure how a rollout *looks* rather than whether it is physically admissible. A natural concern is whether the conclusions above survive when generalization is measured physically instead. Because ACWM-Phys is fully simu-

Category	Environment	Track error ($\times 10^3$)↓ InD	OoD	Δ%	Conservation↓ InD	OoD	Act. fid. ↑
Rigid	Push Cube	8.8	10.0	+14%	0.02	0.03	2.8×
	Stack Cube	11.6	14.9	+28%	0.03	0.05	3.1×
Deformable	Push Rope	9.4	10.4	+11%	0.04	0.09	3.0×
	Cloth Move	15.8	23.2	+47%	0.85	3.00	3.3×
Particle	Push Sand	7.3	9.5	+30%	0.08	0.31	3.9×
	Pour Water	6.1	6.6	+8%	0.06	0.15	4.5×
Kinematics	Robot Arm	5.0	19.4	+288%	0.06	0.25	4.7×
	Reacher	13.8	13.9	+1%	0.05	0.08	5.2×

Table 2: **Physics-grounded evaluation of ACWM-DiT-S** (50 InD / 50 OoD episodes per environment). Track error is the mean GT-vs-rollout point-track displacement; conservation error is the relative drift of a simulator-conserved quantity; action fidelity is $\mathrm{err}(\text{zero action})/\mathrm{err}(\text{true action})$, where $\gg 1$ means the rollout follows the commanded action.

lated, we can test this directly. We report three physics-grounded metrics on 50 InD and 50 OoD episodes per environment (definitions in Appendix A.9): **track error**, the mean displacement between ground-truth and predicted point tracks seeded on the shared conditioning frame ($\times 10^3$, a proxy for pose fidelity); **conservation error**, the relative drift of a quantity the simulator conserves (segmented water/sand mass, rope length, object permanence); and **action fidelity**, the ratio $\mathrm{err}(\text{zero action})/\mathrm{err}(\text{true action})$ under a counterfactual rollout, where a value $\gg 1$ means the prediction is genuinely driven by the action rather than by scene priors.

Finding 2. Measuring physics instead of pixels reproduces the same trend. Ranking environments by physical error recovers the ranking given by perceptual error, and makes the complexity-over-category effect sharper. The central conclusion does not depend on which metric family is used.

Ordering environments from most to least OoD-robust by track-error growth gives Reacher (+1%), Pour Water (+8%), Push Cube (+14%), Cloth Move (+47%), Robot Arm (+288%); ordering them by ΔSSIM in Table 1 gives Reacher (0.000), Push Cube (−0.001), Pour Water (−0.037), Cloth Move (−0.056), Robot Arm (−0.067). The two agree on both endpoints and differ only by one adjacent swap. Notably, Pour Water, a particle environment usually assumed hard, is among the most stable, while Robot Arm is the worst in both metric families.

Finding 3. Models still capture visual statistics, not physical laws. OoD failures such as cubes vanishing, water levels hallucinated, and articulated arms blurring surface exactly where appearance shortcuts can no longer fake the answer. The physics metrics catch these failures when perceptual metrics do not, and can even move in opposite directions.

Conservation error is the sharpest case: near zero on rigid and kinematic environments, it rises $0.85 \rightarrow 3.00$ on Cloth Move and $0.06 \rightarrow 0.15$ on Pour Water. The model lets mass appear and vanish precisely where the physics is hardest, which nothing in Table 1 registers. The Cloth Move PSNR inversion has the same origin: a smoother, less textured rollout is rewarded even as the cloth ends up in the wrong place. Action fidelity stays well above 1 everywhere (2.8×–5.2×), confirming the rollouts are action-driven and the OoD gap is not the model ignoring its conditioning.

5.3 Cross-Architecture Validation

Conclusions about "current ACWMs" should not rest on a single backbone. No released ACWM shares our action space, so checkpoints cannot be scored zero-shot; instead we post-train a second, architecturally distinct model, Cosmos-Predict2.5-2B [1], a pretrained video foundation model, using its official action-injection recipe (Appendix A.10) on Push Cube and Push Rope.

Finding 4. A different architecture shows the same OoD degradation. A model initialized from a large pretrained video foundation model still loses substantial fidelity out of distribution. The gap is a property of the benchmark, not an artifact of the ACWM-DiT backbone.

Environment	Model	Training budget	InD PSNR↑	OoD PSNR↑	ΔPSNR
Push Cube (rigid)	ACWM-DiT-S (≈200M)	100k, from scratch	25.4	25.3	−0.1
	Cosmos-Predict2.5-2B	10k, post-trained	22.9	20.1	−2.8
Push Rope (deformable)	ACWM-DiT-S (≈200M)	100k, from scratch	36.8	35.3	−1.5
	Cosmos-Predict2.5-2B	10k, post-trained	28.7	28.0	−0.7

Table 3: **A second architecture on ACWM-Phys.** Cosmos-Predict2.5-2B, a pretrained video foundation model, post-trained with action conditioning on two environments. Budgets are not matched: ACWM-DiT-S is trained from scratch for 100k steps, Cosmos is post-trained for 10k. Both models retain a non-trivial OoD gap, but they do not order the two environments the same way (Section 5.3).

Cosmos-Predict2.5-2B is uniformly weaker in absolute terms, which is expected at a $10\times$ smaller post-training budget, but the qualitative conclusion transfers: a strong, heavily-pretrained video prior does not close the InD→OoD gap, and both models remain well short of their InD fidelity under a single-factor physical shift. The two architectures do not order these two particular environments identically (ACWM-DiT-S degrades more on the deformable environment, Cosmos more on the rigid one), which is unsurprising given the unmatched budgets and a sample of two; settling whether the finer per-environment ranking is universal requires the full model zoo across all eight environments, which we identify as the main open item for this benchmark.



Figure 4: **Case study: Pour Water.** GT (top) and predicted (bottom) frames at four evenly-spaced timesteps. Two InD episodes (top block) and two OoD episodes (bottom block), with less water (left) and more water (right). The arm closely follows the ground-truth trajectory, but under OoD shift the model underestimates the water amount and part of the fluid disappears.

Case study: Pour Water. The model correctly predicts the pouring trajectory and the general fluid dynamics for in-distribution water levels (Figure 4 in Appendix). Under OoD shifts (fewer or far more water layers than seen during training), the model tends to generate visually plausible but physically inperfect fill levels, highlighting the gap between perceptual quality and true physical understanding.

5.4 Design Ablations

We ablate model scale, action conditioning, latent-space formulation, data volume, and action dimensionality; full tables are in Appendix B. Two conclusions transfer beyond this benchmark.

Finding 5. Cross-attention beats AdaLN once actions are high-dimensional. For low-dimensional controls the two conditioning mechanisms are indistinguishable. Once the action carries several independent degrees of freedom, AdaLN's single global conditioning vector becomes a per-joint bottleneck and cross-attention pulls ahead.

At d_a=2 (Push Cube, Push Rope) the two mechanisms differ by less than 0.3 dB in either direction. At d_a=7–8 the gap opens: cross-attention gains +3.18 dB InD on Robot Arm and +0.65 dB OoD on Cloth Move (Appendix Table 9).

