PhysiCalWorld: IN-CONTEXT PHYSICS CALIBRATION FOR WORLD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing world models typically perceive a scene through current visual frame or a narrow history window. However, these observations may not reveal the physics properties that govern transition dynamics, such as an object's mass distribution, joint damping, or surface friction. Without calibrating these physics factors for world models, predictions tend to regress toward an empirical prior and induce severe drift in novel test scenes. To this end, we propose PhysiCalWorld, which calibrates action-conditioned world models to the instance-specific dynamics via in-context prompting, without fine-tuning or analytical physics modeling. Our method encodes an exploratory interaction clip collected within the test scene into a compact physics descriptor token, which grounds subsequent predictions in the scene's actual physical behavior. PhysiCalWorld is trained end-to-end with the latent prediction loss alone, without physics labels. At inference time, the descriptor token is extracted once and can be reused across rollouts with negligible compute overhead. We evaluate PhysiCalWorld across four environments covering diverse physics variations and show that it substantially improves rollout prediction accuracy and visual planning success rate. The learned descriptors also form a physically structured latent space and generalize beyond the physics ranges seen during training. Project page at https://physical-world-submit.github.io/.

1 INTRODUCTION

World models predict how a scene evolves given its current state and the intended actions of the agents within it. Trained on internet-scale data, video-based world models (Bruce et al., 2024; Yang et al., 2023; Team et al., 2026; Agarwal et al., 2026) are now able to generate photorealistic visual predictions of complex scene dynamics that are difficult to model with traditional physics engines. Modeling the world in image space is thus becoming a promising alternative to hand-built physics modeling in such areas as robotics (Hou et al., 2026) and autonomous driving (Feng et al., 2025).

Existing world models usually generate predictions from the current frame or a short visual history. However, such observations may leave the underlying physical properties unconstrained, forcing the world models to guess a future state among multiple possible ones. For example, an opaque bottle appears identically whether or not it is filled with water, yet it will respond differently under identical actuation. A world model that relies only on empirical priors over object category or appearance will produce incorrect predictions in such scenarios, limiting its applicability in downstream applications that require accurate dynamics (Zhou et al., 2025; Quevedo et al., 2026). This motivates *calibrating* the world model to the hidden physical properties of each scene.

A natural way to resolving this physics ambiguity is to physically interact with the scene to infer its hidden properties, similar to how we humans would do for an object with unknown properties. Inspired by this, prior system identification methods (Wu et al., 2015; Yu et al., 2017; Kumar et al., 2021) attempt to construct accurate physical models with closed-form equations and fitting the predescribed physics parameters to the in-situ interaction data. However, real-world systems exhibit complex, coupled physics factors, thus building a geometric model or formulating closed-form equations can be challenging. Hence, to adapt to per deployment setting, existing world models (Gao et al., 2026) are typically finetuned on a small set of in-house data, which is computation expensive.

To this end, we propose PhysiCalWorld, a method for in-context **Physi**cs **Cal**ibration for **World** models. Inspired by the in-context learning capabilities of large sequence models (Brown et al., 2020; Bar et al., 2022), we prompt world models with a sequence of observation-action pairs collected

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.

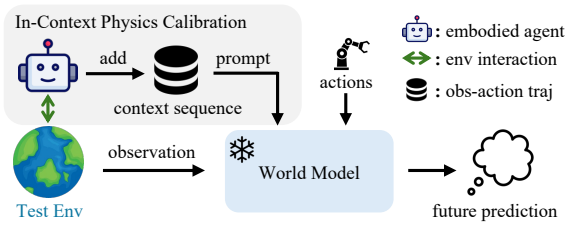


Figure 1: **In-Context Physics Calibration for World Models.** In a test environment with unknown transition dynamics, an embodied agent first executes exploratory actions to collect an in-context interaction sequence. This sequence is then provided as prompt to calibrate the world model such that future predictions adhere to the actual dynamics of the test environment.

in the test environment (*i.e.,* an environment whose physics configuration is unknown) for physics calibration. Our in-context prompting method sidesteps both the need for human efforts to build analytical physics simulators for system identification and expensive model finetuning. Our method requires only an exploratory interaction clip without any gradient descent steps. PhysiCalWorld compresses the interaction sequence into a compact *physics descriptor token* that conditions the world model. The entire system is trained end-to-end with only a video prediction objective, requiring neither ground-truth physics parameters nor a simulator.

At test time, PhysiCalWorld first extracts the descriptor from a few exploratory interactions with the test scene (e.g., via random exploration or scripted policy). This descriptor then conditions all subsequent rollouts in that scene, enabling the world model to generate physically consistent predictions in novel configurations without any test-time weight updates.

In summary, our contributions are three-fold:

- **In-Context Physics Calibration:** We formulate physics calibration of generative world models as in-context conditioning: from a short interaction sequence, the model adapts zero-shot to instance-specific dynamics, with no test-time weight updates.
- **Physics Descriptors Learned from Video Alone:** We introduce a *physics descriptor token* that grounds the world model in hidden physical properties. It is learned end-to-end from video prediction alone, requiring neither ground-truth physics parameters nor a simulator.
- **Performance and Analysis:** Across four environments with single and coupled physics variations, PhysiCalWorld consistently improves rollout accuracy and planning success on two world model backbones. Its single-token descriptor matches or exceeds raw-context conditioning at a fraction of the cost. Without physics labels, the descriptors form a physically structured latent space and extrapolate beyond the training parameter range.

2 Related Work

World Models for Robotics. World models predict future states conditioned on current observations and agent actions, serving as an increasingly foundational framework for robotics (Li et al., 2025c; Hou et al., 2026). For example, they are scalable neural simulators, enabling low-cost policy evaluation without real-world execution (Guo et al., 2026; Gao et al., 2026; Wang et al., 2026e; Team et al., 2025). In addition, through future imagination, world models can substantially improve sample efficiency in reinforcement learning (Hafner et al., 2019; Hansen et al., 2022; Seo et al., 2023; Wu et al., 2023) and facilitate policy learning (Saxena et al., 2026; Luo & Du, 2025). Moreover, optimizing action sequences directly through forward predictions enables embodied agents to complete novel tasks via visual trajectory planning (Zhou et al., 2025; Assran et al., 2025; Wang et al., 2024). However, existing methods typically condition only on the current frame or a narrow observation history, assuming that these observations suffice to identify the underlying transition dynamics of the test scene. As a result, predictions tend to suffer from severe drift if latent physical properties vary across test scenes. In this work, we ground model predictions in the actual test-scene dynamics by introducing an additional in-context interaction prompt collected within the test scene as input to the world model, substantially enhancing the prediction accuracy and visual planning performance.

In-Context Learning in Physical AI. Context can serve embodied agents in various forms. For example, it can adapt a robot policy to test-scene dynamics by conditioning directly on execution history or by explicitly recovering physical parameters (Lee et al., 2020; Kumar et al., 2021; Liu et al., 2025; Zhang et al., 2025). Another role is in-context policy learning, where task demonstrations steer actions either through structured representations such as keypoints and graphs (Di Palo & Johns, 2024; Vosylius & Johns, 2025; Hwang et al., 2024) or raw video-action pairs (Fu et al., 2024; Shah et al., 2025; Ding et al., 2026), and in-context reinforcement learning (Lee et al., 2023; Lu et al., 2023). Most recently, context has been applied to action-conditioned world models by prepending context prefixes to adapt to new camera viewpoints or embodiments (Yang et al., 2026). However, prior

works focus on policy adaptation, require privileged physical priors, or prepend raw context frames at high computational cost. In contrast, we use context to enable physics-grounded future prediction in world models, introducing a compact physics descriptor token to minimize computational overhead while maintaining high prediction accuracy and rollout efficiency.

Physics-Aware Video Generation. Video generative models have advanced rapidly (Yang et al., 2025; Zheng et al., 2025; Chen et al., 2026), with widespread commercial deployment (Seedance et al., 2026; Wan et al., 2025). However, their adherence to physical laws remains limited (Motamed et al., 2026; Upadhyay et al., 2026; Bansal et al., 2025; Meng et al., 2024). Recent efforts seek to improve physical awareness in video models by incorporating explicit physics equations or parameters in the generation process (Zhang et al., 2024; Wang et al., 2026a; Yuan et al., 2026; Wang et al., 2026b), coupling with external physics simulators (Liu et al., 2024; Xie et al., 2025; Foo et al., 2026), leveraging foundation models (Xue et al., 2025; Zhang et al., 2026b), or optimizing handcrafted rewards (Zhang et al., 2026a; Li et al., 2025a). However, these approaches either rely on analytical formulations and are restricted to a pre-defined set of physics properties, or focus on learning general physics and tend to fail in scenarios that require fine-grained instance-level physics grounding. In comparison, our work focuses on calibrating action-conditioned world models to the instance-specific dynamics of a test scene from an exploratory interaction prompt, operating entirely end-to-end without prior assumptions or ground-truth supervision of the environment's underlying physics.

3 METHOD

We consider an embodied agent interacting with an environment over discrete timesteps t. At each step, the agent receives a high-dimensional visual observation $o_t \in \mathcal{O} \subset \mathbb{R}^{H\times W\times 3}$ (and an optional proprioceptive state $p_t \in \mathbb{R}^{d_p}$) and executes a control action $a_t \in \mathcal{A} \subset \mathbb{R}^{d_a}$. Our objective is to learn an action-conditioned world model that predicts future observations directly in a compact latent space and enables both long-horizon rollout and visual planning.

Latent World Models. While our proposed in-context prompting framework is a plug-and-play module applicable to different world model architectures, we base the proposed method on two recent latent world models, DINO-WM (Zhou et al., 2025) and LeWM (Maes et al., 2026), due to their efficiency and strong trajectory planning performance. In these latent world models, each input RGB frame o_t is encoded by a visual encoder E_{v} into a set of visual tokens z_t:

$$z_t = E_{\text{v}}(o_t) \in \mathbb{R}^{N_z\times d_z}, \tag{1}$$

where N_z denotes the number of visual tokens per frame and d_z is the feature dimension. Meanwhile, actions a_t are mapped into action embeddings $e_t^a \in \mathbb{R}^{d'_a}$ via a learned MLP E_a:

$$e_t^a = E_a(a_t). \tag{2}$$

The latent dynamics predictor f_θ (a causal transformer with weights θ) models temporal transitions over a history window of K steps. At step t, the predictor takes a sequence of past visual tokens and action embeddings to predict the next latent tokens:

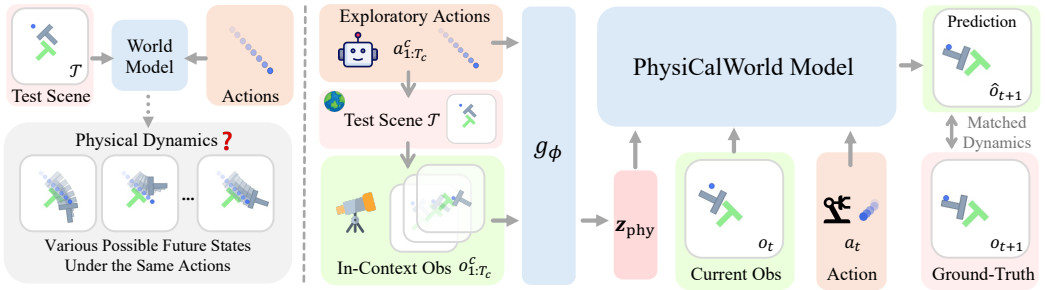
$$\hat{z}_{t+1} = f_\theta\left(\{z_i, e_i^a\}_{i=t-K+1}^{t}\right). \tag{3}$$

For visual evaluation and pixel-space rollouts, we train a lightweight convolutional decoder D_{v} that reconstructs images $\hat{o}_t$ from latent tokens, $\hat{o}_{t+1} = D_{\text{v}}(\hat{z}_{t+1})$, while downstream planning is conducted entirely within the latent space $\mathcal{Z}$.

3.1 PHYSICS CALIBRATION VIA PHYSICS DESCRIPTOR TOKEN

Physics Ambiguity in Visual World Models. The forward transition function $\mathcal{T}$ of an environment depends not only on the current states and actions, but also on a set of underlying physics properties $\xi \in \mathcal{X}$ (e.g., mass, surface friction, or joint damping). Because many of these physics properties rarely correlate with surface appearance, a static visual frame o_t or a brief observation window can hardly identify ξ and hence leave the true transition dynamics underdetermined.

Consequently, action-conditioned future prediction becomes an ill-posed, one-to-many problem. Without sufficient interaction data from the test scenes to calibrate the latent predictor f_θ, its prediction may collapse to the expectation over the training distribution with compounding dynamic drift when the test environment deviates from the nominal mean. While one could explicitly estimate and feed the test-scene physics parameters to f_θ, this usually relies on ad-hoc analytical modeling and also introduces cascading estimation errors. An alternative strategy is to scale up the history window K



(a) Uncalibrated World Model Imagining (b) **Physics Calibration for World Models (Ours)**

Figure 2: **Method Overview.** PhysiCalWorld calibrates action-conditioned world models to the actual dynamics of a test scene via prompting with in-context interaction. In **(a)**, current visual observation alone cannot identify the physical dynamics of the test scene $\mathcal{T}$, forcing the world model to guess among multiple potential future states. In **(b)**, starting from the top left, exploratory actions $a^c_{1:T_c}$ are first executed in the test scene $\mathcal{T}$, yielding observations $o^c_{1:T_c}$. The context encoder g_ϕ then extracts physics information from this obs-action sequence and compresses it into a physics descriptor token $\mathbf{z}_{\text{phy}}$. Conditioned on $\mathbf{z}_{\text{phy}}$, the world model is able to generate future prediction $\hat{o}_{t+1}$ consistent with the test scene's dynamics.

of world models, thus future predictor f_θ can directly extract the physics from previous interaction. However, self-attention complexity in transformer scales quadratically with sequence length, making long-history conditioning computationally prohibitive. Furthermore, continuously carrying long raw frame histories (e.g., thousands of tokens) in every inference step introduces substantial redundancy, as the underlying physics configuration can usually be represented far more compactly.

Physics Descriptor Token. To address these challenges, our method calibrates world models through a single compact representation, termed *physics descriptor token* $\mathbf{z}_{\text{phy}}$, which is extracted from an exploratory interaction sequence collected in the test scene. Formally, given an interaction clip $\mathcal{C} = \{(o^c_i, a^c_i)\}_{i=1}^{T_c}$ of length T_c, the context encoder g_ϕ maps it to a descriptor token $\mathbf{z}_{\text{phy}} = g_\phi(\mathcal{C}) \in \mathbb{R}^{d_{\text{phys}}}$. Internally, a lightweight visual encoder E^v_c extracts per-frame visual features and an action encoder E^a_c maps the control actions into action embeddings. At each timestep i, the two are concatenated along the channel dimension into a joint transition token $[E^v_c(o^c_i) \mid E^a_c(a^c_i)] \in \mathbb{R}^{d_v + d'_a}$. A transformer E^{fuse}_c then fuses these transition tokens across time and pools them via a learnable query token $\mathbf{z}_{\text{cls}}$:

$$\mathbf{z}_{\text{phy}} = g_\phi(\mathcal{C}) \coloneqq E^{\text{fuse}}_c\Big(\mathbf{z}_{\text{cls}},\ \big\{\big[\,E^v_c(o^c_i) \mid E^a_c(a^c_i)\,\big]\big\}_{i=1}^{T_c}\Big)_0, \tag{4}$$

where $[\,\cdot \mid \cdot\,]$ denotes channel-wise concatenation and $(\cdot)_0$ denotes the output at the query-token position. We use ϕ to collectively denote the parameters of g_ϕ, including E^{fuse}_c, E^v_c, E^a_c, and $\mathbf{z}_{\text{cls}}$. The world model's latent dynamics predictor f_θ is subsequently conditioned on $\mathbf{z}_{\text{phy}}$ alongside the default observation-action history window K:

$$\hat{z}_{t+1} = f_\theta\Big(\big\{(z_i, e^a_i)\big\}_{i=t-K+1}^{t},\ \mathbf{z}_{\text{phy}}\Big). \tag{5}$$

Efficiency and Reusability. The design of *physics descriptor token* provides two practical advantages. (1) *minimal rollout overhead*: adding a single additional token $\mathbf{z}_{\text{phy}}$ into the predictor f_θ introduces only a constant and almost negligible computational cost, circumventing the quadratic attention overhead compared to using longer history or simply prepending raw context frames to f_θ (See *+Long-History* and *+Context-Prefix* in Section 4 and Figure 6 for comparison); (2) *one-pass extraction and in-context reuse*: $\mathbf{z}_{\text{phy}}$ is computed via a single forward pass of a lightweight context encoder and can then be cached. If the dynamics of the test scene remains unchanged, the same descriptor can be reused across arbitrarily many future rollouts or planning tasks with no additional encoding cost.

3.2 Training PhysiCalWorld without Physics Labels

To enable physics-calibrated predictions, existing models typically rely on pre-specified analytical equations (Li et al., 2025b; Wang et al., 2026c) or train explicit system identification modules using privileged physical parameters (Yu et al., 2017; Kumar et al., 2021). However, constructing analytical physics formulations often demands substantial human effort and is often intractable for unstructured scenes with coupled dynamics. Meanwhile, obtaining the ground-truth physics parameters requires instrumented environments and often introduces measurement errors, which would then propagate to model training. As a result, these approaches are typically confined to controlled settings and struggle to scale to novel, uncalibrated in-the-wild scenes governed by unknown physical properties.

In contrast, PhysiCalWorld requires neither prior physics equations nor the error-prone explicit parameter estimation during both training and evaluation. While deriving analytical formulations or measuring exact physical parameters of in-the-wild scenes is intractable, constructing data groups where each group consists of interaction data collected within the same physical environment is simple and requires minimum additional efforts during data collection. Such data grouping alone is sufficient to train PhysiCalWorld and learn an effective *physics descriptor token*.

Specifically, the training dataset is categorized into M groups $\mathcal{D}_{\text{train}} = \{\mathcal{G}_j\}_{j=1}^{M}$, where each group $\mathcal{G}_j$ comprises episodes that share the same underlying physics configuration ξ_j. In training, for any transition $(o_t, a_t, o_{t+1}) \sim \mathcal{G}_j$, we independently sample another clip $\mathcal{C}_{\text{tr}}^j = \{(o_k^c, a_k^c)\}_{k=1}^{T_c} \sim \mathcal{G}_j$ from the same group as interaction prompt. Because both sequences share the identical underlying transition dynamics $\mathcal{T}_{\xi_j}$, the context encoder g_ϕ can learn to distill the scene's physical characteristics into the compact descriptor token $\mathbf{z}_{\text{phy}}$ to synthesize the correct prediction. We jointly optimize the context encoder g_ϕ and dynamics predictor f_θ end-to-end using the latent prediction objective:

$$\mathcal{L}(\theta, \phi) = \mathbb{E}_{\mathcal{G}_j \sim \mathcal{D}_{\text{train}},\, (o_t, a_t, o_{t+1}) \sim \mathcal{G}_j,\, \mathcal{C}_{\text{tr}}^j \sim \mathcal{G}_j} \Big[\big\| f_\theta\big(z_t, e_t^a, \mathbf{z}_{\text{phy}}\big) - z_{t+1} \big\|_2^2 \Big], \quad \text{where } z_t = E_v(o_t),\ e_t^a = E_a(a_t),\ z_{t+1} = E_v(o_{t+1}),\ \mathbf{z}_{\text{phy}} = g_\phi\big(\mathcal{C}_{\text{tr}}^j; \phi\big). \tag{6}$$

For simplicity, we assume the input window size K of f_θ to be 1 in Equation 6. This formulation enables PhysiCalWorld to extract instance-specific physical dynamics directly from the interaction clip $\mathcal{C}_{\text{tr}}$ into the compact physics descriptor token $\mathbf{z}_{\text{phy}}$, circumventing the need for ground-truth physics labels as supervision. As in Figure 7, this self-supervised predictive objective can autonomously learn a structural latent space correlated to the underlying physics variations of the scenes.

