T2-Z: TEST-TIME OPTIMIZATION OF ENVIRONMENT LATENTS FOR ROBOTIC WORLD MODELS

Anonymous authors
Paper under double-blind review

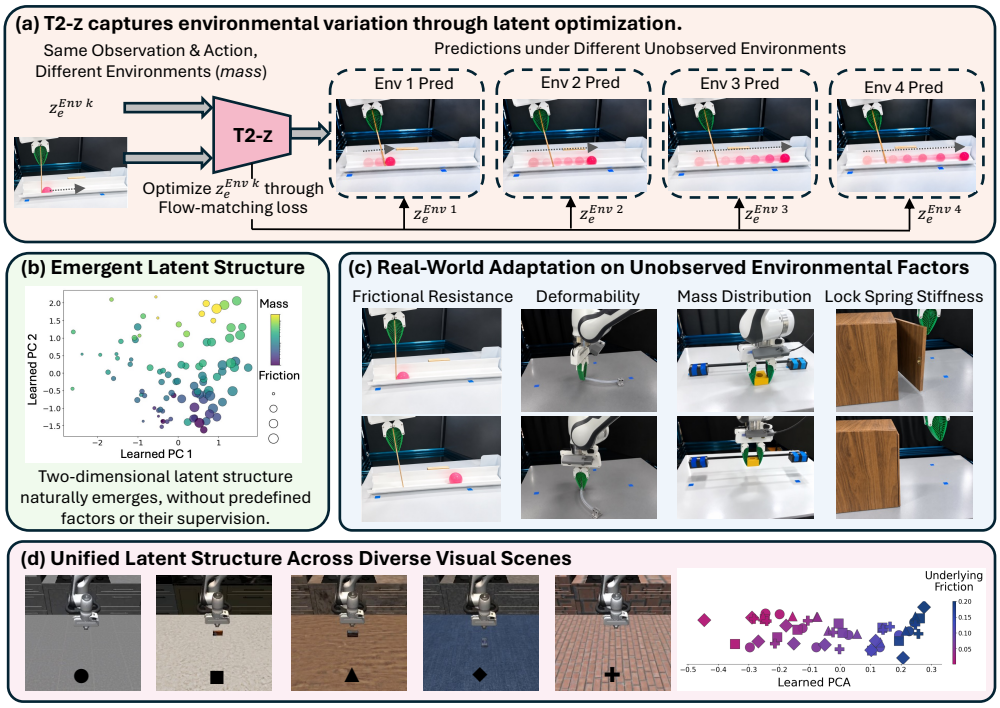


Figure 1: **T2-z for Test-Time Environment Latent Optimization.** **(a)** Given the same observation and action, T2-z directly optimizes environment latents to capture different environmental conditions (*e.g.,* ball mass), leading the world model to predict different outcomes. **(b)** Without knowing how many factors exist or what they represent, PCA of the learned environment latents reveals a two-dimensional structure that closely reflects the underlying factor distribution. **(c)** We evaluate T2-z on real-world robotic tasks spanning diverse unobserved environmental factors. **(d)** Across diverse visual scenes, the learned latents reveal a unified structure that reflects the same underlying environmental factor without being entangled with visual appearance.

ABSTRACT

Predicting how the world responds to an action requires more than what can be seen: robots must account for environmental factors that are neither directly observable nor explicitly specified, from physical properties such as friction and mass to human-defined environmental rules. We introduce T2-z for test-time optimization of environment latents, which augments a robotic world model with a learnable environment latent to condition world predictions. This compact latent captures underlying environmental variation without supervision of what environmental factors exist or what values they take. During training, T2-z alternates between two stages: a world-model optimization stage that fixes the latents and learns general action-conditioned dynamics across environments; and an environment-latent optimization stage that freezes the world model and optimizes the conditioning latents, allowing latent structure to emerge purely through world prediction. At test time, the world model remains frozen and a newly initialized environment latent is optimized from one or more test-time interactions, adapting the model to previously unseen environmental configurations. Across 7 simulated settings and 4 real-world robotic tasks, T2-z reduces object prediction error metric from 15.20 to 7.00 and raises action success from 33.73% to 72.96%. Its emer-

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.

gent latents capture structured representations of individual and compounded factors across diverse visual scenes. Ablations further show that T2-Z can efficiently infer environment latents from as little as a single test-time interaction trajectory.

1 INTRODUCTION

What a robot observes is insufficient to determine how the world will respond. Beneath visually similar scene configurations reside hidden environmental factors, such as friction, mass distribution, and unobserved human-defined interaction rules, that can fundamentally alter the consequences of the same action (fig. 1 (a)). Such factors are often revealed only through interaction with the target environments, as their influence becomes observable through the outcomes they induce. Explicitly representing these factors (Kumar et al., 2021), however, requires specifying in advance what factors may vary, their exact values, and how they should be parameterized, which is difficult to scale to unknown or compounded environmental variation. On the contrary, humans reason through "analysis by synthesis" (Yuille & Kersten, 2006), using internal generative models not only to anticipate future outcomes, but also in reverse to infer latent structure that best accounts for what they observe. This raises a natural question: can hidden environmental factors be similarly captured in a latent representation that is directly optimized from test-time interactions?

We introduce T2-Z for test-time optimization of environmental latents, which augments an action-conditioned robotic world model with a learnable environment latent $\mathbf{z}_e$. $\mathbf{z}_e$ is a randomly-initialized compact representation that is expected to encode the unobserved factors governing how an environment responds to interaction, with no supervision about what environmental factors it should represent. The world model predicts future visual observations conditioned jointly on the action and $\mathbf{z}_e$. We iteratively optimize the world model and $\mathbf{z}_e$ in an alternating two-stage procedure: 1) world-model optimization: with $\mathbf{z}_e$ fixed, we train the world model across all trajectories to learn shared action-conditioned dynamics across environments; 2) environment-latent optimization: with the world model frozen, we optimize the conditioned latent $\mathbf{z}_e$ through prediction error, so that it captures underlying environment-specific variation not explained by the general world model alone. Both stages are optimized with the same flow-matching loss and alternated for multiple rounds until convergence, allowing the environment representation to emerge directly through prediction without predefined physical-factor supervision or a separate environment encoder.

At test time, the world model is frozen and T2-Z turns environment inference into a direct optimization problem. For a previously unseen environmental configuration, we initialize a new $\mathbf{z}_e$ and optimize $\mathbf{z}_e$ to explain the outcomes of one or more observed test-time interactions. Specifically, we update $\mathbf{z}_e$ for multiple optimization steps through the same flow-matching objective used during stage-two training, minimizing prediction error on the observed interactions. Because inference is optimization-based rather than amortized through a learned history encoder, T2-Z is not tied to the trajectory count used for training and can flexibly incorporate any number of test-time trajectories. Once optimized, the latent turns the frozen world model into an environment-adapted test-time simulator, enabling prediction and comparison of candidate action outcomes before execution.

We evaluate T2-Z across 7 simulated settings and 4 real-world robotic tasks. Quantitatively and qualitatively, T2-Z improves world-model prediction across visual quality, physical prediction metrics (*e.g.*, relevant object locations), and downstream action selection. T2-Z reduces physical prediction error from 12.86 to 4.80 in simulation and tracked-state error from 19.30 to 10.85 pixels in the real world, while improving action success from 37.9% to 74.2% and from 26.5% to 70.8%, respectively. PCA of the learned latents reveals structured representations that reflect the underlying environmental factors, including a multi-dimensional organization when more than one factors vary jointly, despite the model never knowing how many factors are present or what they represent (fig. 1(b)). In our multi-background setting, a unified latent structure emerges across diverse scene setups, revealing that the learned representation captures environmental dynamics without becoming entangled with scene appearance (fig. 1(d)). Ablations further show that T2-Z can infer useful environment representations from as few as a single test-time interaction trajectory.

2 RELATED WORKS

Two distinct strands of work are particularly related to our problem.

Robotic world models have made rapid progress in predicting action-conditioned futures (Zhou et al., 2025; 2024; Zhu et al., 2024; Zhen et al., 2025; Zhu et al., 2025; Ye et al., 2026; Huang et al.,

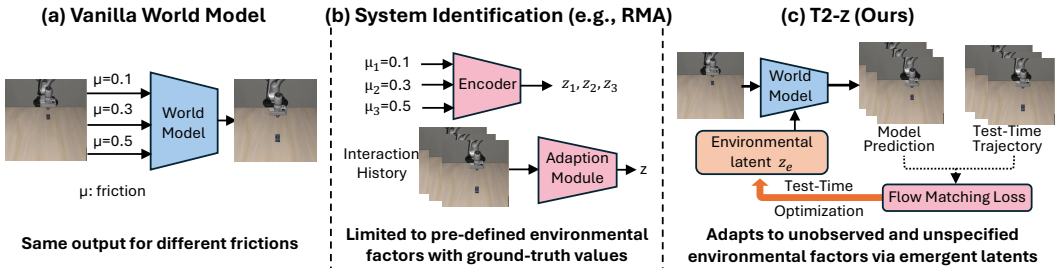


Figure 2: **Comparison with existing approaches.** **(a)** Vanilla world models do not explicitly represent unobserved environmental variation, producing the same prediction under different environmental conditions. **(b)** System-identification approaches such as RMA learn environment representations from predefined factors and their ground-truth values, then infer them from interaction history through a learned encoder. **(c)** T2-z requires neither predefined environmental factors nor a learned environment encoder, instead directly optimizing an emergent environment latent from test-time interaction through prediction error.

2026; Gao et al., 2026). More recent work explores test-time adaptation to improve long-horizon memory or incorporate new behavioral supervision (Wan et al., 2026; Wang et al., 2026; Feng et al., 2026; Yoo et al., 2025; Jang et al., 2026). These approaches predict future outcomes from visual observations alone, without accounting for underlying environmental variation that is not directly observable. (fig. 2 (a)). In contrast, T2-z disentangles unobserved environmental variation from the world model's observable scene representation, through environment latent conditioning.

Environment representations and system identification have demonstrated the value of environment-specific representations for system identification and rapid behavioral adaptation (Yu et al., 2017; Nagabandi et al., 2019; Rakelly et al., 2019; Lee et al., 2020; Kumar et al., 2021; Miki et al., 2022; Nahrendra et al., 2023; Wang et al., 2023; He et al., 2024; Luo et al., 2024). The Rapid Motor Adaptation (RMA) family and related methods typically rely on predefined privileged parameters (Kumar et al., 2021; Miki et al., 2022; Nahrendra et al., 2023; He et al., 2024; Luo et al., 2024). For example, RMA restricts adaptation to 5 factors and uses their ground-truth values to learn an embedding that an adaptation encoder later recovers from interaction history (Kumar et al., 2021). This prespecifies the environment representation and ties inference to a learned history-to-embedding mapping, limiting flexibility under unknown, compounded, or out-of-distribution variation (fig. 2 (b)). These methods also generally rely on a learned history encoder (Chen et al., 2024; Xing et al., 2026), *limiting generalization when the amount of test-time interaction differs from that seen during training*. In contrast, T2-z learns representations of environmental factors implicitly through world prediction error, without predefined ground-truth physical parameters or a learned history encoder.

Our method is also related to the literature on **Test-Time Adaptation** (TTA), which has enabled robots to improve after deployment by adapting policy, prompts, or memory rather than optimizing a dedicated environment representation. Existing approaches include test-time adaptation of diffusion and flow policies (Li et al., 2025; 2026), online policy adaptation (Liu et al., 2026; Jin & Zhang, 2026; Zhu et al., 2026), and adaptation through latent prompts, fast weights, memory, or retrieval (Zhang et al., 2026; Jiang et al., 2026; Feng et al., 2026; Zhao et al., 2026; Park et al., 2026). TTA has also been applied directly to predictive models through video replanning, world-model training, and closed-loop latent adaptation (Ko et al., 2025; Wan et al., 2026; Wang et al., 2026).

3 Method

We introduce T2-z (fig. 2 & algorithm 1), a framework for adapting robotic world models to hidden environmental variation through direct optimization of an environment latent. Our key idea is to separate the general visual predictive capability of a world model from unobserved environment-specific information. During training, we learn these two components through alternating optimization under a shared flow-matching objective. At test time, we freeze the world model and optimize only a newly initialized environment latent from observed test-time interaction trajectories. The optimized latent subsequently conditions the frozen world model to predict the outcomes of diverse actions.

3.1 Problem Setup

We consider a robotic world model that predicts future visual observations $\mathbf{y}$ from the observation $\mathbf{x}$ and action $\mathbf{a}$. We augment the world model with an environment latent $\mathbf{z}_e$ and formulate prediction as:

$$p_\theta\left(\mathbf{y} \mid \mathbf{x}, \mathbf{a}, \mathbf{z}_e\right), \tag{1}$$

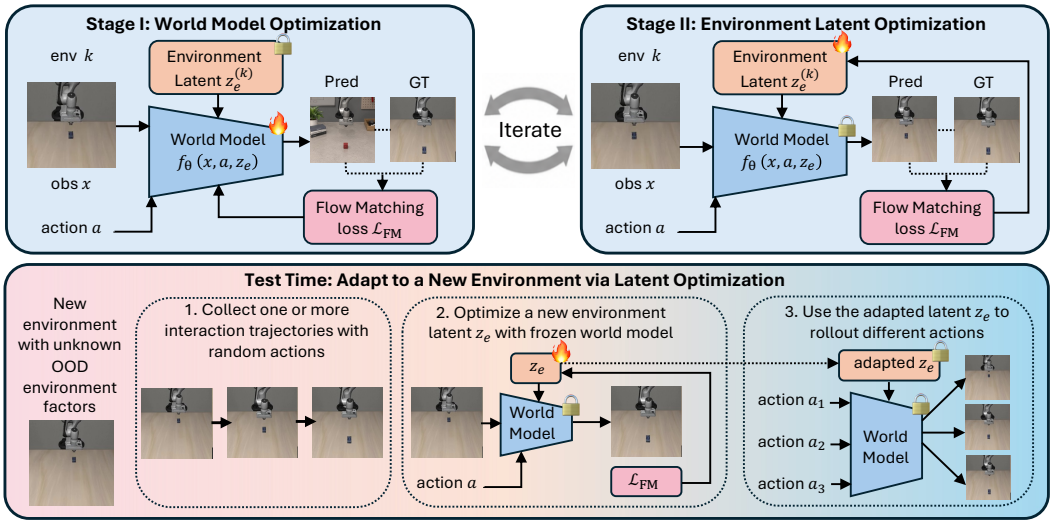


Figure 3: **T2-Z framework.** During training, T2-Z alternates between two stages: (1) world-model optimization, which learns shared action-conditioned dynamics with environment latents fixed, and (2) environment-latent optimization, which optimizes the latents through prediction error with the world model frozen. (3) At test time, the world model remains frozen while a new environment latent is directly optimized from interaction and then reused to predict the outcomes of different actions.

where θ denotes the shared world-model parameters and $\mathbf{z}_e$ is a latent representation of the environment. Importantly, $\mathbf{z}_e$ receives no supervision of which environmental factors it should represent.

Let $\mathbf{y}_t$ denote the noisy/interpolated future observation at flow timestep t, and $\mathbf{u}_t$ its corresponding target flow velocity. The model is optimized using the standard flow-matching objective,

$$\mathcal{L}_{\text{FM}}(\theta, \mathbf{z}_e) = \mathbb{E}\left[\|\mathbf{v}_\theta(\mathbf{y}_t, t \mid \mathbf{x}, \mathbf{a}, \mathbf{z}_e) - \mathbf{u}_t\|_2^2\right]. \tag{2}$$

3.1.1 Training

For each training environment e, we maintain an environment latent $\mathbf{z}_e$, randomly initialized by sampling each dimension from a uniform distribution:

$$\mathbf{z}_e^{(0)} \sim \mathcal{U}(0, 1). \tag{3}$$

We alternate between two stages throughout training: world-model optimization and environment-latent optimization, with k indexing the outer optimization cycles. The world model is updated with low learning rate to learn general dynamics shared across environments, while the environment latents are optimized rapidly for quick adaptation to environment information revealed through interaction. This slow–fast separation follows the perspective of learning at different timescales (Tiwari et al., 2026) and is closely related to test-time learning with slow/fast weights (Hong et al., 2024).