Finding 6. A temporal VAE and richer actions both improve OoD. A causal video VAE outperforms a frame-independent encoder, and expanding an environment to its full action space improves out-of-distribution accuracy at a small in-distribution cost: richer control signals double as richer observations of the underlying physical state.

The Wan 2.1 causal VAE (4× temporal compression) beats the frame-independent FLUX VAE on both Pour Water and Robot Arm (Appendix Table 10). Expanding Cloth Move from the reduced 3-DoF shared action to full per-arm control yields +5.37 dB OoD for −0.81 dB InD (Appendix Table 5).

6 Conclusion and Limitation

We introduced ACWM-Phys, a benchmark spanning four physically diverse interaction regimes, and ACWM-DiT, a latent diffusion transformer baseline trained with flow matching. Current ACWMs achieve strong in-distribution fidelity but degrade substantially out of distribution, and the degradation tracks task complexity rather than physics category. Scoring the same rollouts against physical quantities rather than pixels sharpens this picture: the two metric families agree on the overall trend, yet physical error exposes failures, and occasionally reversals, that pixels do not. We therefore argue that appearance-based evaluation is insufficient evidence for claims about physical generalization, and hope ACWM-Phys serves as a diagnostic tool for future work on architectures that explicitly represent physical structure.

Limitations. Our cross-architecture evidence covers two of eight environments at unmatched budgets, so the per-environment difficulty ranking may not be architecture-independent; a full model zoo is the main open item for this benchmark. Each OoD split varies a single causal factor by design, leaving compositional, material, viewpoint, and cross-task shifts uncovered, though the released generators support them. Finally, ACWM-DiT is bidirectional and thus slow at inference, and simulation does not capture real-world sensing and contact.

AI use statement

We used generative AI tools as general-purpose writing and coding assistants: for polishing the prose of this paper (grammar, phrasing, and LaTeX formatting) and for routine coding support such as boilerplate in data-conversion and plotting scripts. We did not use generative AI tools to generate research ideas, to design the benchmark or its evaluation protocol, to produce or modify any experimental result, or to retrieve or write the related-work citations, all of which were verified manually against the original publications. All AI-assisted text and code were reviewed by the authors, and all reported numbers come from executed experiments. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the aid of generative AI.

Reproducibility statement

ACWM-Phys is built entirely in simulation, so every training and evaluation split is exactly reproducible from a fixed seed. Appendix A.1 specifies the ACWM-DiT architecture and training hyperparameters; Appendix A.2 documents the single-factor shift defining each OoD split; Appendix A.3 and Appendix A.9 define the perceptual and physics-grounded metrics; and Appendix A.10 gives the post-training recipe for the second architecture. Appendix Tables 6 and 7 document the per-environment data-generation and action-space configurations. Code for data generation, training, and evaluation, together with the benchmark data, is provided as anonymized supplementary material and will be released publicly.

REFERENCES

[1] Arslan Ali et al. World simulation with video foundation models for physical AI. *arXiv preprint arXiv:2511.00062*, 2025.

[2] Anurag Bagchi, Zhipeng Bao, Homanga Bharadhwaj, Yu-Xiong Wang, Pavel Tokmakov, and Martial Hebert. Walk through paintings: Egocentric world models from internet priors. *arXiv preprint arXiv:2601.15284*, 2026.

[3] Yixiang Chen, Peiyan Li, Jiabing Yang, Keji He, Xiangnan Wu, Yuan Xu, Kai Wang, Jing Liu, Nianfeng Liu, Yan Huang, et al. Bridgev2w: Bridging video generation models to embodied world models via embodiment masks. *arXiv preprint arXiv:2602.03793*, 2026.

[4] Yanjiang Guo, Lucy Xiaoyang Shi, Jianyu Chen, and Chelsea Finn. Ctrl-world: A controllable generative world model for robot manipulation. *arXiv preprint arXiv:2510.10125*, 2025.

[5] David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2(3): 440, 2018.

[6] Danijar Hafner, Wilson Yan, and Timothy Lillicrap. Training agents inside of scalable world models. *arXiv preprint arXiv:2509.24527*, 2025.

[7] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://arxiv.org/abs/2006.11239.

[8] Jonathan Ho, Tim Salimans, Alexey Gritsenko, William Chan, Mohammad Norouzi, and David J Fleet. Video diffusion models. *Advances in neural information processing systems*, 35:8633–8646, 2022.

[9] Yicong Hong, Yiqun Mei, Chongjian Ge, Yiran Xu, Yang Zhou, Sai Bi, Yannick Hold-Geoffroy, Mike Roberts, Matthew Fisher, Eli Shechtman, Kalyan Sunkavalli, Feng Liu, Zhengqi Li, and Hao Tan. Relic: Interactive video world models with long-horizon memory, 2025. URL https://relic-worldmodel.github.io/.

[10] Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition*, pp. 2366–2369. IEEE, 2010.

[11] Siqiao Huang, Jialong Wu, Qixing Zhou, Shangchen Miao, and Mingsheng Long. Vid2world: Crafting video diffusion models to interactive world models. *arXiv preprint arXiv:2505.14357*, 2025.

[12] Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *arXiv preprint arXiv:2506.08009*, 2025.

[13] Zhennan Jiang, Shangqing Zhou, Yutong Jiang, Zefang Huang, Mingjie Wei, Yuhui Chen, Tianxing Zhou, Zhen Guo, Hao Lin, Quanlu Zhang, et al. Wovr: World models as reliable simulators for post-training vla policies with rl. *arXiv preprint arXiv:2602.13977*, 2026.

[14] Bingyi Kang, Yang Yue, Rui Lu, Zhijie Lin, Yang Zhao, Kaixin Wang, Gao Huang, and Jiashi Feng. How far is video generation from world model: A physical law perspective. *arXiv preprint arXiv:2411.02385*, 2024.

[15] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker3: Simpler and better point tracking by pseudo-labelling real videos. In *International Conference on Computer Vision (ICCV)*, 2025.

[16] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Advances in Neural Information Processing Systems*, 2022. URL https://arxiv.org/abs/2206.00364.

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

[18] Black Forest Labs. Flux. https://github.com/black-forest-labs/flux, 2024.

[19] Minh-Quan Le, Yuanzhi Zhu, Vicky Kalogeiton, and Dimitris Samaras. What about gravity in video generation? post-training newton's laws with verifiable rewards. *arXiv preprint arXiv:2512.00425*, 2025.

[20] Chenyu Li, Oscar Michel, Xichen Pan, Sainan Liu, Mike Roberts, and Saining Xie. Pisa experiments: Exploring physics post-training for video diffusion models by watching stuff drop. *arXiv preprint arXiv:2503.09595*, 2025.

[21] Yunzhu Li, Jiajun Wu, Russ Tedrake, Joshua B Tenenbaum, and Antonio Torralba. Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids. *arXiv preprint arXiv:1810.01566*, 2018.