3.3 Inference with PhysiCalWorld

Collecting the Interaction Prompt. At test time, PhysiCalWorld enables zero-shot adaptation to environments with novel physics configurations without fine-tuning. To ground the physics of an unseen test scene, the agent first executes an exploratory sequence to collect a context prompt $\mathcal{C}_{\text{eval}} = \{(o_k^e, a_k^e)\}_{k=1}^{T_c}$. The control actions $\{a_k^e\}_{k=1}^{T_c}$ can be either randomly sampled or generated by a task-agnostic playful policy, as this interaction prompt is only for inducing the physics dynamics of the scene. However, these actions should produce physical contact with the environment since interactions lacking contact or motion cannot reveal the underlying dynamics.

Physics-grounded Rollout. Given $\mathcal{C}_{\text{eval}}$, the physics descriptor token is computed once by the context encoder g_ϕ and can be held fixed if the test scene dynamics remains unchanged. Calibrated by $\mathbf{z}_{\text{phy}}^{\text{ev}}$, the predictor f_θ can then predict the future latents as:

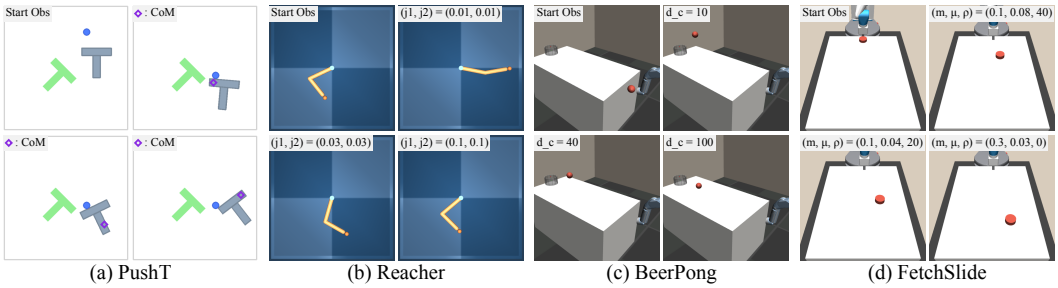
$$\hat{z}_{t+1} = f_\theta\Big(\big\{(z_i, e_i^a)\big\}_{i=t-K+1}^{t},\ \mathbf{z}_{\text{phy}}^{\text{ev}}\Big), \quad \mathbf{z}_{\text{phy}}^{\text{ev}} = g_\phi(\mathcal{C}_{\text{eval}}), \tag{7}$$

where predicted latents $\hat{z}_i$ can be fed back to f_θ as observation to enable long-horizon open-loop rollout (see Algorithm 2 for pseudocode). If the intrinsic physics of the scene does not change, $\mathbf{z}_{\text{phy}}$ can be shared across every rollout step and across every candidate trajectory to be evaluated during visual planning. Therefore, the prompt only needs to be encoded once per scene, regardless of the planning horizon or the number of sampled action sequences. Notably, no physical parameters, analytical models, or task-specific fine-tuning are required at inference: adapting PhysiCalWorld to a new scene only requires collecting a period of in-context interaction.

4 Experiments

Our experiments evaluate four questions: **(1)** Physics Calibration: Does conditioning on in-context interaction clips enable effective physics calibration and yield more accurate future predictions? **(2)** Descriptor Expressiveness and Efficiency: Can PhysiCalWorld learn compact, expressive *physics descriptor tokens* without analytical physics priors and enable high compute efficiency? **(3)** Coupled Physics Dynamics: Can PhysiCalWorld scale to complex scenes whose dynamics are jointly governed by multiple physics properties? **(4)** Backbone Generalization: Can PhysiCalWorld function as an architecture-agnostic, plug-and-play module across diverse world-model backbones?

Baselines. Our in-context prompting framework can be integrated into various world model backbones. We use DINO-WM (Zhou et al., 2025) and LeWM (Maes et al., 2026) as our base world models due to their rollout efficiency and strong trajectory planning performance, alongside Temporal Straightening (TS-WM) (Wang et al., 2026d) as a competitive latent world model baseline.



(a) PushT (b) Reacher (c) BeerPong (d) FetchSlide

Figure 3: **Environment Demonstrations.** We evaluate our method on four simulation environments featuring diverse variations of physics properties. For each environment, we show four images: the top-left image is a random initial observation, while the remaining three images demonstrate how executing the *same* action sequence yields different future states due to different physics parameters of the scenes. Specifically, in (a) Push-T, the center of mass of the T-block is varied; in (b) Reacher, the joint damping of the two-link arm is varied; in (c) BeerPong, the energy dissipation upon table contact is varied; in (d) FetchSlide, *three* physics properties vary concurrently: puck mass, surface friction, and aerodynamic drag.

	Push-T					Reacher				
Method	PSNR↑	SSIM↑	LPIPS↓	IoU↑	SR(%)↑	PSNR↑	SSIM↑	LPIPS↓	IoU↑	SR(%)↑
TS-WM	21.38	0.954	0.0405	53.4	53.3	19.51	0.967	0.0469	30.9	41.3
DINO-WM	21.87	0.955	0.0402	51.5	53.3	19.43	0.968	0.0407	32.6	52.0
+ Long History	22.41	0.959	0.0331	57.6	57.3	20.47	0.971	0.0311	41.2	53.3
+ Context Prefix	22.98	0.961	0.0295	61.4	58.7	20.73	0.972	0.0290	44.1	61.3
+ Estimated Physics†	23.11	0.961	0.0319	59.0	57.3	**21.82**	**0.976**	**0.0221**	**51.5**	**69.3**
+ PhysiCal (Ours)	**23.83**	**0.964**	**0.0267**	**65.1**	**62.7**	21.58	0.975	0.0222	51.0	61.3
LeWM	23.72	0.957	0.0450	64.3	48.0	20.45	**0.964**	0.0404	40.6	60.0
+ PhysiCal (Ours)	**25.50**	**0.964**	**0.0358**	**73.3**	**61.3**	**21.08**	**0.964**	**0.0377**	**45.7**	**64.0**

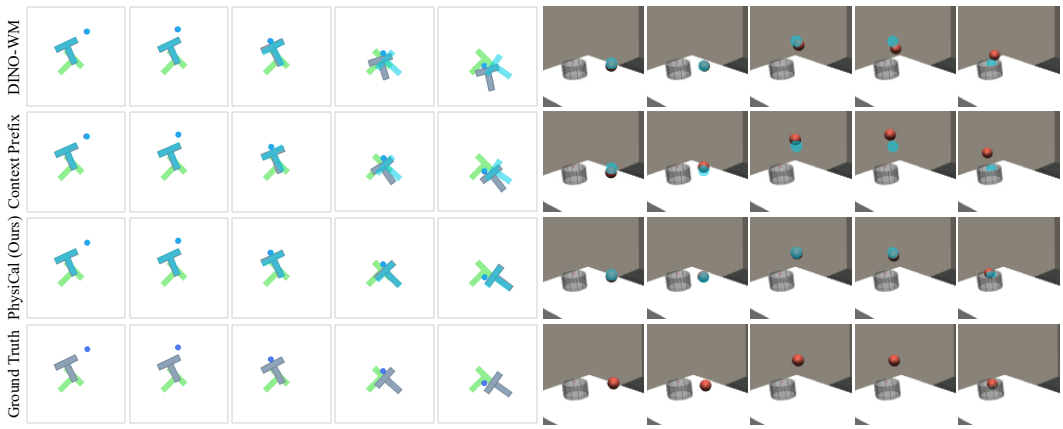
Table 1: **Push-T with Varying Center of Mass of the T-block:** We evaluate on 15 unseen Center of Mass (CoM) and report the quantitative visual metrics and planning success rates (SR). IoU is computed on the combined mask of the agent and the T-block. Different CoMs will significantly affect the motion pattern given the same contact, hence lack of physics calibration will result in low IoU. **Reacher with Different Joint Damping:** The robot arm contains 2 joints and we evaluate on 15 unseen joint damping combinations. Different damping values alter resistance to motion, resulting in different displacement amplitude of the arm under identical torques (see Figure 3).

We train several variants of models for systematic study: **(1)** +*Long History*, which takes in a long history window K to investigate if simply extending history memory in world models can enable physics-calibrated predictions; **(2)** +*Context Prefix*, which directly prepends the raw context interaction prompt to the input of the world model instead of using the proposed physics descriptor token; **(3)** +*Estimated Physics*†, which trains an encoder to predict physics parameters from a context clip and feeds the estimated physics values to the world model (hence this baseline requires additional physics labels for model training); For further analysis, we additionally evaluate **(4)** +*Privileged Physics*‡ in Section 4.3, which takes the ground-truth test-scene physics parameters as input, serving as an oracle ceiling rather than a directly comparable baseline; Finally, we compare LeWM against LeWM+PhysiCal to verify that our method generalizes across different world model backbones. See Appendix A.1 for detailed description of each baseline.

Environments and Evaluation. We illustrate all our environments in Figure 3 and show how different physics configurations will affect the future world states. We train the models on $M = 135$ different configurations and report the results on 15 unseen configurations, with 5 validation episodes per configuration, giving 75 episodes in total. We report four visual metrics (PSNR, SSIM, LPIPS, IoU) in open-loop rollout, which starts from the initial frame and autoregressively predicts all frames of an episode by recursively feeding the model's own predictions back as input. We also evaluate on 75 planning tasks per environment and report the success rate (SR). Each task is defined by a start image and a goal image. We execute the planned actions in the environments. The success criterion is whether the target object reaches the goal within a threshold. See Appendix B for details.

4.1 Environments with Single Physics Variation

Push-T under Varying Center of Mass. The Push-T environment consists of a circular pusher (the agent) and a T-shaped block (See Figure 3). The world model predicts the future latents of the scene



(a) Push-T (b) BeerPong

Figure 4: **Qualitative Open-Loop Rollout Results in Push-T and BeerPong.** We show a subset of the full rollouts due to limited space and crop the synthesized images in BeerPong for clearer view. We use the cyan masks to indicate the ground-truth location of the target objects. Note that the physics configuration of these test scenes is unknown to the models and PhysiCal calibrates the predictions solely based on a given in-context interaction clip (Figure A2 and Figure A3) and yield results closely match the ground-truth.

alongside the pusher's proprioception conditioned on the pusher's actions. In planning, each goal is defined by an image and a target agent state. The block's center of mass (CoM) is randomly sampled. Different CoMs will not change the block's appearance but will alter how the block rotates and slides when pushed, hence resulting in significantly different state even under the identical push actions.

As in Table 1, methods without in-context prompts as input (e.g., DINO-WM and TS-WM) generally underperform, showing that in-context interaction prompts can effectively facilitate physics calibration. We observe that compared to other long-context variants (e.g., +Long History and +Context Prefix, see Appendix A.1 for details), +PhysiCal achieves significantly better performance, especially in challenging open-loop rollout, showing that a single descriptor token can stably ground the physics over multi-round long-horizon rollout. Furthermore, our method also outperforms +Estimated Physics† in all open-loop rollout visual metrics without using any physics labels during training.

Reacher Control under Varying Joint Damping. This environment consists of a two-link robot arm actuated by continuous joint torques. The damping coefficients of the two joints can affect how much the arm moves under the same applied torques. Each planning task optimizes a torque trajectory to steer the arm toward a target state specified by a goal image. The in-context prompts are randomly-sampled actions. We present the quantitative results in Table 1. Our in-context conditioning consistently improves both DINO-WM and LeWM, outperforming all unprivileged baselines and trailing only *+Estimated Physics*, which requires ground-truth joint damping parameters for model training. While +PhysiCal achieves a similar planning success rate as *+Context Prefix*, our method surpasses it across all visual prediction metrics by a margin. Moreover, our compact descriptor token design avoids the quadratic self-attention cost caused by the long context sequences, yielding approximately $10\times$ speedup in planning time (see Figure 6) along with markedly lower training overhead (see Table A5). We specify implementation details in Appendix B.2.

BeerPong under Varying Impact Energy Dissipation. In BeerPong, a robot arm throws a ball toward a target cup on a tabletop. The ball needs to bounce off the table at least once to enter the cup due to the long distance. We vary impact energy dissipation by changing the table's contact damping (e.g., higher damping causes greater energy loss, resulting in lower bounce height). The action input to the world models is the end-effector release velocity. The cup location is randomly initialized in each planning task. The planner then optimizes the release velocity so that the ball can land right inside the cup after bounce. The in-context prompts are collected via randomly-sampled actions. Please see Appendix B.3 for additional details.

We present quantitative results in Table 2. Across both world-model backbones, +PhysiCal consistently improves visual prediction accuracy over the uncalibrated baselines. While our method surpasses *+Context Prefix* across all visual metrics, it achieves a slightly lower planning success rate. This is potentially because the physics variation in BeerPong manifests only when the ball contacts the table, leading to temporally sparse physical cues. Under such observation sparsity, self-attention over the full raw context frames may provide an advantage in identifying collision dynamics.

Method	BeerPong PSNR↑	SSIM↑	LPIPS↓	IoU↑	SR (%)↑	FetchSlide PSNR↑	SSIM↑	LPIPS↓	IoU↑	SR (%)↑
TS-WM	26.97	0.992	0.0129	24.8	60.0	24.07	0.984	0.0131	49.9	26.7
DINO-WM	27.35	0.993	0.0125	27.7	44.0	25.67	0.989	0.0077	58.6	26.7
+ Long History	27.11	0.993	0.0138	24.8	50.7	25.17	0.989	0.0090	51.0	24.0
+ Context Prefix	29.10	0.994	0.0089	40.8	**82.7**	27.14	**0.991**	0.0048	71.5	54.7
+ Estimated Physics†	**29.58**	**0.995**	0.0074	44.0	77.3	26.93	0.990	0.0052	69.6	**69.3**
+ PhysiCal (Ours)	29.49	**0.995**	**0.0067**	**46.3**	80.0	**27.87**	**0.991**	**0.0047**	**73.8**	56.0
LeWM*	25.05	0.983	0.0230	27.2	60.0	25.15	0.981	0.0336	51.1	17.3
+ PhysiCal (Ours)	**27.58**	**0.986**	**0.0170**	**47.7**	**84.0**	**26.43**	**0.982**	**0.0321**	**61.8**	**28.0**

Table 2: **BeerPong with Different Ball-contact Damping.** In BeerPong, we vary table contact damping, which defines the ball's rebound height and trajectory. For planning, the agent needs to throw the ball such that it bounces off the table and lands inside the target cup. IoU is computed on the ball foreground mask. **FetchSlide with Multiple Physics Variations.** This environment simultaneously varies three physical properties: puck mass, table surface friction, and air drag, which jointly determine how the puck slides. In planning, the agent is required to strike the puck to a given location. Estimated Physics† requires environment physics configurations for model training, while ours does not. LeWM*: we use TC-LeWM in BeerPong to enhance performance (see Appendix A.1 for details).

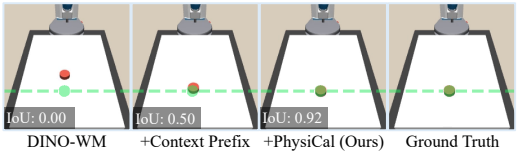


Figure 5: **Qualitative Comparison in FetchSlide.** Visual comparison of open-loop rollout predictions under identical striking actions in an unseen test scene. We only demonstrate the final frame due to space limit. The green mask denotes the ground-truth puck position.

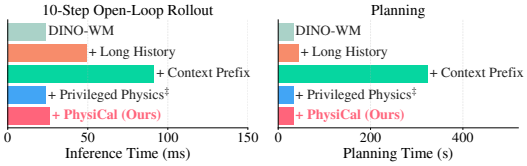


Figure 6: **Inference Speed Comparison of Open-Loop Rollout and Planning in Push-T.** For planning, we use the same Cross-Entropy Method (CEM) setup as DINO-WM. See Appendix C.1 and C.2 for additional results.

4.2 Environments with Multiple Physics Variations

FetchSlide under Multiple Physics Variations. We next investigate whether PhysiCal can calibrate world models in environments where transition dynamics are simultaneously governed by *multiple* physics properties. Hence, in FetchSlide, we concurrently vary three physics parameters: puck mass, table surface friction, and aerodynamic drag. In this environment, a robot arm strikes a flat puck across a tabletop, where these three properties jointly determine the puck's post-strike deceleration and sliding distance. The action space is end-effector delta position, which subsequently defines the striking velocity and angle. In planning, we optimize the strike actions so that the puck can stop at the target position specified by a goal image. The context sequence is collected via random actions.

Quantitative results are presented in Table 2. Despite the added complexity of coupled physical factors, +PhysiCal effectively calibrates predictions without requiring analytical physics equations, yielding significant gains across both backbones. Our method is only surpassed by *+Estimated Physics*†, which is trained with the ground-truth values of all three physical properties. However, precisely measuring such parameters is usually impractical, whereas +PhysiCal can be trained directly via future latent prediction objective. We also provide qualitative results of the last predictions after an open-loop rollout in Figure 5, where uncalibrated DINO-WM noticeably deviates from the target position, whereas PhysiCal generates predictions that faithfully capture the ground-truth puck state.

4.3 Analysis

Physics Calibration *vs.* Privileged Physics‡. We additionally compare our method against an oracle baseline, *+Privileged Physics*‡, serving as an upper-bound reference. Specifically, this baseline directly receives ground-truth physics parameters as supplementary inputs during both training and inference. In contrast, PhysiCal accesses only an unannotated interaction clip from the test scene. Nevertheless, as shown in Table 3, our method achieves competitive performance across most metrics and even surpasses this privileged baseline on Push-T open-loop rollouts.

Generalization to Out-of-Distribution Physics Parameters. We investigate whether PhysiCalWorld can extrapolate to environments where physics parameters lie outside the training distribution (Figure 8). Specifically, in Reacher, we sweep the first joint's damping coefficient beyond its training range while holding the second joint's damping fixed.

Method	Push-T PSNR↑	Push-T IoU↑	Push-T SR↑	Reacher PSNR↑	Reacher IoU↑	Reacher SR↑
DINO-WM	21.87	0.515	53.3	19.43	0.326	52.0
+ Privileged Physics‡	23.36	0.609	60.0	**22.92**	**0.593**	**86.7**
+ PhysiCal (Ours)	**23.83**	**0.651**	**62.7**	21.58	0.510	61.3

Method	FetchSlide PSNR↑	FetchSlide IoU↑	FetchSlide SR↑	BeerPong PSNR↑	BeerPong IoU↑	BeerPong SR↑
DINO-WM	25.67	0.586	26.7	27.35	0.277	44.0
+ Privileged Physics‡	27.68	**0.758**	**66.7**	**30.27**	**0.503**	**86.7**
+ PhysiCal (Ours)	**27.87**	0.738	56.0	29.49	0.463	80.0

Table 3: **Quantitative Comparison of +PhysiCal and Privileged Physics‡.** +PhysiCal calibrates physics only through an interaction clip collected in the test scene, while privileged physics‡ directly use the ground-truth physics parameters of the test scene as input to calibrate the world models.

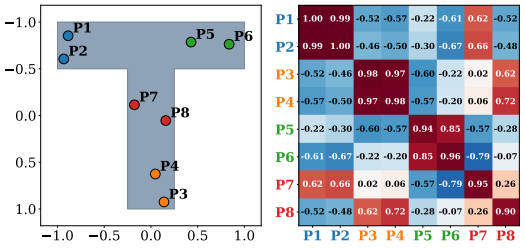


Figure 7: **Cosine Similarity of Physics Descriptor Tokens of Different CoMs in Push-T.** We compute descriptor token $\mathbf{z}_{\text{phy}}$ of 8 unseen centers of mass and visualize their pairwise cosine similarity. Trained without any physics labels, our method can still learn a structural, physics-correlated latent space, e.g., adjacent centers of mass yield high similarity while remote pairs yield negative values.

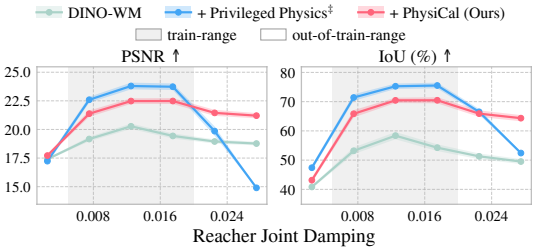


Figure 8: **Extrapolation to Out-of-Distribution Physics Parameters in Reacher.** We report open-loop rollout results. For each circle, we sample 15 damping values within a bin centered at its x-coordinate evaluated on 5 episodes per value, 75 total.

