Scalable Robot Policy Evaluation via Autoregressive Video World Models

Byeongguk Jeon [*1 2] **Seonghyeon Ye** [*1] **JaeHyeok Doo** [1] **Sungdong Kim** [1 2] **Minjoon Seo** [1 2] **Hyungmok Son** [2] **Kimin Lee** [1 2]

Abstract

Video world models offer a scalable alternative to real-world and simulation-based robot policy evaluation, serving as neural simulators. However, video world models suffer from compounding errors over long rollouts and slow inference. To address this gap, we propose WORLDARENA, a scalable evaluation pipeline that combines a few-step autoregressive video world model with a rubric-guided vision-language model (VLM) judge to prevent world-model errors from propagating into evaluation outcomes. We introduce STEP FORCING, which closes the train–test gap by matching noise schedules, contexts, and priors between training and inference. STEP FORCING enables stable long-horizon autoregressive rollouts without teacher distillation or costly self-rollouts at training time. We evaluate eight generalist robot policies on WORLDARENA over 4,186 rollouts, achieving a Pearson correlation of $r = 0.989$ and a Spearman correlation of $\rho = 0.970$ with the real-world RoboArena leaderboard.

1. Introduction

Scalable and reliable evaluation has significantly accelerated progress in foundation models for vision and language (Lin et al., 2014; Russakovsky et al., 2015; Hendrycks et al., 2020; Chiang et al., 2024), and recent generalist robot policies have shown rapid progress to generalize across diverse instructions, objects, and environments (Brohan et al., 2022; Zitkovich et al., 2023; O'Neill et al., 2024; Team et al., 2024; Kim et al., 2024; Bjorck et al., 2025; Black et al., 2024; Physical Intelligence et al., 2025). For a reliable evaluation of generalist robot policies, their capabilities should be measured across diverse environments and objects, but this is challenging in both the real world and simulation. Real-world evaluation requires physical robots and dedicated environments, making it expensive and time-consuming even with distributed infrastructure (Zhou et al., 2025b; Atreya et al., 2025). Simulation-based alternatives (Yu et al., 2020; James et al., 2020; Nasiriany et al., 2024; Wang et al., 2025b; Zhang et al., 2025b; Kim et al., 2026b) require nontrivial engineering for asset and environment setup and suffer from sim-to-real gaps (Zhao et al., 2020; Blanco-Mulero et al., 2024; Tobin et al., 2017).

Video world models built on pretrained video generative models (Blattmann et al., 2023; Brooks et al., 2024; Kong et al., 2024; Chen et al., 2024b; Yang et al., 2024; Ali et al., 2025; Agarwal et al., 2025; Wan et al., 2025) offer a natural alternative: action-conditioned rollouts (Huang et al., 2025a; Zhu et al., 2024) can serve as neural simulators for policy evaluation (Guo et al., 2025a; Team et al., 2025; Quevedo et al., 2025a; Gao et al., 2026; Team, 2025; Team et al., 2026; Yang et al., 2026; Sharma et al., 2026). However, scaling such evaluation demands fast inference and generalization across diverse conditions. Moreover, world-model errors accumulated over long-horizon autoregressive rollouts can compromise the reliability of evaluation outcomes.

We propose WORLDARENA, a scalable evaluation pipeline built on a fast autoregressive video world model trained with STEP FORCING and a rubric-guided vision-language model (VLM) judge. Building on Diffusion Forcing (Chen et al., 2024a), STEP FORCING closes the train–test gap by aligning both the context frames and the noise schedule between training and inference, training the model to recover clean frames from one-step-forwarded priors. Without teacher distillation (Yin et al., 2024b;a) or self-generated contexts (Huang et al., 2025b), STEP FORCING enables stable 300-frame video generation with only 4-step denoising on DROID (Khazatsky et al., 2024), while matching or surpassing world-model baselines that require an order of magnitude more steps. Finally, the judge prevents world-model errors from propagating into evaluation outcomes.

We evaluate eight robot policies on WORLDARENA over 4,186 rollouts from diverse initial views in RoboArena (Atreya et al., 2025), achieving Pearson $r = 0.989$ and Spearman $\rho = 0.970$ with the RoboArena.[1]

[1]KAIST [2]Config. Correspondence to: Byeongguk Jeon <byeongguk@kaist.ac.kr>.

Proceedings of the 43^{rd} International Conference on Machine Learning, Seoul, South Korea. PMLR 306, 2026.

[1]We evaluate the 8 policies that were open-sourced as of the latest RoboArena data dump used in our experiments.

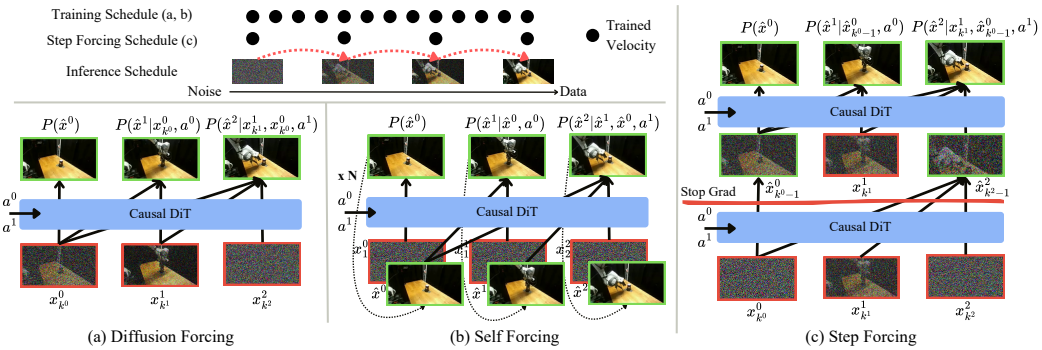


Figure 1. **Left Upper:** STEP FORCING shares the noise schedule between training and inference. **(a)** Diffusion Forcing conditions on noisy ground-truth (red). **(b)** Self Forcing conditions on self-generated context (green) via repeated forward rollouts. **(c)** STEP FORCING conditions on the one-step self-forwarded prior (green) or anchor step (red).

2. Related Work

Reliable evaluation for generalist robot policies requires diverse real-world or simulated rollouts, but both face scalability and sim-to-real gap challenges (Li et al., 2024; Jain et al., 2025; Zhang et al., 2025a). Pretrained video world models offer a scalable alternative via action-conditioned rollouts and automated evaluation (Kim et al., 2026a; Ye et al., 2026; Jang et al., 2025; Liao et al., 2025; Li et al., 2026; Yang et al., 2023; Wu et al., 2024; Zhu et al., 2025; Wang et al., 2025a; Li et al., 2025; Tseng et al., 2025; Quevedo et al., 2025b; Jiang et al., 2025), but slow inference and long-horizon drift limit their practical use. Autoregressive video world models (Bruce et al., 2024; Mao et al., 2025; Zhang et al., 2025c; He et al., 2025; Valevski et al., 2024) mitigate inference cost via key–value caching (Yin et al., 2025; Huang et al., 2025b; Cui et al., 2025), yet remain brittle under multi-view, long-horizon settings (Guo et al., 2025a). Recent work addresses train–test mismatch by training on autoregressive self-generated contexts, either with bidirectional teacher distillation (Huang et al., 2025b) or, concurrently, in a teacher-free manner via history self-resampling (Guo et al., 2025b). However, generating such contexts is computationally expensive.

3. Step Forcing

We introduce STEP FORCING, a training objective for efficient few-step autoregressive video world models. The key idea is to train the model on self-forwarded priors under the same discrete noise schedule used at inference, while anchoring the objective to data-grounded transitions. Figure 1 illustrates the overall procedure.

3.1. Preliminaries: Autoregressive Video World Models

Autoregressive video world models (Bruce et al., 2024; Mao et al., 2025; Zhang et al., 2025c; He et al., 2025; Valevski et al., 2024) factorize future observations $x^{1:N}$ given an initial observation x^0 and action sequence $a^{1:N}$ as

$$p_\theta(x^{1:N} \mid x^0, a^{1:N}) = \prod_{i=1}^{N} p_\theta(x^i \mid x^{0:i-1}, a^i),$$

enabling generation beyond the training horizon and efficient inference with key–value caching (Yin et al., 2025). However, this leads to a train–test context mismatch: Teacher Forcing (Zhou et al., 2025a) and Diffusion Forcing (Chen et al., 2024a) train the model with clean or noised ground-truth contexts $x^{<i}$, whereas inference conditions on self-generated predictions $\hat{x}^{<i}$, causing errors to accumulate over long horizons. Recent work mitigates this mismatch by training directly on self-generated contexts (Huang et al., 2025b; Guo et al., 2025b), but this requires substantial overhead from sequential autoregressive rollouts. Moreover, in action-conditioned world models, replacing $x^{<i}$ with $\hat{x}^{<i}$ shifts the state on which action a^i acts. As a result, supervising $p_\theta(x^i \mid \hat{x}^{<i}, a^i)$ can introduce transitions that are not fully grounded in the action–observation dynamics observed in the data.