[22] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

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

[24] Saman Motamed, Laura Culp, Kevin Swersky, Priyank Jaini, and Robert Geirhos. Do generative video models understand physical principles? In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 948–958, 2026.

[25] Jack Parker-Holder and Shlomi Fruchter. Genie 3: A new frontier for world models. *URL https://deepmind. google/discover/blog/genie-3-a-new-frontier-for-world-models/. Blog post*, 2025.

[26] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4195–4205, 2023.

[27] Georgy Savva, Oscar Michel, Daohan Lu, Suppakit Waiwitlikhit, Timothy Meehan, Dhairya Mishra, Srivats Poddar, Jack Lu, and Saining Xie. Solaris: Building a multiplayer video world model in minecraft. *arXiv preprint arXiv:2602.22208*, 2026.

[28] Dhruv Shah, Benjamin Eysenbach, Nicholas Rhinehart, and Sergey Levine. Rapid exploration for open-world navigation with latent goal models. In *5th Annual Conference on Robot Learning*, 2021. URL https://openreview.net/forum?id=d_SWJhyKfVw.

[29] Wenqiang Sun, Haiyu Zhang, Haoyuan Wang, Junta Wu, Zehan Wang, Zhenwei Wang, Yunhong Wang, Jun Zhang, Tengfei Wang, and Chunchao Guo. Worldplay: Towards long-term geometric consistency for real-time interactive world modeling. *arXiv preprint arXiv:2512.14614*, 2025.

[30] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012. doi: 10.1109/IROS.2012.6386109.

[31] Tong Wan et al. Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

[32] Wan-Video Team. Wan2.1: Open video foundation models. GitHub repository, 2025. URL https://github.com/Wan-Video/Wan2.1. Technical report and weights; project page details evolving.

[33] Jing Wang, Ao Ma, Ke Cao, Jun Zheng, Jiasong Feng, Zhanjie Zhang, Wanyuan Pang, and Xiaodan Liang. Wisa: World simulator assistant for physics-aware text-to-video generation. In *Advances in Neural Information Processing Systems*, 2025.

[34] Zeqing Wang, Xinyu Wei, Bairui Li, Zhen Guo, Jinrui Zhang, Hongyang Wei, Keze Wang, and Lei Zhang. Videoverse: How far is your t2v generator from a world model? *arXiv preprint arXiv:2510.08398*, 2025.

[35] Zijun Wang, Panwen Hu, Jing Wang, Terry Jingchen Zhang, Yuhao Cheng, Long Chen, Yiqiang Yan, Zutao Jiang, Hanhui Li, and Xiaodan Liang. Prophy: Progressive physical alignment for dynamic world simulation. *arXiv preprint arXiv:2512.05564*, 2025.

[36] Zhen Yang et al. CogVideoX: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024.

[37] Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.

[38] Yu Yuan, Xijun Wang, Tharindu Wickremasinghe, Zeeshan Nadir, Bole Ma, and Stanley H. Chan. Newtongen: Physics-consistent and controllable text-to-video generation via neural newtonian dynamics. In *International Conference on Learning Representations*, 2026.

[39] Chenyu Zhang, Daniil Cherniavskii, Antonios Tragoudaras, Antonios Vozikis, Thijmen Nijdam, Derck WE Prinzhorn, Mark Bodracska, Nicu Sebe, Andrii Zadaianchuk, and Efstratios Gavves. Morpheus: Benchmarking physical reasoning of video generative models with real physical experiments. *arXiv preprint arXiv:2504.02918*, 2025.

[40] Ke Zhang, Cihan Xiao, Yiqun Mei, Jiacong Xu, and Vishal M. Patel. Think before you diffuse: Llms-guided physics-aware video generation, 2025. URL https://arxiv.org/abs/2505.21653.

[41] Siyuan Zhou, Hejun Wang, Hu Cheng, Jinxi Li, Dongsheng Wang, Junwei Jiang, Yixiao Jin, Jiayue Huang, Shiwei Mao, Shangjia Liu, Yafei Yang, Hongkang Song, Shenxing Wei, Zihui Zhang, Peng Huang, Shijie Liu, Zhengli Hao, Hao Li, Yitian Li, Wenqi Zhou, Zhihan Zhao, Zongqi He, Hongtao Wen, Shouwang Huang, Peng Yun, Bowen Cheng, Pok Kazaf Fu, Wai Kit Lai, Jiahao Chen, Kaiyuan Wang, Zhixuan Sun, Ziqi Li, Haochen Hu, Di Zhang, Chun Ho Yuen, Bing Wang, Zhihua Wang, Chuhang Zou, and Bo Yang. Physinone: Visual physics learning and reasoning in one suite, 2026. URL https://arxiv.org/abs/2604.09415.

A Appendix

A.1 Details of ACWM-DiT

Overall Architecture ACWM-DiT (Figure 3) is a latent video diffusion transformer trained with flow matching to predict future observation trajectories conditioned on past observations and a sequence of actions.

Video VAE. Following Wan 2.1 [31], we use a pretrained causal VAE that compresses a video $\mathbf{o}_{1:T} \in \mathbb{R}^{T\times 3\times H\times W}$ into latent tokens $\mathbf{z} \in \mathbb{R}^{T_l\times \frac{H}{8}\times \frac{W}{8}\times 16}$, applying $8\times$ spatial compression, $4\times$ temporal compression, and a 16-channel latent space. All VAE weights are frozen throughout training.

Video DiT Backbone. The denoiser is a bidirectional (non-causal) transformer that operates on spatiotemporally patchified latents (spatial patch size 2). Each of the N transformer blocks consists of a spatial self-attention layer (attending over the $\frac{H}{16}\times\frac{W}{16}$ spatial patches within each frame) followed by a temporal self-attention layer attending across T_l latent timesteps at each spatial location. Rotary Position Embeddings (RoPE) are used in both layers. Observation-conditioning frames are passed at noise level $\sigma{=}0$ while predicted frames are noised at the current diffusion step σ; this allows the model to attend bidirectionally over both clean context and noisy future tokens. Conditioning on the diffusion timestep and actions is applied through Adaptive Layer Normalization (AdaLN), which uses the joint signal $\mathbf{c}$ to produce per-step scale and shift parameters (γ, β, α) at every block. We study two model scales: **DiT-S** (hidden dim 768, 10 layers, 12 heads, ≈200M parameters) and **DiT-M** (hidden dim 1024, 16 layers, 16 heads, ≈600M parameters).

Flow Matching. Training follows the latent-space flow matching objective defined in Section 3 with a linear interpolation path and shift parameter $s{=}5.0$. Training losses are weighted by a Gaussian envelope centered at diffusion step 500 to focus supervision on intermediate noise levels. We use 1000 training steps and 50 inference denoising steps.

Action Conditioning Module Actions are integrated via a dedicated *ActionEmbedder* module that maps the pixel-rate action sequence to the latent temporal resolution in two stages. First, an MLP (Linear→SiLU→Linear) projects each action vector from its environment-specific dimension d_a to the DiT hidden dimension d. Second, a 1D strided convolution (kernel size 3, stride $r{=}4$, matching the VAE's $4\times$ temporal compression factor) downsamples the sequence from $1+r(T_l-1)$ pixel-rate steps to T_l latent-rate tokens, producing $\hat{\mathbf{a}} \in \mathbb{R}^{T_l\times d}$.