While *+Privileged Physics* achieves higher accuracy in-distribution, its performance deteriorates sharply when the damping falls below or exceeds the training interval. In particular, its rollout PSNR significantly drops, producing blurry predictions (see Figure A6). In contrast, our method demonstrates robust extrapolation over unseen regimes, probably because our model learns latent pattern to infer dynamics rather than overfitting to rigid scalars.

Interpretability and Latent Structure of the Descriptor $\mathbf{z}_{\text{phy}}$. We analyze whether the physics descriptor learns structured representations of the test-scene physics. Specifically, in Push-T, we collect 10 interaction clips in each of 8 unseen center of mass locations (P_1–P_8) and encode each clip into a physics descriptor token using context encoder g_ϕ. We then compute the pairwise cosine similarities between these tokens and average them by CoM. In Figure 7, descriptors from adjacent CoM locations exhibit strong positive correlation, while CoMs that are far apart yield negative similarity values. This indicates that our training objective enables the context encoder to capture meaningful physics properties correlated with the scene dynamics, even without using ground-truth physics labels during training.

5 Discussion

Limitations. Our approach has several limitations. First, training PhysiCalWorld requires grouping trajectories by the physics configurations under which they were collected. While partitioning data by environment identity is far less demanding than deriving analytical equations, curating grouped data from in-the-wild video remains non-trivial. One potential direction to relax this is leveraging self-supervised representation learning to automatically cluster latent transition dynamics into environment groups. Moreover, the proposed test-time calibration depends on the quality of the interaction prompt given to the model. While we find that simple playful or even random policies often suffice, future work could explore active or curiosity-driven exploration to collect more informative prompts. Finally, our framework assumes the physics properties are stationary: the dynamics remain unchanged within an episode. Extending this approach to non-stationary environments (e.g., fluctuating wind fields) may require additional techniques such as reactive prompting or online descriptor updates.

Conclusion. We believe that world models should go beyond generic visual fidelity toward instance-specific physics-grounded prediction. In this work, we take a step toward this goal by introducing an in-context calibration framework that grounds action-conditioned world models in the actual physics dynamics of novel test scenes. We achieve this by prompting the model with an interaction sequence collected within the test scene, enabling zero-shot calibration without finetuning or analytical physics priors. Evaluations across diverse environments demonstrate that our approach substantially improves predictive accuracy and downstream visual planning.

AI USE STATEMENT

In this work, we used generative AI tools for propose or refine hypotheses, design or provide feedback on research methodology or experiments, implement methods, assist with translation, support qualitative and thematic data analysis, interpret results. We have not used generative AI tools for Generate synthetic data sets, help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, clean and reformat dataset, assist in the writing of proofs, and none of the required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for create or edit software code, creation of artifacts, edit a research paper to improve readability. We have reviewed all AI-assisted work. For example, we checked LLM-generated text when we use it to polish our writing. 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 physics calibration for visual world models using simulated robotic environments. All datasets used in our experiments are generated programmatically from simulation environments and do not contain personal or personally identifiable information. Our experiments do not involve human participants or the collection or analysis of personal data. We do not identify any specific potentially harmful insights, methodologies, or applications arising from this work beyond the general risks associated with autonomous robotic systems.

REPRODUCIBILITY STATEMENT

To facilitate reproducibility, we provide model architectures, baseline implementations, training hyperparameters, context-sampling procedures, and algorithm pseudocode in Appendix A. Appendix B describes the simulation environments, physical-parameter variations, data collection procedures, training and evaluation splits, and evaluation protocols for visual prediction and planning. We further report training hardware and wall-clock durations in Appendix C.1.

REFERENCES

Niket Agarwal, Arslan Ali, Jon Allen, Martin Antolini, Adeline Aubame, Alisson Azzolini, Junjie Bai, Maciej Bala, Yogesh Balaji, Josh Bapst, et al. Cosmos 3: Omnimodal world models for physical ai. *arXiv preprint arXiv:2606.02800*, 2026. 1

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

Hritik Bansal, Zongyu Lin, Tianyi Xie, Zeshun Zong, Michal Yarom, Yonatan Bitton, Chenfanfu Jiang, Yizhou Sun, Kai-Wei Chang, and Aditya Grover. Videophy: Evaluating physical commonsense for video generation. In *International Conference on Learning Representations*, volume 2025, pp. 102075–102121, 2025. 3

Amir Bar, Yossi Gandelsman, Trevor Darrell, Amir Globerson, and Alexei Efros. Visual prompting via image inpainting. *Advances in neural information processing systems*, 35:25005–25017, 2022. 1

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 1

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, et al. Genie: Generative interactive environments. In *Forty-first international conference on machine learning*, 2024. 1

Junsong Chen, Yuyang Zhao, Jincheng Yu, Ruihang Chu, Junyu Chen, Shuai Yang, Xianbang Wang, Yicheng Pan, Zhou Daquan, Huan Ling, et al. Sana-video: Efficient video generation with block linear diffusion transformer. In *International Conference on Learning Representations*, volume 2026, pp. 38973–38997, 2026. 3

Rodrigo de Lazcano, Kallinteris Andreas, Jun Jet Tai, Seungjae Ryan Lee, and Jordan Terry. Gymnasium robotics, 2024. URL https://github.com/Farama-Foundation/Gymnasium-Robotics. 23

Norman Di Palo and Edward Johns. Keypoint action tokens enable in-context imitation learning in robotics. *arXiv preprint arXiv:2403.19578*, 2024. 2

Jian Ding, Xianjie Dai, Roei Herzig, Nussair Hroub, Jinjie Mai, Dengxin Dai, Bernard Ghanem, and Mohamed Elhoseiny. Contextflow: In-context flow matching for robot manipulation. *arXiv preprint arXiv:2609.06852*, 2026. 2

Tuo Feng, Wenguan Wang, and Yi Yang. A survey of world models for autonomous driving. *arXiv preprint arXiv:2501.11260*, 2025. 1

Lin Geng Foo, Mark He Huang, Alexandros Lattas, Stylianos Moschoglou, Thabo Beeler, and Christian Theobalt. Physical simulator in-the-loop video generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4301–4311, 2026. 3

Letian Fu, Huang Huang, Gaurav Datta, Lawrence Yunliang Chen, William Chung-Ho Panitch, Fangchen Liu, Hui Li, and Ken Goldberg. In-context imitation learning via next-token prediction. *arXiv preprint arXiv:2408.15980*, 2024. 2

Shenyuan Gao, William Liang, Kaiyuan Zheng, Ayaan Malik, Seonghyeon Ye, Sihyun Yu, Wei-Cheng Tseng, Yuzhu Dong, Kaichun Mo, Chen-Hsuan Lin, et al. Dreamdojo: A generalist robot world model from large-scale human videos. *arXiv preprint arXiv:2602.06949*, 2026. 1, 2

Yanjiang Guo, Lucy Shi, Jianyu Chen, and Chelsea Finn. Ctrl-world: A controllable generative world model for robot manipulation. In *International Conference on Learning Representations*, volume 2026, pp. 6121–6138, 2026. 2

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pp. 2555–2565. PMLR, 2019. 2

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. *arXiv preprint arXiv:2203.04955*, 2022. 2

Bohan Hou, Gen Li, Jindou Jia, Tuo An, Xinying Guo, Sicong Leng, Haoran Geng, Yanjie Ze, Tatsuya Harada, Philip Torr, et al. World model for robot learning: A comprehensive survey. *arXiv preprint arXiv:2605.00080*, 2026. 1, 2

Minyoung Hwang, Luca Weihs, Chanwoo Park, Kimin Lee, Aniruddha Kembhavi, and Kiana Ehsani. Promptable behaviors: Personalizing multi-objective rewards from human preferences. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16216–16226. IEEE, 2024. 2

Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021. 1, 2, 4

Jonathan Lee, Annie Xie, Aldo Pacchiano, Yash Chandak, Chelsea Finn, Ofir Nachum, and Emma Brunskill. Supervised pretraining can learn in-context reinforcement learning. *Advances in Neural Information Processing Systems*, 36:43057–43083, 2023. 2

Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science robotics*, 5(47):eabc5986, 2020. 2

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*, 2025a. 3

Wenxuan Li, Hang Zhao, Zhiyuan Yu, Yu Du, Qin Zou, Ruizhen Hu, and Kai Xu. Pin-wm: Learning physics-informed world models for non-prehensile manipulation. *arXiv preprint arXiv:2504.16693*, 2025b. 4

Xinqing Li, Xin He, Le Zhang, Min Wu, Xiaoli Li, and Yun Liu. A comprehensive survey on world models for embodied ai. *arXiv preprint arXiv:2510.16732*, 2025c. 2

Chang Liu, Fei Suo, Yanzhou Jin, Zeyu Ping, Yusuke Iwasawa, Yutaka Matsuo, and Yaonan Zhu. Temporally centered sigreg improves leworldmodel representations for robot policy learning, 2026. URL https://arxiv.org/abs/2607.26924. 17

Min Liu, Deepak Pathak, and Ananye Agarwal. Locoformer: Generalist locomotion via long-context adaptation. *arXiv preprint arXiv:2509.23745*, 2025. 2

Shaowei Liu, Zhongzheng Ren, Saurabh Gupta, and Shenlong Wang. Physgen: Rigid-body physics-grounded image-to-video generation. In *European Conference on Computer Vision*, pp. 360–378. Springer, 2024. 3

Chris Lu, Yannick Schroecker, Albert Gu, Emilio Parisotto, Jakob Foerster, Satinder Singh, and Feryal Behbahani. Structured state space models for in-context reinforcement learning. *Advances in Neural Information Processing Systems*, 36:47016–47031, 2023. 2

Yunhao Luo and Yilun Du. Grounding video models to actions through goal conditioned exploration. In *International Conference on Learning Representations*, volume 2025, pp. 92200–92232, 2025. 2

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworld-model: Stable end-to-end joint-embedding predictive architecture from pixels. *arXiv preprint arXiv:2603.19312*, 2026. 3, 5, 17, 18

Fanqing Meng, Jiaqi Liao, Xinyu Tan, Wenqi Shao, Quanfeng Lu, Kaipeng Zhang, Yu Cheng, Dianqi Li, Yu Qiao, and Ping Luo. Towards world simulator: Crafting physical commonsense-based benchmark for video generation. *arXiv preprint arXiv:2410.05363*, 2024. 3

Saman Motamed, Laura Culp, Kevin Swersky, Priyank Jaini, and Robert Geirhos. Do generative video models understand physical principles? In *2026 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 948–958. IEEE, 2026. 3

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 17

Fabian Otto, Onur Celik, Dominik Roth, and Hongyi Zhou. Fancy gym. URL https://github.com/ALRhub/fancy_gym. 22

Julian Quevedo, Ansh Kumar Sharma, Yixiang Sun, Varad Suryavanshi, Percy Liang, and Sherry Yang. Worldgym: World model as an environment for policy evaluation. In *International Conference on Learning Representations*, volume 2026, pp. 78932–78957, 2026. 1

Vaibhav Saxena, Yunhao Luo, Yotto Koga, and Danfei Xu. Nostra: Enabling robust robot imitation via multimodal latent imagination. *IEEE Robotics and Automation Letters*, 2026. 2

Team Seedance, De Chen, Liyang Chen, Xin Chen, Ying Chen, Zhuo Chen, Zhuowei Chen, Feng Cheng, Tianheng Cheng, Yufeng Cheng, et al. Seedance 2.0: Advancing video generation for world complexity. *arXiv preprint arXiv:2604.14148*, 2026. 3

Younggyo Seo, Danijar Hafner, Hao Liu, Fangchen Liu, Stephen James, Kimin Lee, and Pieter Abbeel. Masked world models for visual control. In *Conference on Robot Learning*, pp. 1332–1344. PMLR, 2023. 2

Rutav Shah, Shuijing Liu, Qi Wang, Zhenyu Jiang, Sateesh Kumar, Mingyo Seo, Roberto Martín-Martín, and Yuke Zhu. Mimicdroid: In-context learning for humanoid robot manipulation from human play videos. *arXiv preprint arXiv:2509.09769*, 2025. 2

Yuval Tassa, Yotam Doron, Alistair Muldal, Tom Erez, Yazhe Li, Diego de Las Casas, David Budden, Abbas Abdolmaleki, Josh Merel, Andrew Lefrancq, et al. Deepmind control suite. *arXiv preprint arXiv:1801.00690*, 2018. 22

Gemini Robotics Team, Krzysztof Choromanski, Coline Devin, Yilun Du, Debidatta Dwibedi, Ruiqi Gao, Abhishek Jindal, Thomas Kipf, Sean Kirmani, Isabel Leal, et al. Evaluating gemini robotics policies in a veo world simulator. *arXiv preprint arXiv:2512.10675*, 2025. 2

Robbyant Team, Zelin Gao, Qiuyu Wang, Yanhong Zeng, Jiapeng Zhu, Ka Leong Cheng, Yixuan Li, Hanlin Wang, Yinghao Xu, Shuailei Ma, et al. Advancing open-source world models. *arXiv preprint arXiv:2601.20540*, 2026. 1

Rishi Upadhyay, Howard Zhang, Jim Solomon, Ayush Agrawal, Pranay Boreddy, Shruti Satya Narayana, Yunhao Ba, Alex Wong, Celso M de Melo, and Achuta Kadambi. Worldbench: Disambiguating physics for diagnostic evaluation of world models. *arXiv preprint arXiv:2601.21282*, 2026. 3

Vitalis Vosylius and Edward Johns. Instant policy: In-context imitation learning via graph diffusion. In *International Conference on Learning Representations*, volume 2025, pp. 35011–35030, 2025. 2

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 3

Chen Wang, Chuhao Chen, Yiming Huang, Zhiyang Dou, Yuan Liu, Jiatao Gu, and Lingjie Liu. Physctrl: Generative physics for controllable and physics-grounded video generation. *Advances in Neural Information Processing Systems*, 38:167907–167932, 2026a. 3

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. *Advances in Neural Information Processing Systems*, 38:5388–5416, 2026b. 3

Meizhong Wang, Wanxin Jin, Kun Cao, Lihua Xie, and Yiguang Hong. Contactgaussian-wm: Learning physics-grounded world model from videos. *arXiv preprint arXiv:2602.11021*, 2026c. 4

Ying Wang, Oumayma Bounou, Gaoyue Zhou, Randall Balestriero, Tim GJ Rudner, Yann LeCun, and Mengye Ren. Temporal straightening for latent planning. *arXiv preprint arXiv:2603.12231*, 2026d. 5, 17, 19

Yixuan Wang, Rhythm Syed, Fangyu Wu, Mengchao Zhang, Aykut Onol, Jose Barreiros, Hooshang Nayyeri, Tony Dear, Huan Zhang, and Yunzhu Li. Interactive world simulator for robot policy training and evaluation. *arXiv preprint arXiv:2603.08546*, 2026e. 2

Yuqi Wang, Jiawei He, Lue Fan, Hongxin Li, Yuntao Chen, and Zhaoxiang Zhang. Driving into the future: Multiview visual forecasting and planning with world model for autonomous driving. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14749–14759. IEEE, 2024. 2

Jiajun Wu, Ilker Yildirim, Joseph J Lim, Bill Freeman, and Josh Tenenbaum. Galileo: Perceiving physical object properties by integrating a physics engine with deep learning. *Advances in neural information processing systems*, 28, 2015. 1

Philipp Wu, Alejandro Escontrela, Danijar Hafner, Pieter Abbeel, and Ken Goldberg. Daydreamer: World models for physical robot learning. In *Conference on robot learning*, pp. 2226–2240. PMLR, 2023. 2

Tianyi Xie, Yiwei Zhao, Ying Jiang, and Chenfanfu Jiang. Physanimator: Physics-guided generative cartoon animation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10793–10804. IEEE, 2025. 3

Qiyao Xue, Xiangyu Yin, Boyuan Yang, and Wei Gao. Phyt2v: Llm-guided iterative self-refinement for physics-grounded text-to-video generation. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 18826–18836. IEEE, 2025. 3

Sherry Yang, Yilun Du, Kamyar Ghasemipour, Jonathan Tompson, Leslie Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 2023. 1

Yuncong Yang, Zhengtao Han, Furkan Ozyurt, Zeyuan Yang, Han Yang, Junyi Cao, Haoyu Zhen, Yilun Du, and Chuang Gan. Syncworld: Visual calibration enables world models as zero-shot simulators. *arXiv preprint arXiv:2609.09155*, 2026. 2

Zhuoyi Yang, Jiayan Teng, Wendi Zheng, Ming Ding, Shiyu Huang, Jiazheng Xu, Yuanming Yang, Wenyi Hong, Xiaohan Zhang, Guanyu Feng, et al. Cogvideox: Text-to-video diffusion models with an expert transformer. In *International Conference on Learning Representations*, volume 2025, pp. 83048–83077, 2025. 3

Wenhao Yu, Jie Tan, C Karen Liu, and Greg Turk. Preparing for the unknown: Learning a universal policy with online system identification. *arXiv preprint arXiv:1702.02453*, 2017. 1, 4

Yu Yuan, Xijun Wang, Tharindu Wickremasinghe, Zeeshan Nadir, Bole Ma, and Stanley Chan. Newtongen: Physics-consistent and controllable text-to-video generation via neural newtonian dynamics. In *International Conference on Learning Representations*, volume 2026, pp. 12847–12878, 2026. 3

Qiyuan Zhang, Biao Gong, Shuai Tan, Zheng Zhang, Yujun Shen, Xing Zhu, Yuyuan Li, Kelu Yao, Chunhua Shen, and Changqing Zou. Physrvg: Physics-aware unified reinforcement learning for video generative models. *arXiv preprint arXiv:2601.11087*, 2026a. 3

Tianyuan Zhang, Hong-Xing Yu, Rundi Wu, Brandon Y Feng, Changxi Zheng, Noah Snavely, Jiajun Wu, and William T Freeman. Physdreamer: Physics-based interaction with 3d objects via video generation. In *European Conference on Computer Vision*, pp. 388–406. Springer, 2024. 3

Xiangdong Zhang, Jiaqi Liao, Shaofeng Zhang, Fanqing Meng, Xiangpeng Wan, Junchi Yan, and Yu Cheng. Videorepa: Learning physics for video generation through relational alignment with foundation models. *Advances in Neural Information Processing Systems*, 38:122647–122676, 2026b. 3

Xilun Zhang, Shiqi Liu, Peide Huang, William Jongwon Han, Yiqi Lyu, Mengdi Xu, and Ding Zhao. Dynamics as prompts: In-context learning for sim-to-real system identifications. *IEEE Robotics and Automation Letters*, 10(4):3190–3197, 2025. 2

Zangwei Zheng, Xiangyu Peng, Yuxuan Lou, Chenhui Shen, Tom Young, Xinying Guo, Binluo Wang, Hang Xu, Hongxin Liu, Mingyan Jiang, et al. Open-sora 2.0: Training a commercial-level video generation model in $200 k. *arXiv preprint arXiv:2503.09642*, 2025. 3

Gaoyue Zhou, Hengkai Pan, Yann Lecun, and Lerrel Pinto. Dino-wm: World models on pre-trained visual features enable zero-shot planning. In *International Conference on Machine Learning*, pp. 79115–79135. PMLR, 2025. 1, 2, 3, 5, 17, 18, 21, 28

CONTENTS

1 Introduction **1**

2 Related Work **2**

3 Method **3**
3.1 Physics Calibration via Physics Descriptor Token 3
3.2 Training PhysiCalWorld without Physics Labels 4
3.3 Inference with PhysiCalWorld 5

4 Experiments **5**
4.1 Environments with Single Physics Variation 6
4.2 Environments with Multiple Physics Variations 8
4.3 Analysis 8

5 Discussion **9**

A Implementation Details **17**
A.1 Baselines 17
A.2 Implementation of PhysiCalWorld 18
A.3 Training and Inference Pseudocode of PhysiCalWorld 19

B Environment Training and Evaluation Details **21**
B.1 Push-T 21
B.2 Reacher 22
B.3 BeerPong 22
B.4 FetchSlide 23
B.5 Success-threshold selection and sensitivity 24

C Efficency Analyzation **25**
C.1 Model Training Efficency 25
C.2 Model Inference Efficiency 25

D Additional Quantitative Results **27**
D.1 Next-Frame Prediction Performance 27
D.2 MPC Results Across Execution Budgets 28

E Additional Qualitative Results **29**
E.1 Context Clips for Main-Text Examples 29
E.2 Uncropped BeerPong Visualization 30
E.3 Qualitative Extrapolation Results 30
E.4 Additional Open-Loop Rollout Results 32
E.5 Additional Planning Results 36

APPENDIX

In this appendix, we first present our implementation details in Appendix A, such as the baseline design, model architecture, hyperparameters, as well as training and evaluation setups. Next, we provide the details of the four environments we used in Appendix B. Lastly, we show additional quantitative and qualitative results in Appendices D–E.