Algorithm 1 T2-Z Training and Test-Time Optimization

Require: Training sets $\{\mathcal{D}_e\}_{e=1}^{E}$, world model p_θ, loss $\mathcal{L}_{\text{FM}}$, learning rate η_θ and η_z

Initialize θ and $\mathbf{z}_e \sim \mathcal{U}(0, 1)$ for each environment e

// Training: alternating optimization

while not converged **do**

// Stage I: world-model optimization

  Freeze environment latents $\{\mathbf{z}_e\}_{e=1}^{E}$

  for world-model optimization steps **do**

    $\theta \leftarrow \theta - \eta_\theta \nabla_\theta \mathcal{L}_{\text{FM}}(\theta, \mathbf{z}_e)$

  end for

// Stage II: environment-latent optimization

  Freeze world-model parameters θ

  for environment-latent optimization steps **do**

    $\mathbf{z}_e \leftarrow \mathbf{z}_e - \eta_z \nabla_{\mathbf{z}_e} \mathcal{L}_{\text{FM}}(\theta, \mathbf{z}_e)$

  end for

end while

// Test-time optimization

Require: Trained world model θ^*, observed interactions $\mathcal{D}_{\text{test}}$, Optimization Step s, learning rate $\eta_z^{(s)}$

Freeze θ^* and initialize $\mathbf{z}_{\text{test}}^{(0)}$

for $s = 0, \ldots, S-1$ **do**

  $\mathcal{L}_{\text{test}} \leftarrow \frac{1}{|\mathcal{D}_{\text{test}}|} \sum_{(\mathbf{x},\mathbf{a},\mathbf{y}) \in \mathcal{D}_{\text{test}}} \mathcal{L}_{\text{FM}}(\theta^*, \mathbf{z}_{\text{test}}^{(s)})$

  $\mathbf{z}_{\text{test}}^{(s+1)} \leftarrow \mathbf{z}_{\text{test}}^{(s)} - \eta_z^{(s)} \nabla_{\mathbf{z}_{\text{test}}^{(s)}} \mathcal{L}_{\text{test}}$

end for

return $\mathbf{z}_{\text{test}}^{*} \leftarrow \mathbf{z}_{\text{test}}^{(S)}$

Stage I: World-Model Optimization. In the first stage, we fix all $\mathbf{z}_e$ and optimize only the world-model parameters. Let $\mathcal{D}_e$ denote the interaction trajectories collected under environment e.

$$\theta^{(k+1)} = \arg\min_{\theta} \mathbb{E}_{e,\,(\mathbf{x},\mathbf{a},\mathbf{y})\sim\mathcal{D}_e} \left[\mathcal{L}_{\mathrm{FM}} \left(\theta, \mathbf{z}_e^{(k)} \right) \right]. \tag{4}$$

During this stage, $\mathbf{z}_e$ is held constant, while the same world model is optimized over trajectories spanning different environments and actions. This encourages θ to capture predictive structure that is shared across the training distribution, including how visual states evolve to different robot actions.

Stage II: Environment-Latent Optimization. In the second stage, we reverse the optimization: the world model is frozen, and only the environment latents are updated. For each environment e,

$$\mathbf{z}_e^{(k+1)} = \arg\min_{\mathbf{z}_e} \mathbb{E}_{(\mathbf{x},\mathbf{a},\mathbf{y})\sim\mathcal{D}_e} \left[\mathcal{L}_{\mathrm{FM}} \left(\theta^{(k+1)}, \mathbf{z}_e \right) \right]. \tag{5}$$

Because θ is fixed, the only way to reduce prediction error is to change $\mathbf{z}_e$ to better match the interaction outcomes observed in environment e. As a result, environments that induce systematically different outcomes drive their latents toward different regions of the latent space.

Alternating Optimization Training starts with Stage I, then alternates between Stage II and Stage I, each optimized for a fixed number of steps while keeping the other component fixed. As the world model improves, latents are optimized against an increasingly capable predictive model; conversely, the refined latents provide more informative conditioning for subsequent world-model optimization.

3.1.2 Test-Time Optimization

At test time, we adapt to a previously unseen environment by optimizing only its environment latent while keeping the trained world model fixed. We initialize a new latent $\mathbf{z}_{\mathrm{test}}^{(0)}$ and observe one or more interaction trajectories on randomly-sampled actions. We denote the resulting test-time interaction set as

$$\mathcal{D}_{\mathrm{test}} = \{(\mathbf{x}_i, \mathbf{a}_i, \mathbf{y}_i)\}_{i=1}^{N}, \tag{6}$$

where each tuple consists of an observation $\mathbf{x}_i$, an action $\mathbf{a}_i$, and the resulting observations $\mathbf{y}_i$.

We then optimize $\mathbf{z}_{\mathrm{test}}$ using the same flow-matching objective employed during training:

$$\mathbf{z}_{\mathrm{test}}^{(s+1)} = \mathbf{z}_{\mathrm{test}}^{(s)} - \eta_z^{(s)} \nabla_{\mathbf{z}_{\mathrm{test}}^{(s)}} \frac{1}{|\mathcal{D}_{\mathrm{test}}|} \sum_{(\mathbf{x},\mathbf{a},\mathbf{y})\in\mathcal{D}_{\mathrm{test}}} \mathcal{L}_{\mathrm{FM}} \left(\theta^*, \mathbf{z}_{\mathrm{test}}^{(s)}; \mathbf{x}, \mathbf{a}, \mathbf{y} \right). \tag{7}$$

where θ^* denotes the frozen world-model parameters and $\eta_z^{(s)}$ follows a coarse-to-fine learning-rate strategy, with larger learning rates in early optimization steps and smaller learning rates in later steps. Because the world model is fixed, prediction errors over the observed interactions can only be reduced by updating $\mathbf{z}_{\mathrm{test}}$. The latent is therefore optimized toward a representation under which the frozen world model reproduces the interaction outcomes observed in the current environment.

Once optimized, $\mathbf{z}_{\mathrm{test}}^*$ is reused to condition the world model for actions that were not observed during latent optimization. Given a different action $\mathbf{a}'$, we predict its outcome as

$$\hat{\mathbf{y}}' \sim p_{\theta^*} \left(\mathbf{y} \mid \mathbf{x}, \mathbf{a}', \mathbf{z}_{\mathrm{test}}^* \right). \tag{8}$$

The resulting conditioned world model serves as an environment-adapted test-time simulator, allowing it to predict the consequences of different actions under the inferred environment.

Unlike amortized history-encoder approaches, T2-Z uses direct optimization. This avoids committing to a fixed history mapping and searches for the representation that best explains the current environment. The formulation can naturally incorporate any number of test-time trajectories that differs from training, by optimizing a latent against all available evidence. Moreover, T2-Z is not restricted to the range of factor values during training, providing greater flexibility under out-of-distribution configurations. Ablation in fig. 5 shows this optimization remains stable across varying test-time trajectory numbers, and is interaction-efficient with a single trajectory sufficient in most settings.

4 Experiments

4.1 Experimental Setup

Tasks We evaluate T2-Z across 7 simulated settings and 4 real-world robotic tasks. In simulation, **Friction** requires predicting how far an object moves after a push under different surface friction; **Gravity** varies gravitational strength and requires predicting the object motion; **Mass Collision**

Table 1: **Simulation performance across environmental variations.** For Friction, Gravity, Mass Collision, Mass × Friction, and Diverse Scenes, physical error is measured in pixels; for Light Switch, it is lamp-score error; and for Mass Balance, it is bar-tilt error in degrees. The average is computed over all seven tasks.

Task	Method	Physical Error (Mean) ↓	Physical Error (Final) ↓	LPIPS ↓	PSNR ↑	Action Success ↑
Mass Balance	Standard World Model	1.1844	4.5571	0.0244	32.969	40.0%
	LoRA TTA	1.7975	7.3105	0.0631	24.895	50.0%
	DINOv2 Context	1.2636	4.9480	**0.0239**	**33.263**	30.0%
	TTT-KQV	2.5075	10.1759	0.0312	32.114	30.0%
	T2-z (Ours)	**0.6403**	**2.0842**	0.0240	32.814	**80.0%**
Friction	Standard World Model	11.0626	24.4116	0.0667	31.113	32.0%
	LoRA TTA	8.1902	19.1902	0.0582	**32.031**	48.0%
	DINOv2 Context	11.8142	26.1703	0.0681	30.697	48.0%
	TTT-KQV	10.8137	23.8675	0.0649	30.779	32.0%
	T2-z (Ours)	**3.8540**	**7.5931**	**0.0495**	31.976	**72.0%**
Gravity	Standard World Model	21.4246	34.8779	0.0345	32.509	23.5%
	LoRA TTA	7.9005	13.7137	0.0231	33.328	47.1%
	DINOv2 Context	**5.9300**	**8.8290**	**0.0195**	**34.095**	76.5%
	TTT-KQV	18.9572	30.9355	0.0291	33.308	35.3%
	T2-z (Ours)	7.3408	11.0653	0.0202	33.389	**82.4%**
Mass Collision	Standard World Model	21.9329	25.7364	0.0904	29.805	38.9%
	LoRA TTA	15.8704	24.5877	0.0691	30.049	33.3%
	DINOv2 Context	8.5264	10.2101	0.0540	31.039	38.9%
	TTT-KQV	20.9753	29.2930	0.0842	29.385	55.6%
	T2-z (Ours)	**6.4522**	**8.1321**	**0.0538**	**31.150**	**61.1%**
Light Switch	Standard World Model	10.3500	18.8200	0.0187	32.619	50.0%
	LoRA TTA	8.2800	14.7700	0.0175	33.123	50.0%
	DINOv2 Context	10.0000	18.3700	0.0160	33.074	50.0%
	TTT-KQV	10.4800	18.8700	0.0161	32.692	50.0%
	T2-z (Ours)	**3.4300**	**4.7400**	**0.0150**	**33.517**	**87.5%**
Mass × Friction	Standard World Model	16.0219	29.2454	0.0610	31.077	35.0%
	LoRA TTA	16.8282	26.9302	0.0701	29.865	26.7%
	DINOv2 Context	11.8116	19.0756	0.0565	**31.367**	60.8%
	TTT-KQV	18.9511	29.1411	0.0726	29.739	33.3%
	T2-z (Ours)	**8.1481**	**13.7222**	**0.0500**	30.673	**69.7%**
Diverse Backgrounds	Standard World Model	8.0260	16.1831	0.1059	28.201	45.8%
	LoRA TTA	6.5805	16.2328	0.1074	27.986	58.3%
	DINOv2 Context	7.7367	16.0065	0.1064	28.000	58.3%
	TTT-KQV	7.7783	15.5869	0.1116	27.691	58.3%
	T2-z (Ours)	**3.7145**	**6.3403**	**0.0888**	**28.804**	**66.7%**
Average	Standard World Model	12.8575	21.9759	0.0574	31.185	37.9%
	LoRA TTA	9.3496	17.5336	0.0584	30.182	44.8%
	DINOv2 Context	8.1546	14.8014	0.0492	31.648	51.8%
	TTT-KQV	12.9233	22.5528	0.0585	30.815	42.1%
	T2-z (Ours)	**4.7971**	**7.6682**	**0.0430**	**31.760**	**74.2%**

varies colliding object mass and predicts the resulting collision motion. **Light Switch** assigns different human-defined hidden interaction rules that determine whether a robot action turns the light on or off; **Mass Balance** predicts the tilt of a dumbbell under different mass distribution; **Mass × Friction** jointly varies object mass and surface friction at the same time; **Diverse Visual Backgrounds** evaluates the same friction variations across 30 diverse scene setups, testing whether the learned environment representation remains consistent across visual changes.. In the real world, **Door Close** varies the spring-lock stiffness, which changes how the door responds to closing actions; **Ball Friction** requires predicting the motion of a pushed ball of unknown mass; **Stick Balance** requires predicting the motion of a stick with objects of different masses attached to its ends; and **Soft Pull** requires predicting the deformation a soft object under different rotating actions.

Implementation Details. We adapt BWM Team (2026), an action-conditioned world model built on Wan2.2-TI2V-5B. The environment latent has a dimension of 32. Training starts with 300 world-model updates, followed by 200 latent updates and alternating 200-200 updates. Stage I/II use LR 10^{-5} / 0.03 respectively. At test time, we freeze the world model and use a coarse-to-fine LR schedule of $3.0/1.5/0.5/0.15$, 10 steps each, for 40 steps. T2-z uses only one test-time trajectory on randomly-sampled action for Friction, Gravity, Mass Collision, Mass × Friction, Ball Friction, and Soft Pull, and two for Light Switch, Mass Balance, Door Close, and Stick Balance.

Evaluation Metrics. (1) **Physical Error** measures task-relevant state prediction: object-location error in pixels for Friction, Gravity, Mass Collision, and Diverse Backgrounds; lamp activation error for Light Switch; and bar-tilt angle error in degrees for Mass Balance. For real-world tasks, we measure tracked-state error in pixels. We report **Physical Error (Mean)**, averaged over predicted frames, and **Physical Error (Final)**, measured at the final predicted frame. (2) **Video Generation Quality** is measured by LPIPS and PSNR between predicted and ground-truth videos. LPIPS measures perceptual discrepancy in a learned feature space, capturing differences in visual structure and

appearance, and serves as our primary measure of generation quality. In contrast, PSNR primarily reflects pixel-level reconstruction fidelity and is less informative about semantically meaningful differences or whether object motion is physically correct. (3) **Action Success** treats the world model as a simulator: given candidate actions, we predict their outcomes and select the action most likely to achieve the task goal, such as pushing a ball to a target, balancing a stick, or closing a door. We execute the chose action in real world and observe if meets the success criterions in Supp section C.

Baselines. We compare T2-Z with four baselines using the same world-model backbone. **Standard World Model** uses no environment latent or test-time adaptation; **LoRA TTA** adapts attention LoRA parameters from test-time interactions; **DINOv2 Context** amortizes a 32-D environment context from interaction trajectories; and **TTT-KQV** adapts fast weights within the world model. All adaptive baselines receive the same test-time interaction trajectories as T2-Z.

Please refer to the Supplementary for more details about architecture (section A), task definitions (section C), baseline implementation (section D), and compute efficiency comparison (section E.4).

4.2 Simulation Experiments

Single-Factor Environmental Variation. We first evaluate T2-Z under individual unobserved environmental factors, spanning continuous physical properties, mass-dependent interactions, and discrete human-defined rules. As shown in table 1, T2-Z achieves the highest action success on all five tasks, increasing the five-task average from 36.9% for the Standard World Model to 76.6%, while reducing average LPIPS from 0.0469 to 0.0325. It also substantially improves task-relevant prediction, reducing mean error from 11.06 to 3.85 pixels on Friction, 21.93 to 6.45 pixels on Mass Collision, and 1.18° to 0.64° on Mass Balance. These gains arise because T2-Z directly optimizes an environment latent from observed interactions, capturing environment-specific variation more flexibly than non-adaptive or amortized baselines.

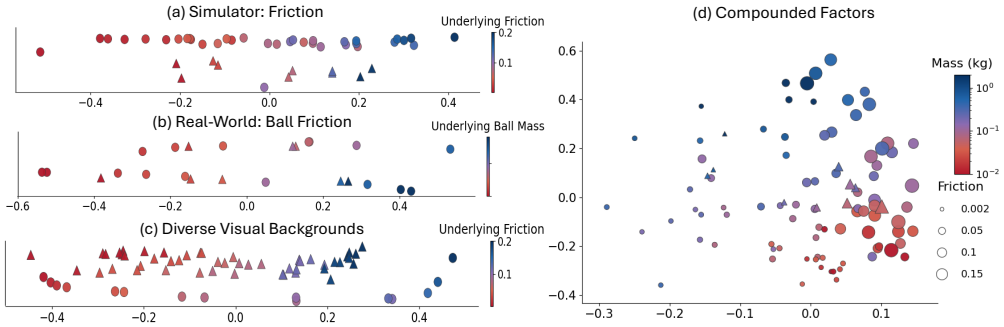


Figure 4: **PCA of learned environment latents.** We visualize learned environment latents using PCA onto x-y axes, with ◯ denoting training environments and △ denoting test environments. Ground-truth factors (denoted by color or size) are shown only for visualization and are never provided to the model. **(a–b)** show that structured latent representations emerge in both simulated and real-world single-factor settings. **(c)** shows that a unified latent structure emerges across diverse visual backgrounds, indicating that the representation does not become entangled with scene appearance. **(d)** shows that a two-dimensional latent structure emerges when mass and friction vary jointly, with x-axis primarily corresponding to friction and y-axis to mass, despite the model never knowing how many underlying factors are present or what they represent.

Emergent Structure of Environment Latents. We examine whether direct optimization gives rise to structured environment representations without supervision of the identity or value of underlying factors. In fig. 4, we apply principal component analysis (PCA) to the learned environment latents and reduce them to two dimensions for visualization. We annotate the projected points using their corresponding ground-truth factor values to reveal their relationship with the learned representation; these values are never provided to the model. Figure 4(a) shows the single-factor friction experiment. The learned latents form a one-dimensional structure for a single factor that corresponds closely to the underlying friction values, as reflected by the point colors: similar values are located nearby, while friction varies consistently x-axis. PCA visualizations for other tasks are provided in the Supplementary Material section E.3.