The action embedding is summed with the per-step timestep embedding to form the joint conditioning signal $\mathbf{c} = \mathbf{c}_t + \hat{\mathbf{a}} \in \mathbb{R}^{T_l\times d}$, which is then broadcast into every DiT block via AdaLN, modulating the scale and shift applied after each layer normalization. This design couples action information with the diffusion timestep and propagates it uniformly across all spatial and temporal attention operations without requiring additional cross-attention layers or token-sequence modifications. For the standard training setting we use no action dropout ($p_{\text{drop}}{=}0$); ablations on classifier-free guidance are left to future work.

A.2 Out-of-Distribution Split Design

Each ACWM-Phys environment defines a physically motivated OoD split that targets a specific axis of generalization rather than using random perturbations. Below we summarize the shift type and parameter range for each environment. All OoD trajectories are generated by the same simulator as the training data, but with configurations explicitly excluded from the training distribution.

Rigid-Body. For `Push Cube`, training covers a central workspace region, while OoD episodes place the cube initial positions and push targets near table corners or edges, requiring spatial extrapolation of rigid-body dynamics. We also include a harder variant, `push_cube_4cube`, with four or more cubes to test generalization to unseen object counts. For `Stack Cube`, the target placement direction defines the shift: training uses a subset of directions, while OoD episodes require stacking toward held-out cardinal directions, recorded by the `ood_label` field.

Deformable. For `Push Rope`, the OoD shift is rope length, which affects stiffness and deformability. Training uses rope lengths in $[2.0, 2.8]$ m, while OoD episodes fix the length to 3.1 m, producing qualitatively different deformation dynamics. For `Cloth Move`, OoD episodes change the cloth size and initial configuration, placing the cloth in spatial arrangements not observed during training.

Particle. For `Push Sand`, OoD episodes use different random seeds from training/InD testing, resulting in unseen initial particle layouts and density configurations. For `Pour Water`, the shift is the water quantity: InD episodes cover a nominal water-level range, while OoD episodes use substantially lower or higher volumes, testing extrapolation of fluid dynamics.

Kinematics. For `Robot Arm`, OoD episodes require reaching targets in an expanded workspace. The joint-angle action range expands from $[-0.95, 0.58]$ in InD to $[-1.15, 0.88]$ in OoD, inducing more extreme arm configurations than those seen during training. For `Reacher`, OoD goals are sampled from corner sectors of the reachable space excluded during training, with the action range expanding from $[-3.3, 3.5]$ to $[-3.7, 4.2]$ rad.

A.3 MASKED-MSE (M-MSE)

Standard MSE treats all pixels equally and can therefore be dominated by static background regions, especially in environments where only a small portion of the scene is physically dynamic. To better focus evaluation on moving regions, we introduce **Masked-MSE (M-MSE)**, a motion-aware weighted mean squared error.

Given a ground-truth video $\mathbf{o}_{1:T}$ and a predicted video $\hat{\mathbf{o}}_{1:T}$, we first compute a per-pixel motion map using only the ground-truth video:

$$m_{h,w} = \max_{t \in [1,T],\, c} \left| \mathbf{o}_{t,c,h,w} - \mathbf{o}_{1,c,h,w} \right|,$$

which measures the maximum absolute deviation of each pixel from the first frame across all timesteps and channels. We then define a soft per-pixel weight:

$$w_{h,w} = 0.01 + m_{h,w},$$

where the small floor value prevents zero weights for perfectly static pixels while remaining negligible compared to typical motion magnitudes. In practice, the weight is therefore dominated by $m_{h,w}$ and assigns very small weights to static background pixels.

M-MSE is defined as the resulting weighted mean squared error:

$$\text{M-MSE} = \frac{\sum_{t,c,h,w} w_{h,w} \left(\hat{\mathbf{o}}_{t,c,h,w} - \mathbf{o}_{t,c,h,w} \right)^2}{\sum_{t,c,h,w} w_{h,w}}.$$

Because $w_{h,w}$ is proportional to the amount of motion at each pixel, errors in dynamic foreground regions are strongly up-weighted, while static background pixels contribute only minimally. This makes M-MSE more sensitive to physically meaningful prediction failures that may be underemphasized by standard MSE.

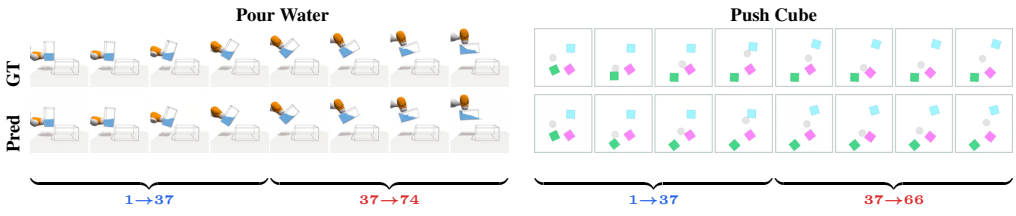

Figure 5: **Auto-regressive Generation.** The model generates frames 1→37 (blue) conditioned on the first frame, then generates frames 37→T (red) conditioned on the last predicted frame of the first window. GT (top) and predicted (bottom) frames at four evenly-spaced timesteps per window.

Benchmark	Type	Action-conditioned	Rich Physics	Data Scale	ID+OOD
Solaris [27]	Multi-Player Minecraft	✓	✗	9k videos	✗
Dreamer4 [6]	Minecraft	✓	✗	2.5K hour gameplay	✗
DROID [17]	Expert manipulation	✓	✗	76K trajectories	✗
RECON [28]	Egocentric navigation	✓	✗	5K trajectories	✗
Physics-IQ [24]	Passive physics evaluation	✗	✓	396 videos	✗
WISA [33]	Passive physics evaluation	✗	✓	32K videos	✗
PhysInOne [41]	Synthetic physics interaction	✗	✓	2M videos	✗
ACWM-Phys (ours)	**Synthetic physics interaction**	✓	✓	**15k videos**	✓

Table 4: **Comparison of ACWM-Phys with existing action-conditioned and physics-related benchmarks.** Benchmarks used for training action-conditioned world models generally lack rich and diverse physical interactions, whereas existing physics-focused benchmarks are typically not action-conditioned. Also, most prior benchmarks do not include both in-distribution and out-of-distribution evaluations

Table 5: **Action dimensionality ablation.** Cloth Move compares the reduced 3-DoF shared action against the full 8-DoF per-arm action space. Push Cube compares a single pusher (d_a=2) against two independent pushers (d_a=4).