A IMPLEMENTATION DETAILS

A.1 BASELINES

In this section, we provide a detailed description for each baseline we used in our paper. Please see the description of each method below:

a) **DINO-WM** (Zhou et al., 2025): We follow the implementation in the official codebase, using a frozen DINOv2 (Oquab et al., 2023) backbone as the visual encoder E_{v}. The input to the dynamics predictor f_θ is an observation-action sequence of $K = 3$ steps, including $K - 1$ historical transitions alongside the current observation and action. The predictor predicts one-step future observation $\hat{o}_{t+1}$. Because this baseline receives no in-context interaction prompt, it must infer transition dynamics solely from this local history window. We observe that DINO-WM exhibits severe rollout drift when physical properties vary across test scenes.

b) **DINO-WM + Context Prefix:** This baseline uses the same model architecture and training objective as DINO-WM. The only difference is that it additionally feeds the latent embeddings of the whole in-context interaction prompt to the dynamics predictor f_θ. As a result, for this baseline, the total input window size of f_θ is $T_c + K$. Since transformer self-attention scales quadratically with the number of input tokens, this uncompressed prefix adds substantially more computation to both model training and inference (see Figure 6 and Table A7 for runtime comparisons).

c) **DINO-WM + Long History:** This baseline uses the same model architecture and training objective as DINO-WM. The only difference is that it extends the target observation history from 3 to 15 consecutive frames (*i.e.,* $K = 15$). The goal of this baseline is to study whether extending history can enable effective physics-calibration in world models.

d) **LeWM** (Maes et al., 2026): LeWM is a latent world model directly trained from scratch (without using frozen pre-trained models like DINOv2), hence it learns task-specific latent visual representations and action-conditioned dynamics. We follow the implementation in its official codebase. Based on LeWM, we also include **LeWM + PhysiCal (Ours)**, where we integrate the proposed physical calibration module into LeWM, to show that our proposed physics calibration method is a plug-and-play module that can potentially extend to any latent world models.

e) **TC-LeWM** (Liu et al., 2026): This method applies SIGReg to temporally centered residuals rather than the full latent distribution, preventing the suppression of fine-grained moving components. We include this to enhance the LeWM baseline, as vanilla LeWM's latent representations fail to capture the ball in the Beerpong environment (yielding very low IoU and planning success). Similarly, we include **TC-LeWM + PhysiCal (Ours)** to evaluate whether our proposed method can be applied to this improved LeWM baseline.

f) **Temporal Straightening (TS-WM)** (Wang et al., 2026d): we adopt this TS-WM variant: DINOv2 (patch) + proj configuration with $16 \times 16 \times 8$ spatial features, as it achieves the strongest overall performance reported in the original paper. Similar to DINO-WM, this variant uses a frozen DINOv2 visual encoder and a causal transformer dynamics predictor, but inserts an additional lightweight projection head between the visual encoder and the predictor. During training, TS-WM introduces a curvature regularizer to enforce a more structured latent representation, and such regularization has demonstrated stronger planning performance. We include this baseline to study whether additional regularization can promote physics grounding in world models.

g) **DINO-WM + Estimated Physics†:** This baseline first estimates the physics configurations of the scenes (e.g., the center of mass in Push-T or the joint damping in Reacher) from a context clip and feed the estimated value as condition for the dynamic predictor f_θ. Specifically, we train a separate physics estimator based on the context encoder architecture as our method but use the ground-truth physics labels as supervision. At inference time, the estimated parameters are provided to the dynamics predictor as additional input for conditioning. This baseline compares

explicit, supervised physics estimation with our end-to-end in-context calibration method. Note that this baseline requires predefined physics labels for training, while our method does not.

h) **DINO-WM + Privileged Physics[‡]:** This baseline is based on DINO-WM but can access the ground-truth physics configurations of the scenes during both training and inference. Specifically, the dynamics predictor directly receives these true physical parameters as supplementary inputs alongside standard observation and action tokens. Since neither our method nor any other baseline can access the ground-truth physics configuration, this model functions as an oracle upper bound for reference rather than a directly comparable baseline.

A.2 Implementation of PhysiCalWorld

Software and Hardware. Our experiments are implemented in Python 3.10 on Red Hat Enterprise Linux 8.10. We train and evaluate our models using NVIDIA L40S GPUs and NVIDIA RTX PRO 6000 Blackwell GPUs interchangeably.

Base World Models. We implemented PhysiCalWorld on two latent world-model backbones, DINO-WM (Zhou et al., 2025) and LeWM (Maes et al., 2026), denoted as DINO-WM+PhysiCal and LeWM+PhysiCal, as in Table 1 and Table 2. The base architecture and optimization hyperparameters used for the two backbones are summarized in Table A1.

Configuration	DINO-WM-based models	LeWM-based models
Vision Encoder E_v	Frozen DINOv2 (ViT-Small/14)	Learnable ViT-Tiny/14
Visual Token Dimension d_z	384	192
Visual Tokens per Frame	256 (16×16)	1
Dynamics Predictor f_θ	Causal Transformer	Causal Transformer
Layers	6	6
Attention Heads	16	16
FFN Hidden Dimension d_{mlp}	2048	2048
Predictor Dropout	0.1	0.1
Optimizer	AdamW	AdamW
Model Learning Rate	1×10^{-4}	5×10^{-5}
Weight Decay	0.01	1×10^{-3}
Gradient Clipping Norm	1.0	1.0
Global Batch Size	48	128
Observation History Length K	3 steps	3 steps
Proprioceptive Loss Weight	0.5	0.5

Table A1: **Model Architectures and Training Hyperparameters.** Comparison of the base configurations used by the DINO-WM and LeWM model families.

Context Encoder. The context encoder g_ϕ maps a 12-frame ($T_c = 12$) interaction sequence to a 64-dimension physics descriptor $\mathbf{z}_{\text{phy}}$. Each frame is embedded by a ResNet-based visual encoder E_c^v into 64 two-dimensional keypoints. The world-model action is embedded by a single linear layer $E_c^a : \mathbb{R}^{d_a} \to \mathbb{R}^{10}$. For Push-T, the 4D observable pusher state is additionally embedded by $E_c^p : \mathbb{R}^4 \to \mathbb{R}^{10}$. The resulting per-step features are concatenated and mapped to one 512-dimension latent embedding. A four-layer Transformer encoder with eight attention heads takes all the latent embeddings as input and pools them with a learnable query token, which is then mapped to the physics descriptor token $\mathbf{z}_{\text{phy}}$ via an MLP. Table A2 summarizes the corresponding hyperparameters.

Context Sampling and Backbone Integration. For each prediction segment sampled from a physical-configuration group G_j, we independently sample the interaction context from a different episode in the same group. Thus, the context shares the same underlying physical dynamics as the prediction target without containing trajectory-specific future observations. All contexts contain $T_c = 12$ observations sampled at the model prediction gap of the corresponding world model.

For Push-T and Reacher, we uniformly sample the $T_c = 12$-frame context window within the prompt episode. For BeerPong, we always sample the context from the beginning of the episode, because the ball–table impact is temporally sparse, and starting from the beginning ensures that the context contains the ball–table interaction that reveals the contact damping. For FetchSlide, we also start the context from the beginning of the episode because the trajectories are relatively short. If fewer than 12 sampled frames are available, we pad the context to length T_c by repeating the final available frame.

Hyperparameter	Value
ResNet GroupNorm Groups	8
Spatial Projection Channels	64
Number of Spatial Keypoints	64
Visual Feature Dimension	128
Action Embedding Dimension	10
Proprioceptive Embedding Dimension	10
Temporal Encoder Depth	4
Attention Heads	8
Hidden Dimension	512
FFN Hidden Dimension	1024
Physics Descriptor Dimension	64
Context Length T_c	12 frames
Learning Rate	Same as the base world model

Table A2: **Context encoder hyperparameters.** Hyperparameters used for the in-context physics encoder g_ϕ.

For DINO-WM+PhysiCal, $\mathbf{z}_{\text{phys}}$ is broadcast across all spatial patches and temporal steps and concatenated with the predictor features along the channel dimension. For LeWM+PhysiCal, the 64-dimensional descriptor is concatenated with the 192-dimensional per-step latent representation before the causal predictor, while action conditioning follows the original LeWM implementation through AdaLN-zero.

Temporal Straightening. For Temporal Straightening (TS-WM) (Wang et al., 2026d), we follow the original implementation and use the DINOv2 (patch) + projection configuration with the default curvature regularization and optimization settings reported by the authors, as it is the best-performing model reported in the original paper. We summarize the corresponding hyperparameters in Table A3.

Hyperparameter	Value
Projection Intermediate Channels	192
Projection Output Channels	8
Projected Feature Map Shape	$16 \times 16 \times 8$
Curvature Loss Weight λ_{curv}	0.1
Projection Head Learning Rate	1×10^{-5}
Predictor Learning Rate	5×10^{-4}

Table A3: **Temporal Straightening hyperparameters.** Baseline-specific hyperparameters used for TS-WM.

A.3 Training and Inference Pseudocode of PhysiCalWorld

We present the pseudocode for training PhysiCalWorld in Algorithm 1, which is also described in Section 3.2. Training PhysiCalWorld requires neither ground-truth physics labels nor an analytical dynamics model; the only supervision structure it consumes is the group partition of the dataset (where data collected within environments sharing identical physics are grouped together). Because every episode in $\mathcal{G}_j$ shares the same underlying transition dynamics $\mathcal{T}_{\xi_j}$, if both the context prompt and the train target segment are sampled from the same group, g_ϕ can minimize the prediction loss by encoding context-specific physical dynamics ξ_j from the prompt into $\mathbf{z}_{\text{phy}}$. Note that we draw the interaction prompt from an episode distinct from the trajectory segment used for future prediction so as to prevent the prompt carrying information about the future.

During training, the predictor is teacher-forced on the visual encoder representations under causal masking (e.g., z_0 cannot attend to z_1), with prediction loss gradients flowing jointly into both θ and ϕ. The visual decoder D_{v} is trained *separately* using an ℓ_2 reconstruction loss to map predicted latents $\hat{z}_t$ back to pixel-space images.

Algorithm 1 Training PhysiCalWorld

Require: train dataset $\mathcal{D}_{\text{train}} = \{\mathcal{G}_j\}_{j=1}^{M}$ of M physics configurations, context encoder g_ϕ, visual encoder E_{v}, action encoder E_{a}, dynamics predictor f_θ, input window length K, prompt length T_c, training steps N
for $i = 1 \to N$ **do**
  $\mathcal{G}_j \sim \mathcal{D}_{\text{train}}$, sample a data group from dataset $\mathcal{D}_{\text{train}}$
  $(o_{0:K}, a_{0:K-1}) \sim \mathcal{G}_j$, sample a trajectory segment of length $K+1$ for training
  # Sample interaction prompt from distinct episode in $\mathcal{G}_j$
  $\mathcal{C}_{\text{tr}}^j = \{(o_k^c, a_k^c)\}_{k=1}^{T_c} \sim \mathcal{G}_j$, with episode_id($\mathcal{C}_{\text{tr}}^j$) $\neq$ episode_id($o_{0:K}$)
  $\mathbf{z}_{\text{phy}} \leftarrow g_\phi(\mathcal{C}_{\text{tr}}^j)$ # encode physics descriptor token
  $z_{0:K} \leftarrow E_{\text{v}}(o_{0:K})$ # encode visual observation tokens
  $e_{0:K-1}^a \leftarrow E_a(a_{0:K-1})$ # encode action embeddings
  # Predict future latents $\hat{z}_{1:K}$ with causal masks (e.g., z_0 cannot attend to z_1)
  $\hat{z}_{1:K} \leftarrow f_\theta\left(\{(z_t, e_t^a)\}_{t=0}^{K-1},\ \mathbf{z}_{\text{phy}}\right)$
  $\mathcal{L} \leftarrow \frac{1}{K}\sum_{t=1}^{K} \|\hat{z}_t - z_t\|_2^2$
  Backprop to update (θ, ϕ) and E_a using $\mathcal{L}$
end for
return $g_\phi, E_v, E_a, f_\theta$

We outline the procedure for open-loop rollout inference with PhysiCalWorld in Algorithm 2, as detailed in Section 3.3. At test time, the descriptor is extracted once prior to the rollout loop and subsequently held fixed. Inside the loop, autoregressive rollout runs similarly to the base world model except for the addition of the conditioning physics descriptor token, preserving per-step latency at the level of the unconditioned base predictor (Appendix C.2). Latent-to-pixel decoding through D_{v} is only required for visual evaluation; our visual planning directly operates in the latent space, $\hat{z}$.

Algorithm 2 Open-loop Rollout with PhysiCalWorld

Models: trained context encoder g_ϕ, visual encoder E_{v}, action encoder E_{a}, dynamics predictor f_θ, visual decoder D_{v}
Input: interaction prompt $\mathcal{C}_{\text{eval}} = \{(o_k^c, a_k^c)\}_{k=1}^{T_c}$, initial observation o_0, input actions $a_{0:T-1}$, input window length K, open-loop rollout horizon T
extract the physics descriptor token of the test scene
$\mathbf{z}_{\text{phy}}^{\text{ev}} \leftarrow g_\phi(\mathcal{C}_{\text{eval}})$
encode the initial observation and set $\hat{z}_0 = z_0$ so that line 11 is well defined at $t = 0$
$z_0 \leftarrow E_{\text{v}}(o_0)$; $\hat{z}_0 \leftarrow z_0$
for $t = 0 \to T-1$ **do**
  # encode the t-th action embedding
  $e_t^a \leftarrow E_{\text{a}}(a_t)$
  # predict future latents autoregressively
  $\hat{z}_{t+1} \leftarrow f_\theta\left(\{(\hat{z}_i, e_i^a)\}_{i=\max(0,\, t-K+1)}^{t},\ \mathbf{z}_{\text{phy}}^{\text{ev}}\right)$
  # decode the predicted latent to image
  $\hat{o}_{t+1} \leftarrow D_{\text{v}}(\hat{z}_{t+1})$
end for
return predicted future latents $\hat{z}_{1:T}$ and images $\hat{o}_{1:T}$

	Push-T	Reacher	FetchSlide	BeerPong
Training / evaluation physical configurations	135 / 15	135 / 15	135 / 15	135 / 15
Episodes per configuration	101	20	240	60
Total dataset episodes	15,150	3,000	36,000	9,000
Control interval (s)	0.10	0.04	0.04	0.01
world model # of frameskip	5	5	5	15
world model per-step interval (s)	0.50	0.20	0.20	0.15
Environment action dimension	2	2	2	2
World-model action input dimension	10	10	8	2

Table A4: **Task and dataset summary.** All environments use 224×224 RGB observations.

B Environment Training and Evaluation Details

We provide a detailed description of our experiments in each environment, including environment setup, training and evaluation details. Table A4 summarizes the dataset scale and hyperparameters of the world models in each environment. The resolution of the image observation is 224×224 for all environments.

B.1 Push-T

Task and Physics Variation. Push-T is a planar manipulation task in a 512×512 pixel workspace, where a circular pusher of radius 15 pixels moves a rigid T-shaped block toward a goal configuration specified by an RGB image. The block consists of a 120×30 pixel crossbar and a 30×90 pixel stem. We uniformly sample the block's local center of mass (CoM) inside its T-shaped interior and hold it fixed within each episode. Changing the CoM changes the moment arms of contact forces and the resulting translational and rotational response, while the block appearance remains fixed.

Dataset and Action Space. The dataset contains 150 CoM configurations, with 135 used for training and 15 held out for evaluation, totaling 15,150 episodes with 101 episodes per configuration. We train a CoM-agnostic PPO policy that targets pushing the T-block to the light-green area and use this policy to collect trajectories for training, since a random policy cannot ensure sufficient and meaningful interaction. In theory, any policy that can produce sufficient contacts suffices to be used as the data collection policy. Each raw action specifies an absolute 2D pusher target in simulator pixel coordinates, tracked by a PD controller at a control frequency of 10 Hz. Following Zhou et al. (2025), we inject independent Gaussian noise into the output action during policy execution. The action has ten different standard deviations: $\sigma \in \{0.1, 0.3, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 7.0, 15.0\}$ pixels. This online noise injection enhances dataset coverage and generalization, which is particularly essential for downstream planning using CEM, as its initial optimization phase relies heavily on exploring noisy action trajectories. We subsample observations every 5 control steps and concatenate the 5 2D actions in between into a 10D world-model action, yielding a prediction interval of 0.5 seconds. Models receive 224×224 RGB images and a 4D pusher proprioceptive input (x, y, v_x, v_y).

Training and evaluation. Please see Table A1, Table A2, and Table A3 for detailed model hyperparameters, and Table A5 for training wall-clock durations. Visual prediction IoU is computed on the union of the agent and block foreground masks. Each Planning task pairs a single current observation with a goal observation recorded 15 control steps later in the same trajectory. Success requires the Euclidean position error over the concatenated agent and block coordinates to be below 20 pixels and the wrapped block-orientation error to be below $\pi/9$ (20°). For open-loop execution, we set the planning horizon to 25 control steps (i.e., 5-step future prediction in the world model). In Table 1, we report the success rate under this fixed 25-step budget when directly executing the planned action sequence from the initial observation. To leverage the world model for dynamic planning, we also experiment with model predictive control (MPC), in which the model actively replans if the task has not succeeded. Table A9 evaluates cumulative success rates with at most two replans (i.e., budgets of 25, 50, and 75 control steps). We observe that the success rate of our method consistently increases with larger step budgets, as iterative replanning allows for continuous error correction.

B.2 REACHER

Task and physical variation. We modify the Reacher environment from the DeepMind Control Suite (Tassa et al., 2018), where a planar two-link arm moves toward a goal configuration specified by an RGB image. The proximal and distal links have lengths of 0.12 and 0.10 meters, respectively. We vary the two joint damping coefficients over $[0.005, 0.020]\,\mathrm{N}\cdot\mathrm{m}\cdot\mathrm{s/rad}$ using Latin-hypercube sampling and hold them fixed within each episode. Changing joint damping alters the two-link arm's resistance to motion without changing the visual appearance.

Dataset and action space. The dataset contains 150 joint-damping configurations, with 135 used for training and 15 held out for evaluation, totaling 3,000 episodes with 20 episodes per configuration. Each episode contains 100 control steps and starts from randomly sampled joint angles with zero initial joint velocities. Unlike Push-T, random torque commands naturally induce diverse joint motions and provide broad coverage of the joint-state and velocity space, so no task-specific data-collection policy is required. Each 2D action is sampled independently from $[-1, 1]^2$ and specifies normalized motor commands corresponding to joint torques in $[-0.05, 0.05]\,\mathrm{N}\cdot\mathrm{m}$. Each command is held for two 0.02-second simulation steps, giving a control frequency of 25 Hz. Similar to Push-T, we subsample observations every 5 control steps and concatenate the five intervening 2D actions into a 10D world-model action, yielding a prediction interval of 0.2 seconds. Unlike PushT, Reacher do not provide proprioception inputs.

Training and evaluation. Visual prediction IoU is computed on the union of the two arm-link foreground masks. Each Planning task pairs a single current observation with a goal observation recorded 25 control steps later in the same trajectory. Success requires the angular error of each joint to be below 10° ($\pi/18$ radians). Similar to PushT, we use a 25-control-step planning horizon (i.e. 5-step future prediction in the world model), and report the 25-step budget planning result in Table 1. We additionally evaluate MPC with at most two replans, yielding cumulative budgets of 25, 50, and 75 control steps, as reported in Table A9. Note that repeated replanning substantially reduces the effect of accumulated model error in Reacher, allowing even imperfect models to iteratively approach the target and resulting in high MPC success rates. We observed that LeWM-based models perform particularly well on Reacher, which may be attributed to its encoder being task-specific and can focus on essential structures such as the angles of the joints, whereas DINOv2 is trained as a general-purpose image encoder.