3.2. Algorithm

We propose STEP FORCING, training the world model to denoise from one-step self-forwarded priors under a fixed discrete denoising schedule. We formulate denoising with rectified flow (Lipman et al., 2022; Liu et al., 2022), where a clean frame x^i and noise ϵ^i are connected by the linear path $x_t^i = t\epsilon^i + (1-t)x^i$. We define the inference noise schedule $0 = t_0 < \cdots < t_S = 1$ and use the same schedule for one-step self-forwarding during training. This schedule alignment reduces the train–test mismatch and enables efficient few-step denoising (e.g., $S = 4$) without relying on distillation-based few-step samplers (Yin et al., 2025; Huang et al., 2025b).

Formally, given a video-action trajectory $(x^{0:N}, a^{1:N}) \sim \mathcal{D}$ and rectified-flow velocity v_θ (Liu et al., 2022), we form

Table 1. Diagnostic comparison of action-conditioned training on BAIR Robot Pushing.

Method	Steps	SSIM ↑		LPIPS ↓	
		ID	OOD	ID	OOD
Teacher Forcing (Zhou et al., 2025a)	8	0.7942	0.7118	0.0554	0.1058
Diffusion Forcing (Chen et al., 2024a)	8	0.7657	0.6861	0.0690	0.1117
Resampling Forcing (Guo et al., 2025b)	8	0.7891	0.6996	0.0572	0.1082
Self Forcing (Huang et al., 2025b)	4	0.7929	0.6953	0.0548	0.1083
STEP FORCING	4	**0.8063**	**0.7374**	**0.0525**	**0.0768**

per-frame noisy latents $x^i_{k^i}$ from independently sampled noise levels $k^i \sim \mathcal{U}\{1, \dots, S\}$, following Diffusion Forcing (Chen et al., 2024a). Given the noisy context $h^i := \{x^j_{k^j}\}_{j<i}$, we take a single Euler step under stop gradient to obtain a *self-forwarded prior*:

$$\hat{x}^i_{k^i-1} = x^i_{k^i} - (t_{k^i} - t_{k^i-1})\,\mathrm{sg}\big[v_\theta(x^i_{k^i}, t_{k^i}, h^i, a^i)\big], \quad (1)$$

where the stop-gradient, $\mathrm{sg}[\cdot]$, prevents v_θ from being optimized to shape its own future inputs. One-step self-forwarding mitigates the train-test gap by training the model to learn from its own imperfect generation. However, unlike self-generated contexts, it enables *parallel* context generation, significantly reducing the training cost and enabling stable training. Also, to ground supervision in the data distribution, with probability p we apply an *anchor step* that bypasses self-forwarding and sets the prior directly to the noisy ground-truth latent, $\hat{x}^i_{k^i-1} = x^i_{k^i}$. By preventing action-observation mismatch from self-generated context, anchor step preserves ground-truth dynamics, which is essential for action-conditioned video generation. We then predict the clean frame from $\hat{x}^i_{k^i-1}$ conditioned on the self-forwarded context $\hat{h}^i := \{\hat{x}^j_{k^j-1}\}_{j<i}$, and minimize the squared error to x^i; the overall procedure is illustrated in Figure 1, while the full equations are provided in Algorithm 1.

3.3. Diagnostic Comparison on BAIR Robot Pushing

We use BAIR Robot Pushing (Ebert et al., 2017) as a small-scale diagnostic setup for comparing different training objectives in action-conditioned autoregressive world modeling. We examine whether each objective preserves action–observation dynamics while maintaining stable long-horizon rollouts. Specifically, we measure both in-horizon (ID) and extrapolation (OOD) frames. Experimental details are provided in Appendix B.

STEP FORCING achieves the best SSIM (Wang et al., 2004) and LPIPS (Zhang et al., 2018) in both windows (Table 1; Figure 2). More importantly, the baselines expose the failure modes motivating our design. Teacher Forcing preserves data-grounded transitions but suffers from error accumulation during long-horizon autoregressive rollout. Diffusion Forcing improves robustness to noisy contexts, but its generated rollouts can still drift over extrapolation horizons.

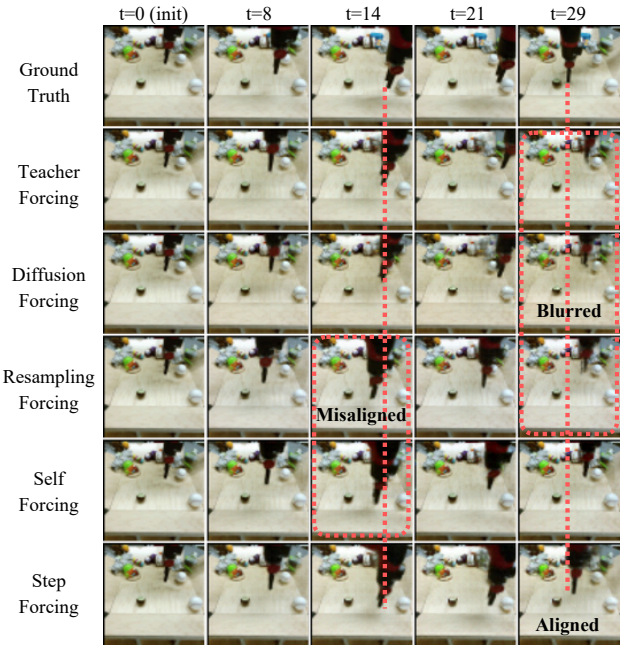


Figure 2. STEP FORCING maintains action controllability and visual quality over long-horizon rollouts.

Resampling Forcing and Self Forcing produce visually plausible videos by training on self-generated contexts, but this weakens action following because the same action is supervised from model-induced states rather than data-grounded states. In contrast, STEP FORCING reduces the train–test context gap through one-step self-forwarded priors while using anchor steps to preserve action–observation dynamics.

4. WORLDARENA

We present WORLDARENA, a fully automated evaluation pipeline for generalist robot policies that scales across diverse environments without requiring physical rollouts or human intervention. WORLDARENA consists of two components: (1) a fast autoregressive video world model trained with STEP FORCING, and (2) a task-progress-aware VLM judge that scores the generated rollouts.

4.1. Adapting Pretrained Video Models into Fast Autoregressive World Models

To adapt pretrained bidirectional video models into fast autoregressive video world models, we make three architectural and training modifications: (1) we replace bidirectional attention with frame-level causal attention, (2) we encode actions with a two-layer MLP and inject them into each frame through cross-attention, and (3) we use per-frame independent noise scheduling during training (Chen et al., 2024a). We first train the model with rectified flow matching (Lipman et al., 2022) under the Diffusion Forcing scheme (Chen et al., 2024a), converting the bidirectional video model into an autoregressive video world model. We then apply STEP

Table 2. 300-frame video generation conditioned on a multi-view frame and the action sequence for 256 RoboArena trajectories. We report metrics averaged equally over all three views.

Method	Denoising steps	Pixel-level metrics			Perceptual metrics		
		PSNR ↑	MSE ↓	SSIM ↑	LPIPS ↓	FID ↓	FVD ↓
Ctrl-World (Guo et al., 2025a)	50	16.61	0.024	0.703	0.320	34.64	188.9
Diffusion Forcing (Chen et al., 2024a)	4	15.81	0.029	0.683	0.327	33.18	242.0
Diffusion Forcing (Chen et al., 2024a)	8	15.69	0.030	0.678	0.327	29.98	210.5
Diffusion Forcing (Chen et al., 2024a)	16	15.52	0.031	0.672	0.331	29.16	190.2
Diffusion Forcing (Chen et al., 2024a)	32	15.36	0.032	0.661	0.338	30.07	**179.0**
Step Forcing	4	**16.78**	**0.023**	**0.715**	**0.293**	**28.54**	203.8

FORCING to enable stable long-horizon autoregressive rollouts with few-step inference. During inference, we evaluate the policy in a closed loop within the world model. At each step, the policy predicts an action given the current observation, and the world model generates the next observation conditioned on the past context and the predicted action. This generated observation is then fed back to the policy to predict the subsequent action. To accelerate this autoregressive decoding, following prior video diffusion models (Gao et al., 2024; Cui et al., 2025; Huang et al., 2025b; Yin et al., 2025), we use key–value (KV) caching with a sliding window context.

4.2. Automatically Scoring Rollouts from World Models

After the world model generates action-conditioned rollouts, a scoring model is needed to evaluate the policy performance conditioned on the rollout. Previous works rely on a VLM that generates a binary score given the generated rollout (Quevedo et al., 2025a; Li et al., 2025). However, a binary success metric conflates policy failures with world model artifacts. For example, when a policy picks up the correct object but fails to place it due to world model error (object disappearance), a binary scorer would evaluate the trajectory as a failure. Instead, we introduce a task-progress-aware evaluation using a predefined 0–5 rubric prompted to a VLM judge. To reliably disentangle true policy failures from world-model artifacts, we instruct the VLM to isolate world-model error detection to the wrist view—where physical inconsistencies predominantly manifest—while assessing actual task progress strictly through the fixed external views. This multi-view evaluation strategy allows us to allocate partial scores reflecting the policy's valid progress before the world-model corruption. Unlike binary metrics, this approach fairly credits policy capabilities under imperfect world-model rollouts. We provide the full VLM prompts and rubric definitions in Appendix D.

5. Experiments

We initialize our video world model from Wan2.1-T2V-1.3B (Wan et al., 2025) and train on DROID (Khazatsky et al., 2024). Full architectural details, training schedule, and inference setup are in Appendix E.