Environment	Action Space	d_a	InD		OoD	
			MSE↓	PSNR↑	MSE↓	PSNR↑
Cloth Move	Reduced (shared Δxyz)	3	10.667	19.72	23.820	16.23
	Full (per-arm Δpose + grasp)	8	12.861	18.91	6.922	21.60
Push Cube	Single pusher	2	2.919	25.35	2.950	25.30
	Two pushers	4	6.034	22.19	9.997	20.00

A.4 Dataset Statistics and Action Space Definitions

Tables 6 and 7 summarize the dataset sizes, action dimensionalities, and trajectory horizons across all eight ACWM-Phys environments, together with the detailed semantics of each action space.

A.5 Metrics vs. Diffusion Steps

Figures 6 and 7 show SSIM and PSNR as a function of the number of inference diffusion steps across all eight ACWM-Phys environments. In general, performance saturates quickly: gains from 5 to 50 steps are marginal for most tasks, suggesting that 5–10 steps suffice at inference time.

A.6 Dataset Visualizations

Figure 8 shows representative ground-truth frames from all eight ACWM-Phys environments for both in-distribution (InD) and out-of-distribution (OoD) test splits. Each row displays eight evenly-spaced frames from a single episode, illustrating the diversity of physical interactions and the nature of the OoD distribution shift in each environment.

A.7 Per-Environment Case Studies

For each environment, we show one in-distribution (InD) and one out-of-distribution (OoD) episode with GT (top) and Pred (bottom) rows at four evenly-spaced timesteps. For kinematics environments (Robot Arm, Reacher), an additional Overlay row blends the GT at 45% opacity with a blue tint over the prediction, making positional errors directly visible without relying on side-by-side comparison alone.

Push Rope. The model faithfully predicts rope deformation under InD conditions. Under OoD stiffness shifts, the predicted rope shape diverges from ground truth at later timesteps, with the model underestimating rope rigidity and producing slightly straighter configurations than observed.

Table 6: **ACWM-Phys dataset statistics.** Each environment provides in-distribution (InD) and out-of-distribution (OoD) test splits with controlled distribution shifts. Action dim refers to the dimensionality of the action vector fed to ACWM-DiT; full action definitions are listed in Table 7.

Category	Environment	Train	InD Test	OoD Test	Act. Dim	Horizon
Rigid Dynamics	Push Cube	1,987	100	100	2	∼66
	Stack Cube	1,987	100	100	7	∼57
Deformable	Push Rope	1,987	100	100	2	64
	Cloth Move*	1,987	100	100	3	64
Particle	Push Sand	1,784	100	100	7	64
	Pour Water	1,000	50	50	4	300
Kinematics	Robot Arm	1,987	100	100	7	64
	Reacher	1,987	100	100	2	50
Total	8 environments	**14,706**	**750**	**750**	2–7	50–300

*Cloth Move uses reduced 3-dim actions; 8-dim variant used in action-dimension ablation.

Table 7: **Action space definitions across ACWM-Phys environments.** Actions are either *absolute* target states or *delta* (incremental) commands.

Environment	Type	Dim	Semantics
Push Cube	Absolute position	2	Target (x, y) pusher position on workspace
Stack Cube	Delta EE pose	7	$(\Delta x, \Delta y, \Delta z, \Delta r_x, \Delta r_y, \Delta r_z)$ + gripper open/close
Push Rope	Delta position	2	2D horizontal pole displacement $(\Delta x, \Delta z)$
Cloth Move	Delta position	3	$(\Delta x, \Delta y, \Delta z)$ dual-arm displacement*
Push Sand	Delta board pose	7	Board 3D position + 4D orientation delta
Pour Water	Delta cup pose	4	$(\Delta x, \Delta y, \Delta z, \Delta\theta)$ cup tilt delta
Robot Arm	Delta joint angles	7	Per-joint angle delta $\Delta\mathbf{q} \in \mathbb{R}^7$
Reacher	Joint torques	2	Torques (τ_1, τ_2) applied to two links

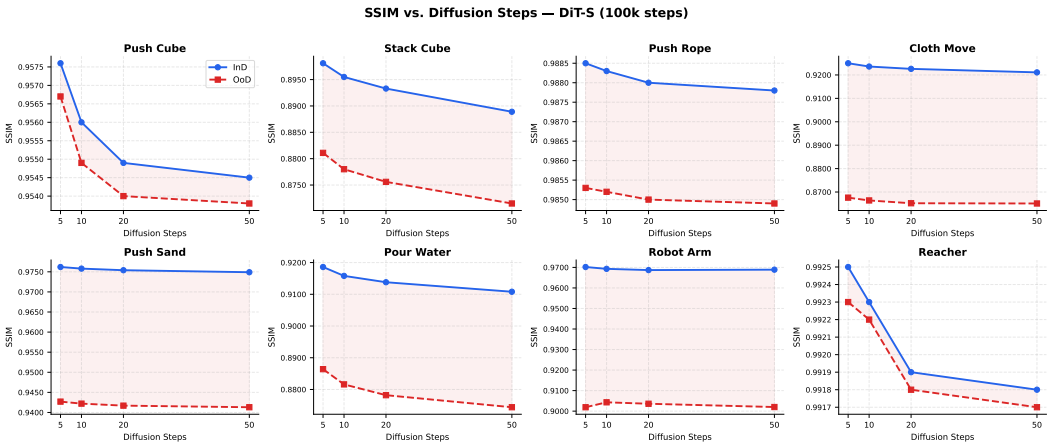

Figure 6: **SSIM vs. diffusion steps** for ACWM-DiT-S (100k training steps). Blue circles: InD test; red squares: OoD test. Higher SSIM is better (↑).

Cloth Move. The model does not capture the dynamics well for both InD and OoD. Under OoD cloth sizes (smaller or larger than the training range), the model hallucinates incorrect cloth extents and misplaces the deformation boundary at the sphere, producing plausible but geometrically inaccurate draping.

Push Sand. The model generalizes partially to OoD doubled particle counts: overall granular flow direction and pile topology are qualitatively preserved. However, the model sometimes predicts

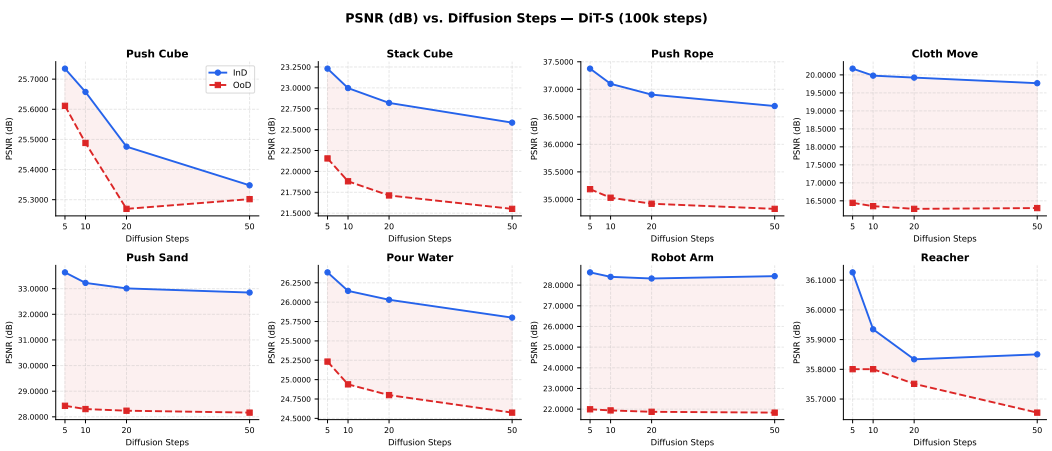

Figure 7: **PSNR vs. diffusion steps** for ACWM-DiT-S (100k training steps). Blue circles: InD test; red squares: OoD test. Higher PSNR is better (↑).