B.3 BEERPONG

Task and physical variation. We modify the BeerPong environment from Fancy Gym (Otto et al.), where a simulated Barrett Whole-Arm Manipulator (WAM) throws a ball toward a cup across a table. We omit an explicit gripper and instead kinematically attach the ball to the end effector during the first five control steps, after which it is released with the end effector's instantaneous velocity. The robot starts from a fixed configuration, while the cup position is sampled along the table's longitudinal axis and remains fixed within each episode. We uniformly sample the ball-contact damping parameter over $[10, 40), \mathrm{N}\cdot\mathrm{s/m}$ and hold it fixed within each episode. This changes energy dissipation during contact and the resulting bounce trajectory without changing visual appearance. We empirically choose the contact-damping range as lower values produce insufficient rebound while higher values lead to excessively unstable ball trajectories; within this range, most episodes remain feasible, with some throws capable of reaching the cup after one or two bounces on the table. The table measures approximately 0.90×1.68 meters, the cup has a nominal inner radius of 0.10 meters, and the ball has a radius of 0.049 meters and a mass of 0.10 kilograms.

Dataset and action space. The dataset contains 150 contact-damping configurations, with 135 used for training and 15 held out for evaluation, totaling 9,000 episodes with 60 episodes per configuration. Each episode contains 300 control steps. A 2D throw command is sampled uniformly from $[2.5, 3.0) \times [6.0, 8.0)\,\mathrm{m/s}$ and specifies target forward and upward end-effector velocities. After releasing the ball, the robot arm retracts to a fixed configuration to prevent its post-throw pose from leaking information about the throw command. Each control step spans two 0.005-second simulation steps, giving a control frequency of 100 Hz. We subsample observations every 15 control steps, yielding a prediction interval of 0.15 seconds, and provide the normalized initial 2D throw command as a fixed condition at every world-model step. Models receive 224×224 RGB images without proprioception inputs.

Training and evaluation. Visual prediction IoU is computed on the ball foreground mask. Each Planning task pairs a single initial observation with a recorded final observation of a successful

in-cup throw. Success requires the ball to stay in the cup. We report success on 75 shared targets. Visual prediction IoU is computed on the ball foreground mask. All planning tasks share the same initial robot and ball configuration, differing only in the randomly sampled cup position. The goal observation depicts the ball inside the cup at that position, and the planner must find a throw command that reaches the corresponding goal. Success requires the ball to remain inside the cup after settling. Unlike Push-T and Reacher, BeerPong is a one-shot planning task, in which the planner optimizes a single 2D throw command.

B.4 FETCHSLIDE

Task and physical variation. We modify the FetchSlide-v4 environment from Gymnasium Robotics (de Lazcano et al., 2024), where a Fetch manipulator strikes a cylindrical puck toward a goal position on a table. The puck has a radius of 0.05 meters and a height of 0.04 meters. We independently sample puck mass $m \in [0.04, 0.36)\,\mathrm{kg}$, surface friction $\mu \in [0.0305, 0.0435)$, and fluid density $\rho \in [0, 40)\,\mathrm{kg/m^3}$ from uniform distributions and hold them fixed within each episode. For the same strike action, lower puck mass results in a higher post-strike velocity, while subsequent deceleration depends jointly on surface friction and aerodynamic drag, with stronger friction and higher fluid density increasing deceleration and larger mass reducing the effect of aerodynamic drag.

Dataset and action space. The dataset contains 150 physical configurations, with 135 used for training and 15 held out for evaluation, totaling 36,000 episodes with 240 episodes per configuration. Each strike is specified by a 2D action sampled uniformly from $[0.35, 0.85) \times [-0.8, 0.8)$. Following the original setting of FetchSlide-v4 (de Lazcano et al., 2024), action is expanded into a 4D Cartesian command $(a_x, a_y, 0, 0)$, where the first three components specify end-effector displacement scaled by 0.05 meters and the final component controls the gripper, which is always closed. The same command is applied for the first two control steps. Each control step spans 0.04 seconds. We subsample observations every five control steps, yielding a prediction interval of 0.2 seconds, and concatenate the two initial 4D commands into a normalized 8D condition provided at every world-model step. After striking the puck, the robot arm moves to a fixed position to prevent information leakage from the post-strike pose. Models receive 224×224 RGB images without proprioception input.

Training and evaluation. Visual prediction IoU is computed on the puck foreground mask. All planning tasks share the same initial robot and puck configuration and differ only in the target puck position. Similar to Beerpong, FetchSlide is a one-shot planning task, where the planner optimizes a single 2D strike command, after which the puck evolves freely until it settles. Success requires the Euclidean distance between the final and target puck positions in the table plane to be at most 0.03 meters. We evaluate all methods on the same set of 75 planning targets.

B.5 SUCCESS-THRESHOLD SELECTION AND SENSITIVITY

To better understand the planning performance of the Reacher and FetchSlide environments, reported respectively in Table 1 and Table 2, we additionally present success rate *v.s* error-threshold curves in Figure A1.

Let the error threshold be τ, measured in degrees for Reacher and centimeters for FetchSlide. For Reacher, success requires both joint angular position errors to satisfy $\max_{j\in\{1,2\}} |q_j^{25} - q_j^{\text{goal}}| < \tau$. For FetchSlide 2D, the success criteria is that terminal puck-position error $\|\mathbf{p}_{xy}^{\text{final}} - \mathbf{p}_{xy}^{\text{goal}}\|_2 \leq \tau$. For the results shown in Table 1 and Table 2, the error tolerance τ is 10° for Reacher and $3\,\text{cm}$ for FetchSlide.

As shown in Figure A1, in both the Reacher and the FetchSlide environments, our method improves performance over the baseline world models (DINO-WM and LeWM), as evidenced by comparing DINO-WM (deep blue) and DINO-WM + ours (green). We improved significantly from unprivileged models in FetchSlide. The gains are modest in the Reacher environment, likely due to its highly constrained state space, which consists of only two rotational joints.

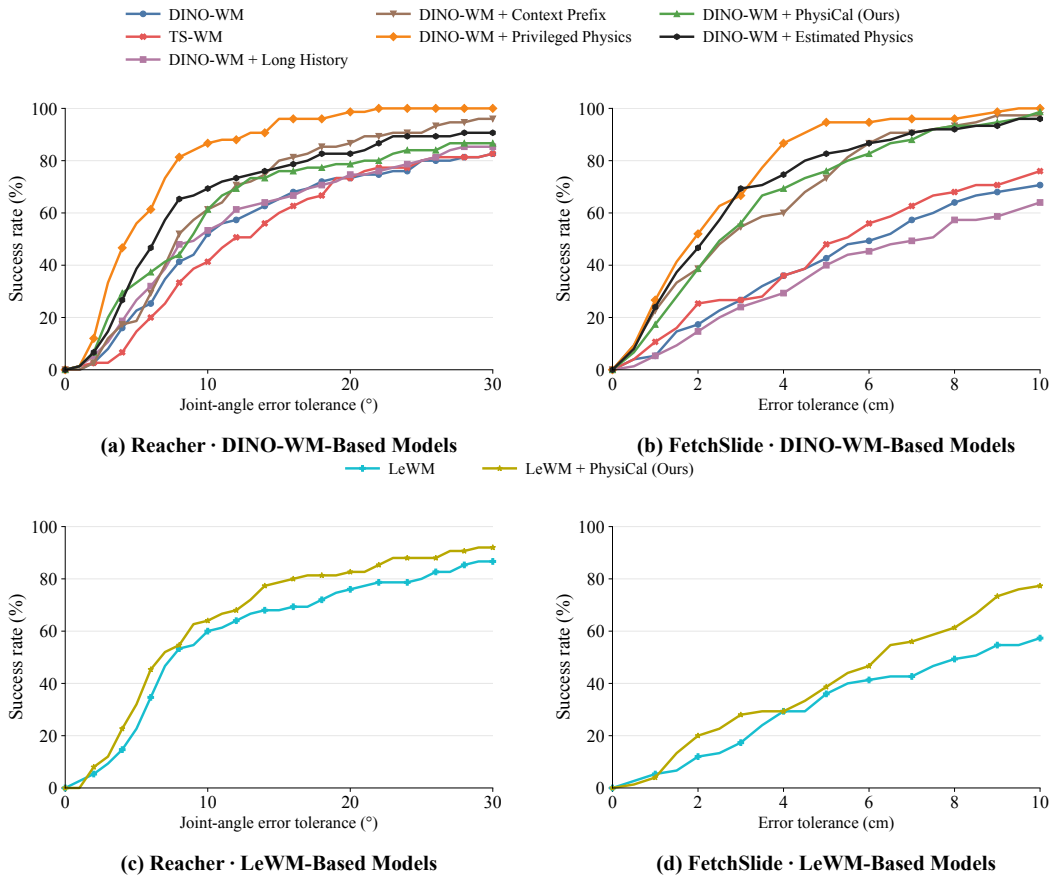


Figure A1: **Success-threshold sensitivity.** Evaluation of planning success rates across varying error tolerances for DINO-WM variants (top) and LeWM variants (bottom). (a) **Reacher (DINO-WM-based models)** using maximum absolute joint-angle error across joints. (b) **FetchSlide (DINO-WM-based models)** using terminal puck-XY position error. (c) **Reacher (LeWM-based models)** across joint-angle tolerances. (d) **FetchSlide (LeWM-based models)** across puck-XY position tolerances.

C Efficency Analyzation

C.1 Model Training Efficency

We report the training time for all world models evaluated in this work, comparing the number of epochs, GPU configurations, and wall-clock training hours across tasks. Training time is measured from the initialization of the formal training log to the saving of the selected checkpoint. We use two types of GPUs: L40S denotes NVIDIA L40S, and Blackwell denotes NVIDIA RTX PRO 6000 Blackwell.

While our method introduces additional training cost over the corresponding vanilla world-model backbone, it remains substantially more efficient than Long-History models and Context Prefix models. Long History and Context Prefix attend over 15 frames, whereas our method still uses only 4 interaction frames, just as the baseline. The longer sequence substantially increases the cost of attention and leads to much higher per-epoch training time. Due to the high per-epoch training time for the two baselines, for Long History we trained only 5 epochs in Push-T, and for Context Prefix we trained only 5 epochs in Push-T, 10 epochs in Reacher and 3 epochs in BeerPong.

	Push-T			**Reacher**		
Method	Epochs	GPUs	Time (hours)	Epochs	GPUs	Time (hours)
TS-WM	10	4 × L40S	24.9	20	4 × L40S	8.7
DINO-WM	10	4 × L40S	36.5	20	4 × L40S	13.2
+ Long History	5	2 × PRO 6000	65.9	20	2 × PRO 6000	30.1
+ Context Prefix	5	4 × L40S	52.0	10	2 × PRO 6000	18.6
+ Privileged Physics	10	4 × L40S	36.5	20	4 × L40S	12.8
+ PhysiCal (Ours)	10	4 × L40S	49.7	20	4 × L40S	19.7
LeWM	10	4 × L40S	24.6	20	4 × L40S	4.7
+ PhysiCal (Ours)	10	4 × L40S	37.4	20	4 × L40S	13.8

Table A5: **World-model training time on Push-T and Reacher.**

	BeerPong			**FetchSlide**		
Method	Epochs	GPUs	Time (hours)	Epochs	GPUs	Time (hours)
TS-WM	5	2 × PRO 6000	34.6	5	2 × PRO 6000	10.8
DINO-WM	5	2 × PRO 6000	32.2	5	2 × PRO 6000	19.8
+ Long History	5	4 × L40S	88.1	5	4 × L40S	49.5
+ Context Prefix	3	2 × PRO 6000	52.6	5	4 × L40S	44.7
+ Privileged Physics	5	4 × L40S	27.9	5	4 × L40S	15.9
+ PhysiCal (Ours)	5	4 × L40S	45.3	5	4 × L40S	25.6
LeWM*	5	2 × PRO 6000	37.2	5	2 × PRO 6000	8.7
+ PhysiCal (Ours)	5	2 × PRO 6000	49.2	5	2 × PRO 6000	20.3

Table A6: **World-model training time on BeerPong and FetchSlide.** LeWM*: we use TC-LeWM in BeerPong to enhance performance (see Appendix A.1 for details).

C.2 Model Inference Efficiency

We measure latent model inference on Push-T using one NVIDIA RTX PRO 6000 Blackwell Server Edition GPU and FP32 with TF32 disabled. In the next-frame prediction, we use 15 observed frames for Long History and three for the remaining methods, while open-loop rollouts start from three observed frames, and Long History grows its history autoregressively up to its 15-frame window. Context Prefix and our method use the same 12-frame context from an episode with the same physical parameters. For planning inference time, we use the same CEM setting as the experiments and report the mean of ten repeated solves for one fixed target after one warm-up solve.

We report mean wall-clock latency over 100 timed runs following 20 warm-up runs across ten validation episodes. Warm-up excludes one-time initialization cost so that we can record stabilized

inference time. GPU is synchronized before and after each timed run to ensure all asynchronous CUDA execution is done before we record the time.

Table A7 shows that our method incurs only a 2.30 ms one-time condition-encoding cost while retaining nearly the same teacher-forced latency as DINO-WM (7.82 vs. 5.58 ms), while Context Prefix and Long History need much longer (13.91 ms and 13.91 ms). For open-loop rollouts, our method requires 16.03 ms for 5-step prediction and 45.83 ms for 20-step prediction, remaining close to DINO-WM (13.78 ms and 44.48 ms, respectively), compared with Context Prefix (48.36 ms and 177.61 ms) and Long History (18.05 ms and 131.27 ms). These results indicate that compressing context into a compact representation preserves near-baseline inference cost, avoiding the computational penalties of uncompressed prefixes or long observation histories.

Method	context encoding (ms)	Next-Frame (ms)	Rollout-5f (ms)	Rollout-10f (ms)	Rollout-20f (ms)	Planning (s)
DINO-WM	–	5.58	13.78	24.01	44.48	32.66
+ Context Prefix	4.89	13.91	48.36	91.44	177.61	323.09
+ Long History (15 frames)	–	13.91	18.05	48.97	131.27	43.96
+ Privileged Physics[‡]	–	5.56	13.67	23.82	44.05	32.55
+ PhysiCal (Ours)	2.30	7.82	16.03	26.31	45.83	32.67

Table A7: **Push-T model inference latency.** (1) **Condition encoding**: Measures the one-time latency to encode the 12 conditioning frames into model representations. Our method uses the lightweight context encoder, while the Context Prefix method uses the primary vision backbone (i.e., DINOv2). Methods without context conditioning do not require this step. (2) **Next-Frame Prediction (Teacher Forcing)**: Measures the latency given unencoded real observation history (3 frames for standard models, 15 frames for Long History) predicts the single next latent step under recorded actions, with context conditioning pre-cached if applicable. (3) **Open-Loop Rollout (5, 10, and 20 frames)**: Measures the total latency of predicting H future latent frames ($H \in \{5, 10, 20\}$) from three unencoded observed frames (IC3), including initial observation and context encoding once. Long History retains its 15-frame capacity and fills its history autoregressively.

D Additional Quantitative Results

D.1 Next-Frame Prediction Performance

We provide additional next-frame prediction results on all four environments in Table A8. We report PSNR, SSIM, LPIPS, and foreground IoU to evaluate both pixel-level reconstruction quality and foreground prediction accuracy. Across all four environments, incorporating PhysiCal consistently improves the next-frame prediction performance of both DINO-WM and LeWM.

We provide the next-frame prediction result for all four environments, summarized in Table A8.

	Push-T				**Reacher**			
Method	PSNR↑	SSIM↑	LPIPS↓	IoU(%)↑	PSNR↑	SSIM↑	LPIPS↓	IoU(%)↑
TS-WM	24.95	0.975	0.0172	80.3	22.75	0.977	0.0188	57.7
DINO-WM	27.12	0.981	0.0097	85.9	23.79	0.981	0.0125	65.3
+ Long History	27.38	0.982	0.0093	86.6	**26.98**	**0.986**	**0.0062**	**78.2**
+ Context Prefix	27.07	0.981	0.0096	86.2	24.65	0.982	0.0107	69.7
+ Estimated Physics	**28.58**	**0.985**	**0.0076**	**89.3**	25.21	0.984	0.0096	71.6
+ PhysiCal (Ours)	28.29	0.984	0.0080	88.7	25.02	0.984	0.0097	71.3
LeWM	27.50	0.974	0.0292	85.8	24.13	**0.974**	**0.0222**	66.9
+ PhysiCal (Ours)	**28.74**	**0.977**	**0.0261**	**88.4**	**24.39**	0.973	0.0242	**67.4**
	BeerPong				**FetchSlide**			
Method	PSNR↑	SSIM↑	LPIPS↓	IoU(%)↑	PSNR↑	SSIM↑	LPIPS↓	IoU(%)↑
TS-WM	33.71	0.996	0.0043	62.5	29.15	0.990	0.0071	84.7
DINO-WM	35.81	0.997	0.0026	73.7	34.35	0.996	0.0021	93.4
+ Long History	**36.98**	**0.998**	**0.0018**	**78.6**	**35.49**	**0.997**	**0.0016**	94.4
+ Context Prefix	35.88	0.997	0.0026	74.8	34.73	0.996	0.0017	93.7
+ Estimated Physics	36.50	**0.998**	0.0022	76.5	34.57	0.996	0.0018	93.4
+ PhysiCal (Ours)	36.42	**0.998**	0.0021	76.6	35.45	**0.997**	0.0018	**94.7**
LeWM*	30.31	0.987	0.0153	67.2	29.77	**0.986**	**0.0283**	78.6
+ PhysiCal (Ours)	**31.61**	**0.989**	**0.0140**	**72.8**	**29.94**	0.985	0.0286	**79.3**

Table A8: **Next-frame prediction on four environments.** We report PSNR, SSIM, LPIPS, and foreground IoU (%). LeWM*: we use TC-LeWM in BeerPong to enhance performance (see Appendix A.1 for details). Bold and underlined values indicate the best and second-best results within each group separately: the six-method DINO-WM/TS-WM group and the two-method LeWM group.

D.2 MPC Results Across Execution Budgets

Following DINO-WM (Zhou et al., 2025), in addition to the open-loop CEM planning results, we also evaluate model predictive control (MPC), where the model replans from the newly observed state after executing a sequence of optimized actions. Specifically, in Push-T and Reacher, CEM optimizes an action sequence spanning 25 control steps (i.e., 5 model steps) using the world model. The full sequence is then executed in the environment, after which the agent replans if not successful. We allow at most two additional replans.

The result is shown in Table A9. PhysiCal consistently improves performance on Push-T under all maximum numbers of control steps, showing the advantage of our method. On Reacher, PhysiCal provides a particularly strong advantage within the first 25 execution steps. However, the open-loop advantage narrows with replanning because the task is highly feedback-correctable, allowing MPC to compensate for systematic model bias.

Method	**Push-T**			**Reacher**		
Max. control steps	25	50	75	25	50	75
TS-WM	53.3	54.7	56.0	41.3	65.3	68.0
DINO-WM	53.3	54.7	56.0	52.0	76.0	82.7
+ Long History	57.3	61.3	64.0	53.3	89.3	97.3
+ Context Prefix	58.7	60.0	60.0	61.3	85.3	92.0
+ Estimated Physics†	57.3	62.7	65.3	69.3	85.3	90.7
+ Privileged Physics‡	60.0	62.7	64.0	**86.7**	**100.0**	**100.0**
+ PhysiCal (Ours)	**62.7**	**66.7**	**68.0**	61.3	80.0	93.3
LeWM	48.0	48.0	49.3	60.0	84.0	90.7
+ PhysiCal (Ours)	**61.3**	**62.7**	**62.7**	**64.0**	**86.7**	**92.0**

Table A9: **MPC success rates on Push-T and Reacher under different maximum numbers of control steps.** Max. control steps denotes the maximum number of control steps allowed across all replanning iterations. Incorporating PhysiCal substantially improves both DINO-WM- and LeWM-based models across all allowed numbers of replans.