5.1. Multi-View Long-Horizon Generation

We assess long-horizon video generation quality by sampling 256 multi-view initial observations from RoboArena and generating 300-frame videos conditioned on each observation and the corresponding action sequence. We report pixel-level metrics (PSNR, MSE, SSIM (Wang et al., 2004)) and perceptual metrics (LPIPS (Zhang et al., 2018), FID (Heusel et al., 2017), FVD (Unterthiner et al., 2018)) averaged across the three views. As shown in Table 2, our method achieves the best performance on all metrics except FVD. On FVD, it outperforms Diffusion Forcing up to 16 denoising steps and remains competitive with only 4 steps (15 FPS on H200), demonstrating stabilized few-step denoising over long horizons enabled by training on self-forwarded priors. Qualitative comparisons and a detailed speed–performance analysis are provided in Appendices F.1 and F.2.

5.2. Correlation with Real-World Evaluation

To investigate whether WORLDARENA provides reliable evaluation results, we simulate RoboArena evaluation episodes within our world model and measure the correlation with real-world evaluation. For each policy, we reuse the initial observations from RoboArena episodes that contain all three views: two fixed external views and one wrist view.[2] Starting from these initial observations, the policy is executed in closed loop within the world model: at each step, the policy predicts an action from the current generated observation, and the world model predicts the next observation conditioned on that action. This replaces the physical environment in the RoboArena evaluation protocol with our world model while keeping the same policies and initial conditions. Each rollout proceeds for 30 seconds and is scored by a VLM judge. We use GPT-4o as the default judge.

[2]We include all three-view initial observations available in the latest RoboArena data dump (Feb. 26, 2026).

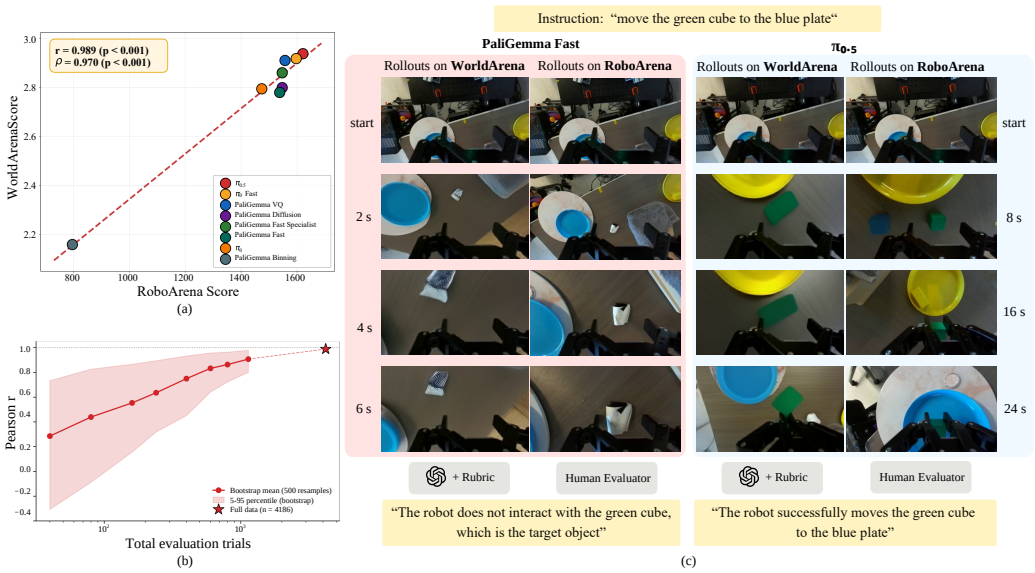


Figure 3. (a) Pearson and Spearman correlations between WORLDARENA and the RoboArena leaderboard. (b) Pearson correlation under varying numbers of evaluation trials, showing that greater diversity in environments and tasks improves the correlation. (c) WORLDARENA closed-loop rollouts compared with RoboArena rollouts using the same policies and tasks, sharing only the initial view. Rollouts proceed from top to bottom: PaliGemma Fast fails in both settings by selecting the wrong object, whereas $\pi_{0.5}$ succeeds in both.

WORLDARENA shows strong positive correlation with the RoboArena leaderboard across the eight evaluated policies, achieving Pearson $r = 0.989$ and Spearman $\rho = 0.970$ (Figure 3a). The resulting policy ranking closely matches the real-world leaderboard under GPT-4o scoring. Replicating the RoboArena benchmark in our neural simulator (across 8 different policies) takes only 100 H100 GPU hours, including the computation and communication costs for each policy action generation and world-model rollout, substantially reducing the cost of large-scale policy evaluation. We further analyze how the number of evaluation trials affects correlation with the real-world leaderboard (Figure 3b). The correlation improves as the number of trials increases, highlighting the importance of broad coverage for reliable policy comparison. Qualitative examples in Figure 3c show that WORLDARENA captures both successful executions and policy-specific failure modes. For example, WORLDARENA models the successful behavior of $\pi_{0.5}$ while also reproducing the wrong-object selection failure observed for PaliGemma Fast in RoboArena. Additional implementation details, qualitative examples, and failure-case analysis are provided in Appendix G.2.

6. Ablations

Component ablation of STEP FORCING. We ablate the main components of STEP FORCING on DROID, with a focus on the wrist view, which is particularly challenging due to its fast, action-sensitive motion. All results are evaluated with 4-step denoising during autoregressive rollout. The full method achieves an FVD of 231.00. Removing the self-forwarded prior, removing the anchor step, and replacing the discrete aligned schedule with continuous timesteps increase FVD to 258.50, 294.00, and 327.00, respectively. These results show that self-forwarding reduces context mismatch, anchor steps mitigate prior drift, and schedule alignment is critical for stable few-step autoregressive rollout. We also observe consistent findings on BAIR Robot Pushing setup; see Table 4 in Appendix B.

Effect of the VLM Evaluation Rubric. We ablate the task-progress rubric (§4) by replacing WORLDARENA scores with binary scores. We measure Spearman correlation between scores and the RoboArena leaderboard ranking across eight policies. The task-progress rubric achieves $\rho = 0.970$, while binary success rate reduces this to $\rho = 0.922$. The rubric's sensitivity to partial task progress is important for reliable policy ranking.

7. Limitations

Long-horizon robot manipulation remains challenging for video world models due to object inconsistency under sustained contact. Scaling embodied interaction data, including human–object interaction data, is a promising direction.

8. Conclusion

Toward reliable world-model-based policy evaluation, we present WORLDARENA, a scalable pipeline combining a fast autoregressive video world model trained with STEP FORCING and a rubric-guided VLM judge.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Agarwal, N., Ali, A., Bala, M., Balaji, Y., Barker, E., Cai, T., Chattopadhyay, P., Chen, Y., Cui, Y., Ding, Y., et al. Cosmos world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025.

Ali, A., Bai, J., Bala, M., Balaji, Y., Blakeman, A., Cai, T., Cao, J., Cao, T., Cha, E., Chao, Y.-W., et al. World simulation with video foundation models for physical ai. *arXiv preprint arXiv:2511.00062*, 2025.

Atreya, P., Pertsch, K., Lee, T., Kim, M. J., Jain, A., Kuramshin, A., Eppner, C., Neary, C., Hu, E., Ramos, F., et al. Roboarena: Distributed real-world evaluation of generalist robot policies. *arXiv preprint arXiv:2506.18123*, 2025.

Bardhan, J., Drozdik, P., Sivic, J., and Petrik, V. Persistent robot world models: Stabilizing multi-step rollouts via reinforcement learning. *arXiv preprint arXiv:2603.25685*, 2026.

Bjorck, J., Castañeda, F., Cherniadev, N., Da, X., Ding, R., Fan, L., Fang, Y., Fox, D., Hu, F., Huang, S., et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

Black, K., Brown, N., Driess, D., Esmail, A., Equi, M., Finn, C., Fusai, N., Groom, L., Hausman, K., Ichter, B., et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Blanco-Mulero, D., Barbany, O., Alcan, G., Colomé, A., Torras, C., and Kyrki, V. Benchmarking the sim-to-real gap in cloth manipulation. *IEEE Robotics and Automation Letters*, 9(3):2981–2988, 2024.

Blattmann, A., Dockhorn, T., Kulal, S., Mendelevitch, D., Kilian, M., Lorenz, D., Levi, Y., English, Z., Voleti, V., Letts, A., et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023.

Brohan, A., Brown, N., Carbajal, J., Chebotar, Y., Dabis, J., Finn, C., Gopalakrishnan, K., Hausman, K., Herzog, A., Hsu, J., et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

Brooks, T., Peebles, B., Holmes, C., DePue, W., Guo, Y., Jing, L., Schnurr, D., Taylor, J., Luhman, T., Luhman, E., Ng, C., Wang, R., and Ramesh, A. Video generation models as world simulators. 2024.

Bruce, J., Dennis, M. D., Edwards, A., Parker-Holder, J., Shi, Y., Hughes, E., Lai, M., Mavalankar, A., Steigerwald, R., Apps, C., et al. Genie: Generative interactive environments. In *International Conference on Machine Learning*, 2024.