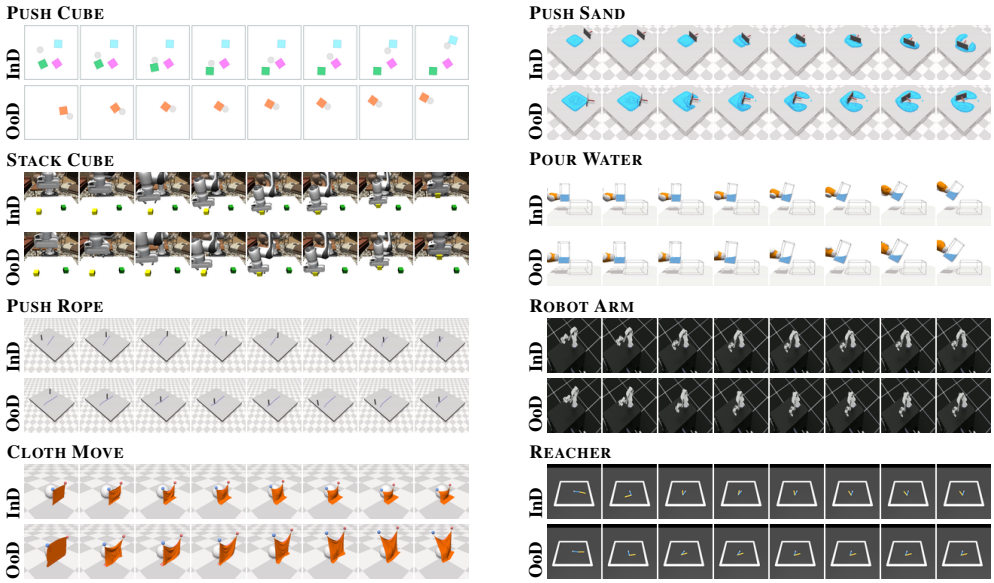

Figure 8: **Dataset visualizations for all eight ACWM-Phys environments.** Left: rigid-body and deformable tasks. Right: particle and kinematics tasks. For each environment, InD (top) and OoD (bottom) ground-truth frames are shown at eight evenly-spaced timesteps from a representative episode.

fewer particles than are present—the sand pile appears to *shrink*—reflecting that precise particle count is not internalized and degrades under large distribution shifts in granular density.

Stack Cube. InD stacking trajectories e.g. pick-up, transport, and placement are accurately predicted. Under OoD target placement shifts, the model predicts a plausible but positionally incorrect stack, indicating limited spatial extrapolation beyond training placement configurations.

Robot Arm. The overlay row (blue-tinted GT ghost over prediction) reveals systematic end-effector position errors under OoD workspace expansion. InD predictions closely match GT joint-angle trajectories; OoD predictions reproduce plausible arm motion but with a consistent spatial offset, consistent with the large ΔPSNR and ΔSSIM reported in Table 1.

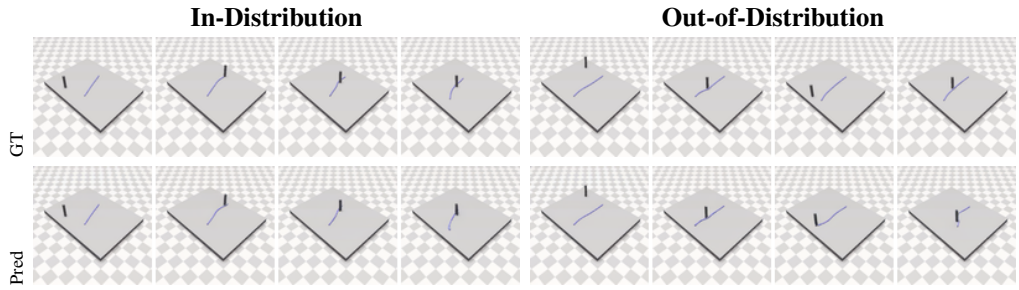

Figure 9: **Push Rope case study.** InD (left) and OoD with longer rope (right).

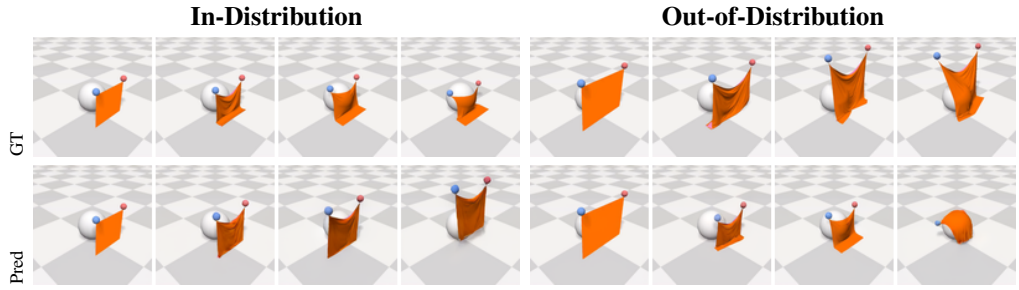

Figure 10: **Cloth Move case study.** InD (left) and OoD cloth-size shift (right).

Reacher. The model achieves near-perfect prediction for the two-link planar arm both InD and OoD. The overlay row confirms negligible positional error even under OoD corner-sector goals unseen during training, consistent with the minimal $\Delta\text{SSIM} = 0.000$ in the main evaluation.

A.8 Case Study: Push Cube

Figure 15 contrasts two OoD regimes: a single cube pushed to an out-of-distribution workspace position, and a scene with 4+ cubes (unseen during training). The model accurately tracks rigid-body trajectories in InD cases and generalizes well to OoD settings overall, although cubes occasionally disappear abruptly in some OoD videos, a failure that the conservation metric of Table 2 is designed to catch and that no perceptual metric registers.

A.9 Physics-Grounded Metrics

This section defines the three metrics reported in Table 2. All are computed on 50 InD and 50 OoD episodes per environment, using the same conditioning frame for the ground-truth and the predicted rollout so that the two sequences are directly comparable.

Track error. We seed a dense grid of query points on the shared conditioning frame and propagate them through both the ground-truth video and the model rollout with CoTracker3 [15]. The metric is the mean Euclidean displacement between corresponding tracks, averaged over points and time and reported as $\times 10^3$ in normalized image coordinates. Unlike MSE or SSIM, it is insensitive to texture and blur and responds only to *where* matter ends up, which makes it a proxy for pose error that does not require privileged state at evaluation time.