E Additional Qualitative Results

In this section, we provide additional qualitative results for world-model prediction and planning. We first present the interaction clips used in the main-text examples in Appendix E.1 and the uncropped BeerPong visualization in Appendix E.2. We then show an extrapolation example in Appendix E.3, followed by additional open-loop rollouts in Appendix E.4 and planning results in Appendix E.5 across all four environments. The extrapolation, open-loop, and planning examples are accompanied by their corresponding interaction clips.

E.1 Context Clips for Main-Text Examples

We present the in-context interaction clips corresponding to the main-text Push-T, BeerPong, and FetchSlide examples in Figure A2, Figure A3, and Figure A4, respectively. For each example, all context-conditioned models receive the same clip. PhysiCalWorld encodes this clip into a physics descriptor token, while Context Prefix directly attends to the encoded context sequence.

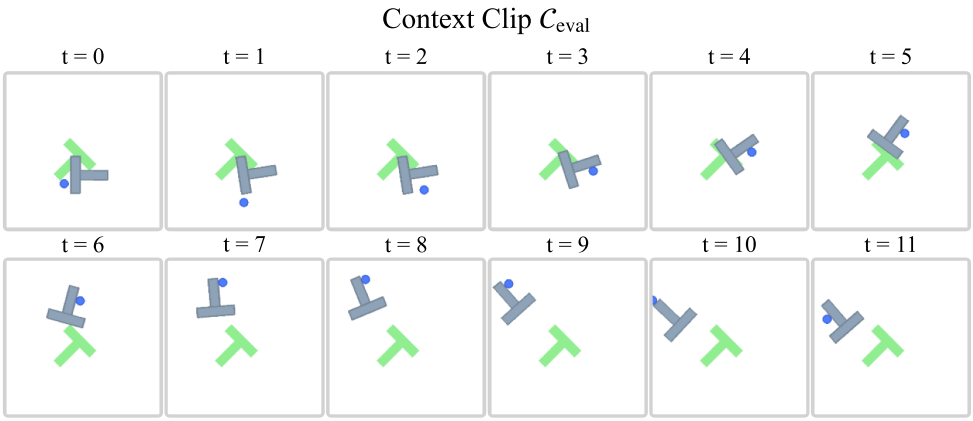


Figure A2: **Qualitative Example of an Exploratory In-Context Interaction Clip C_{eval} in the Push-T Environment.** We provide the in-context interaction clip used by PhysiCalWorld to generate the main-text rollout results in Figure 4(a). Collected using a CoM-agnostic PPO policy, this clip is encoded by the context encoder g_ϕ into a *physics descriptor token*, which is used to calibrate the world model's predictions.

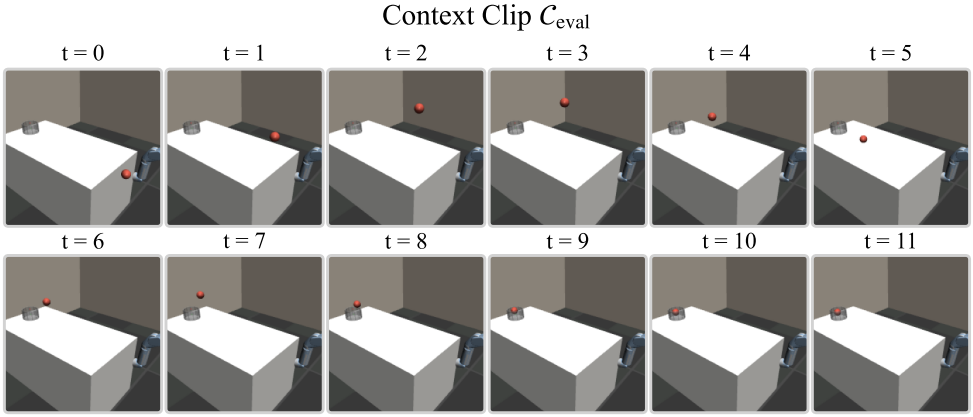


Figure A3: **Qualitative Example of an Exploratory In-Context Interaction Clip C_{eval} in the BeerPong Environment.** We show the in-context interaction clip used by PhysiCalWorld to produce the main-text rollout results in Figure 4(b). This clip is collected using a randomly sampled throw command and subsequently encoded by the context encoder g_ϕ into a *physics descriptor token*, which is used to calibrate the world model's predictions.

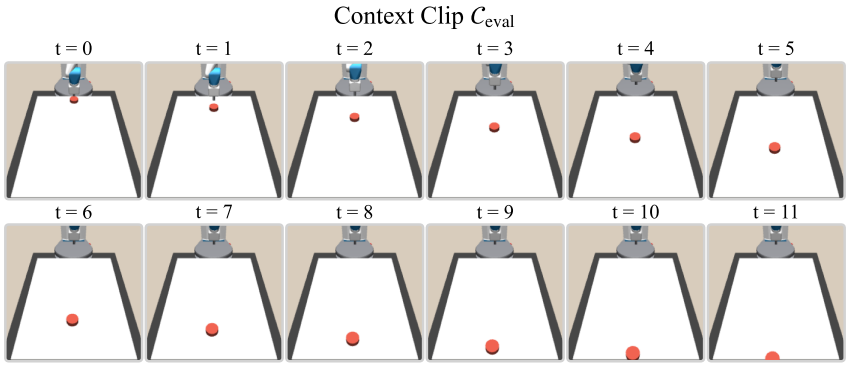


Figure A4: **Qualitative Example of an Exploratory In-Context Interaction Clip $\mathcal{C}_{\text{eval}}$ in the FetchSlide Environment.** We present the in-context interaction clip used by PhysiCalWorld to generate the main-text prediction results in Figure 5. Collected using a randomly sampled strike command, this clip is encoded by the context encoder g_ϕ into a *physics descriptor token*, which is used to calibrate the world model's predictions.

E.2 Uncropped BeerPong Visualization

We provide the uncropped view of the BeerPong rollouts shown in Figure 4(b). The full view in Figure A5 places the predicted ball trajectories within the surrounding scene, allowing their motion relative to the table and cup to be examined.

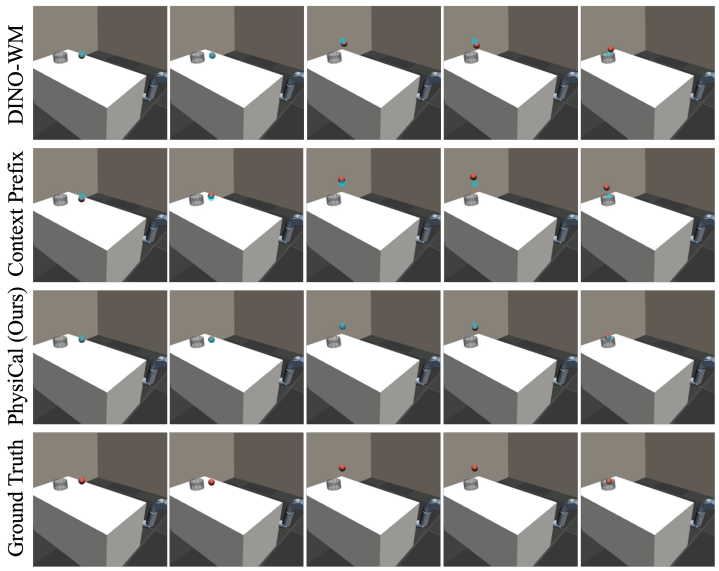


Figure A5: **Uncropped Open-Loop Rollouts in the BeerPong Environment.** We present the full-scene view of the BeerPong rollouts shown in Figure 4(b), displaying the predicted ball trajectories relative to the table and cup. DINO-WM + Ours accurately predicts the ball's bounce, while DINO-WM and DINO-WM + Context Prefix overestimate its rebound height. The semi-transparent cyan overlay marks the ground-truth ball position.

E.3 Qualitative Extrapolation Results

We provide a qualitative example complementing the Reacher extrapolation results in Figure 8. Following the main-text experiment, the first joint's damping lies outside the training range, while the second joint's damping remains fixed. We present the open-loop predictions in Figure A6 and the interaction clip used by PhysiCalWorld in Figure A7.

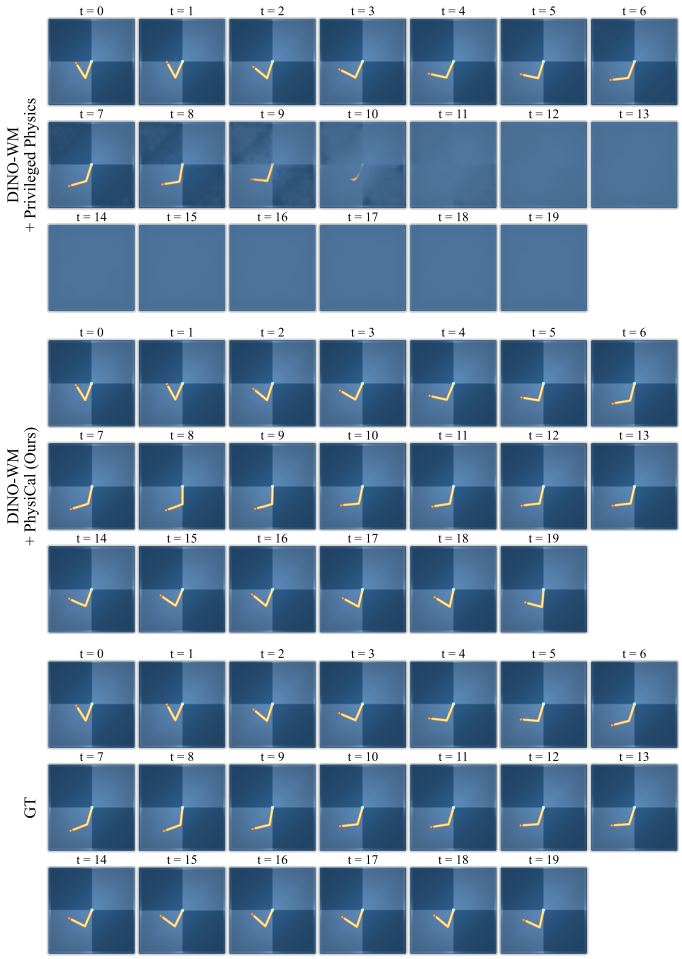


Figure A6: **Qualitative Open-Loop Rollouts in the Reacher Environment under Out-of-Distribution Physics.** We present an extrapolation example corresponding to the quantitative results in Figure 8, with the first joint's damping outside the training range and the second joint's damping held fixed. Both models predict future observations autoregressively without additional ground-truth observations. DINO-WM + Privileged Physics predicts accurately from $t = 0$ to $t = 6$, but background blurring emerges at $t = 7$ and progressively intensifies through $t = 10$. From $t = 11$ to $t = 19$, its predictions collapse into a nearly uniform blue field, with no discernible arm or scene structure. In contrast, DINO-WM + Ours preserves the scene structure and continues to predict the arm's motion throughout the rollout. The corresponding interaction clip is shown in Figure A7.

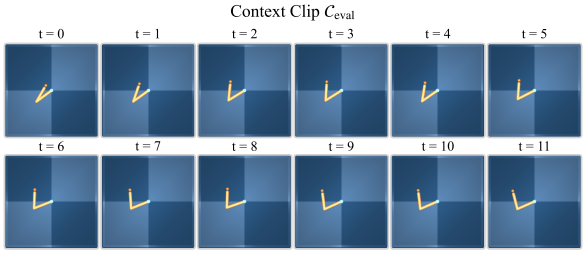


Figure A7: **Qualitative Example of an Exploratory In-Context Interaction Clip $\mathcal{C}_{\text{eval}}$ in the Reacher Environment under Out-of-Distribution Physics.** We provide the in-context interaction clip used by PhysiCalWorld to generate the extrapolation results in Figure A6. Collected using randomly sampled joint-torque commands, this clip is encoded by the context encoder g_ϕ into a *physics descriptor token*, which is used to calibrate the world model's predictions.

E.4 Additional Open-Loop Rollout Results

We provide additional open-loop rollout comparisons for Push-T, Reacher, BeerPong, and FetchSlide in Figure A8, Figure A10, Figure A12, and Figure A14, respectively. Starting from the initial observations, each model predicts future observations autoregressively under the recorded actions, without receiving further ground-truth observations. Each example is accompanied by the interaction clip shared by the context-conditioned models.

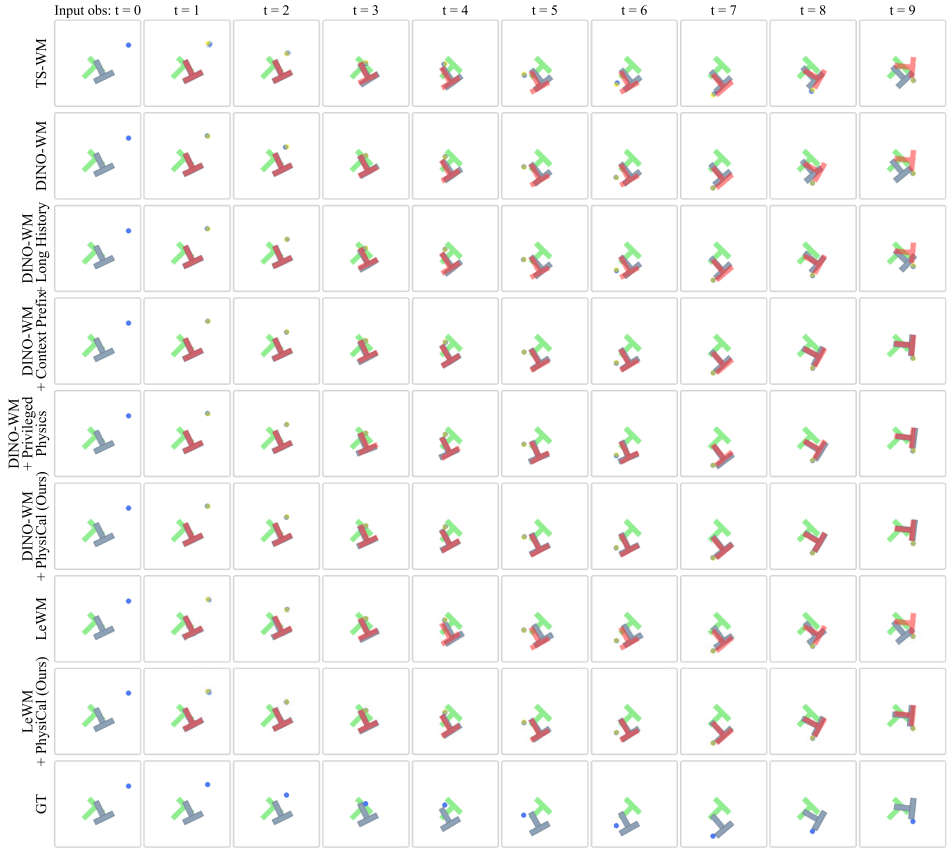


Figure A8: **Qualitative Open-Loop Rollouts in the Push-T Environment.** We present open-loop rollouts, where models autoregressively predict future observations without additional ground-truth observations. All context-conditioned models use the interaction clip shown in Figure A9. DINO-WM + Ours, LeWM + Ours, DINO-WM + Privileged Physics, and DINO-WM + Context Prefix closely match the ground-truth motion, while the remaining baselines exhibit substantial prediction errors. Although Context Prefix also predicts accurately, its direct attention over the full context sequence makes inference substantially slower.

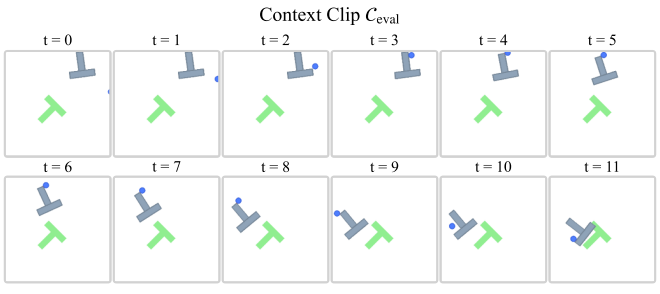


Figure A9: **Qualitative Example of an Exploratory In-Context Interaction Clip $\mathcal{C}_{\text{eval}}$ in the Push-T Environment.** This in-context interaction clip is used by PhysiCalWorld to generate the open-loop rollout results in Figure A8. The clip is collected using a CoM-agnostic PPO policy and encoded by the context encoder g_ϕ into a *physics descriptor token* to calibrate the world model's predictions.



Figure A10: **Qualitative Open-Loop Rollouts in the Reacher Environment.** We show autoregressive predictions of the arm's motion without providing further ground-truth observations during the rollout. The context-conditioned models share the interaction clip presented in Figure A11. DINO-WM + Ours, LeWM + Ours, and DINO-WM + Privileged Physics accurately predict the arm's motion. Context Prefix overestimates the motion, whereas the remaining baselines underestimate it, suggesting that it uses the context to adjust its predictions but remains less accurate than our models.

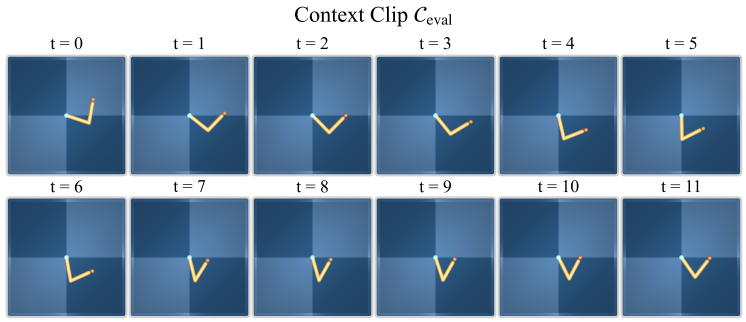


Figure A11: **Qualitative Example of an Exploratory In-Context Interaction Clip C_{eval} in the Reacher Environment.** We provide the in-context interaction clip used by PhysiCalWorld to produce the open-loop rollout results in Figure A10. Generated by randomly sampled joint-torque commands, this clip is encoded by the context encoder g_ϕ into a *physics descriptor token*, which is used to calibrate the world model's predictions.

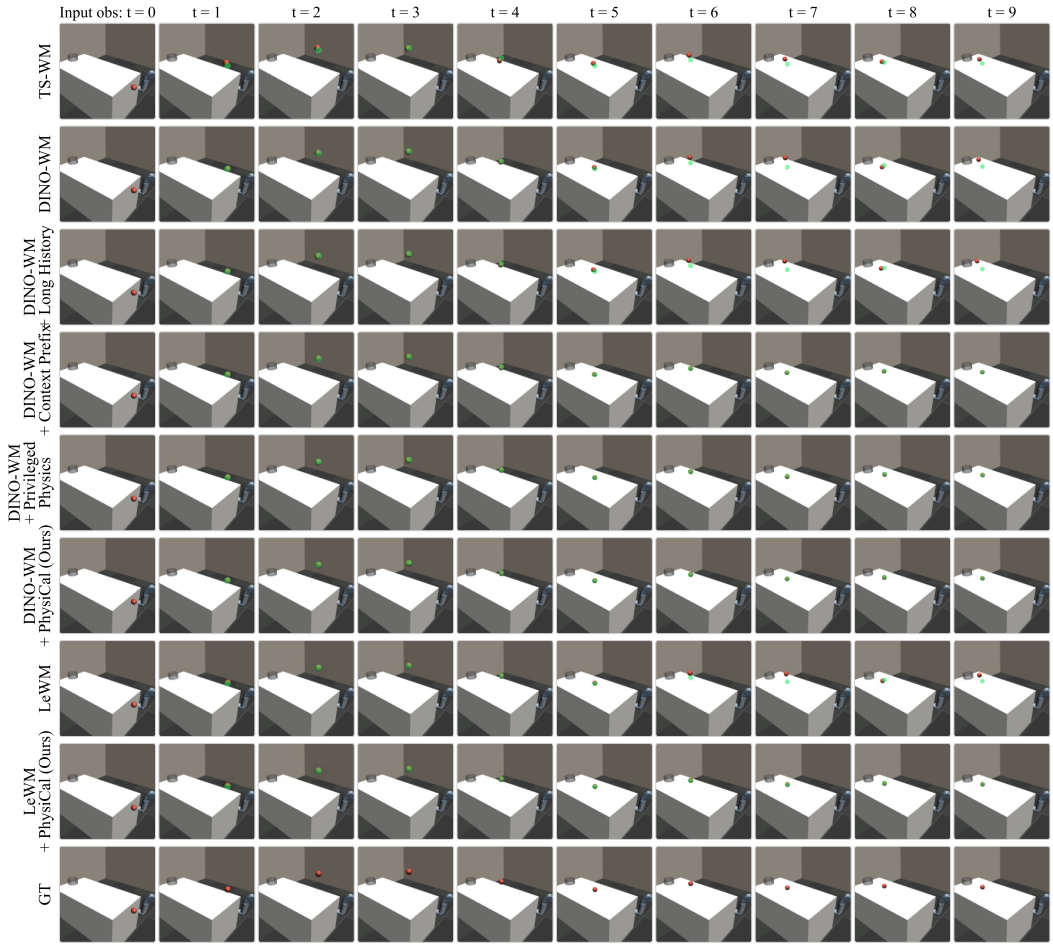


Figure A12: **Qualitative Open-Loop Rollouts in the BeerPong Environment.** We provide open-loop predictions of the ball's trajectory, generated autoregressively without subsequent ground-truth observations. All models that use context receive the interaction clip in Figure A13. DINO-WM + Ours, LeWM* + Ours, DINO-WM + Privileged Physics, and DINO-WM + Context Prefix accurately reproduce the ball's bounce, while the remaining baselines overestimate its rebound. The accuracy of Context Prefix, however, comes at the cost of substantially slower inference due to direct attention over the full context sequence. *LeWM denotes TC-LeWM in BeerPong.