Chen, B., Martí Monsó, D., Du, Y., Simchowitz, M., Tedrake, R., and Sitzmann, V. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *Advances in Neural Information Processing Systems*, 37:24081–24125, 2024a.

Chen, H., Zhang, Y., Cun, X., Xia, M., Wang, X., Weng, C., and Shan, Y. Videocrafter2: Overcoming data limitations for high-quality video diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024b.

Chiang, W.-L., Zheng, L., Sheng, Y., Angelopoulos, A. N., Li, T., Li, D., Zhu, B., Zhang, H., Jordan, M., Gonzalez, J. E., et al. Chatbot arena: An open platform for evaluating llms by human preference. In *International Conference on Machine Learning*, 2024.

Cui, J., Wu, J., Li, M., Yang, T., Li, X., Wang, R., Bai, A., Ban, Y., and Hsieh, C.-J. Self-forcing++: Towards minute-scale high-quality video generation. *arXiv preprint arXiv:2510.02283*, 2025.

Denton, E. and Fergus, R. Stochastic video generation with a learned prior. In *International Conference on Machine Learning*, 2018.

Ebert, F., Finn, C., Lee, A. X., and Levine, S. Self-supervised visual planning with temporal skip connections. *CoRL*, 12(16):23, 2017.

Gao, K., Shi, J., Zhang, H., Wang, C., Xiao, J., and Chen, L. Ca2-vdm: Efficient autoregressive video diffusion model with causal generation and cache sharing. *arXiv preprint arXiv:2411.16375*, 2024.

Gao, S., Liang, W., Zheng, K., Malik, A., Ye, S., Yu, S., Tseng, W.-C., Dong, Y., Mo, K., Lin, C.-H., et al. Dreamdojo: A generalist robot world model from large-scale human videos. *arXiv preprint arXiv:2602.06949*, 2026.

Guo, Y., Shi, L. X., Chen, J., and Finn, C. Ctrl-world: A controllable generative world model for robot manipulation. *arXiv preprint arXiv:2510.10125*, 2025a.

Guo, Y., Yang, C., He, H., Zhao, Y., Wei, M., Yang, Z., Huang, W., and Lin, D. End-to-end training for autoregressive video diffusion via self-resampling. *arXiv preprint arXiv:2512.15702*, 2025b.

He, X., Peng, C., Liu, Z., Wang, B., Zhang, Y., Cui, Q., Kang, F., Jiang, B., An, M., Ren, Y., et al. Matrix-game 2.0: An open-source real-time and streaming interactive world model. *arXiv preprint arXiv:2508.13009*, 2025.

Hendrycks, D., Burns, C., Basart, S., Zou, A., Mazeika, M., Song, D., and Steinhardt, J. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*, 2020.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in Neural Information Processing Systems*, 30, 2017.

Huang, S., Wu, J., Zhou, Q., Miao, S., and Long, M. Vid2world: Crafting video diffusion models to interactive world models. *arXiv preprint arXiv:2505.14357*, 2025a.

Huang, X., Li, Z., He, G., Zhou, M., and Shechtman, E. Self forcing: Bridging the train-test gap in autoregressive video diffusion. *arXiv preprint arXiv:2506.08009*, 2025b.

Jain, A., Zhang, M., Arora, K., Chen, W., Torne, M., Irshad, M. Z., Zakharov, S., Wang, Y., Levine, S., Finn, C., et al. Polaris: Scalable real-to-sim evaluations for generalist robot policies. *arXiv preprint arXiv:2512.16881*, 2025.

James, S., Ma, Z., Arrojo, D. R., and Davison, A. J. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.

Jang, J., Ye, S., Lin, Z., Xiang, J., Bjorck, J., Fang, Y., Hu, F., Huang, S., Kundalia, K., Lin, Y.-C., et al. Dreamgen: Unlocking generalization in robot learning through video world models. *arXiv preprint arXiv:2505.12705*, 2025.

Jiang, Y., Chen, S., Huang, S., Chen, L., Zhou, P., Liao, Y., He, X., Liu, C., Li, H., Yao, M., et al. Enerverse-ac: Envisioning embodied environments with action condition. *arXiv preprint arXiv:2505.09723*, 2025.

Khazatsky, A., Pertsch, K., Nair, S., Balakrishna, A., Dasari, S., Karamcheti, S., Nasiriany, S., Srirama, M. K., Chen, L. Y., Ellis, K., et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

Kim, M. J., Pertsch, K., Karamcheti, S., Xiao, T., Balakrishna, A., Nair, S., Rafailov, R., Foster, E., Lam, G., Sanketi, P., et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Kim, M. J., Gao, Y., Lin, T.-Y., Lin, Y.-C., Ge, Y., Lam, G., Liang, P., Song, S., Liu, M.-Y., Finn, C., et al. Cosmos policy: Fine-tuning video models for visuomotor control and planning. *arXiv preprint arXiv:2601.16163*, 2026a.

Kim, Y., Pumacay, W., Rayyan, O., Argus, M., Han, W., VanderBilt, E., Salvador, J., Deshpande, A., Hendrix, R., Jauhri, S., et al. Molmospaces: A large-scale open ecosystem for robot navigation and manipulation. *arXiv preprint arXiv:2602.11337*, 2026b.

Kong, W., Tian, Q., Zhang, Z., Min, R., Dai, Z., Zhou, J., Xiong, J., Li, X., Wu, B., Zhang, J., et al. Hunyuanvideo: A systematic framework for large video generative models. *arXiv preprint arXiv:2412.03603*, 2024.

Langley, P. Crafting papers on machine learning. In *International Conference on Machine Learning*, 2000.

Li, L., Zhang, Q., Luo, Y., Yang, S., Wang, R., Han, F., Yu, M., Gao, Z., Xue, N., Zhu, X., et al. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026.

Li, X., Hsu, K., Gu, J., Pertsch, K., Mees, O., Walke, H. R., Fu, C., Lunawat, I., Sieh, I., Kirmani, S., et al. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024.

Li, Y., Zhu, Y., Wen, J., Shen, C., and Xu, Y. Worldeval: World model as real-world robot policies evaluator. *arXiv preprint arXiv:2505.19017*, 2025.

Liao, Y., Zhou, P., Huang, S., Yang, D., Chen, S., Jiang, Y., Hu, Y., Cai, J., Liu, S., Luo, J., et al. Genie envisioner: A unified world foundation platform for robotic manipulation. *arXiv preprint arXiv:2508.05635*, 2025.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, 2014.

Lipman, Y., Chen, R. T., Ben-Hamu, H., Nickel, M., and Le, M. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Liu, X., Gong, C., and Liu, Q. Flow straight and fast: Learning to generate and transfer data with rectified flow. *arXiv preprint arXiv:2209.03003*, 2022.

Mao, X., Lin, S., Li, Z., Li, C., Peng, W., He, T., Pang, J., Chi, M., Qiao, Y., and Zhang, K. Yume: An interactive world generation model. *arXiv preprint arXiv:2507.17744*, 2025.

Nasiriany, S., Maddukuri, A., Zhang, L., Parikh, A., Lo, A., Joshi, A., Mandlekar, A., and Zhu, Y. Robocasa: Large-scale simulation of everyday tasks for generalist robots. *arXiv preprint arXiv:2406.02523*, 2024.

O'Neill, A., Rehman, A., Maddukuri, A., Gupta, A., Padalkar, A., Lee, A., Pooley, A., Gupta, A., Mandlekar, A., Jain, A., et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *IEEE International Conference on Robotics and Automation*, 2024.

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

Physical Intelligence, Black, K., Brown, N., Darpinian, J., Dhabalia, K., Driess, D., Esmail, A., Equi, M., Finn, C., Fusai, N., et al. $\pi_{0.5}$: a vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Quevedo, J., Liang, P., and Yang, S. Evaluating robot policies in a world model. *arXiv preprint arXiv:2506.00613*, 2025a.

Quevedo, J., Sharma, A. K., Sun, Y., Suryavanshi, V., Liang, P., and Yang, S. Worldgym: World model as an environment for policy evaluation. *arXiv preprint arXiv:2506.00613*, 2025b.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115(3): 211–252, 2015.

Sharma, A. K., Sun, Y., Lu, N., Zhang, Y., Liu, J., and Yang, S. World-gymnast: Training robots with reinforcement learning in a world model. *arXiv preprint arXiv:2602.02454*, 2026.

Srivastava, N., Mansimov, E., and Salakhudinov, R. Unsupervised learning of video representations using lstms. In *International Conference on Machine Learning*, 2015.

Su, J., Ahmed, M., Lu, Y., Pan, S., Bo, W., and Liu, Y. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, pp. 127063, 2024.

Team, G., Wang, B., Ni, C., Huang, G., Zhao, G., Li, H., Li, J., Lv, J., Liu, J., Feng, L., et al. Gigabrain-0.5 m*: a vla that learns from world model-based reinforcement learning. *arXiv preprint arXiv:2602.12099*, 2026.

Team, G. R., Devin, C., Du, Y., Dwibedi, D., Gao, R., Jindal, A., Kipf, T., Kirmani, S., Liu, F., Majumdar, A., et al. Evaluating gemini robotics policies in a veo world simulator. *arXiv preprint arXiv:2512.10675*, 2025.