Compounded Environmental Factors. We next evaluate T2-Z under the more challenging setting of compounded environmental factors, where object mass and surface friction vary simultaneously and jointly reflect in object motion. Importantly, T2-Z is never informed that there are two underlying factors, nor given their identities or values. table 1 shows T2-Z achieves the lowest physical errors, while improving action success from 35.0% to 69.7%. More strikingly, fig. 4(d) shows

Table 2: **Real-world robotic task performance.** T2-Z outperforms baselines across real-world tasks.

Task	Method	Physical Error (Mean) ↓	Physical Error (Final) ↓	LPIPS ↓	PSNR ↑	Action Success ↑
Door Close	Standard World Model	21.82	41.22	0.05700	29.96	22.22%
	LoRA TTA	17.35	32.14	0.04946	30.24	22.22%
	DINOv2 Context	21.12	38.90	0.05490	30.28	22.22%
	TTT-KQV	19.87	35.93	0.05164	30.95	22.22%
	T2-Z (Ours)	**7.95**	**13.47**	**0.03208**	**32.15**	**55.56%**
Ball Friction	Standard World Model	28.36	64.42	0.03153	36.04	0.00%
	LoRA TTA	37.48	72.09	0.03805	35.44	50.00%
	DINOv2 Context	24.90	52.54	0.02934	**36.32**	12.50%
	TTT-KQV	29.80	67.37	0.03224	36.29	25.00%
	T2-Z (Ours)	**14.47**	**24.57**	**0.02401**	36.23	**62.50%**
Stick Balance	Standard World Model	5.29	11.40	0.04716	31.87	20.00%
	LoRA TTA	3.63	7.24	0.04111	32.40	40.00%
	DINOv2 Context	2.80	7.54	**0.03640**	**33.58**	20.00%
	TTT-KQV	4.49	10.40	0.03948	33.38	25.00%
	T2-Z (Ours)	**2.27**	**5.01**	0.04175	32.05	**83.33%**
Soft Pull	Standard World Model	21.71	35.90	0.07207	33.32	63.64%
	LoRA TTA	18.63	26.20	0.06862	33.63	63.64%
	DINOv2 Context	21.19	29.17	0.07087	33.47	72.73%
	TTT-KQV	**18.23**	29.69	**0.06439**	**34.03**	72.73%
	T2-Z (Ours)	18.71	**23.59**	0.07580	32.52	**81.82%**
Average	Standard World Model	19.30	38.24	0.05194	32.80	26.47%
	LoRA TTA	19.27	34.42	0.04931	32.93	43.97%
	DINOv2 Context	17.50	32.04	0.04788	33.41	31.86%
	TTT-KQV	18.10	35.85	0.04694	**33.66**	36.24%
	T2-Z (Ours)	**10.85**	**16.66**	**0.04341**	33.24	**70.80%**

that the learned latents naturally form a two-dimensional structure corresponding to the two underlying factors, showing that T2-Z discovers the structure of compounded environmental factors through prediction alone, without knowing in advance how many factors are present or what they represent.

Robustness across Diverse Visual Scenes. We test whether the learned environment representation captures underlying dynamics without becoming entangled with visual appearance. Specifically, we ask whether a unified latent structure emerges across visually diverse scenes despite substantial differences in appearance. We construct a multi-background setting in which the same underlying friction factor varies across diverse scene layouts and visual backgrounds. table 1 shows that T2-Z outperforms all baselines across every metric. fig. 4(c) further shows that latents from different backgrounds organize consistently rather than separating according to scene appearance. This indicates that environment-latent can isolate environment-specific information that is shared across visually different observations, enabling a unified latent structure to emerge across diverse scenes.

4.3 Real-World Robotic Experiments

We further evaluate T2-Z on four real-world tasks. As shown in table 2, compared with the Standard World Model, T2-Z reduces average mean/final physical error from $19.30/38.24$ to $10.85/16.66$ pixels and LPIPS from 0.05194 to 0.04341, while increasing action success from 26.47% to 70.80%. On Stick Balance, for example, action success increases from 20.00% to 83.33%. fig. 4(b) reveals a clear one-dimensional latent structure without ground-truth friction supervision or post-hoc annotation. T2-Z is particularly important in the real world, where underlying environmental factors are typically unknown and unmeasured and interactions contain additional uncontrolled variation. Notably, T2-Z achieves even larger improvements over the Standard World Model than in simulation, indicating greater benefits as environmental variation becomes less controlled.

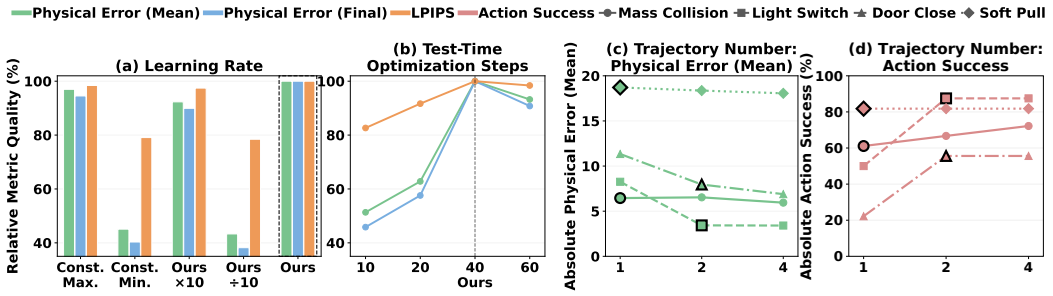


Figure 5: **Ablation studies. (a)** Test-Time Optimization Learning-rate strategy. *Constant Maximum/Minimum* remove coarse-to-fine scheduling, while *Ours* $\times 10/\div 10$ scale our schedule uniformly. **(b)** Ablating the test-time optimization budget shows that performance peaks at 40 steps. **(c,d)** Effect of the number of test-time interaction trajectories. Bold markers denote our chosen trajectory count, after which performance largely saturates.

4.4 Ablative & Qualitative Studies

We ablate the key design choices of T2-Z in fig. 5 and table 3, including latent optimization, interaction evidence, latent parameterization, and alternating training. In fig. 5(a), we vary the learning-rate strategy. Our schedule performs best, validating both its coarse-to-fine form and learning-rate scale. Figure 5(b) varies test-time optimization from 10–60 steps; performance improves up to 40 steps and then saturates, motivating our default. Figure 5(c,d) ablate the number of test-time interaction trajectories. Performance largely saturates beyond the selected number of trajectories, indicating that additional interaction provides limited benefit. Recall that T2-Z uses only one test-time trajectory for 7 tasks, and two for 4 tasks, demonstrating that T2-Z can infer useful environment information with as few as one or two interactions. Additional ablations in table 3 study latent parameterization and training. All the ablated models underperform our model, supporting our alternating schedule and latent mechanism. We provide the latent-dimensionality ablation in Supplementary table 12.

In fig. 6, we show qualitative results across simulation and real-world tasks. T2-Z closely matches ground-truth outcomes, indicating that the inferred latent successfully conditions prediction on the current environment. By contrast, the baselines fail to capture these environment-specific differences, producing largely random motions and incorrect final configurations across environments. Please refer to the Supplementary Material for more qualitative examples (section F).

Table 3: **Additional Ablation study.** *Joint Model–Latent Training* optimizes the world model and latent together without separate stages; *Direct Token* directly optimizes the full-dimensional conditioning token; *32-D Random* $[-0.05, 0.05]$ changes the latent initialization range; and *Alternating Latent/Model* change the number of updates before switching between the two optimization stages, compared with our default 200/200.

Method	Physical Error (Mean) ↓	Physical Error (Final) ↓	LPIPS ↓	PSNR ↑	Action Success ↑
Joint Model-Latent Training	7.03	15.66	0.0657	31.203	68%
Direct Token (3072-D)	94.50	80.49	0.7626	15.218	24%
32-D Random $[-0.05, 0.05]$	4.81	9.73	0.0495	32.172	60%
Alternating Latent/Model (100/100)	5.41	11.50	**0.0484**	32.208	56%
Alternating Latent/Model (400/400)	4.31	8.80	0.0507	**32.430**	68%
T2-Z	**3.85**	**7.59**	0.0495	31.976	**72%**

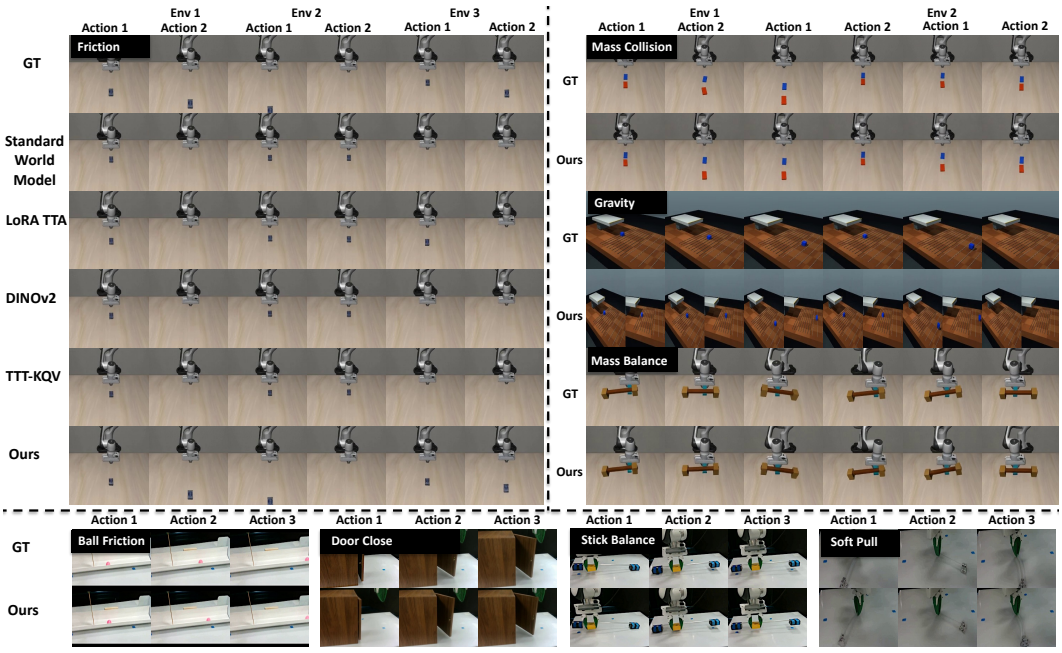


Figure 6: **Qualitative results in simulation and the real world.** Within each task, we apply the same actions across different environments, which produce different ground-truth outcomes due to underlying environmental variation. T2-Z closely matches these environment-dependent outcomes.

Conclusion We introduce T2-Z which augments robotic world models with enviornment latent conditioning. T2-Z alternates between world-model optimization that learns action-conditioned dyanmics, and environment-latent optimization that captures hidden environment factors. For a new environment, the world model is frozen and only the latent is optimized. Across 7 simulation and 4 real-world tasks, T2-Z outperforms baselines a lot in both video prediction and action selection.

AI USE STATEMENT

We used generative AI tools to assist with literature review, language refinement and code inspection. The authors verified the references, scientific claims, experimental results, and AI-assisted artifacts, and take responsibility for the final submission.

REPRODUCIBILITY STATEMENT

The paper and supplementary material describe the model architecture, training settings, datasets, task definitions, and evaluation protocols. Anonymized code and configuration files are provided as supplementary materials to specify further implementation details.

REFERENCES

BWM Team. BWM: A low-cost high-fidelity world simulator for robot learning. *arXiv preprint arXiv:2607.29302*, 2026. 6, 15

Shiyi Chen, Zeyu Wan, Shiyang Yan, Chun Zhang, Weiyi Zhang, Qiang Li, Debing Zhang, and Fasih Ud Din Farrukh. Slr: Learning quadruped locomotion without privileged information. In *Conference on Robot Learning*, pp. 3212–3224, 2024. 3

Karan Dalal, Daniel Koceja, Jiarui Xu, Yue Zhao, Shihao Han, Ka Chun Cheung, Jan Kautz, Yejin Choi, Yu Sun, and Xiaolong Wang. One-minute video generation with test-time training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 17702–17711, 2025. 21

Yusen Feng, Bingchen Han, Jiangran Lyu, Kai Liu, Yixin Zheng, Yuxuan Wan, Weiheng Liu, Sun Han, Ruiqin Li, Yulong Zhang, Fangfu Liu, Xuesong Shi, Libin Liu, Yizhou Wang, Zhizheng Zhang, and He Wang. WAM-TTT: Steering world-action models by watching human play at test time. *arXiv preprint arXiv:2607.06988*, 2026. 3

Shenyuan Gao, William Liang, Kaiyuan Zheng, Ayaan Malik, Seonghyeon Ye, Sihyun Yu, Wei-Cheng Tseng, Yuzhu Dong, Kaichun Mo, Chen-Hsuan Lin, Qianli Ma, Seungjun Nah, Loic Magne, Jiannan Xiang, Yuqi Xie, Ruijie Zheng, Dantong Niu, You Liang Tan, K. R. Zentner, George Kurian, Suneel Indupuru, Pooya Jannaty, Jinwei Gu, Jun Zhang, Jitendra Malik, Pieter Abbeel, Ming-Yu Liu, Yuke Zhu, Joel Jang, and Linxi Jim Fan. Dreamdojo: A generalist robot world model from large-scale human videos. *arXiv preprint arXiv:2602.06949*, 2026. 3

Haoran He, Peilin Wu, Chenjia Bai, Hang Lai, Lingxiao Wang, Ling Pan, Xiaolin Hu, and Weinan Zhang. Bridging the sim-to-real gap from the information bottleneck perspective. In *Conference on Robot Learning*, pp. 473–495, 2024. 3

Yining Hong, Beide Liu, Maxine Wu, Yuanhao Zhai, Kai-Wei Chang, Linjie Li, Kevin Lin, Chung-Ching Lin, Jianfeng Wang, Zhengyuan Yang, Yingnian Wu, and Lijuan Wang. Slowfast-vgen: Slow-fast learning for action-driven long video generation, 2024. URL https://arxiv.org/abs/2410.23277. 4

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022. 20

Wenlong Huang, Yu-Wei Chao, Arsalan Mousavian, Ming-Yu Liu, Dieter Fox, Kaichun Mo, and Li Fei-Fei. Pointworld: Scaling 3d world models for in-the-wild robotic manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20765–20779, 2026. 2

Jinwoo Jang, Minjong Yoo, Sihyung Yoon, and Honguk Woo. Test-time mixture of world models for embodied agents in dynamic environments. In *International Conference on Learning Representations*, 2026. 3

Yunfan Jiang, Yevgen Chebotar, Ruijie Zheng, Fengyuan Hu, Yunhao Ge, Jimmy Wu, Tianyuan Dai, Scott Reed, Li Fei-Fei, Yuke Zhu, and Linxi Jim Fan. RoboTTT: Context scaling for robot policies. *arXiv preprint arXiv:2607.15275*, 2026. 3

Ruofan Jin and Zaixi Zhang. Agentic-vla: Efficient online adaptation for vision-language-action models. *arXiv preprint arXiv:2605.22896*, 2026. 3

Po-Chen Ko, Jiayuan Mao, Yu-Hsiang Fu, Hsien-Jeng Yeh, Chu-Rong Chen, Wei-Chiu Ma, Yilun Du, and Shao-Hua Sun. Implicit state estimation via video replanning. *arXiv preprint arXiv:2510.17315*, 2025. 3

Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots. In *Robotics: Science and Systems*, 2021. doi: 10.15607/RSS.2021.XVII.011. 2, 3

Kimin Lee, Younggyo Seo, Seunghyun Lee, Honglak Lee, and Jinwoo Shin. Context-aware dynamics model for generalization in model-based reinforcement learning. In *International Conference on Machine Learning*, pp. 5757–5766, 2020. 3

Zezeng Li, Rui Yang, Ruochen Chen, ZhongXuan Luo, and Liming Chen. ADPro: a test-time adaptive diffusion policy for robot manipulation via manifold and initial noise constraints. *arXiv preprint arXiv:2508.06266*, 2025. 3

Zezeng Li, Enda Xiang, Thuy Tran, Di Huang, Momath Thiam, and Liming Chen. PriGo: Test-time primitive guidance to diffusion and flow policies for adaptive robotic manipulation. *arXiv preprint arXiv:2607.07076*, 2026. 3

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2023. 17

Changyu Liu, Yiyang Liu, Taowen Wang, Qiao Zhuang, James Chenhao Liang, Wenhao Yang, Renjing Xu, Qifan Wang, Dongfang Liu, and Cheng Han. On-the-fly vla adaptation via test-time reinforcement learning. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, 2026. 3

Shixin Luo, Songbo Li, Ruiqi Yu, Zhicheng Wang, Jun Wu, and Qiuguo Zhu. Pie: Parkour with implicit-explicit learning framework for legged robots. *IEEE Robotics and Automation Letters*, 9 (11):9986–9993, 2024. doi: 10.1109/LRA.2024.3459797. 3

Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Science Robotics*, 7(62):eabk2822, 2022. doi: 10.1126/scirobotics.abk2822. 3

Anusha Nagabandi, Ignasi Clavera, Simin Liu, Ronald S. Fearing, Pieter Abbeel, Sergey Levine, and Chelsea Finn. Learning to adapt in dynamic, real-world environments through meta-reinforcement learning. In *International Conference on Learning Representations*, 2019. 3

I Made Aswin Nahrendra, Byeongho Yu, and Hyun Myung. Dreamwaq: Learning robust quadrupedal locomotion with implicit terrain imagination via deep reinforcement learning. In *IEEE International Conference on Robotics and Automation*, pp. 5078–5084, 2023. doi: 10.1109/ICRA48891.2023.10161144. 3

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. 21

Jeongeun Park, Juhan Park, Taekyung Kim, Sungjoon Choi, Dongyoon Han, and Sangdoo Yun. Retrieve, don't retrain: Extending vision language action models to new tasks at test time. *arXiv preprint arXiv:2606.15631*, 2026. 3

Kate Rakelly, Aurick Zhou, Chelsea Finn, Sergey Levine, and Deirdre Quillen. Efficient off-policy meta-reinforcement learning via probabilistic context variables. In *International Conference on Machine Learning*, pp. 5331–5340, 2019. 3

Rishabh Tiwari, Kusha Sareen, Lakshya A. Agrawal, Joseph E. Gonzalez, Matei Zaharia, Kurt Keutzer, Inderjit S. Dhillon, Rishabh Agarwal, and Devvrit Khatri. Learning, fast and slow: Towards llms that adapt continually. *arXiv preprint arXiv:2605.12484*, 2026. 4

Chi Wan, Kangrui Wang, Yuan Si, Pingyue Zhang, and Manling Li. Worldagen: Unified state-action prediction with test-time world model training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18584–18592, 2026. doi: 10.1609/aaai.v40i22.38925. 3

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, Jianyuan Zeng, Jiayu Wang, Jingfeng Zhang, Jingren Zhou, Jinkai Wang, Jixuan Chen, Kai Zhu, Kang Zhao, Keyu Yan, Lianghua Huang, Mengyang Feng, Ningyi Zhang, Pandeng Li, Pingyu Wu, Ruihang Chu, Ruili Feng, Shiwei Zhang, Siyang Sun, Tao Fang, Tianxing Wang, Tianyi Gui, Tingyu Weng, Tong Shen, Wei Lin, Wei Wang, Wei Wang, Wenmeng Zhou, Wente Wang, Wenting Shen, Wenyuan Yu, Xianzhong Shi, Xiaoming Huang, Xin Xu, Yan Kou, Yangyu Lv, Yifei Li, Yijing Liu, Yiming Wang, Yingya Zhang, Yitong Huang, Yong Li, You Wu, Yu Liu, Yulin Pan, Yun Zheng, Yuntao Hong, Yupeng Shi, Yutong Feng, Zeyinzi Jiang, Zhen Han, Zhi-Fan Wu, and Ziyu Liu. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025. 15

Junjie Wang, Yao Mu, Dong Li, Qichao Zhang, Dongbin Zhao, Yuzheng Zhuang, Ping Luo, Bin Wang, and Jianye Hao. Prototypical context-aware dynamics generalization for high-dimensional model-based reinforcement learning. In *ICLR Workshop on Scene Representations for Autonomous Driving*, 2023. 3

Ying Wang, Oumayma Bounou, Yann LeCun, and Mengye Ren. Adajepa: An adaptive latent world model. *arXiv preprint arXiv:2606.32026*, 2026. 3

Jiaxu Xing, Zhiyuan Zhu, Yunfan Ren, Ismail Geles, Yifan Zhai, Rudolf Reiter, and Davide Scaramuzza. Continual robot policy learning via variational neural dynamics. *arXiv preprint arXiv:2606.27353*, 2026. 3

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, Ayaan Malik, Kyungmin Lee, William Liang, Nadun Ranawaka, Jiasheng Gu, Yinzhen Xu, Guanzhi Wang, Fengyuan Hu, Avnish Narayan, Johan Bjorck, Jing Wang, Gwanghyun Kim, Dantong Niu, Ruijie Zheng, Yuqi Xie, Jimmy Wu, Qi Wang, Ryan Julian, Danfei Xu, Yilun Du, Yevgen Chebotar, Scott Reed, Jan Kautz, Yuke Zhu, Linxi Jim Fan, and Joel Jang. World action models are zero-shot policies, 2026. 2

Minjong Yoo, Jinwoo Jang, Sihyung Yoon, and Honguk Woo. World model implanting for test-time adaptation of embodied agents. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 72556–72573, 2025. 3

Wenhao Yu, Jie Tan, C. Karen Liu, and Greg Turk. Preparing for the unknown: Learning a universal policy with online system identification. In *Robotics: Science and Systems*, 2017. 3

Alan Yuille and Daniel Kersten. Vision as bayesian inference: Analysis by synthesis? *Trends in Cognitive Sciences*, 10(7):301–308, 2006. 2

Wenbo Zhang, Jianxiong Li, Shuai Yang, Sijin Chen, Jiajun Liu, Lingqiao Liu, and Xiao Ma. TTT-VLA: Test-time latent prompt optimization for vision-language-action models. *arXiv preprint arXiv:2606.03127*, 2026. 3

Jianchao Zhao, Huoren Yang, Yusong Hu, Yuyang Gao, Qiguan Ou, Cong Wan, SongLin Dong, Zhiheng Ma, and Yihong Gong. Retrieve-then-steer: Online success memory for test-time adaptation of generative vlas. *arXiv preprint arXiv:2605.10094*, 2026. 3

Haoyu Zhen, Qiao Sun, Hongxin Zhang, Junyan Li, Siyuan Zhou, Yilun Du, and Chuang Gan. Learning 4d embodied world models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5337–5347, 2025. 2

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 79115–79135, 2025. 2

Siyuan Zhou, Yilun Du, Jiaben Chen, Yandong Li, Dit-Yan Yeung, and Chuang Gan. Robodreamer: Learning compositional world models for robot imagination. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 61885–61896, 2024. 2

Chuning Zhu, Raymond Yu, Siyuan Feng, Benjamin Burchfiel, Paarth Shah, and Abhishek Gupta. Unified world models: Coupling video and action diffusion for pretraining on large robotic datasets. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.015. 2

Fangqi Zhu, Hongtao Wu, Song Guo, Yuxiao Liu, Chilam Cheang, and Tao Kong. IRASim: Learning interactive real-robot action simulators. *arXiv preprint arXiv:2406.14540*, 2024. 2

Shaoting Zhu, Baijun Ye, Jiaxuan Wang, Jiakang Chen, Ziwen Zhuang, Linzhan Mou, Runhan Huang, and Hang Zhao. TTT-Parkour: Rapid test-time training for perceptive robot parkour. *arXiv preprint arXiv:2602.02331*, 2026. 3

Supplementary Material for

T2-Z: Test-Time Optimization of Environment Latents for Robotic World Models

CONTENTS

A Model Architecture and Optimization **15**

A.1 Architecture and latent conditioning 15

A.2 Training and test-time adaptation 15

B Experimental Setup **16**

B.1 Training data and resource budgets 16

B.2 Simulator and real-world setup 17

C Evaluation Protocols **17**

C.1 Shared protocol and metrics 17

C.2 Simulated tasks 17

C.3 Real-world tests 19

D Baseline Implementation Details **20**

D.1 Standard World Model 20

D.2 LoRA TTA 20

D.3 DINOv2 Context 21

D.4 TTT-KQV 21

E Additional Experiments **22**

E.1 Test-time interaction-count ablation 22

E.2 Latent dimensionality 22

E.3 Latent-space PCA across tasks 22

E.4 Computational efficiency 22

F Additional Qualitative Results **25**

References **37**

A Model Architecture and Optimization

A.1 Architecture and latent conditioning

Inputs and inherited backbone. For an episode in environment e, the model receives one initial RGB frame I_0 and an aligned sequence $a = (a_t)_t$ of 14-D robot actions, and predicts the subsequent video frames. We initialize the video diffusion transformer (DiT) and action encoder from BWM (BWM Team, 2026), retaining its Wan2.2(Wan et al., 2025)-based action-conditioning pathway. The frozen video VAE encodes the initial frame into the video-latent stream and decodes predicted latents back to RGB. Text conditioning is disabled; the initial frame is not an extra cross-attention token. In short, I_0 enters the video-latent stream, a follows BWM's two action-conditioning paths, and the environment latent augments both paths.

Environment-indexed latent conditioning. We associate each environment e with a freely optimized vector $z_e \in \mathbb{R}^{32}$. Episodes assigned to an environment share its latent; their initial frames and actions vary, so predicting their different outcomes constrains the environment representation. There is no observation-to-latent encoder and no supervision from the numerical physical parameter. Let $D = 3072$ be the DiT width. Two separate projectors turn z_e into model-width vectors:

$$\begin{aligned} q_b(z_e) &= W_{b,2}\,\mathrm{GELU}(W_{b,1}\,\mathrm{LN}(z_e)), \quad b \in \{\mathrm{token}, \mathrm{mod}\}, \\ W_{b,1} &: 32 \to 128, \qquad W_{b,2} : 128 \to D. \end{aligned} \tag{9}$$

The two linear layers in each projector have no bias; the projectors do not share weights. The 128-D width is internal to these MLPs: both outputs are D-dimensional.

Cross-attention branch. BWM's action encoder maps each action to a D-dimensional conditioning token. Let $A_{\mathrm{tok}}(a)$ denote this token sequence and $u_e = q_{\mathrm{token}}(z_e)$ the single environment token. With text disabled, each DiT block receives the conditioning sequence $C_e = [u_e; A_{\mathrm{tok}}(a)]$. For video patch tokens $X_\ell \in \mathbb{R}^{N\times D}$ entering block ℓ, the attention operation can be written schematically as

$$\mathrm{CA}(X_\ell, C_e) = \mathrm{softmax}\left(\frac{(X_\ell W_Q)(C_e W_K)^\top}{\sqrt{d_h}}\right)(C_e W_V)W_O. \tag{10}$$

Here N is the number of video patches, d_h is one attention head's width, and W_Q, W_K, W_V, W_O are the block's learned attention matrices. The actual operation is multi-head; the equation shows where the new token enters. Every video patch can attend to the environment token as well as the action tokens through keys and values.

Modulation branch. BWM also groups four aligned 14-D actions at a time and maps each group to a D-dimensional action-modulation vector a_g^{mod}. Index g identifies the corresponding latent-video temporal group, and s identifies a spatial patch within it. Let τ be the current flow-matching timestep and $b_{g,s}(\tau) \in \mathbb{R}^D$ be Wan's *base timestep embedding* at that video patch, before action or environment conditioning. The initial-frame latent uses the unnoised timestep, while predicted-frame latents use τ. The environment-modulation vector is $v_e = q_{\mathrm{mod}}(z_e) \in \mathbb{R}^D$. The implementation adds these three vectors and applies the existing time projection:

$$\tilde{b}_{g,s}(\tau) = b_{g,s}(\tau) + a_g^{\mathrm{mod}} + v_e, \qquad M_{g,s} = \mathrm{TimeProj}(\tilde{b}_{g,s}(\tau)) \in \mathbb{R}^{6\times D}. \tag{11}$$

The action and environment vectors are broadcast to every spatial patch in group g. The six D-dimensional outputs of $M_{g,s}$ supply the DiT blocks' adaptive-normalization shifts, scales, and residual gates; the modified timestep embedding also conditions the output head. Thus z_e influences both attention and per-patch modulation, while leaving BWM's original action pathways in place.

A.2 Training and test-time adaptation

Training schedule. Training uses the inherited flow-matching video-prediction loss, excluding the initial conditioning frame. A latent block freezes the shared model and updates the environment table; a model block freezes the table and updates the DiT, action encoder, and two projectors. The basic cycle alternates a 200-update latent block and a 200-update model block. An initial 300-update model block precedes the curriculum. The shared-model optimizer is AdamW with

Table 4: Model components and optimization roles. Test-time adaptation updates only the environment latent.

Component	Initialization or structure	Training update
Video DiT	BWM-initialized Wan2.2 backbone	Model blocks
Action encoder	BWM-initialized 14-D action pathway	Model blocks
Video VAE	Wan2.2 video encoder/decoder	Frozen
Environment table	Learnable 32-D latents indexed by environment	Latent blocks
Latent projectors	Two $32 \to 128 \to 3072$ MLPs	Model blocks

learning rate 10^{-5}, weight decay 0.01, 100 warmup updates, gradient clipping 0.5, and bfloat16 training. The environment-latent optimizer uses learning rate 0.03 and zero weight decay. Latents are independently initialized from $\mathcal{U}(0, 1)$ and clipped to $[0, 1]$ after training updates.

Test-time adaptation. For a held-out environment, a 32-D latent is optimized from its test-time interaction trajectories; all shared weights stay frozen. Query outcomes never enter this update. Both simulation and real-world main evaluations use 40 latent updates with learning rates $3.0, 1.5, 0.5, 0.15$, each for ten updates. The resulting latent is reused across query actions from that environment. Exact task-specific execution configurations are released with the code.

B Experimental Setup

B.1 Training data and resource budgets

Tables 5 and 6 report counts of physical environments and recorded episodes, rather than latent entries. Each video chunk contains one initial conditioning frame and the remaining frames are prediction targets. The batch notation is *training environments* $\times$ *action chunks per environment* per optimizer update.

Table 5: Simulation training scale by physical environment. Diverse Backgrounds uses 30 scene backgrounds. Budget is H200 GPUs $\times$ hours per method.

Task	Physical envs. $\times$ episodes/env.	Frames	Stride	Batch	Budget
Friction	35×10	41	1	8×4	2×24
Gravity	20×10	61	1	8×4	2×24
Mass Collision	20×9	61	1	8×8	2×24
Light Switch	12×10	33	1	8×8	2×24
Mass Balance	20×15	41	2	8×8	2×24
Mass $\times$ Friction	84×9	61	1	16×6	4×48
Diverse Backgrounds	15×50	41	1	12×6	4×24

Light Switch episodes contain continuous button-press trajectories. Diverse Backgrounds varies friction across physical environments while the different scene backgrounds provide repeated interactions.

Table 6: Real-world training scale by physical environment and native/model-input RGB resolution. Budget is H200 GPUs $\times$ hours per method.

Task	Physical envs. / episodes	Frames	Stride	Native RGB	Input RGB	Batch	Budget
Ball Friction	8 / 240	61	1	560×248	320×160	6×8	2×24
Door Close	10 / 438	49	1	640×480	256×192	6×8	2×24
Stick Balance	9 / 421	33	3	640×480	256×192	6×8	2×24
Soft Pull	9 / 146	33	3	512×256	320×160	10×6	2×24

B.2 Simulator and real-world setup

Simulator. We construct the custom tasks in a LIBERO-based simulator (Liu et al., 2023) with a Franka Panda arm. Simulated RGB videos are recorded at 20 fps and 224×224 resolution. The action representation has 14 channels.

Real world. The physical tasks use a Franka robot arm and a fixed Intel RealSense D435i camera. We use only its main RGB view, captured at 640×480 and 20 fps; the task-specific saved video crops and model-input sizes appear in Table 6. Model inputs are letterboxed to the listed sizes. All tasks use 14-channel action vectors; the target-action representation is task dependent.

C Evaluation Protocols

C.1 Shared protocol and metrics

Shared comparison. For action-selection tasks, we sample candidate actions and use each model's predicted videos to make an action decision. The selected action is then executed in the environment, and success is determined from its actual outcome using the task-specific criteria below. Each evaluated candidate-action set contains at least one action that satisfies the target condition. All methods receive the same query inputs and candidate actions, and all methods that adapt at test time receive the same test-time interaction trajectories. The Standard World Model does not adapt at test time, and no method sees a query outcome before making its prediction.

For tracked object positions p_t and predictions $\hat{p}_t$ over T evaluated future frames, average displacement error (ADE) and final displacement error (FDE) are

$$\mathrm{ADE} = \frac{1}{T}\sum_{t=1}^{T} \|\hat{p}_t - p_t\|_2, \qquad \mathrm{FDE} = \|\hat{p}_T - p_T\|_2. \tag{12}$$

Both are reported in image pixels. Tasks with scalar physical states use the corresponding mean/final absolute errors stated below. PSNR and SSIM measure full-frame appearance. LPIPS uses the official pretrained AlexNet perceptual metric on RGB frames scaled to $[-1, 1]$, without an object mask; per-frame values are averaged over the evaluated future frames and then over query videos.

A query video denotes one action-conditioned video chunk evaluated at test time. The counts below sum query videos across the evaluated environments; K is the number of test-time interaction trajectories per test environment.