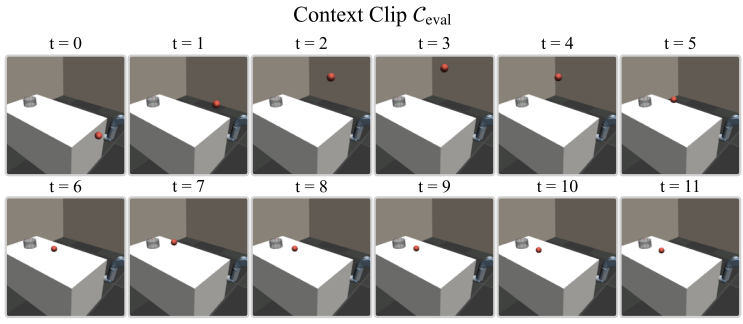


Figure A13: **Qualitative Example of an Exploratory In-Context Interaction Clip $\mathcal{C}_{\text{eval}}$ in the BeerPong Environment.** We present the in-context interaction clip used by PhysiCalWorld to generate the open-loop rollout results in Figure A12. The context encoder g_ϕ encodes this clip, collected using a randomly sampled throw command, into a *physics descriptor token* that calibrates the world model's predictions.

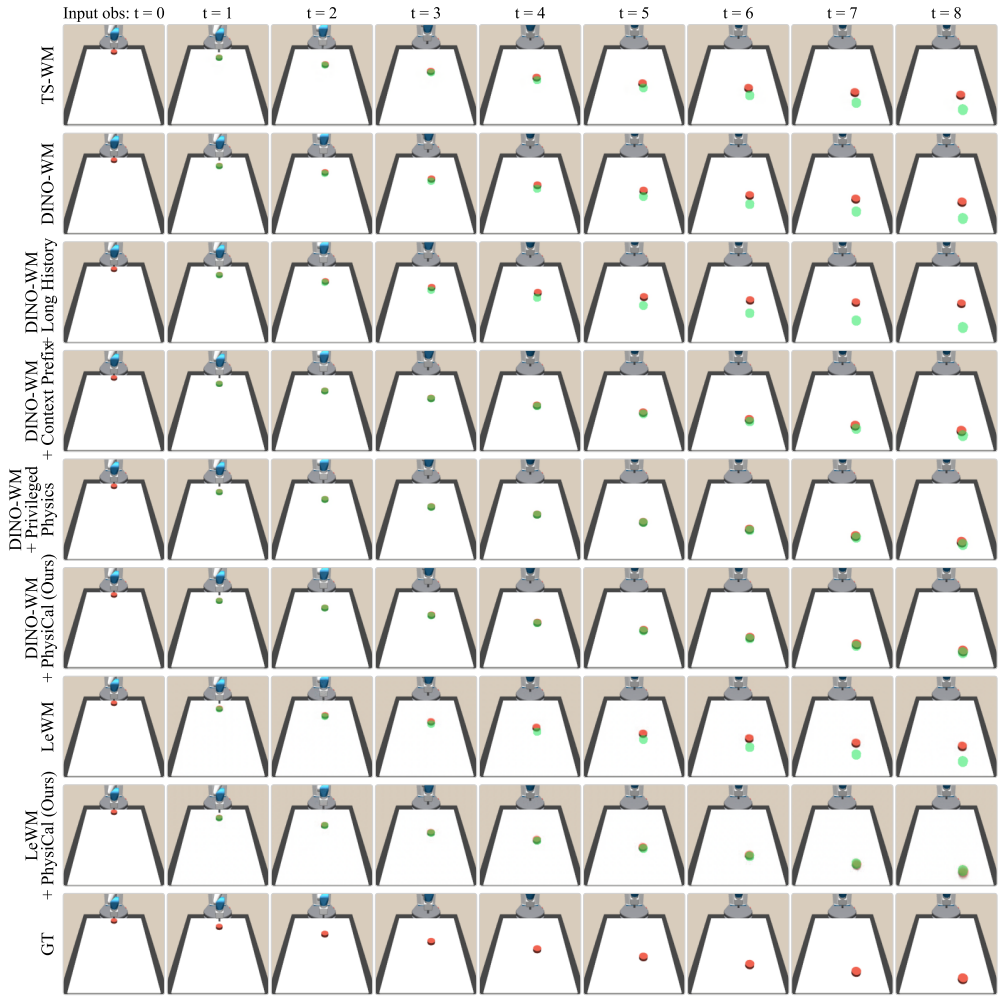


Figure A14: **Qualitative Open-Loop Rollouts in the FetchSlide Environment.** We present an eight-frame open-loop rollout, in which each model predicts future observations autoregressively without receiving additional ground-truth observations. The interaction clip shown in Figure A15 is shared by all context-conditioned models. DINO-WM + Ours, LeWM + Ours, and DINO-WM + Privileged Physics closely match the ground-truth puck motion. DINO-WM + Context Prefix slightly underestimates its movement, while the remaining baselines without context exhibit more pronounced underestimation.

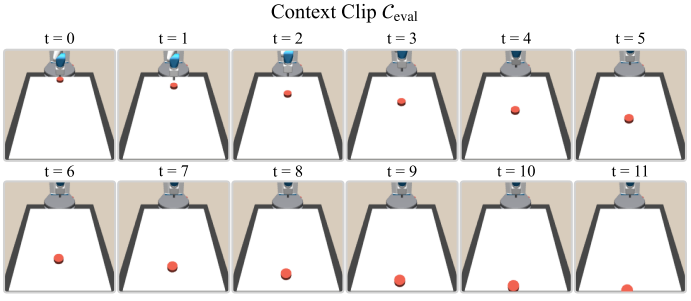


Figure A15: **Qualitative Example of an Exploratory In-Context Interaction Clip $\mathcal{C}_{\text{eval}}$ in the FetchSlide Environment.** We provide the in-context interaction clip used by PhysiCalWorld to generate the results in Figure A14. This clip is collected using a randomly sampled strike command and subsequently encoded by the context encoder g_ϕ into a *physics descriptor token*, which is used to calibrate the world model's predictions.

E.5 Additional Planning Results

We present additional planning comparisons across all four environments using DINO-WM- and LeWM-based models. For each example, CEM optimizes actions toward a goal observation using the world model, and the resulting actions are executed in the environment. We compare predicted trajectories with actual execution to illustrate how modeling errors affect goal achievement. Each planning example is followed by the corresponding interaction clip.

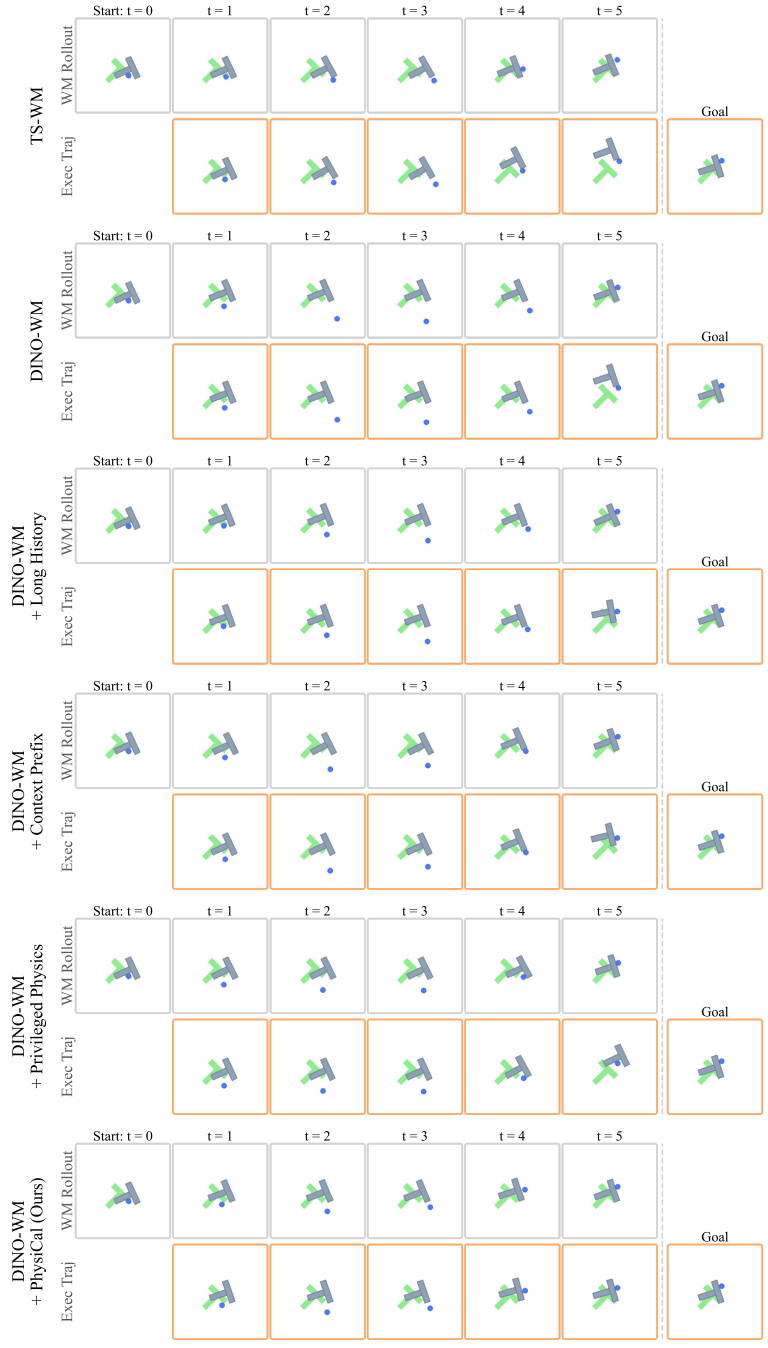


Figure A16: **Qualitative Planning Rollouts in the Push-T Environment, DINO-WM-Based Models.** We use CEM to optimize action sequences toward a goal observation and execute them in the environment, comparing model predictions with the resulting motion. All context-conditioned models use the interaction clip shown in Figure A17. DINO-WM + Ours closely matches the executed motion, while the remaining models misrepresent the T-block's translational and rotational dynamics.

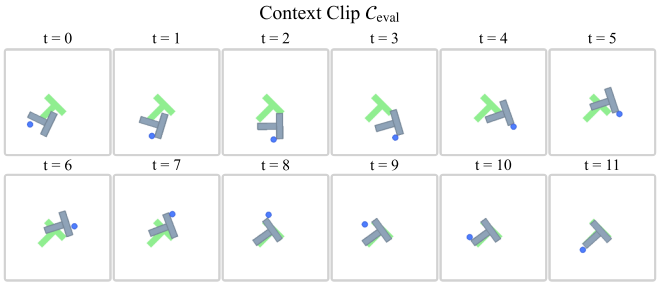


Figure A17: **Qualitative Example of an Exploratory In-Context Interaction Clip $\mathcal{C}_{\text{eval}}$ in the Push-T Environment.** We provide the in-context interaction clip used by PhysiCalWorld to generate the planning results in Figure A16. Collected using a CoM-agnostic PPO policy, this clip is encoded by the context encoder g_ϕ into a *physics descriptor token*, which is used to calibrate the world model's predictions during planning.

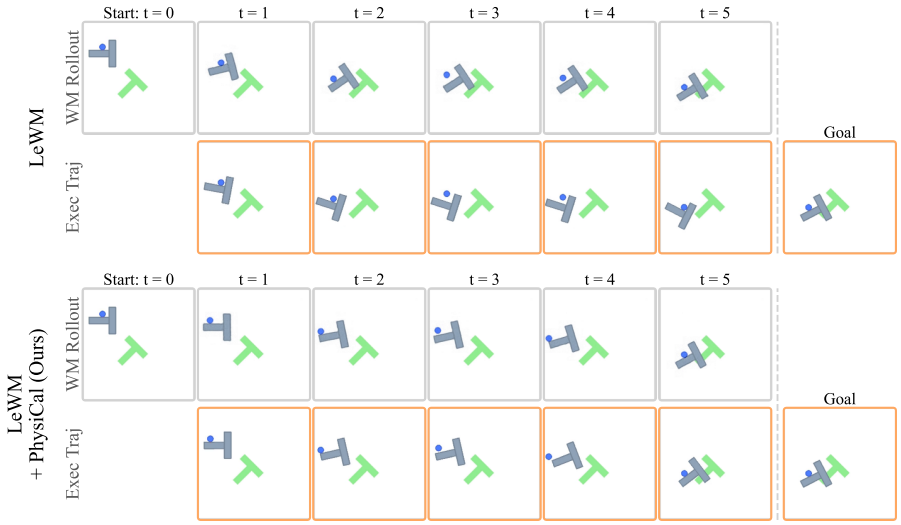


Figure A18: **Qualitative Planning Rollouts in the Push-T Environment, LeWM-Based Models.** Action sequences generated by CEM to reach a goal observation are executed in the environment and compared with the world models' predictions. LeWM + Ours uses the interaction clip presented in Figure A19 and accurately predicts the resulting motion. In contrast, LeWM underestimates the effect of the push at $t = 3$, with the T-block moving and rotating substantially farther than predicted at $t = 4$.

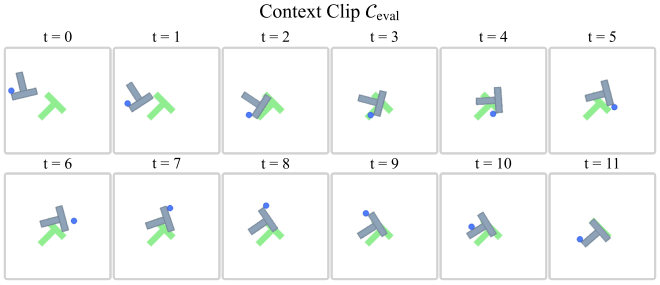


Figure A19: **Qualitative Example of an Exploratory In-Context Interaction Clip $\mathcal{C}_{\text{eval}}$ in the Push-T Environment.** We show the in-context interaction clip used by PhysiCalWorld to produce the planning results in Figure A18. This clip is collected using a CoM-agnostic PPO policy and subsequently encoded by the context encoder g_ϕ into a *physics descriptor token*, which is used to calibrate the world model's predictions during planning.

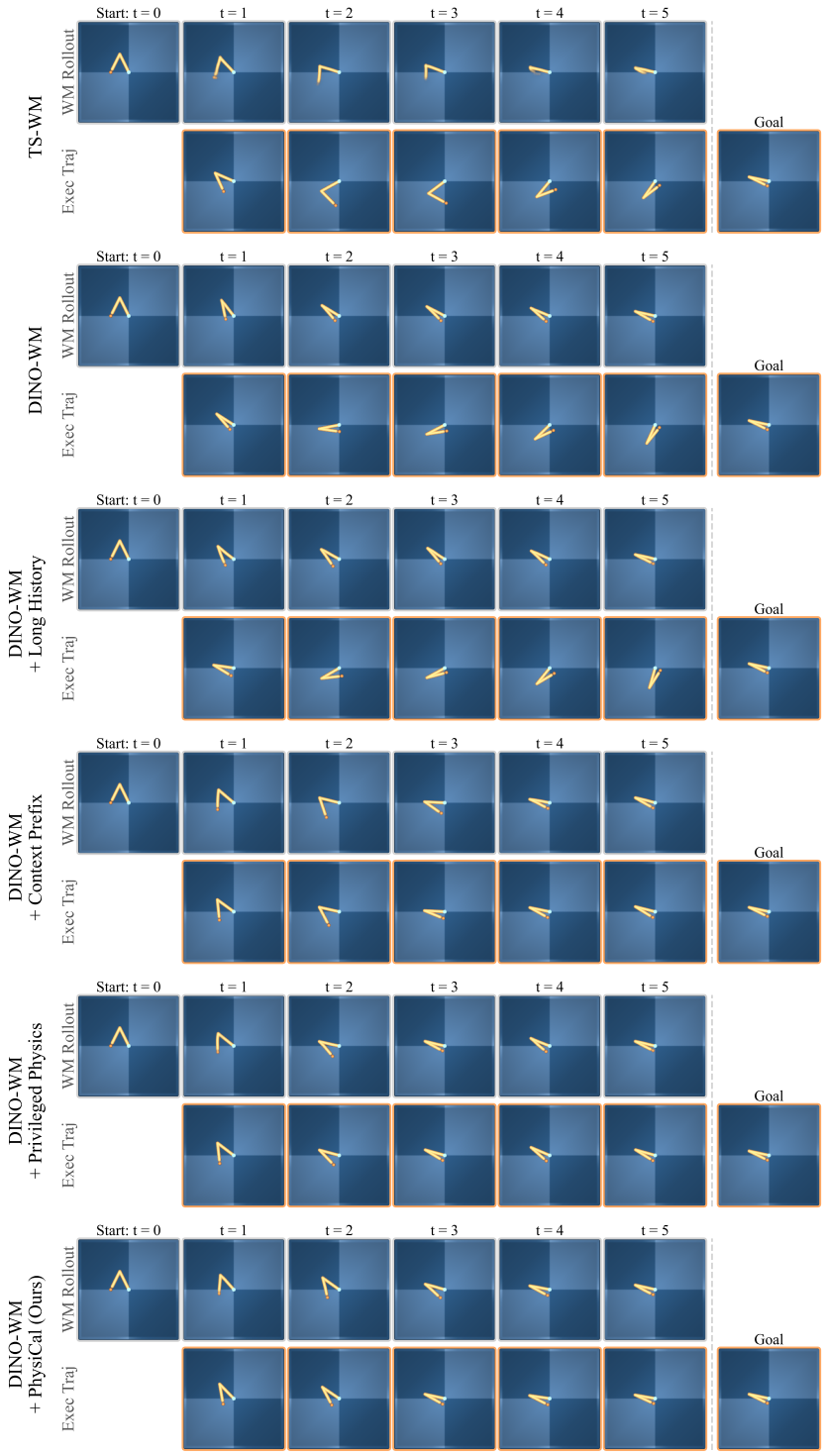


Figure A20: **Qualitative Planning Rollouts in the Reacher Environment, DINO-WM-Based Models.** We execute CEM-generated action sequences in the environment to reach a goal observation and compare the resulting trajectories with model predictions. The context-conditioned models share the interaction clip in Figure A21. DINO-WM + Ours, DINO-WM + Privileged Physics, and DINO-WM + Context Prefix accurately predict the arm's motion, while the remaining baselines underestimate it. Although Context Prefix predicts accurately, attending directly to the full context sequence makes inference substantially slower.