Team, O. M., Ghosh, D., Walke, H., Pertsch, K., Black, K., Mees, O., Dasari, S., Hejna, J., Kreiman, T., Xu, C., et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Team, X. W. M. 1x world model: Evaluating bits, not atoms. Technical report, 1X Technologies, 2025. Accessed: 2026-02-23.

Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P. Domain randomization for transferring deep neural networks from simulation to the real world. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017.

Tseng, W.-C., Gu, J., Zhang, Q., Mao, H., Liu, M.-Y., Shkurti, F., and Yen-Chen, L. Scalable policy evaluation with video world models. *arXiv preprint arXiv:2511.11520*, 2025.

Unterthiner, T., Van Steenkiste, S., Kurach, K., Marinier, R., Michalski, M., and Gelly, S. Towards accurate generative models of video: A new metric & challenges. *arXiv preprint arXiv:1812.01717*, 2018.

Valevski, D., Leviathan, Y., Arar, M., and Fruchter, S. Diffusion models are real-time game engines. *arXiv preprint arXiv:2408.14837*, 2024.

Wan, T., Wang, A., Ai, B., Wen, B., Mao, C., Xie, C.-W., Chen, D., Yu, F., Zhao, H., Yang, J., et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Wang, L., Zhao, K., Liu, C., and Chen, X. Learning real-world action-video dynamics with heterogeneous masked autoregression. *arXiv preprint arXiv:2502.04296*, 2025a.

Wang, Y. R., Ung, C., Tannert, G., Duan, J., Li, J., Le, A., Oswal, R., Grotz, M., Pumacay, W., Deng, Y., et al. Roboeval: Where robotic manipulation meets structured and scalable evaluation. *arXiv preprint arXiv:2507.00435*, 2025b.

Wang, Z., Bovik, A., Sheikh, H., and Simoncelli, E. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13 (4):600–612, 2004.

Wu, J., Yin, S., Feng, N., He, X., Li, D., Hao, J., and Long, M. ivideogpt: Interactive videogpts are scalable world models. *Advances in Neural Information Processing Systems*, 37:68082–68119, 2024.

Yang, J., Lin, K., Li, J., Zhang, W., Lin, T., Wu, L., Su, Z., Zhao, H., Zhang, Y.-Q., Chen, L., et al. Rise: Self-improving robot policy with compositional world model. *arXiv preprint arXiv:2602.11075*, 2026.

Yang, M., Du, Y., Ghasemipour, K., Tompson, J., Schuurmans, D., and Abbeel, P. Learning interactive real-world simulators. *arXiv preprint arXiv:2310.06114*, 2023.

Yang, Z., Teng, J., Zheng, W., Ding, M., Huang, S., Xu, J., Yang, Y., Hong, W., Zhang, X., Feng, G., et al. Cogvideox: Text-to-video diffusion models with an expert transformer. *arXiv preprint arXiv:2408.06072*, 2024.

Ye, S., Ge, Y., Zheng, K., Gao, S., Yu, S., Kurian, G., Indupuru, S., Tan, Y. L., Zhu, C., Xiang, J., et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.

Yin, T., Gharbi, M., Park, T., Zhang, R., Shechtman, E., Durand, F., and Freeman, B. Improved distribution matching distillation for fast image synthesis. *Advances in Neural Information Processing Systems*, 37:47455–47487, 2024a.

Yin, T., Gharbi, M., Zhang, R., Shechtman, E., Durand, F., Freeman, W. T., and Park, T. One-step diffusion with distribution matching distillation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024b.

Yin, T., Zhang, Q., Zhang, R., Freeman, W. T., Durand, F., Shechtman, E., and Huang, X. From slow bidirectional to fast autoregressive video diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

Yu, T., Quillen, D., He, Z., Julian, R., Hausman, K., Finn, C., and Levine, S. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, 2020.

Zhang, K., Sha, S., Jiang, H., Loper, M., Song, H., Cai, G., Xu, Z., Hu, X., Zheng, C., and Li, Y. Real-to-sim robot policy evaluation with gaussian splatting simulation of soft-body interactions. *arXiv preprint arXiv:2511.04665*, 2025a.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., and Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018.

Zhang, S., Xu, Z., Liu, P., Yu, X., Li, Y., Gao, Q., Fei, Z., Yin, Z., Wu, Z., Jiang, Y.-G., et al. Vlabench: A large-scale benchmark for language-conditioned robotics manipulation with long-horizon reasoning tasks. In *IEEE/CVF International Conference on Computer Vision*, 2025b.

Zhang, Y., Peng, C., Wang, B., Wang, P., Zhu, Q., Kang, F., Jiang, B., Gao, Z., Li, E., Liu, Y., et al. Matrix-game: Interactive world foundation model. *arXiv preprint arXiv:2506.18701*, 2025c.

Zhao, W., Queralta, J. P., and Westerlund, T. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *IEEE symposium series on computational intelligence (SSCI)*, 2020.

Zhou, D., Sun, Q., Peng, Y., Yan, K., Dong, R., Wang, D., Ge, Z., Duan, N., and Zhang, X. Taming teacher forcing for masked autoregressive video generation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025a.

Zhou, Z., Atreya, P., Tan, Y. L., Pertsch, K., and Levine, S. Autoeval: Autonomous evaluation of generalist robot manipulation policies in the real world. *arXiv preprint arXiv:2503.24278*, 2025b.

Zhu, F., Wu, H., Guo, S., Liu, Y., Cheang, C., and Kong, T. Irasim: Learning interactive real-robot action simulators. *arXiv preprint arXiv:2406.14540*, 2024.

Zhu, F., Wu, H., Guo, S., Liu, Y., Cheang, C., and Kong, T. Irasim: A fine-grained world model for robot manipulation. In *IEEE/CVF International Conference on Computer Vision*, 2025.

Zitkovich, B., Yu, T., Xu, S., Xu, P., Xiao, T., Xia, F., Wu, J., Wohlhart, P., Welker, S., Wahid, A., et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, 2023.

Algorithm 1 STEP FORCING

Require: Dataset $\mathcal{D}$ of (video, action) pairs
Require: Denoising schedule $\{t_0, \dots, t_S\}$ with $0 = t_0 < \cdots < t_S = 1$, where S denotes the number of denoising steps.
Require: Velocity network v_θ, number of frames N, anchor probability p

repeat
  Sample $(x, a) \sim \mathcal{D}$
  Sample $k^i \sim \mathcal{U}\{1, \dots, S\}$, $\epsilon^i \sim \mathcal{N}(0, I)$ for $i \in \{1, \dots, N\}$
  $x^i_{k^i} \leftarrow t_{k^i}\, \epsilon^i + (1 - t_{k^i})\, x^i$ {noisy frames}
  $h^i \leftarrow \{x^j_{k^j}\}_{j<i}$ {context frames}
  With probability p, set $\Delta t^i \leftarrow 0$ *(anchor)*; otherwise $\Delta t^i \leftarrow t_{k^i} - t_{k^i-1}$ *(self-forward)*
  $\tilde{t}^i \leftarrow t_{k^i} - \Delta t^i$ {actual noise level of prior}
  $\hat{x}^i_{k^i-1} \leftarrow x^i_{k^i} - \Delta t^i \cdot \mathrm{sg}\big[v_\theta(x^i_{k^i}, t_{k^i}, h^i, a^i)\big]$ {one-step self-forwarded prior}
  $\hat{h}^i \leftarrow \{\hat{x}^j_{k^j-1}\}_{j<i}$ {self-forwarded context frames}
  $\hat{x}^i \leftarrow \hat{x}^i_{k^i-1} - \tilde{t}^i\, v_\theta(\hat{x}^i_{k^i-1}, \tilde{t}^i, \hat{h}^i, a^i)$ {clean prediction}
  Update θ by descending $\nabla_\theta \frac{1}{N} \sum_{i=1}^{N} \|x^i - \hat{x}^i\|_2^2$
until converged

A. Step Forcing Algorithm

A.1. Algorithm Details

We provide the detailed STEP FORCING training procedure in Algorithm 1, with per-frame action conditioning.

B. Implementation Details: Action-conditioned BAIR

Dataset. We use the BAIR Robot Pushing dataset (Ebert et al., 2017): 43,264 training clips and 256 test clips of 30 frames each, 64×64 RGB pixels in $[-1, 1]$, accompanied by a per-frame 4-dimensional robot action. Training samples a random 15-frame window per clip; evaluation takes the first 30 frames of each test clip and uses K=1 as the conditioning prefix. All experiments use the original 64×64 resolution.

Architecture Details. We use a causal video DiT (Peebles & Xie, 2023) adapted to the 3-channel RGB pixel space (Table 3). Each frame is patchified by a 4×4 Conv2d into 16×16=256 spatial tokens. Each DiT block contains spatial self-attention (per frame), causal temporal self-attention (across frames), and an MLP, with all three components AdaLN-modulated by a per-frame conditioning vector. Temporal position is encoded with Rotary Position Embeddings (RoPE) (Su et al., 2024) applied to the temporal-attention Q, K projections. The action vector is projected through a two-layer MLP and added to the per-frame condition that drives AdaLN.