Table 7: Simulation evaluation counts: test-time interaction trajectories and query videos.

Task	K	Query videos
Friction	1	90
Gravity	1	90
Mass Collision	1	80
Light Switch	2	60
Mass Balance	2	130
Mass $\times$ Friction	1	80
Diverse Backgrounds	1	90

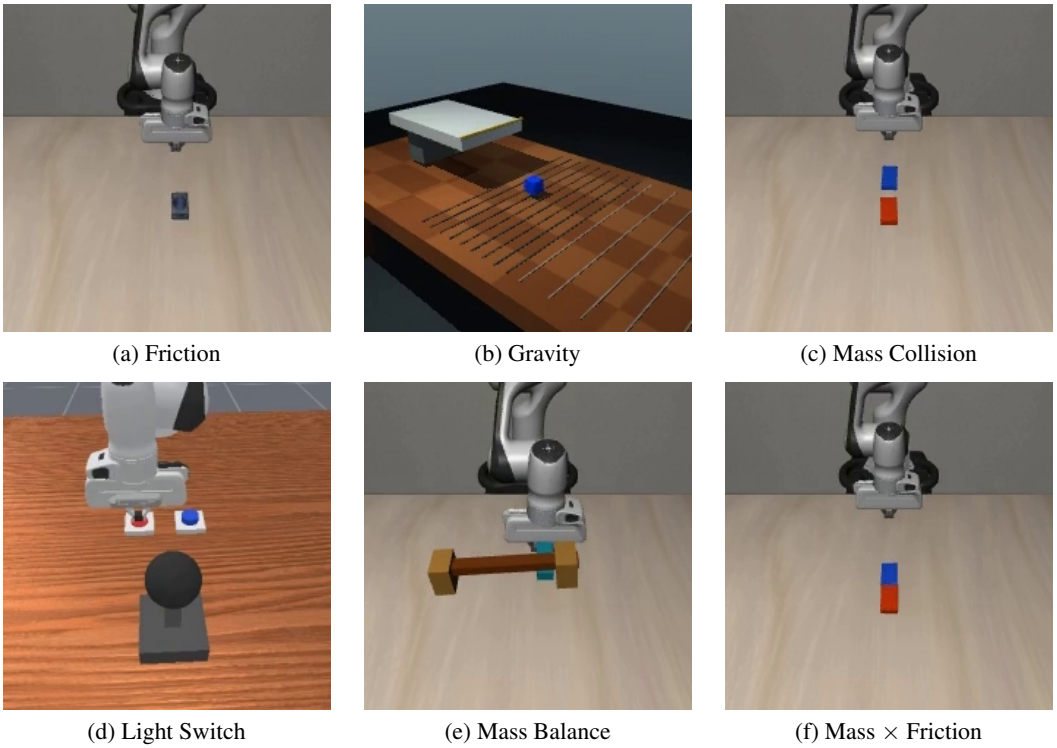
C.2 Simulated tasks

Friction. We change the table's surface-friction coefficient while the robot pushes the same block with different action strengths. We track the pushed block's image centroid. Action Success means that its observed displacement reaches the requested short, medium, or long target region.

Gravity. We vary the simulator's gravitational acceleration, changing the path of a blue block after it is thrown. Different actions correspond to different launch speeds of the block. We track the projectile's centroid. Success means that the observed projectile endpoint reaches the requested near, middle, or far target region.

Table 8: Simulation object-centric quantities extracted from the predicted and ground-truth RGB videos. ADE/FDE are in image pixels.

Task	Tracked quantity	Object-centric error
Friction	Pushed-block centroid	ADE/FDE
Gravity	Blue projectile centroid	ADE/FDE
Mass Collision	Red struck-object centroid	ADE/FDE
Light Switch	Yellow-pixel fraction in lamp region	Mean/final absolute error (pp)
Mass Balance	Balance-bar tilt angle	Mean/final angle error ($^\circ$)
Mass $\times$ Friction	Red struck-object centroid	ADE/FDE
Diverse Backgrounds	Pushed-block centroid	ADE/FDE



(a) Friction (b) Gravity (c) Mass Collision

(d) Light Switch (e) Mass Balance (f) Mass $\times$ Friction

Figure 7: Representative ground-truth frames from the simulated tasks. Diverse Backgrounds uses the Friction push task with varied scene backgrounds.

Mass Collision. We vary the mass of the red target object while a driver object collides with it under different actions. We track the *struck* red object's centroid, not the driver's. Success means that the struck object reaches the requested displacement region.

Light Switch. We vary the hidden causal rule specifying whether the red button, the blue button, both, or neither can change the lamp state. The object-centric signal is *color*, not average brightness: a fixed RGB rule detects yellow pixels in the lamp region $\mathcal{R}$. Each frame's score is

$$y_t = \frac{1}{|\mathcal{R}|} \sum_{p \in \mathcal{R}} \mathbf{1}[\mathrm{RGB}_t(p) \text{ passes the yellow-color rule}]. \tag{13}$$

We report the absolute difference between predicted and observed scores in percentage points (pp). For T evaluated future frames,

$$E^{\text{light}}_{\text{mean}} = \frac{100}{T} \sum_{t=1}^{T} |\hat{y}_t - y_t|, \qquad E^{\text{light}}_{\text{final}} = 100|\hat{y}_T - y_T|. \tag{14}$$

A score of at least 0.35 denotes "on" for the terminal-state goal; success means that the observed terminal lamp state matches the requested on/off target.

Mass Balance. We vary the left/right mass ratio of the balance apparatus; the robot's support position is the action used to probe that fixed ratio. We track the bar angle in degrees; its ADE/FDE analogues are mean/final angular errors. Success means that the observed terminal bar angle lies in the horizontal balance interval.

Mass $\times$ Friction. We combine different red-object masses with different table-friction coefficients, so both physical factors can change across environments. We track the red struck object's centroid. For each target line, the desired action is the least forceful sampled action whose observed endpoint crosses that line; the selector estimates this minimum from the predicted videos.

Diverse Backgrounds (multi-background). We vary the table-friction coefficient as in Friction while rendering the same push task under different scene backgrounds; background is a visual change, not a new action. We again track the pushed block's centroid; success means reaching the requested displacement region. This tests whether the shared environment representation follows dynamics rather than appearance.

C.3 Real-world tests

Table 9: Real-world evaluation counts: test-time interaction trajectories and query videos.

Task	K	Query videos
Ball Friction	1	80
Door Close	2	100
Stick Balance	2	45
Soft Pull	1	24

Table 10: Real-world object-centric measurements from the main camera. ADE/FDE are reported in native-image pixels.

Task	Tracked quantity	Object-centric error
Ball Friction	Visible ball center	ADE/FDE
Door Close	Outer door-edge position	ADE/FDE
Stick Balance	Two end markers, averaged	ADE/FDE
Soft Pull	Dragged object's visible center	ADE/FDE

Ball Friction. We vary the mass of steel beads placed inside the ball, changing its total mass and the sliding-friction force under a robot swing (Figure 8a). The ball variants in the latent-space visualizations correspond to these different steel-bead loads. We track the ball's visible center for native-pixel ADE/FDE. Success means that the selected swing moves the ball to the right-hand blue-marker target.

Door Close. The door has a spring-loaded latch; we replace the spring to change how much closing action is needed to engage it (Figure 8b). The tracked outer door edge, rather than a centroid or hinge angle, defines native-pixel ADE/FDE. Success means that the selected action fully closes and latches the door.

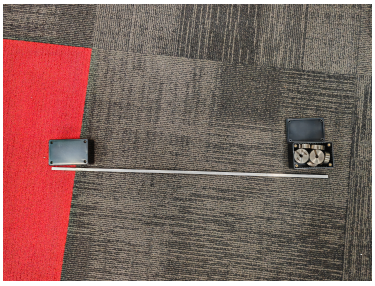
Stick Balance. We change the weights attached to the stick's ends, altering its mass distribution and thus the angle at which it settles after lifting (Figure 8c). We track the two material markers at its ends and average their positional errors for ADE/FDE. We sample a set of candidate actions and use the model to predict which actions will maintain balance. One action is selected uniformly at random from the predicted balancing set and executed on the real robot. A trial is successful if the stick actually maintains balance. Each sampled candidate set contains at least one action that



(a) Steel-bead loads for the ball variants



(b) Door latch and alternative springs



(c) Stick with adjustable end masses



(d) Soft tubes with different stiffness and curvature

Figure 8: Physical designs of the four real-world tasks. The Ball task changes the number of steel beads inside the ball (not visible in the video frame), and the Door task uses a spring-loaded latch.

achieves balance. If the model predicts no balancing action, the trial is counted as a failure. We report the fraction of successful trials.

Soft Pull. The task uses soft tubes with different stiffness coefficients. The robot may grasp from the left or right, and the tubes also have different natural curvatures (Figure 8d). The central prediction question is approximately how far the grasp must rotate before the tube starts dragging the object. The visible object center defines native-pixel ADE/FDE. Given a candidate action, the model predicts the rotation angle $\hat{\theta}$ at which the object starts sliding. We then conduct two real-robot tests, rotating to $\hat{\theta} - 6°$ and $\hat{\theta} + 6°$, respectively. A trial is successful only if the object does not slide in the first test and slides in the second; otherwise, it is counted as a failure. We report the fraction of successful trials.

D Baseline Implementation Details

All four baselines use the task-level training data and per-method GPU budgets in Tables 5–6. They use the same query videos and, for action-selection tasks, the same candidate actions and success criteria as T2-Z. The three adaptive baselines receive exactly the same test-time interaction trajectories as our latent optimizer. The tables give the common task-level batch specification; DINOv2 Context uses a subset of sampled trajectories for conditioning and all sampled trajectories as queries during training, whereas TTT-KQV processes them as ordered within-environment streams.

D.1 Standard World Model

This baseline fine-tunes the BWM-initialized DiT and action encoder on pooled task trajectories without an environment latent or test-time adaptation. It uses AdamW with model learning rate 10^{-5}, weight decay 0.01, 100 warmup updates, gradient clipping 0.5, and bfloat16.

D.2 LoRA TTA

Starting from the Standard World Model, rank-8, $\alpha = 8$ LoRA adapters (Hu et al., 2022) are placed on the DiT attention Q/K/V/output projections. The backbone stays frozen during test-time adaptation. Adapters are reset for each environment and optimized on its test-time interaction trajectories, then held fixed across that environment's query actions. The Friction configuration uses 50 AdamW

adaptation updates at learning rate 10^{-4}, zero weight decay, and gradient clipping 1; task-specific adaptation configurations are released with the code.

D.3 DINOv2 Context

A frozen DINOv2-B/14 encoder (Oquab et al., 2024) extracts conditioning-frame features, while a learned 128-D action GRU summarizes the intervening 14-D actions. For each of N task-configured sampled frames, the CLS and mean patch features are projected to 256 dimensions. The $N-1$ adjacent visual–action transition features (640-D each) are concatenated; a $640(N-1) \to 1024 \to 1024$ MLP and final projection produce the 32-D context. Multiple conditioning-trajectory summaries are averaged before that final projection. At each training update, we sample a set of trajectories Q_e from a training environment and select a subset $C_e \subseteq Q_e$ for conditioning. All trajectories in Q_e serve as training queries. The number of conditioning trajectories is sampled from the task-configured choices (typically one or two; one to four for Soft Pull), so the predictor learns to use different conditioning-set sizes. Every query in Q_e is predicted using only the context inferred from C_e; training minimizes the mean query-video flow-matching loss. The context head, BWM DiT, action encoder, and two context projectors are trained jointly, while DINOv2 remains frozen. The model and context-head learning rates are 10^{-5} and 10^{-4}, respectively. At test time, context comes from a forward pass over the observed test-time interaction trajectories, with no gradient update.

D.4 TTT-KQV

We adapt a video test-time-training fast-weight module (Dalal et al., 2025) to the BWM backbone. Gated two-layer GELU MLP modules with expansion factor four are inserted after eight approximately evenly spaced attention blocks. Training randomly shuffles the chunks from each sampled environment in a batch. Within each training stream, all chunks use the same sampled flow-matching timestep (but independent noise), matching the same-timestep processing of observed interactions and query videos at inference. At position i, the preceding chunks form the observed prefix and the current chunk is the query. Fast weights reset to a learned shared initialization at the start of each stream. For each chunk, the module causally writes its key–value tokens before reading its output. Let x_t denote an input token, h an attention head, and d_h its feature dimension; t indexes tokens, not flow-matching timesteps. Learned key and value projections produce $k_{h,t}, v_{h,t} \in \mathbb{R}^{d_h}$. The fast state $W = \{W_h\}_h$ contains each head's MLP weights and biases. Its per-token binding loss and learned step scaling are

$$\begin{aligned} \ell_{h,t}(W) &= \frac{1}{2d_h} \left\| k_{h,t} + \mathrm{LN}(\mathrm{MLP}_{W_h}(k_{h,t})) - v_{h,t} \right\|_2^2, \\ \eta_{h,t} &= \frac{\eta_{\text{base}}}{d_h}\, \sigma(w_h^\top x_t + b_h), \qquad \eta_{\text{base}} = 0.1. \end{aligned} \tag{15}$$

Here LN normalizes within each head, and σ is the sigmoid function. The factor $1/d_h$ in the loss averages squared errors over feature dimensions; the separate factor $1/d_h$ in $\eta_{h,t}$ scales the update. For update j, let $\mathcal{B}_j$ be the current minibatch of at most 64 tokens. The differentiable fast-state update is

$$W^{j+1} = W^j - \nabla_W \sum_h \sum_{t \in \mathcal{B}_j} \eta_{h,t} \ell_{h,t}(W) \Big|_{W=W^j}. \tag{16}$$

We use a base inner learning rate of 0.1 with learned token-wise, head-specific scaling, rather than a constant update step size. The scaling parameters w_h, b_h are shared slow weights; the inner update changes only W. Gates start at 0.1. The outer loss averages the video flow-matching losses over all stream positions and sampled environments. Its gradients update the shared slow weights, using backbone LR 10^{-5} and TTT-module LR 10^{-4}; the fast state itself is ephemeral. At inference, every denoising timestep resets the fast state and replays the observed test-time interaction trajectories at that same timestep. Each query then takes an independent update–then–read branch on its own noisy tokens. Query ground truth never enters the update, and fast state is not carried between queries or denoising timesteps. Exact stream packing and task-specific execution settings are released with the code.

E Additional Experiments

E.1 Test-time interaction-count ablation

Table 11: Numerical results for Figure 5(c,d) at $K = 1, 2, 4$ test-time interaction trajectories. Bold marks the selected K. Physical error is in pixels except for Light Switch (lamp-score error in percentage points).

	Physical error (mean) ↓			Action success (%) ↑		
Task	$K = 1$	$K = 2$	$K = 4$	$K = 1$	$K = 2$	$K = 4$
Mass Collision	**6.452**	6.534	5.946	**61.1**	66.7	72.2
Light Switch	8.26	**3.43**	3.41	50.0	**87.5**	87.5
Door Close	11.337	**7.954**	6.889	22.2	**55.6**	55.6
Soft Pull	**18.710**	18.363	18.060	**81.8**	81.8	81.8

E.2 Latent dimensionality

Table 12: Environment-latent dimensionality on Friction under the same $K = 1$ evaluation protocol (ten held-out physical environments; 90 query videos). Object ADE/FDE are main-view centroid errors in pixels; video metrics use the concatenated main and wrist views. Action success is evaluated on 25 reachable target decisions. Best values are bold.

Latent dim.	ADE↓	FDE↓	PSNR↑	SSIM↑	LPIPS↓	Success↑
8	8.92	19.26	31.293	0.9474	0.0646	44%
16	7.63	15.90	29.557	0.9470	0.0614	64%
32 (ours)	**3.85**	**7.59**	**31.976**	**0.9513**	**0.0495**	**72%**
64	6.86	13.98	31.692	0.9504	0.0551	64%
128	7.08	14.51	31.372	0.9483	0.0581	64%

E.3 Latent-space PCA across tasks

Figure 9 shows the latent-space analysis for four simulation and four real-world tasks. Circles show training-time environment latents, and triangles show latents adapted from test-time interaction trajectories. Colors indicate the underlying physical setting or causal class for visualization only; these labels are not provided to the model. For each task independently, we center its training-time 32-D latents by their mean $\bar{z}$, compute the leading two right singular vectors $V_{1:2}$ of the centered matrix without feature rescaling, and plot $(z - \bar{z})V_{1:2}$. Test-time latents are projected using the same mean and basis; they do not enter the PCA fit. Thus the PC axes are task-specific rather than shared across panels.