Conservation error. Each environment has one quantity the simulator conserves: segmented fluid or granular mass for Pour Water and Push Sand, contour length for Push Rope and Cloth Move, and object count (permanence) for Push Cube and Stack Cube. We segment the quantity in every predicted frame, and report the mean relative drift $\frac{1}{T}\sum_t |q_t - q_0|/q_0$. A model that renders a plausible scene while letting water evaporate or a cube blink out is penalized here and nowhere else.

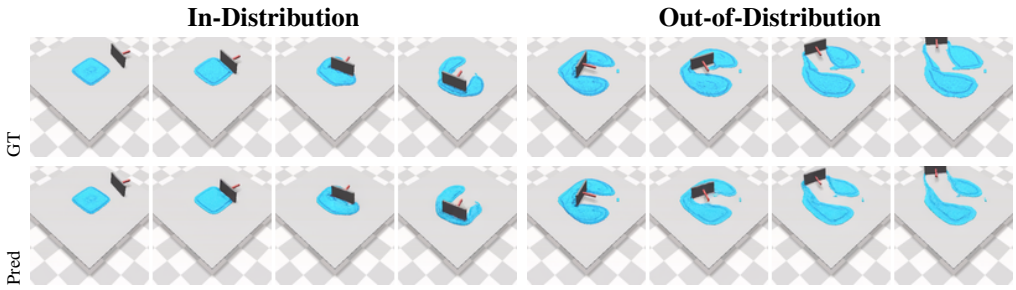

Figure 11: **Push Sand case study.** InD (left) and OoD doubled-particle-count (right).

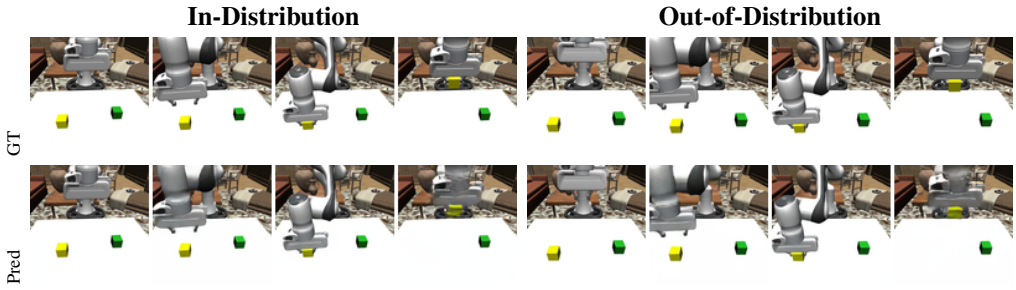

Figure 12: **Stack Cube case study.** InD (left) and OoD placement-shift (right).

Action fidelity. To verify that a rollout is causally driven by its action sequence rather than by scene priors, we re-run each episode with the action input zeroed and measure $\mathrm{err}(\text{zero action})/\mathrm{err}(\text{true action})$ under the track metric. A ratio near 1 would mean the action conditioning is ignored; the observed range of $2.8\times$–$5.2\times$ confirms that predictions respond to the commanded action. This also doubles as the protocol by which action-free, pixel-to-pixel world models can be scored on ACWM-Phys: such a model is evaluated in the zero-action condition and compared against the same track and conservation references.

A.10 Cross-Architecture Post-Training Details

No publicly released ACWM shares the action space of ACWM-Phys, so existing checkpoints cannot be evaluated zero-shot. To obtain a second architecture we post-train Cosmos-Predict2.5-2B [1] following its official action-conditioning recipe: actions are flattened within each 12-step chunk, embedded by an MLP, and added to the DiT's global AdaLN/timestep conditioning signal. Porting ACWM-Phys to its data format required roughly 100 lines of conversion code, which we release alongside the benchmark as a template for adding further models.

We emphasize the two caveats visible in Table 3. The training budgets are not matched (ACWM-DiT-S is trained from scratch for 100k steps while Cosmos is post-trained from a pretrained checkpoint for 10k), so absolute PSNR values are not directly comparable across rows. And the coverage is two of eight environments. These results establish that a strong pretrained video prior does not eliminate the OoD gap; they do not establish that the per-environment difficulty ranking is architecture-independent.

B Ablation Studies

We conduct ablation studies along the following axes to provide further insights: model scale, action-conditioning mechanism, latent-space formulation, training data volume, and action dimensionality.

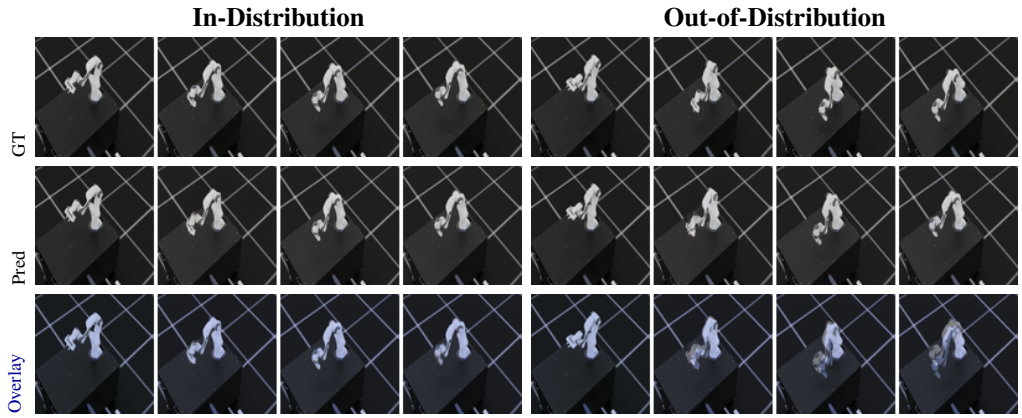

Figure 13: **Robot Arm case study.** InD (left) and OoD workspace-expansion (right). Overlay row: GT (blue tint, 45% opacity) over prediction highlights positional error.

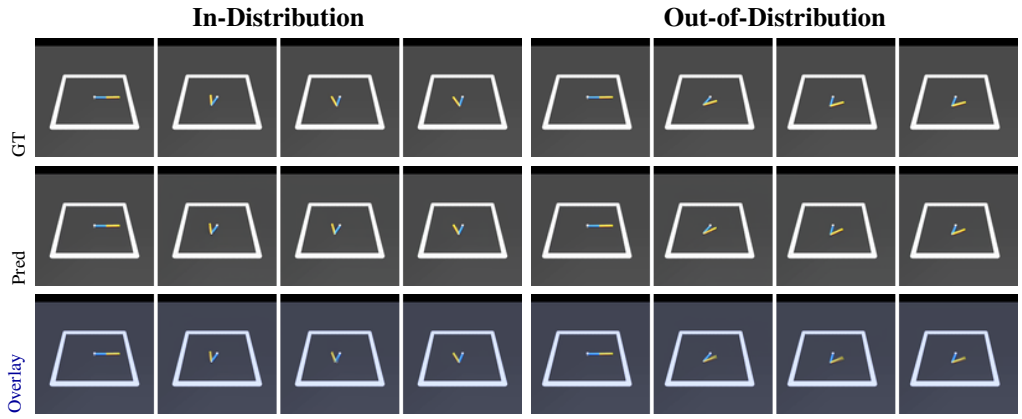

Figure 14: **Reacher case study.** InD (left) and OoD corner-sector goals (right). Overlay nearly coincides with Pred, confirming strong geometric generalization.