Table 3. Backbone hyperparameters.

	student / main	teacher
embed_dim	256	448
depth	6	12
num_heads	8	14
patch_size	4	4
RoPE max_seq	128	128
attention	causal	bidirectional
action MLP	$4, 512, 256$	$4, 896, 448$
parameter count	9.2M	54.6M

Loss Function. All five methods share the same backbone and the same loss target (v-prediction).

- **Diffusion Forcing** (Chen et al., 2024a): per-frame independent noise levels $t_i \sim \mathcal{U}(0.001, 0.999)$. A single forward

pass through the causal DiT predicts v_i for every frame and the loss is averaged.

- **Teacher Forcing** (Zhou et al., 2025a): per-frame independent $t_i \sim \mathcal{U}(0.001, 0.999)$ as in DF, but each target frame n_i should be predicted conditional on *strict-past clean* frames $c_{<i}$, not on the (noisy) future. Naively this is one forward pass per target i (T passes per step). Following Resampling Forcing (Guo et al., 2025b), we use the 2T-sequence dense trick: pack inputs as $[c_0, ..., c_{T-1}, n_0, ..., n_{T-1}]$, share temporal indices between c_i and n_i, and apply the block-attention mask shown below. This delivers one supervised prediction per frame in a single forward.

- **Resampling Forcing** (Guo et al., 2025b): Because the official code is not publicly available, we implemented faithfully according to the paper. Each training step has two phases: (i) *Resampling*: sample a shared timestep $t_s \sim$ LogitNormal(shift=0.6), corrupt all frames at t_s with i.i.d. Gaussian noise, then *sequentially* 1-step Euler-denoise per frame in `no_grad` to obtain resampled clean estimates $\tilde{x}_{0..T-1}$; (ii) *Supervised step*: sample per-frame independent $t_i \sim \mathcal{U}(0.001, 0.999)$, pack $[\tilde{x}_0, \ldots, \tilde{x}_{T-1}, n_0, \ldots, n_{T-1}]$ with shared RoPE indices, and apply the same 2T-dense mask as in TF.

- **Self Forcing** (Huang et al., 2025b): implemented based on the official code, with the following modifications motivated by our small-scale BAIR setup:

  1. *initial view conditioning.* For a fair comparison, we use the same 9M-parameter causal DiT backbone as the other BAIR baselines, whereas the original Self Forcing implementation is built on Wan-1.3B-i2v. Accordingly, the conditioning interface differs slightly: the original implementation uses channel-axis image-to-video conditioning, while our BAIR setup uses frame-axis conditioning. Specifically, the ground-truth conditioning frame is concatenated along the temporal axis and pinned at t=0. To keep the score networks in-distribution during distillation, we prepend this conditioning frame at t=0 to every `real_score` and `fake_score` call.
  2. *Numerical stabilization.* For numerical stability, we clamp the DMD2 normalizer $\|x_{0,\text{real}} - x_{0,\text{student}}\|_1$ from below by a small constant ε=0.1. This avoids division-by-zero when the teacher nearly perfectly reconstructs the student output.

  All other hyperparameters follow the paper: LR_G=2×10^{-6}, LR_F=4×10^{-7}, AdamW β_1=0, β_2=0.999, weight_decay=0.01, EMA=0.99 from step 200, critic:generator update ratio 5:1.

- **STEP FORCING**: trained at a fixed discrete schedule S=4, with per-frame timestep sampled from $\{0.25, 0.50, 0.75, 1.0\}$. We supervise the model on fully-resampled denoising trajectories with anchor dropout p_{dropout}=0.5, as described in Section 3.

2T-sequence dense mask (TF). Let T be the training-window length. We concatenate T clean and T noisy frames along the temporal axis and share RoPE positions $[0, \ldots, T-1, 0, \ldots, T-1]$. The boolean attention mask $M \in \{0,1\}^{2T \times 2T}$ is:

$$M_{q,kv}=1 \text{ iff } \begin{cases} q < T,\ kv \leq q & \text{(clean half causal)}, \\ q=T+i,\ kv < i & \text{(noisy } n_i \text{ sees strict-past clean)}, \\ q=T+i,\ kv=q & \text{(noisy self-attention)}. \end{cases}$$

The mask blocks noisy-to-noisy attention, ensuring that each noisy target is predicted without conditioning on other noisy targets under the dense layout. This yields the same per-target conditioning structure as the naive T-pass TF objective, up to a constant factor in the loss, while requiring only a single forward pass per step.

Training. We use AdamW with a constant learning rate of 2×10^{-4}, $\beta = (0.9, 0.999)$, weight decay 10^{-4}, and global gradient-norm clipping at 1.0. We train for 100k steps with batch size 16.

Training Cost Comparison. We compare the per-step training cost of each method in our implementation. All models have 9.2M parameters and are trained on a single NVIDIA H100. "Fwd/step" denotes the number of model forward passes required per optimizer step, including no-gradient forwards used for autoregressive resampling.

Method	Fwd/step	sec/step	step/s
Diffusion Forcing	1	86 ms	11.6
STEP FORCING	2	122 ms	8.2
Teacher Forcing	1	180 ms	5.5
Resampling Forcing	$T{+}1$ (16)	$\sim$1.3 s	0.77
Self Forcing (Huang et al., 2025b)	$\sim$344	2.4 s	0.42

[3]

Teacher Forcing uses a $2T$-length dense sequence, which increases temporal-attention cost relative to Diffusion Forcing. STEP FORCING uses two forward passes per training step: one no-gradient self-forward pass and one gradient update pass, both over a length-T sequence. Resampling Forcing performs an autoregressive resampling loop with T sequential no-gradient forwards, one per target frame, before the supervised update. For Self Forcing, we follow the original DMD distillation procedure, which requires multi-step autoregressive rollouts together with repeated score and critic evaluations; this leads to a higher per-step cost in the BAIR setting.

Implementation note. The wall-clock numbers above reflect our straightforward implementation of each method. In particular, our autoregressive-style baselines use dense causal-mask forwards without key–value cache amortization. Thus, the reported costs should be interpreted as a practical comparison under a common implementation rather than as the most optimized possible runtime for each algorithm. Further engineering, such as caching key–value tensors across autoregressive resampling steps, could reduce the absolute cost of RF- or SF-style methods. However, these methods still require sequential autoregressive resampling during training, whereas STEP FORCING only adds a single no-gradient self-forward pass. Therefore, we expect the qualitative cost ordering to remain similar:

$$\text{DF} < \text{STEP FORCING} < \text{TF} \ll \text{RF} \ll \text{SF}.$$

Overall, this comparison highlights that autoregressive resampling-based training objectives can be substantially more expensive than objectives that avoid long sequential resampling loops.

Inference. At evaluation time, we autoregressively generate 30 frames starting from $K{=}1$ clean conditioning frame and the ground-truth action sequence. For simplify the implementation we do not use KV-caching.

Evaluation protocol. We evaluate on 64 held-out test clips (BAIR test split, deterministic seed). Each rollout is compared frame-by-frame against the ground-truth clip with:

- **MSE**: pixel-space mean squared error on $[-1, 1]$.
- **SSIM** (Wang et al., 2004): window size 11, `pytorch_msssim` default; computed on rescaled $[0, 1]$ frames.
- **LPIPS** (Zhang et al., 2018): AlexNet backbone, computed directly on $[-1, 1]$ frames.

Full Results. Table 4 extends the main paper table with oracle teacher checkpoints (9M / 30M / 55M) and the FVD column. The teachers are evaluated under their own training protocol, using in-painting at $T{=}15$ and $S{=}16$. Therefore, they are not directly comparable to the causal student rows, but serve as an oracle ceiling reference.

C. Stochastic Moving MNIST Diagnostic

C.1. Dataset

SMMNIST clips are generated on the fly. Two digits, sampled from the MNIST training set (test set at evaluation) and resized to $28{\times}28$, are placed at uniformly random positions in a $64{\times}64$ frame. Initial velocities are drawn from $\mathcal{U}[-3, 3]^2$

[3] *sec/step* is wall-clock time measured on a single H100; *Fwd/step* is an analytical count of model forward passes per optimizer step, including no-gradient forwards. DF uses one gradient forward. STEP FORCING uses two forwards: one no-gradient self-forward and one gradient forward. TF uses one forward over the $2T$-packed sequence. RF uses $T{+}1$ forwards: T no-gradient resampling forwards plus one supervised forward. Since our training window length is $T{=}15$, this gives 16 forwards per optimizer step. For SF (DMD), with a 5:1 critic-to-generator update ratio, each optimizer step uses six autoregressive rollouts of $F \cdot S$ generator forwards plus score/critic evaluations, giving $6 \cdot 56 + 3 + 5 = 344$ at $F{=}14$ and $S{=}4$.

Table 4. Full BAIR high-motion (top-64) evaluation. Causal students autoregressively roll out 29 frames from one conditioning frame at S=4, while teachers use in-painting at T=15 and S=16. Frame 0 is excluded from ID metrics.