E.4 Computational efficiency

We profile a representative Friction setting on one H200 GPU. For training, each method is warmed up for two optimizer steps and timed for ten further steps. Figure 10 reports the median compute-only optimizer time per update, excluding dataset reads, and peak allocated GPU memory rather than full-run training time. Each measured update has six loss-bearing video chunks; method-specific conditioning computation remains included. The world-model and latent updates of our method are separate phases and are therefore shown separately. LoRA TTA starts from the Standard World Model and has no distinct world-model training row.

For inference, all methods receive the same test-time interaction trajectory and five disjoint query video chunks. We use 25 denoising steps and exclude model loading, dataset reads, and MP4 writing from timing; query generation includes pipeline sampling and VAE decoding. Interaction processing is performed once and reused across queries. DINOv2 Context encodes the interaction trajectory without gradient updates, our method runs 40 latent updates, and LoRA TTA runs 50 adapter updates. TTT-KQV precomputes one post-interaction fast state at each denoising timestep and reuses

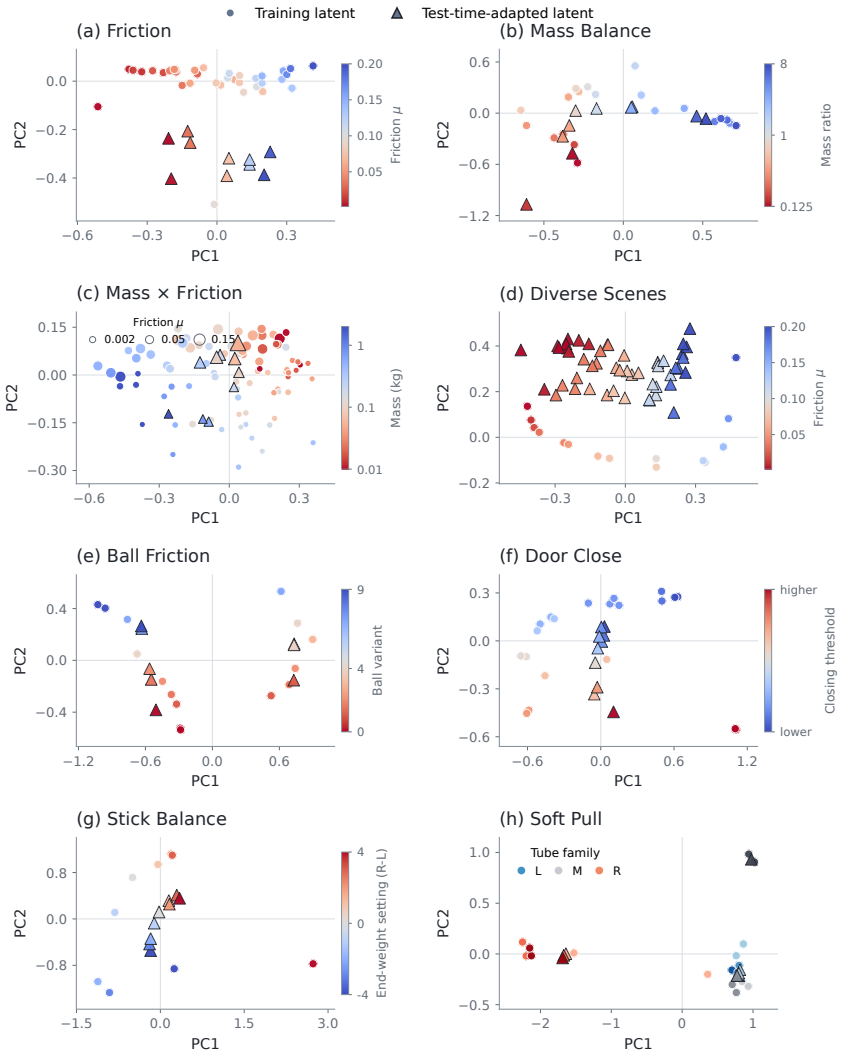


Figure 9: Latent-space PCA across simulation (a–d) and real-world tasks (e–h). Circles denote training-time latents and triangles denote test-time-adapted latents. Color indicates the underlying physical setting; point size additionally indicates friction in Mass × Friction. These factor annotations are used only for visualization and are not provided to the model. The Soft Pull test-time latents shown here use four test-time interaction trajectories.

those states for all five queries. Figure 11 separates interaction processing from five-query generation and shows how processing time changes with $K \in \{1, 2, 4\}$; the five-method total-time comparison uses $K = 1$ throughout. At $K = 1$, our method takes 18.09 s including latent adaptation and five query generations, versus 32.73 s for LoRA TTA and 37.57 s for cached TTT-KQV.

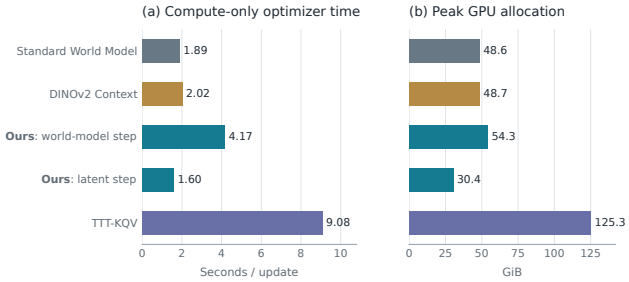


Figure 10: Training resource profile. Time is the median compute-only optimizer duration over ten measured updates after two warmup updates, excluding dataset reads; memory is peak allocated GPU memory. Each update uses six loss-bearing video chunks.

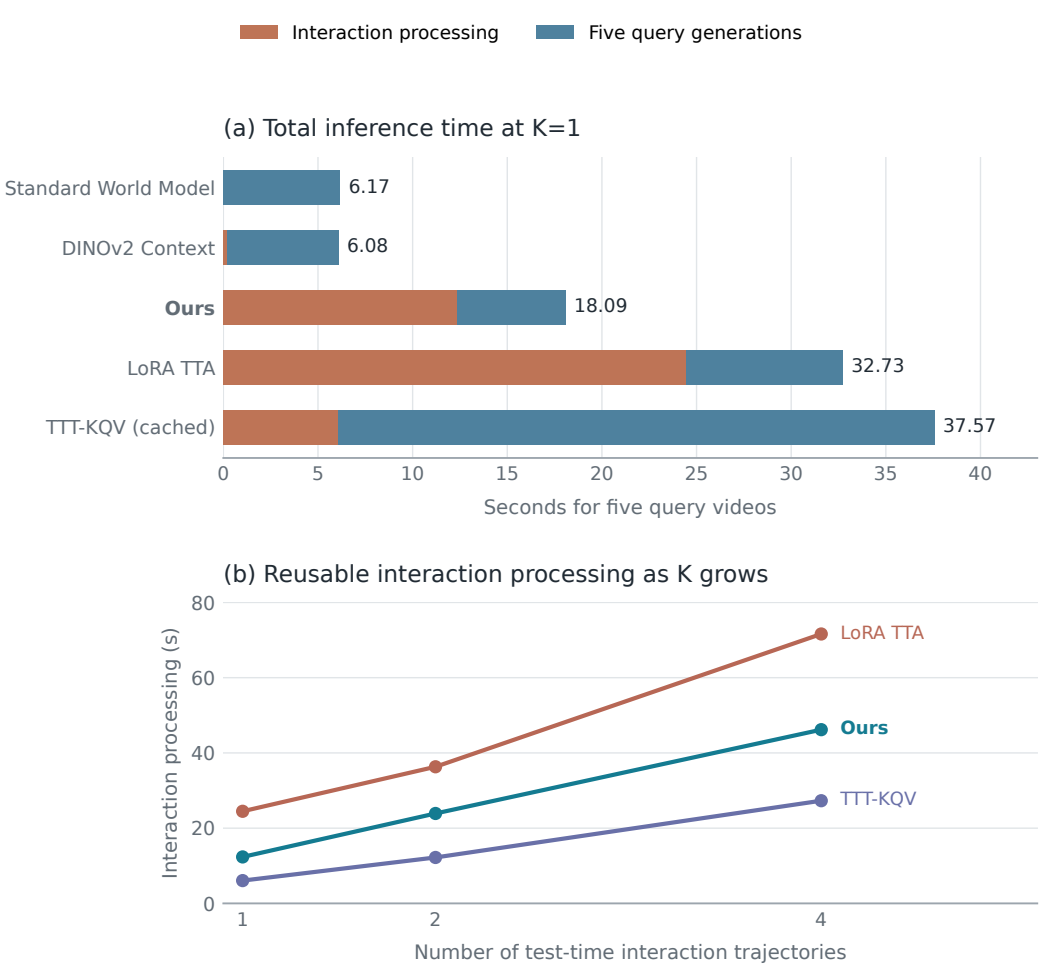


Figure 11: Compute-only inference profile on one H200 GPU, excluding model and sample loading and MP4 writing. Top: processing of one test-time interaction trajectory plus generation of the same five query videos. Bottom: interaction processing time for $K = 1, 2, 4$ with the update counts held fixed within each method. The bottom panel does not compare query-generation time at $K > 1$.

F Additional Qualitative Results

Figures 12–22 show selected held-out test actions. Each task shows two test environments and four actions per environment; Soft Pull shows two actions per environment. Each column uses the same initial observation and candidate action across methods; rows show ground truth, the four baselines, and our prediction. Simulation panels show the main camera only.

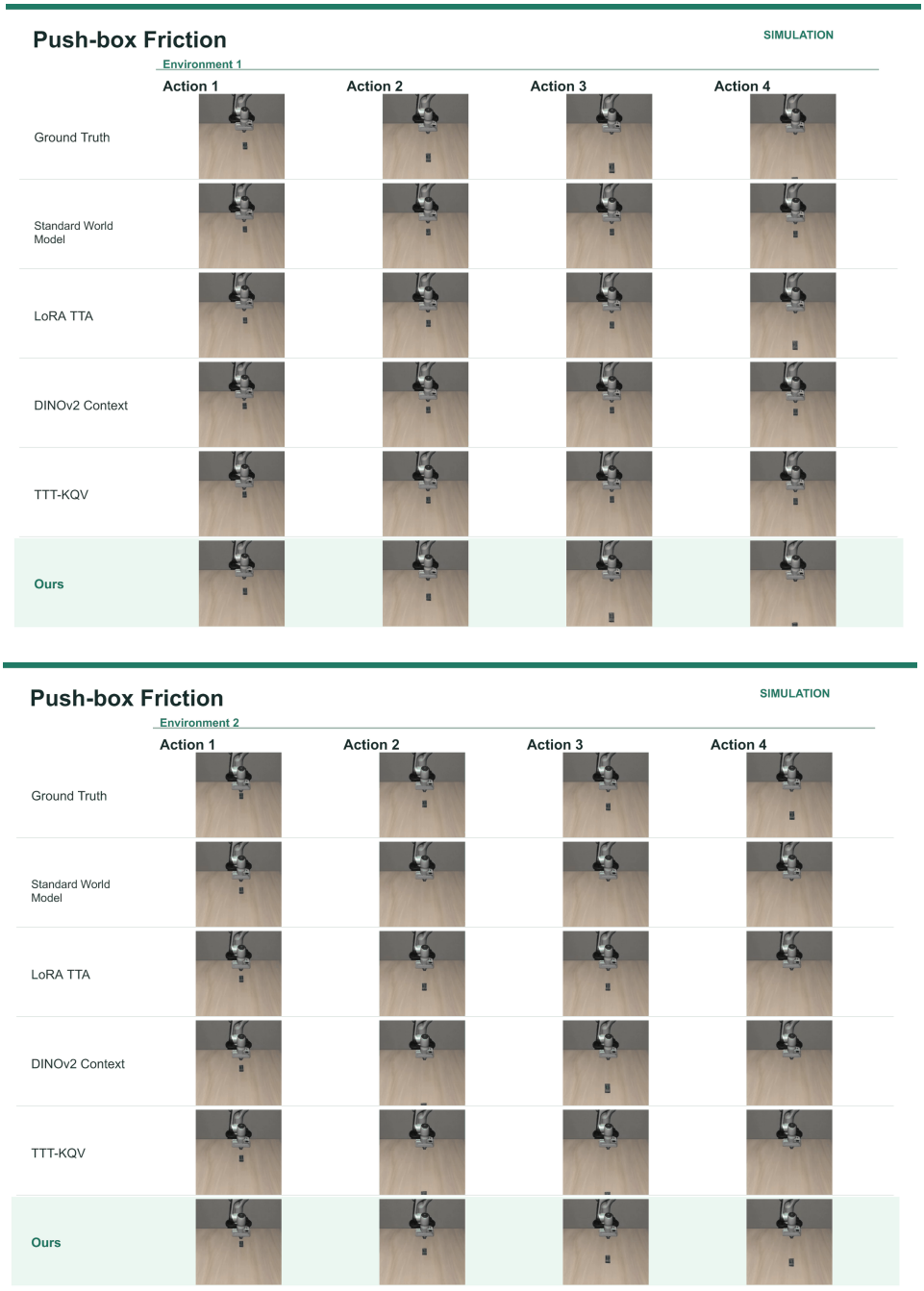


Figure 12: Friction: selected test actions.

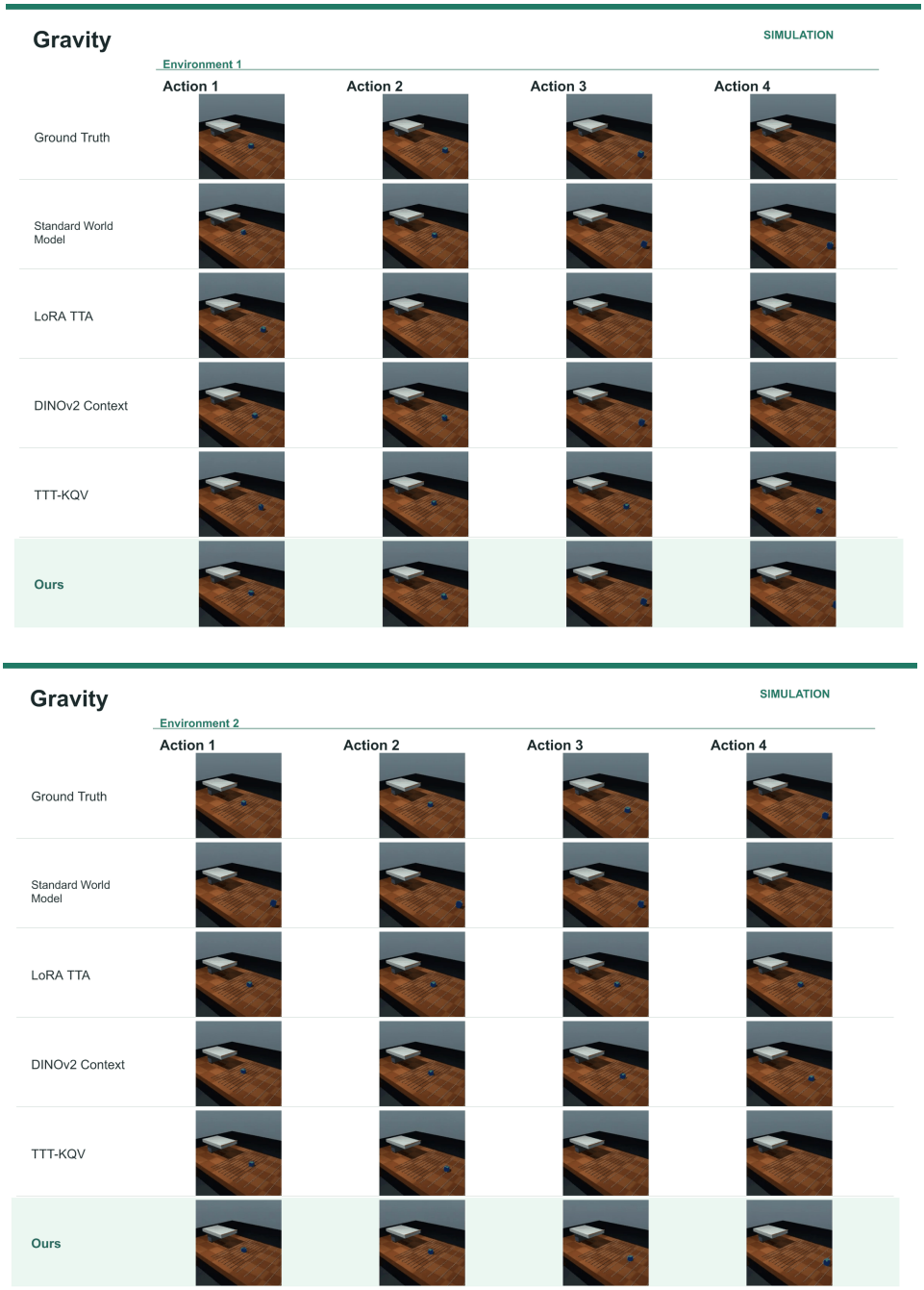


Figure 13: Gravity: selected test actions.