Model Scale. Scaling model capacity improves OoD PSNR/MSE more than it improves InD accuracy. We report this as a measured trend only; it is not by itself evidence that larger models represent physical structure, since these are pixel metrics. Gains diminish from DiT-M to DiT-L at the current data scale.

Table 8 compares DiT-S (≈200M), DiT-M (≈600M), and DiT-L (≈800M) on Cloth Move (deformable) and Robot Arm (kinematics), two environments that represent qualitatively different physical interaction regimes. Scaling from DiT-S to DiT-M consistently improves both InD and OoD performance, with larger gains on OoD. Gains from DiT-M to DiT-L are more modest, indicating diminishing returns at this data scale.

Table 8: **Model scale ablation** on Cloth Move (3-DoF action, 50k steps) and Robot Arm (50k steps).

Model	Params	Cloth Move				Robot Arm			
		InD		OoD		InD		OoD	
		MSE↓	PSNR↑	MSE↓	PSNR↑	MSE↓	PSNR↑	MSE↓	PSNR↑
DiT-S	≈200M	10.776	19.68	22.416	16.49	1.452	28.38	6.552	21.84
DiT-M	≈600M	10.248	19.89	20.031	16.98	1.247	29.04	4.891	23.11
DiT-L	≈800M	9.985	20.01	18.876	17.24	1.183	29.27	4.453	23.51



Figure 15: **Case study: Push Cube.** GT (top) and predicted (bottom) frames at four evenly-spaced timesteps. Two InD episodes (top block) and two OoD episodes (bottom block) show diverse cube configurations, with one cube (left) and four cubes (right). The model accurately tracks cube positions and push trajectories across both distributions.

Action Conditioning. Cross-attention is mainly beneficial for high-dimensional action spaces. For low-dimensional controls (d_a=2), AdaLN performs similarly or slightly better, while cross-attention gives clear gains on Robot Arm and modest OoD gains on Cloth Move.

We compare AdaLN-based action conditioning with a cross-attention variant that injects action tokens through dedicated cross-attention layers. As shown in Table 9, cross-attention brings no benefit on Push Cube and Push Rope (d_a=2), where AdaLN already captures simple displacement controls effectively. In contrast, for Robot Arm (d_a=7), cross-attention substantially improves both InD and OoD performance, suggesting better binding between joint commands and articulated motion. For Cloth Move (d_a=8), cross-attention slightly hurts InD performance but modestly improves OoD performance. Overall, cross-attention is most useful when actions are high-dimensional and require structured spatial-temporal grounding.

Table 9: **Action conditioning ablation**: AdaLN vs. cross-attention across four environments spanning low and high action dimensionality. ↑ means clearly better than AdaLN.

Method	Push Cube (d_a=2)				Push Rope (d_a=2)				Robot Arm (d_a=7)				Cloth Move (d_a=8)			
	InD		OoD		InD		OoD		InD		OoD		InD		OoD	
	MSE	PSNR	MSE	PSNR	MSE	PSNR	MSE	PSNR	MSE	PSNR	MSE	PSNR	MSE	PSNR	MSE	PSNR
AdaLN (ours)	2.919	25.35	2.950	25.30	0.214	36.70	0.329	34.83	1.434	28.43	6.559	21.83	9.393	20.27	5.464	22.62
Cross-Attn	3.105	25.08	3.033	25.18	0.216	36.65	0.334	34.77	0.691↑	31.61↑	4.596↑	23.38↑	11.512	19.39	4.713↑	23.27↑

Latent-Space Choice. A temporally-aware causal VAE outperforms frame-independent VAE

The Wan 2.1 causal VAE [32] applies 4× temporal compression, coupling consecutive frames in the latent space. We ablate this against a frame-independent image VAE (FLUX VAE [18]), which encodes each frame independently (1× temporal compression). Table 10 reports results on Pour Water and Robot Arm. WanVAE outperforms FluxVAE on both InD and OoD scenarios, indicating that temporally-aware latent representations are beneficial even for highly stochastic particle dynamics.

Data Efficiency. Data efficiency is governed by geometric variability: Pour Water degrades gracefully with less data due to its single repeatable pouring motion, while Push Cube degrades sharply as it requires diverse trajectory coverage over a wide workspace.

Table 10: **Latent-space formulation ablation.** We compare a temporally-aware causal video VAE (Wan 2.1, $4\times$ temporal compression) with a frame-independent image VAE (FLUX, $1\times$ temporal compression) on `Pour Water` and `Robot Arm`.

VAE	Temp.	Pour Water				Robot Arm			
		InD		OoD		InD		OoD	
		MSE↓	PSNR↑	MSE↓	PSNR↑	MSE↓	PSNR↑	MSE↓	PSNR↑
FLUX VAE	$1\times$	3.241	24.89	3.895	24.09	2.314	26.50	10.143	20.07
WanVAE (ours)	$4\times$	**2.630**	**25.80**	**3.488**	**24.57**	**1.434**	**28.43**	**6.559**	**21.83**

We train DiT-S on Push Cube and Pour Water using 100%, 50%, and 25% of the available training trajectories (Table 11). Both environments show degradation as data is reduced. Push Cube degrades sharply, indicating that diverse cube configurations and push directions require broad trajectory coverage to generalize. Pour Water is substantially more data-efficient: at 50% data the InD drop is only 0.26 dB, and even at 25% it retains reasonable performance (-1.69 dB), likely because its dynamics are governed by a single repeatable pouring motion with limited geometric variability.

Table 11: **Training data scaling ablation** on Push Cube and Pour Water. Models trained on 100%, 50%, and 25% of available trajectories.

Data	Trajs	Push Cube				Pour Water			
		InD		OoD		InD		OoD	
		MSE↓	PSNR↑	MSE↓	PSNR↑	MSE↓	PSNR↑	MSE↓	PSNR↑
100% (full)	1987 / 1000	2.919	25.35	2.950	25.30	2.630	25.80	3.488	24.57
50%	994 / 500	3.724	24.29	3.981	24.00	2.793	25.54	3.623	24.41
25%	497 / 250	5.318	22.74	5.891	22.30	3.882	24.11	4.689	23.29

Action Dimensionality. Higher-dimensional actions make prediction harder, but can improve OoD generalization when they provide more informative control signals and richer observation.

Table 5 compares action-space variants for Cloth Move and Push Cube. For Cloth Move, we use the full action space by allowing the two grippers to move independently, providing more detailed control over the deformable object. This substantially improves OoD MSE, suggesting that richer action signals help the model infer two-arm cloth dynamics. In contrast, for Push Cube, adding a second pusher increases interaction complexity without providing the same informative benefit, resulting in higher MSE in both InD and OoD settings.