Method	Fwd/step	S	In-distribution (1–14)			Out-of-distribution (15–29)			FVD ↓
			MSE ↓	SSIM ↑	LPIPS ↓	MSE ↓	SSIM ↑	LPIPS ↓	
Teachers (oracle reference)									
Teacher 9M	1	16	0.0658	0.7820	0.0583	—	—	—	20.42
Teacher 30M	1	16	0.0610	0.8023	0.0501	—	—	—	15.98
Teacher 55M	1	16	**0.0587**	**0.8069**	**0.0489**	—	—	—	**14.46**
Causal students									
Teacher Forcing (Zhou et al., 2025a)	1	8	0.0620	0.7942	0.0554	0.1185	0.7118	0.1058	**17.70**
Diffusion Forcing (Chen et al., 2024a)	1	8	0.0758	0.7657	0.0690	0.1113	0.6861	0.1117	22.47
Resampling Forcing (Guo et al., 2025b)	T+1	8	0.0618	0.7891	0.0572	0.1265	0.6996	0.1082	19.15
Self Forcing (Huang et al., 2025b)	$\sim$344[†]	4	0.0588	0.7929	0.0548	0.1246	0.6953	0.1083	21.31
Step Forcing (ours)	**2**	4	**0.0512**	**0.8063**	**0.0525**	**0.0778**	**0.7374**	**0.0768**	19.26
Ablations of Step Forcing									
– continuous-t	2	4	0.0660	0.7895	0.0670	0.0922	0.7318	0.0959	18.43
– continuous-t	2	8	0.0700	0.7859	0.0651	0.0966	0.7274	0.0952	18.03
– continuous-t	2	16	0.0720	0.7834	0.0640	0.0998	0.7226	0.0951	18.31
– no self-forward (p=1)	1	4	0.1329	0.7001	0.1081	0.1587	0.6298	0.1401	29.91
– no anchor (p=0)	2	4	0.1869	0.5998	0.1933	0.4141	0.3745	0.3334	49.25
– DF (discrete)	1	4	0.0911	0.7002	0.0913	0.1583	0.5797	0.1451	43.63

[†]Self Forcing's $\sim$344 forwards per training step decomposes as one generator update (56 autoregressive rollout forwards + 3 score evaluations) plus five critic updates, each consisting of 56 autoregressive rollout forwards + 1 score evaluation. This gives $59 + 5 \times 57 = 344$.

(px/frame). At each step each digit moves ballistically; when it hits a wall the position is reflected and a fresh velocity is resampled from the same uniform distribution, matching the stochastic Moving MNIST protocol of Denton & Fergus (2018). Pixels are scaled to $[-1, 1]$.

C.2. Architecture and Optimization

All methods share an identical causal video DiT (Table 5): frame-level patch tokenization, spatial self-attention within each frame, *causal* temporal attention across frames, and AdaLN conditioning on the per-frame noise level. The model has 9M parameters. We use AdamW with learning rate 2×10^{-4}, weight decay 10^{-4}, gradient clipping at 1.0, and batch size 16. Resampling Forcing is the exception: its T-pass training (§C.5) at batch 16 exhausts H100 80GB memory, so we use batch 4 for the same 50,000 optimizer steps. RF therefore sees fewer total samples (200k vs 800k); we observed empirically that this does not disadvantage the method—loss converges to a comparable level by step 30k—and the train-cost column in Table 6 is independent of batch size by construction.

C.3. Per-Method Training Schedules

All methods are trained for a 50k-step total budget; the only difference is the noise-level schedule and (for post-trained methods) the initialization checkpoint.

Diffusion Forcing (DF). 50k steps from scratch with per-frame independent continuous noise levels $t \sim \mathcal{U}(0.001, 0.999)$ and velocity-prediction loss (Chen et al., 2024a).[4]

Teacher Forcing (TF). 50k steps from scratch. Per sample, one target frame $i \in [1, T_{\text{train}})$ and one continuous noise level t_i are drawn; past frames are presented at t=0 (clean ground truth) and only frame i is supervised. TF serves as the simplest baseline that exhibits exposure bias under autoregressive rollouts at inference.

Resampling Forcing (RF). 50k steps from scratch with the paper-faithful loss of Guo et al. (2025b). Each step (i) samples

[4]We adopt the Diffusion Forcing paradigm under the linear (rectified flow) parameterization shared by all methods in this study; the DF paper uses a VP/DDPM schedule, which is equivalent to continuous t in the limit of fine discretization.

a shared resampling timestep $t_s \sim \text{LogitNormal}_{s=0.6}$ (paper Eq. 6–7) and runs T_{train} sequential one-step Euler updates under `no_grad` to produce a *self-rollout history* $\tilde{x}_{0:T-1}$ at $t{=}0$ (frame i conditioned on the already resampled $\tilde{x}_{<i}$, paper Eq. 5); (ii) for every target frame $i \in [0, T_{\text{train}})$, builds the input pattern $x_{<i}{=}\tilde{x}_{<i}$ at $t{=}0$ and $x_i{=}$noised $x_{\text{clean},i}$ at $t_i \sim \mathcal{U}(0.001, 0.999)$, predicts the velocity, and accumulates the per-target v-loss; (iii) averages over targets and takes one optimizer step. The paper executes step (ii) in a single forward pass via sparse causal attention; our toy uses T separate forwards, which is algorithm-equivalent but adds an implementation-specific constant-factor overhead (see §C.5).

Self Forcing (SF). 40k TF pretraining + 10k SF post-training (50k total). Let i^* denote the sampled target frame index and s^* the target noise level. Each step (i) rolls out frames $[K_{\text{init}}, i^*)$ under `no_grad` via S_{train} Euler steps per frame, (ii) partially denoises the target frame i^* to noise level s^*, and (iii) takes one gradient step at (i^*, s^*) with x_0-prediction loss. We set $K_{\text{init}}{=}4$ (clean conditioning frames), $H_{\text{train}}{=}12$ (training rollout horizon), and $S_{\text{train}}{=}4$ (Euler steps per frame at training time), matching the inference noise schedule.

Step Forcing (ours). 50k steps from scratch with the two-stage update of Section 3: discrete $S{=}4$ rectified-flow schedule, self-forwarded prior with stop-gradient, zero-step prior anchor with dropout probability $p_{\text{dropout}}{=}0.5$.

C.4. Inference and Evaluation

We generate $T_{\text{eval}}{=}32$ frames autoregressively, conditioned on $K_{\text{init}}{=}4$ ground-truth frames. Each new frame is denoised from pure noise ($t{=}1$) to $t{=}0$ in S_{inf} Euler steps along the linear interpolation schedule. Continuous-time methods (DF, TF, RF) use $S_{\text{inf}}{=}16$, the inference budget at which DF and TF yield indistinguishable metrics; few-step methods (SF, Step Forcing) use $S_{\text{inf}}{=}4$ matching their training schedule. We report SSIM and LPIPS, separately aggregated over in-distribution frames ($0 \le t < T_{\text{train}}$) and out-of-horizon frames ($T_{\text{train}} \le t \le T_{\text{train}}{+}8$, i.e. a fixed 8-frame OOD horizon), averaged over 32 evaluation clips drawn with seeds disjoint from training. We bound the OOD horizon at $T_{\text{train}}{+}8$.

C.5. Training Cost Calculation

Table 6 reports per-method *train cost* as the expected number of model forward passes per optimizer step, normalized to Teacher Forcing. The values are computed analytically from each loss function's structure (rather than from wall-clock timings, which depend on hardware and noise) and verified against the code paths in our implementation:

- **Teacher Forcing / Diffusion Forcing**: 1 forward per step. A single (B, T, C, H, W) noisy input is mapped to a velocity prediction; the loss on the predicted target produces one backward pass.
- **Step Forcing (ours)**: 2 forwards per step. The first is a stop-gradient self-forward of x_k to construct the prior $\hat{x}_{k-1}$; the second is the gradient-bearing forward of $\hat{x}_{k-1}$ used in the clean-prediction loss.
- **Resampling Forcing**: $T + 1 = 17$ forwards per step (paper-claimed). The history-conditioned resampling phase requires T sequential one-step Euler updates under `no_grad` (one per frame, because frame i's resample is conditioned on the already-resampled $\tilde{x}_{<i}$), followed by 1 gradient-bearing prediction forward in which all T targets are supervised in parallel via sparse causal attention (`flex_attention`). Our dense-attention toy DiT lacks this kernel and instead executes the prediction phase as T separate dense forward passes, raising actual training compute to $2T = 32$ forwards per step. We report the paper-claimed algorithmic cost in Table 6; this choice favours the baseline.
- **Self Forcing**: $\mathbb{E}[(i^* - K_{\text{init}}) \cdot S_{\text{train}}] + \mathbb{E}[S_{\text{train}} - s^*] + 1$. With $K_{\text{init}} = 4$, $H_{\text{train}} = 12$, $S_{\text{train}} = 4$ (so $i^* \sim \mathcal{U}\{4, \dots, 15\}$, $s^* \sim \mathcal{U}\{1, \dots, 4\}$): $5.5 \cdot 4 + 1.5 + 1 = 24.5$. The expected 22 context-build forwards (autoregressive rollout of frames $K_{\text{init}} \to i^* - 1$ at S_{train} Euler steps each) dominate, plus 1.5 pre-target denoising forwards on average and 1 gradient pass.