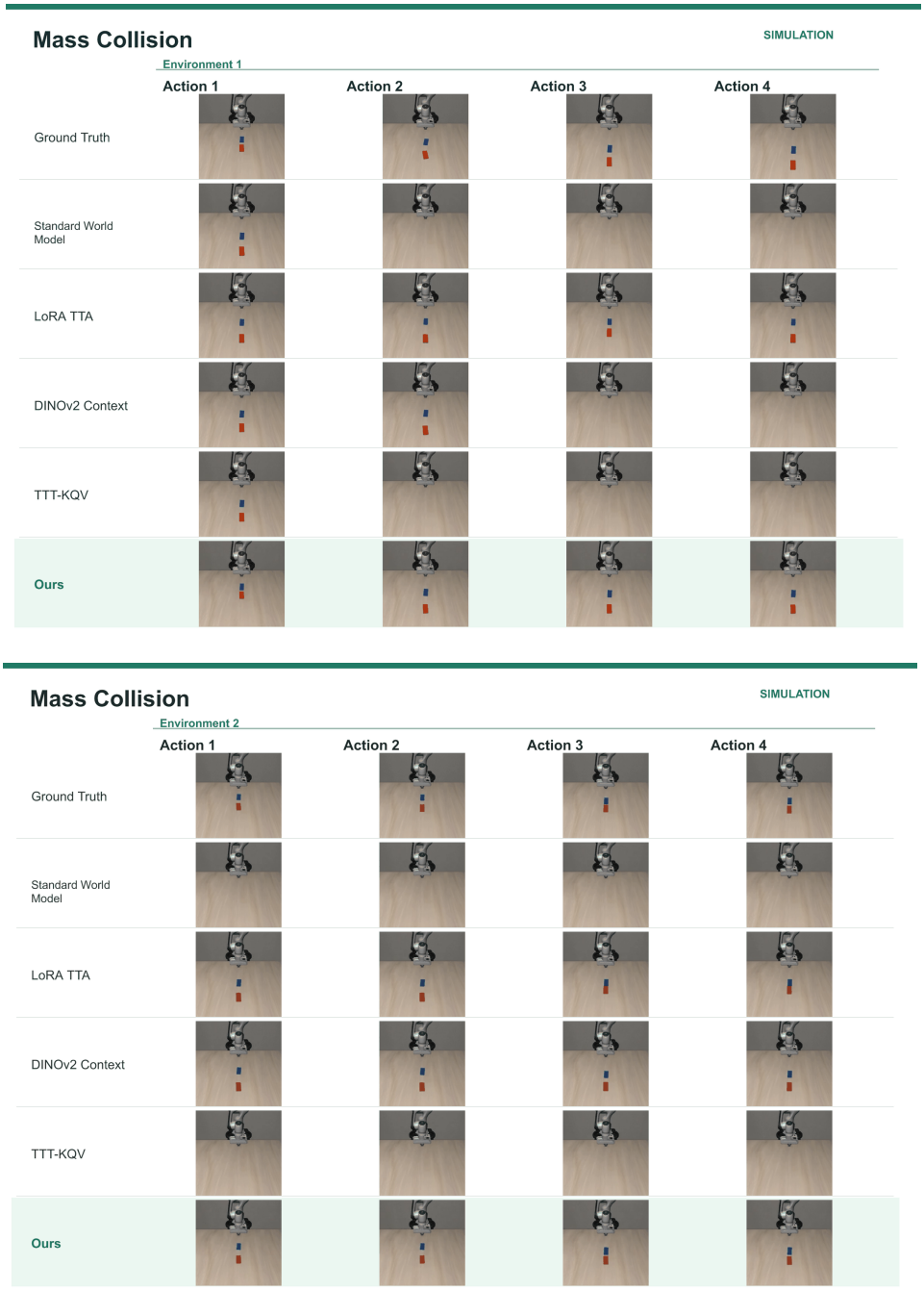


Figure 14: Mass Collision: selected test actions.

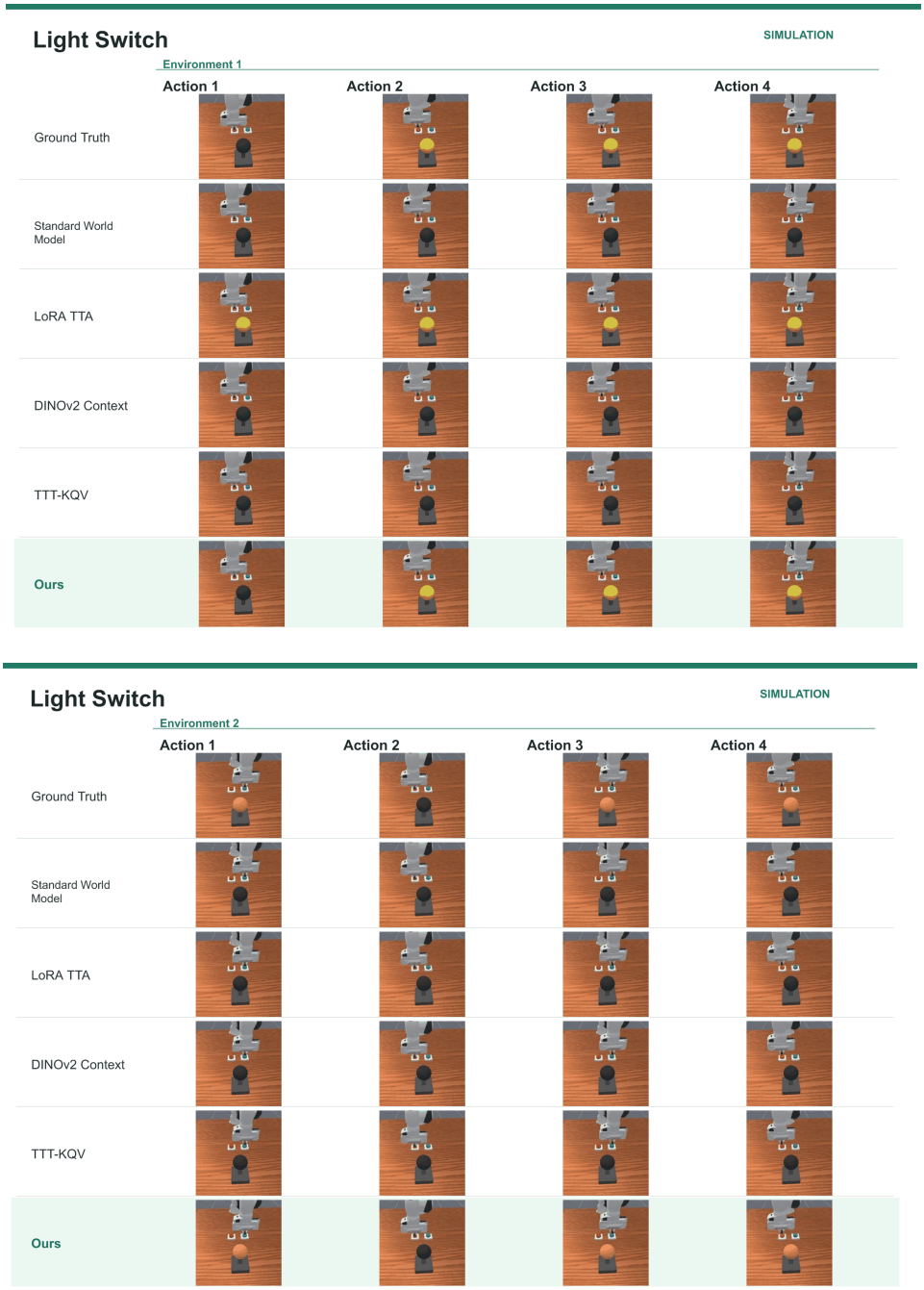


Figure 15: Light Switch: selected test actions.

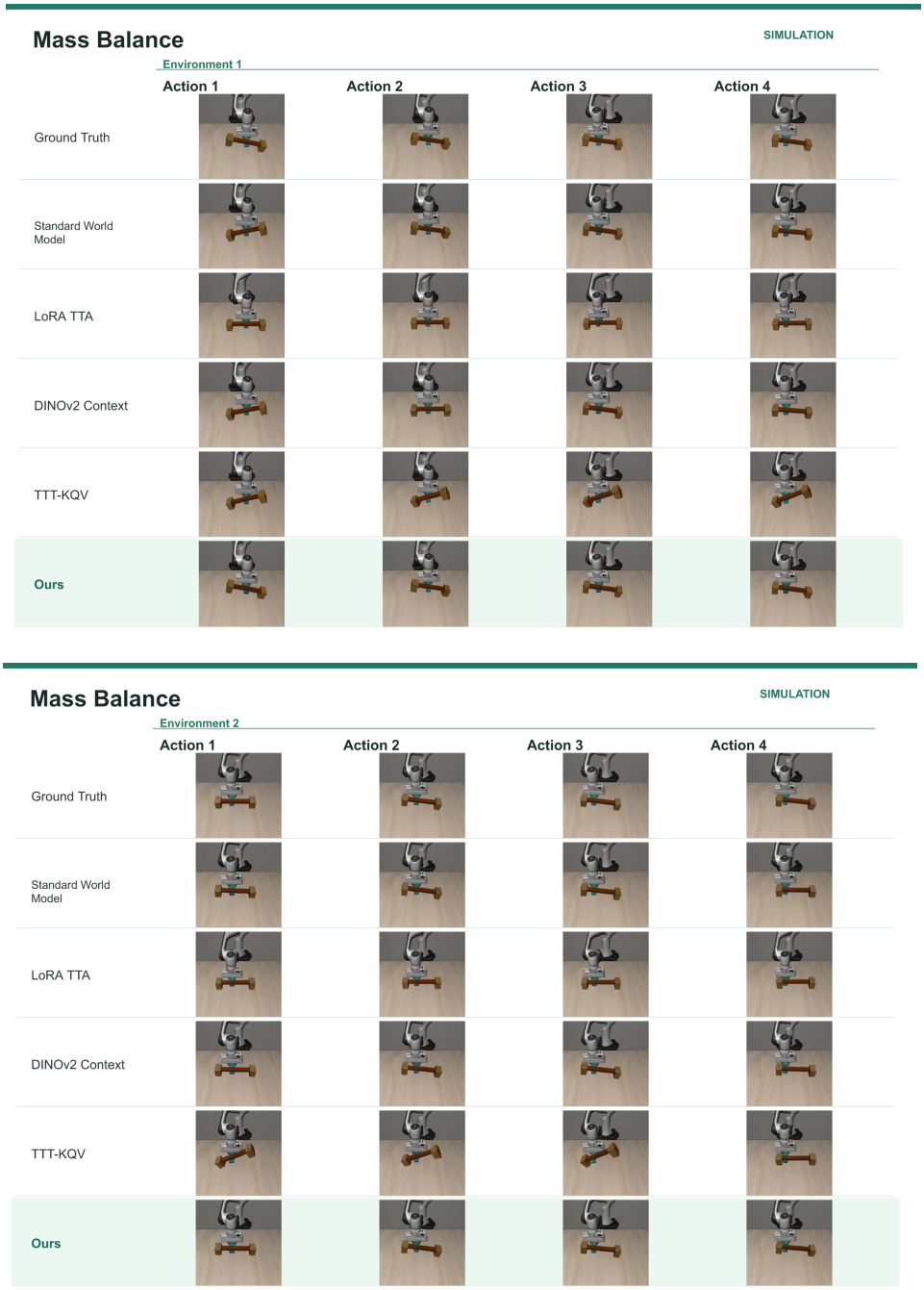


Figure 16: Mass Balance: selected test actions.

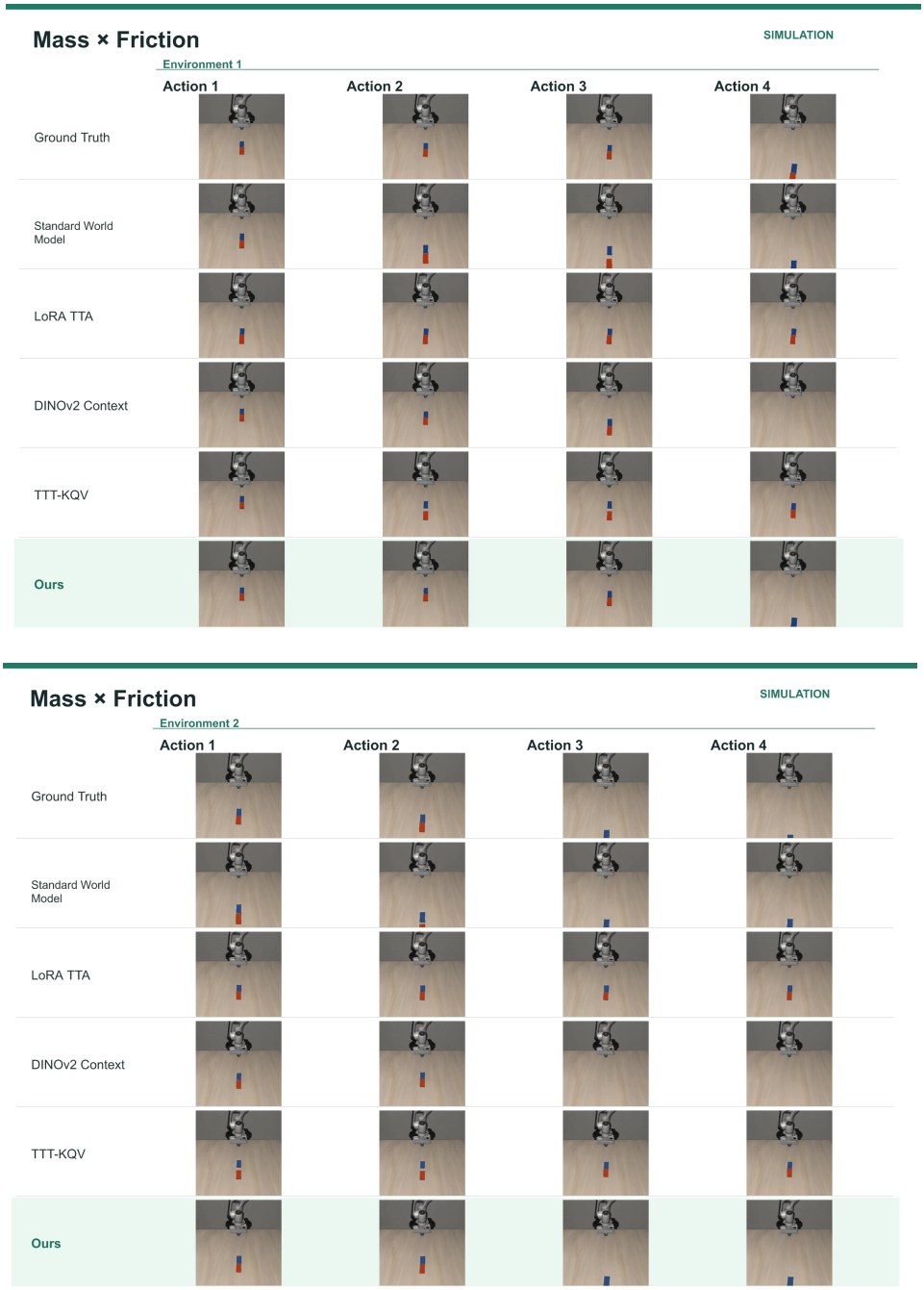


Figure 17: Mass × Friction: selected test actions.