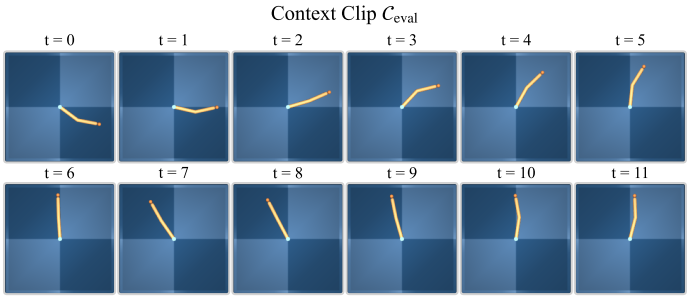


Figure A21: **Qualitative Example of an Exploratory In-Context Interaction Clip $\mathcal{C}_{\text{eval}}$ in the Reacher Environment.** We present the in-context interaction clip used by PhysiCalWorld to generate the planning results in Figure A20. Generated by randomly sampled joint-torque commands, this clip is encoded by the context encoder g_ϕ into a *physics descriptor token*, which is used to calibrate the world model's predictions during planning.

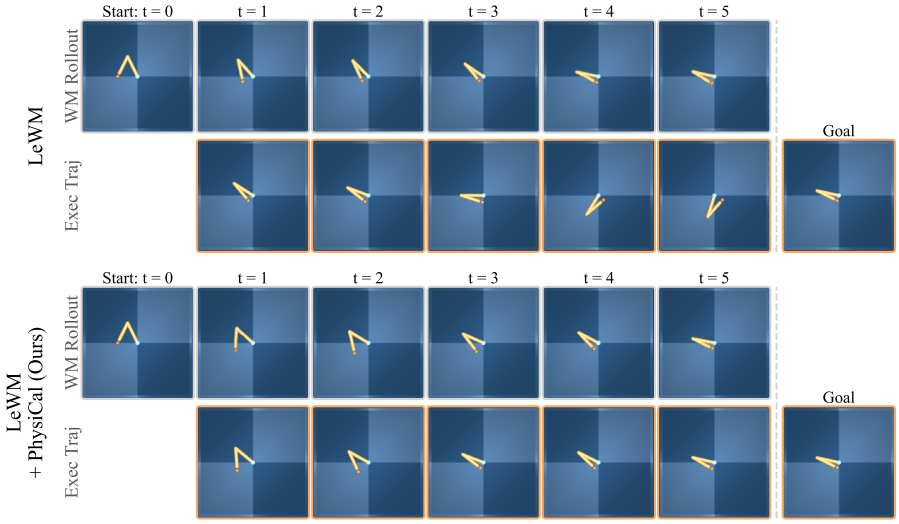


Figure A22: **Qualitative Planning Rollouts in the Reacher Environment, LeWM-Based Models.** CEM optimizes actions toward a goal observation; these actions are then executed in the environment to compare predicted and actual arm motion. Using the interaction clip shown in Figure A23, LeWM + Ours closely matches the executed trajectory. LeWM instead underestimates the arm's movement, leading the planner to apply excessive torque.

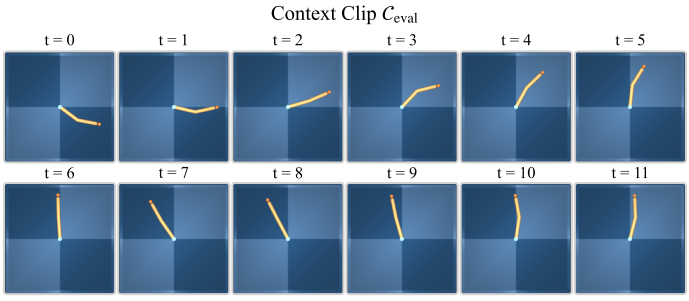


Figure A23: **Qualitative Example of an Exploratory In-Context Interaction Clip $\mathcal{C}_{\text{eval}}$ in the Reacher Environment.** This in-context interaction clip is used by PhysiCalWorld to produce the planning results in Figure A22. The context encoder g_ϕ encodes this clip, collected using randomly sampled joint-torque commands, into a *physics descriptor token* that calibrates the world model's predictions during planning.

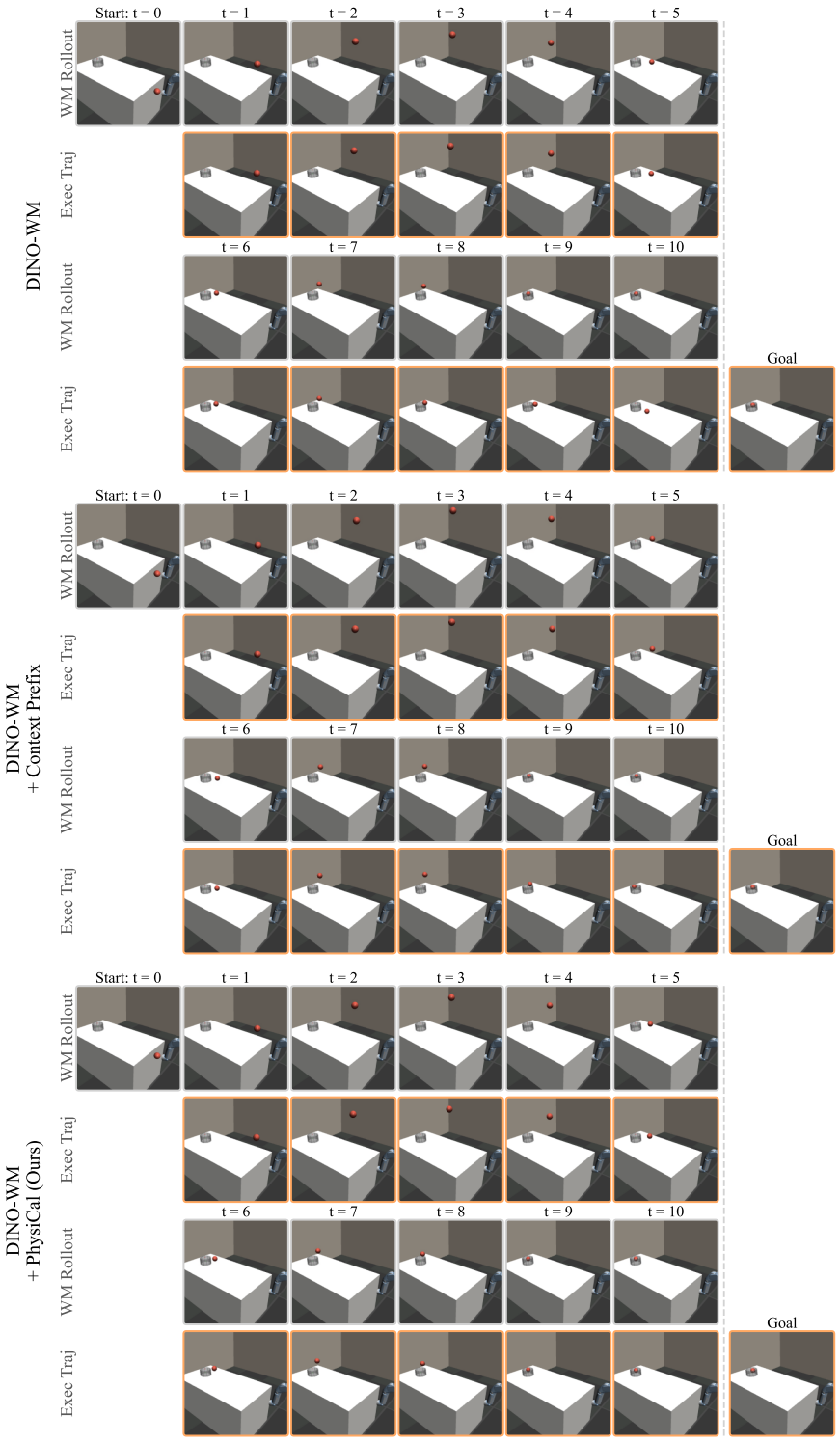


Figure A24: **Qualitative Planning Rollouts in the BeerPong Environment, DINO-WM-Based Models.** We use CEM to optimize a throw command toward the goal and execute it in the environment, comparing the predicted and actual ball trajectories. All context-conditioned models use the interaction clip shown in Figure A25. DINO-WM + Ours and DINO-WM + Context Prefix successfully land the ball in the cup, while DINO-WM overestimates the rebound height and its planned throw falls short. Although Context Prefix predicts accurately, attending to the full context sequence makes inference substantially slower.

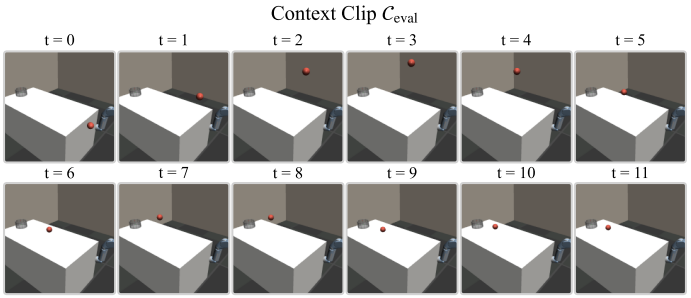


Figure A25: **Qualitative Example of an Exploratory In-Context Interaction Clip $\mathcal{C}_{\text{eval}}$ in the BeerPong Environment.** We provide the in-context interaction clip used by PhysiCalWorld to generate the planning results in Figure A24. Collected using a randomly sampled throw command, this clip is encoded by the context encoder g_ϕ into a *physics descriptor token*, which is used to calibrate the world model's predictions during planning.

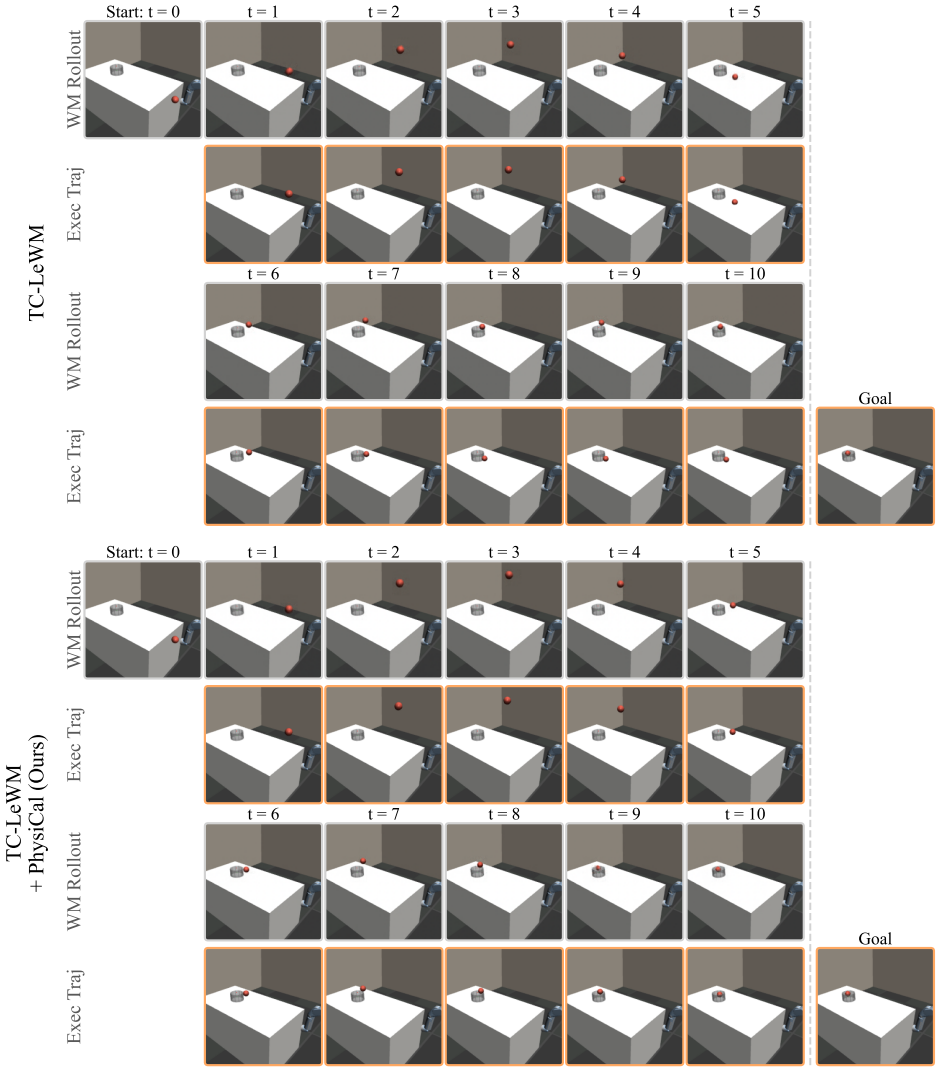


Figure A26: **Qualitative Planning Rollouts in the BeerPong Environment, LeWM*-Based Models.** We execute CEM-generated throw commands in the environment and compare the resulting ball trajectories with model predictions. LeWM* + Ours uses the interaction clip presented in Figure A27 and accurately predicts a successful throw into the cup. In contrast, LeWM* overestimates the ball's rebound, resulting in a throw that falls short. *LeWM denotes TC-LeWM in BeerPong.

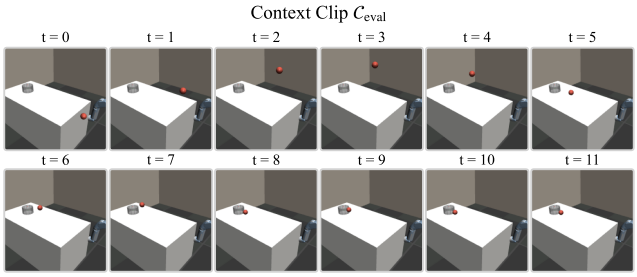


Figure A27: **Qualitative Example of an Exploratory In-Context Interaction Clip $\mathcal{C}_{\text{eval}}$ in the BeerPong Environment.** We show the in-context interaction clip used by PhysiCalWorld to produce the planning results in Figure A26. The clip is collected using a randomly sampled throw command and encoded by the context encoder g_ϕ into a *physics descriptor token* to calibrate the world model's predictions during planning.

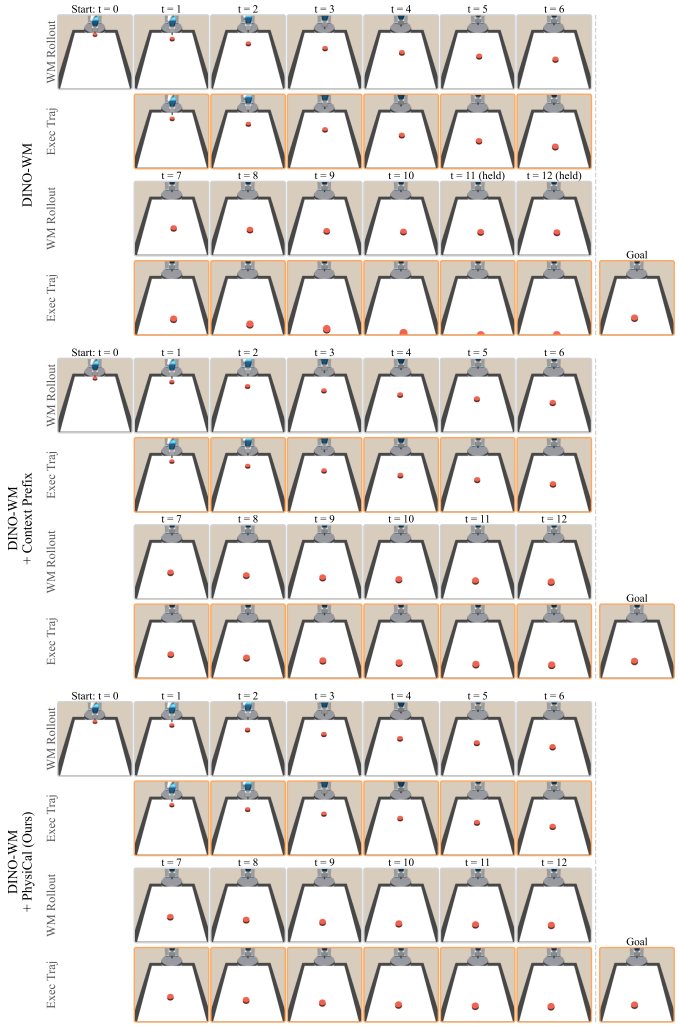


Figure A28: **Qualitative Planning Rollouts in the FetchSlide Environment, DINO-WM-Based Models.** CEM optimizes a strike command toward the goal, which is then executed in the environment to compare predicted and actual puck motion. The context-conditioned models share the interaction clip in Figure A29. DINO-WM + Ours closely matches execution, while DINO-WM and Context Prefix overestimate the puck's deceleration and consequently overshoot the goal. This error is more pronounced in DINO-WM; Context Prefix partially corrects it using context but remains less accurate than our model. "held" indicates that the final execution frame is held after the puck falls off the table.

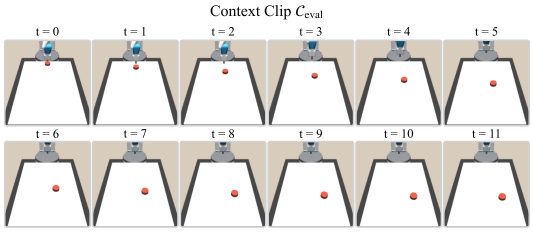


Figure A29: **Qualitative Example of an Exploratory In-Context Interaction Clip $\mathcal{C}_{\text{eval}}$ in the FetchSlide Environment.** We present the in-context interaction clip used by PhysiCalWorld to generate the planning results in Figure A28. Collected using a randomly sampled strike command, this clip is encoded by the context encoder g_ϕ into a *physics descriptor token*, which is used to calibrate the world model's predictions during planning.

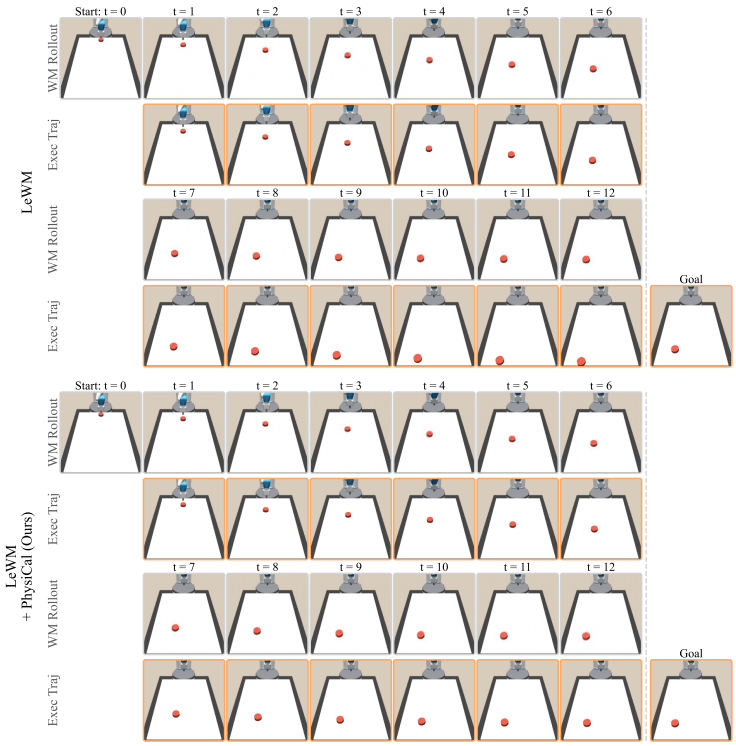


Figure A30: **Qualitative Planning Rollouts in the FetchSlide Environment, LeWM-Based Models.** We compare model predictions with environment execution of strike commands generated by CEM to reach the goal. Using the interaction clip shown in Figure A31, LeWM + Ours accurately predicts the puck's motion. LeWM instead overestimates its deceleration, leading to an overly strong strike that sends the puck beyond the goal.

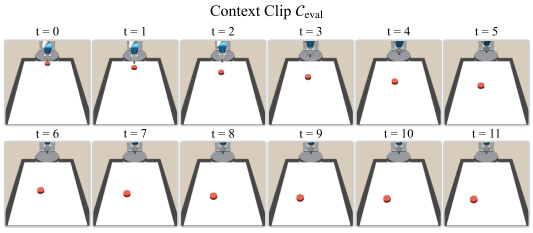


Figure A31: **Qualitative Example of an Exploratory In-Context Interaction Clip $\mathcal{C}_{\text{eval}}$ in the FetchSlide Environment.** This in-context interaction clip is used by PhysiCalWorld to generate the planning results in Figure A30. The context encoder g_ϕ encodes this clip, collected using a randomly sampled strike command, into a *physics descriptor token* that calibrates the world model's predictions during planning.