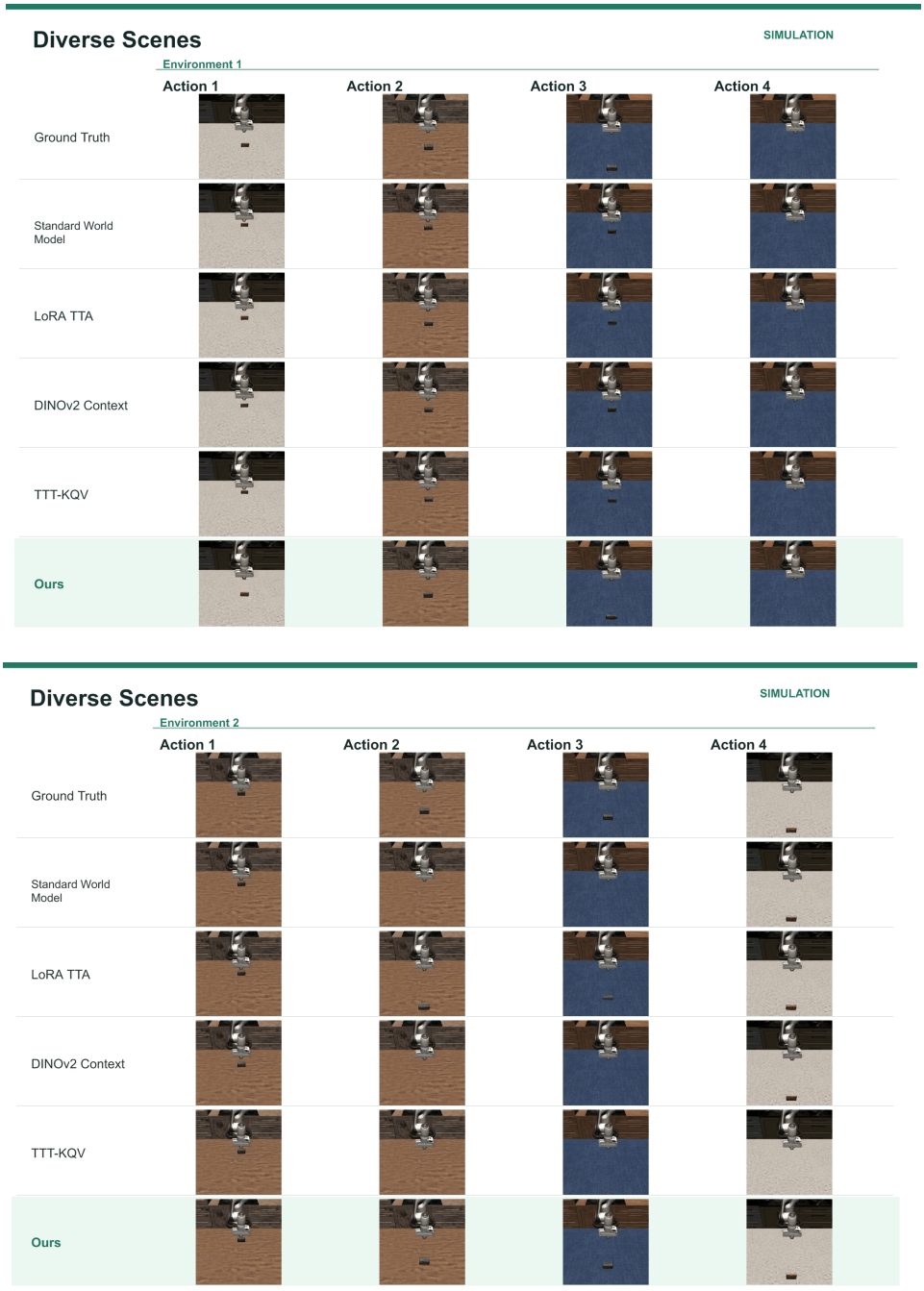


Figure 18: Diverse Backgrounds: selected test actions.

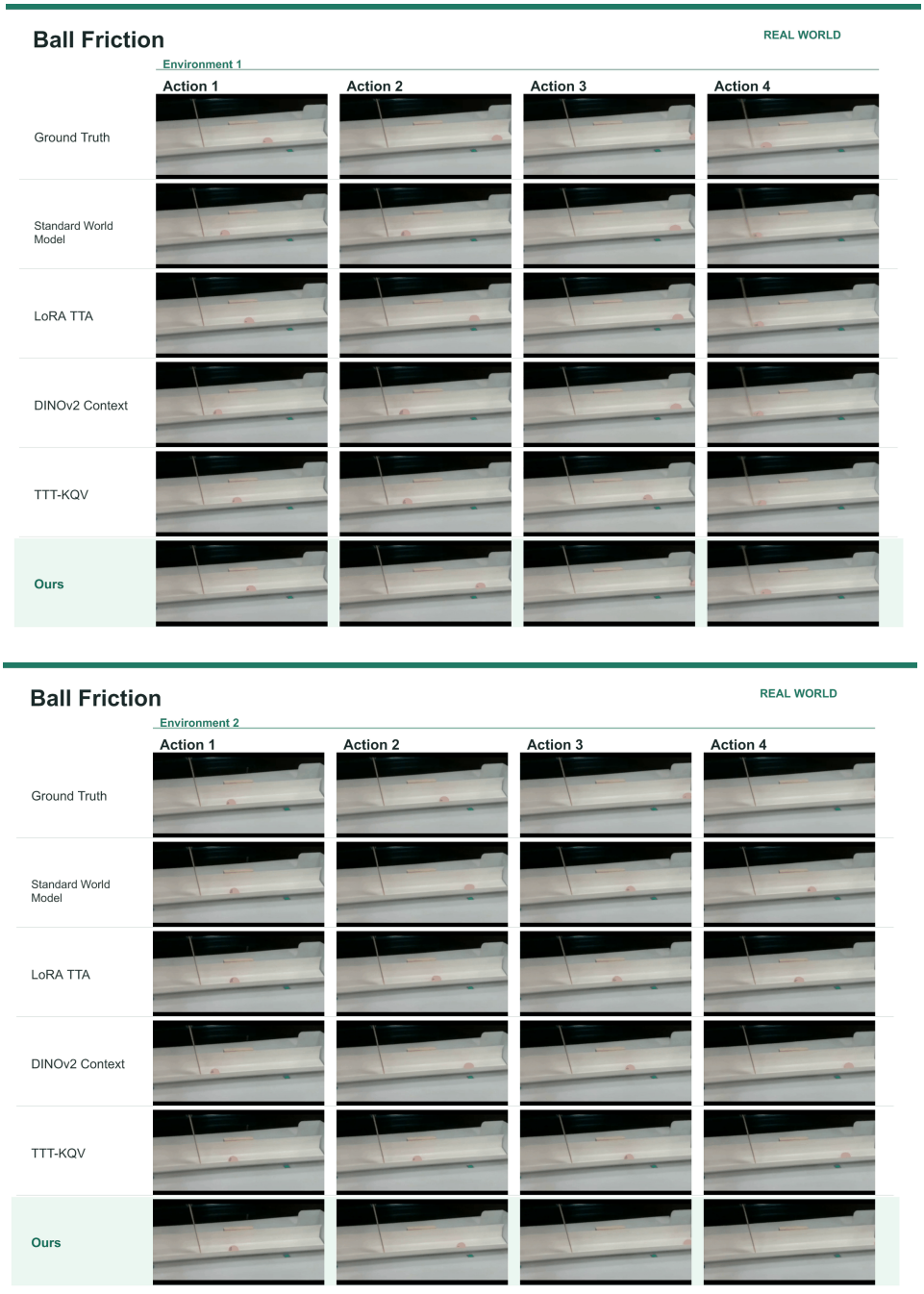


Figure 19: Ball Friction: selected test actions.

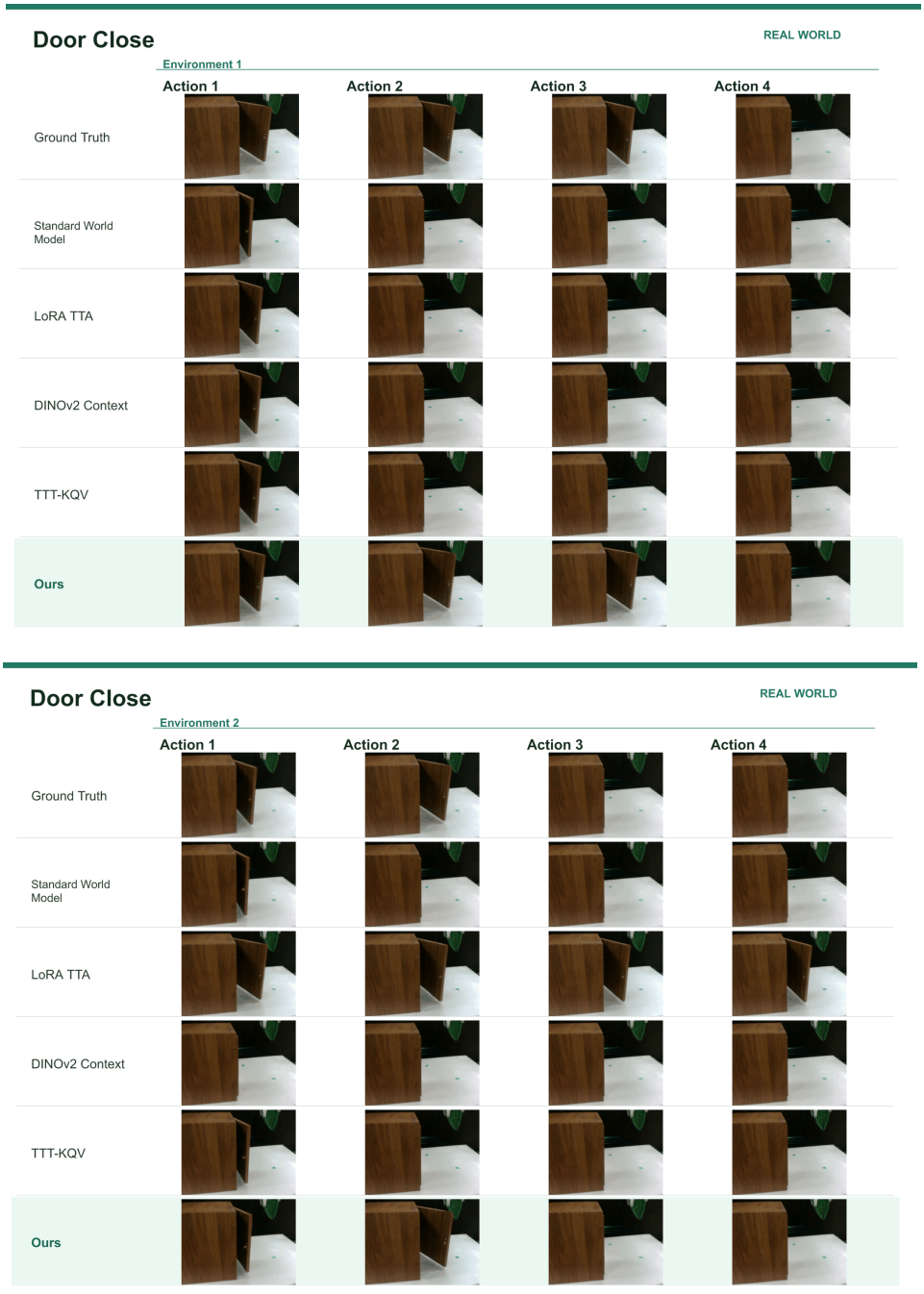


Figure 20: Door Close: selected test actions.

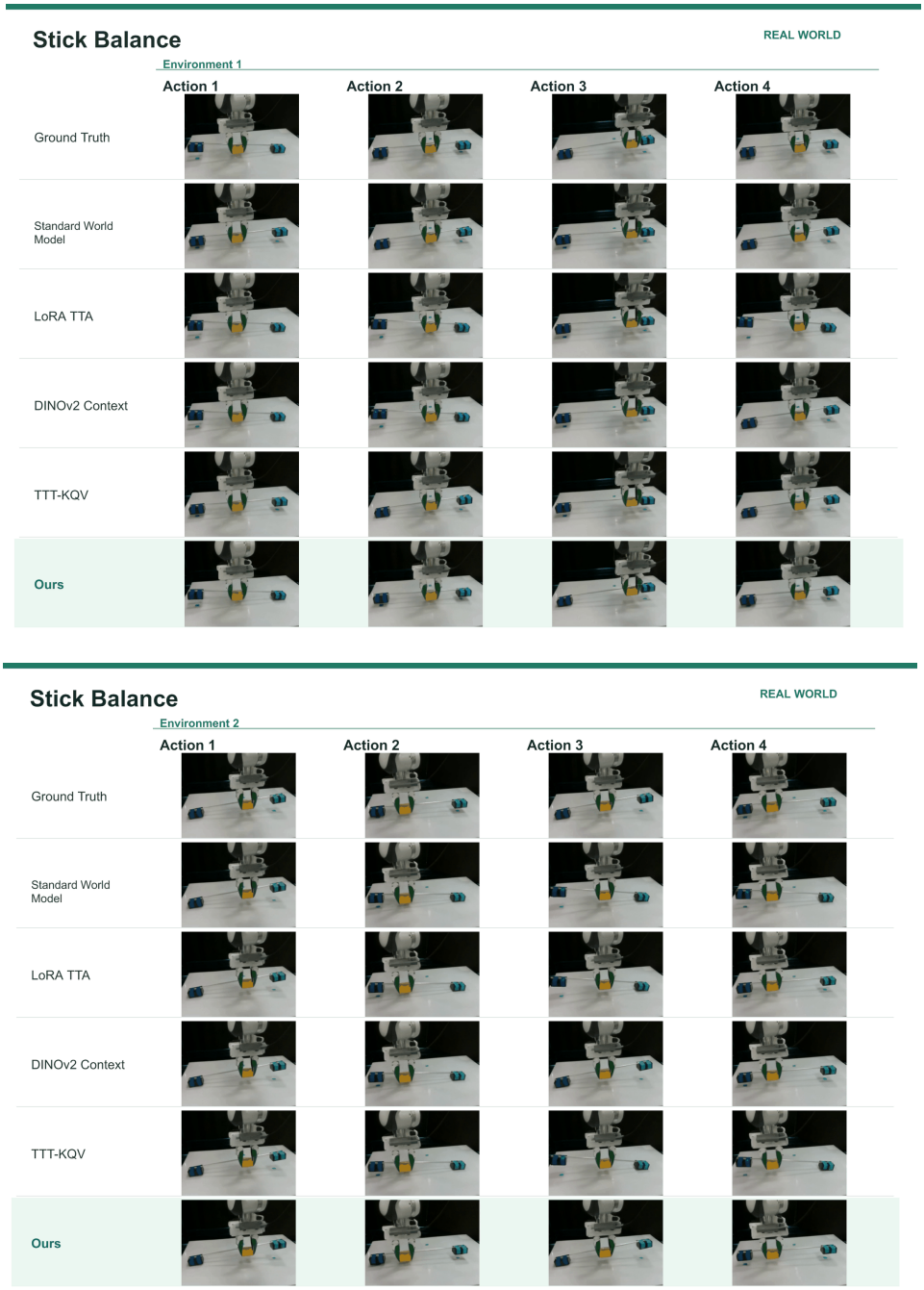


Figure 21: Stick Balance: selected test actions.

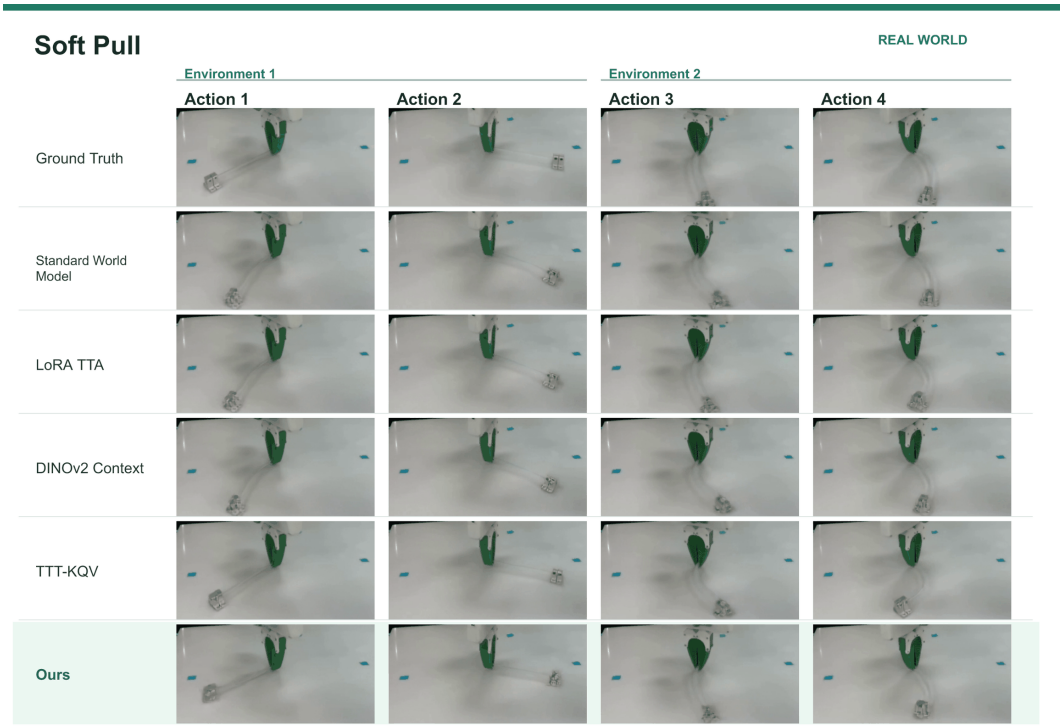


Figure 22: Soft Pull: selected test actions.

REFERENCES

BWM Team. BWM: A low-cost high-fidelity world simulator for robot learning. *arXiv preprint arXiv:2607.29302*, 2026.

Karan Dalal, Daniel Koceja, Jiarui Xu, Yue Zhao, Shihao Han, Ka Chun Cheung, Jan Kautz, Yejin Choi, Yu Sun, and Xiaolong Wang. One-minute video generation with test-time training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17702–17711, 2025.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems, Datasets and Benchmarks Track*, 2023.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.