Self Forcing in Table 6 uses a *single-target* adaptation, in which one gradient signal is produced per training step. The original formulation (Huang et al., 2025b) additionally supervises multiple target frames per iteration, sharing the rollout to amortize cost across multiple gradient signals; this would increase *forwards/step* but reduce *forwards-per-gradient-target*. We adopt the single-target form because it is the simplest faithful rendering of the algorithm in a one-stream causal DiT and isolates the per-step compute attributable to the train–test alignment objective. Resampling Forcing, in contrast, is implemented in the *all-target* form (one gradient signal per frame per step, averaged), matching the paper exactly except for the dense-vs.-sparse-attention kernel choice discussed above.

C.6. Hyperparameter Summary

Table 5 summarizes the hyperparameters used in the SMMNIST diagnostic.

Table 5. Hyperparameters for the SMMNIST diagnostic.

Data	
Frame resolution	64×64
Digit resolution	28×28
Digits per clip	2
Initial velocity	$\mathcal{U}[-3, 3]^2$ px/frame
Wall behavior	reflection + velocity resample
T_{train} / T_{eval} / K_{init}	16 / 24 / 4
Architecture (causal video DiT)	
Patch size	4
Embedding dim	256
Depth	6
Attention heads	8
Parameter count	9.08M
Optimization (all methods)	
Optimizer	AdamW
Learning rate	2×10^{-4}
Weight decay	10^{-4}
Gradient clip (ℓ_2)	1.0
Batch size	16 (RF: 4, multi-pass OOM otherwise)
Total training steps	50,000
Diffusion / forcing schedule	
Parameterization	rectified flow (v-prediction)
Forward process	$x_t = (1-t)\, x_0 + t\, \epsilon$
Continuous t (DF, TF, RF)	$\mathcal{U}(0.001, 0.999)$
Discrete S (SF, Step Forcing)	4
Step Forcing p_{dropout}	0.5
Self Forcing $K_{\text{init}}, H_{\text{train}}, S_{\text{train}}$	$4, 12, 4$
RF resampling timestep t_s	$\text{LogitNormal}_{s=0.6}$
RF prediction timestep t_i	$\mathcal{U}(0.001, 0.999)$
Inference / evaluation	
S_{inf}, continuous-time methods	16
S_{inf}, few-step methods	4
Sampler	Euler
Number of evaluation clips	32
Metrics	SSIM, LPIPS

C.7. Experimental Results

Setup. We use Stochastic Moving MNIST (Srivastava et al., 2015; Denton & Fergus, 2018) as a controlled toy diagnostic. All methods share the same 9.1M-parameter causal video DiT, trained from scratch for 50k steps on 16-frame windows. We compare STEP FORCING against Teacher Forcing, Diffusion Forcing (Chen et al., 2024a), Self Forcing (Huang et al., 2025b), and Resampling Forcing (Guo et al., 2025b). Since this toy setup does not pretrain a separate teacher, we replace Self Forcing's distillation loss with supervised flow-matching against ground truth; this differs from the original formulation and should be read as a distillation-free variant.

Results. STEP FORCING attains the best in-distribution and out-of-horizon SSIM and LPIPS at $4\times$ fewer denoising steps than the baselines, while using substantially fewer training-time forward passes (Appendix C). It improves over Diffusion Forcing in few-step inference and matches Resampling Forcing in fidelity at lower training cost. Self Forcing relies on bidirectional teacher distillation; without it, its few-step generation degrades, while STEP FORCING remains stable (Figure 4). This does not establish absolute algorithmic superiority but demonstrates that STEP FORCING enables stable few-step training without distillation.

Table 6. Quantitative comparison on Stochastic Moving MNIST. Rows are grouped by inference budget ($S_{\text{inf}} = 16$ vs. 4). *Train cost* is the expected number of model forward passes per optimizer step, normalized to Teacher Forcing, as *claimed by each method's paper* (assuming sparse causal attention for RF, sparse-rollout SF, etc.); derivation in Appendix C.5. Our toy DiT lacks these sparse-attention kernels and thus pays a multi-pass constant-factor overhead at training time, but the algorithmic cost reported here is the implementation-independent quantity. ID denotes in-distribution frames ($t < T_{\text{train}} = 16$); OOD denotes out-of-horizon frames ($16 \le t \le 24$, i.e. up to $T_{\text{train}} + 8$). **Bold**: best per column.

Method	Train cost	ID-SSIM ↑	ID-LPIPS ↓	OOD-SSIM ↑	OOD-LPIPS ↓
Inference: 16 denoising steps					
Teacher Forcing	1.0×	0.719	0.118	0.498	0.307
Diffusion Forcing	1.0×	0.724	0.124	0.535	0.317
Resampling Forcing	17.0×	0.723	0.116	**0.577**	0.297
Inference: 4 denoising steps					
Resampling Forcing	17.0×	0.713	0.112	0.576	0.309
Self Forcing	24.5×	0.647	0.291	0.297	0.572
Step Forcing (Ours)	2.0×	**0.771**	**0.102**	0.551	**0.271**

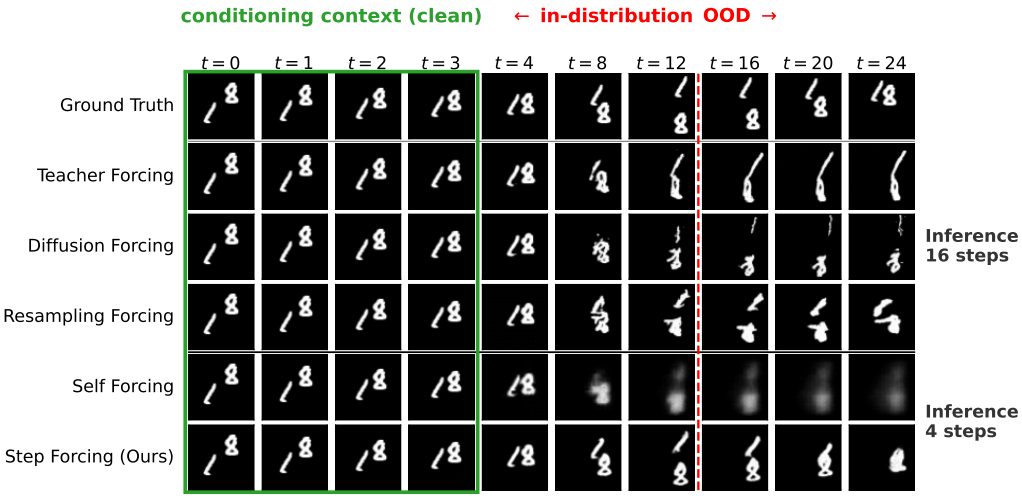


Figure 4. Qualitative comparison on Stochastic Moving MNIST (Denton & Fergus, 2018). Each model is conditioned on $K_{\text{init}} = 4$ clean frames (green box) and rolls out autoregressively to $t = 24$; the dashed red line at $t = T_{\text{train}} = 16$ marks the in-distribution / OOD boundary. Step Forcing remains stable across both regimes, whereas the baselines degrade either inside or beyond the training horizon.

D. WORLDARENA Details

This appendix details the VLM scoring procedure used by WORLDARENA (Appendices D.1, D.2, and D.3).

D.1. VLM-as-a-Judge: Rubric and Strategy

Human scoring requires human-in-the-loop evaluation, limiting scalability. To maximize scalability, we replace human annotators with a pretrained VLM as a judge, which generates a score when conditioned on rollouts produced by the video world model. However, since world model rollouts inevitably contain model-induced artifacts, naive automatic scoring may compromise reliability. To maintain reliable evaluation, we introduce two design choices.

(1) Fine-grained evaluation rubric. We design a six-level rubric (Figure 5) that assigns scores based on both task progress and the stage at which world-model errors occur. Concretely:

- **5 (Success)**: Task goal is achieved.
- **4 (Near success, world-model fail during interaction)**: The robot reaches and interacts with the target.
- **3 (World-model fail upon interaction)**: World-model failure after target contact.
- **2 (Attempt without interaction)**: The robot approaches the target object.

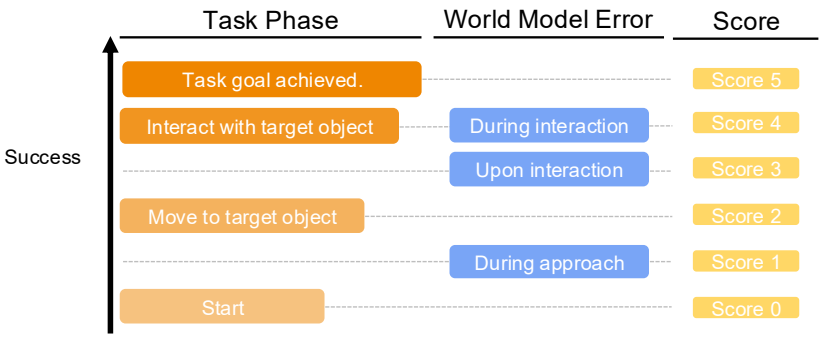


Figure 5. **Design principles of the evaluation rubric.** We design the rubric such that scores primarily reflect task progress. In addition, we assign different penalties depending on when world-model errors occur, so failures that arise earlier in the rollout receive lower scores than those that happen after substantial task progress.

- **1 (World-model fail before interaction)**: World-model failure while approaching the target.
- **0 (Failure)**: All other cases.

(2) Excluding the wrist view from success judgments. World-model artifacts predominantly appear in the wrist view, since it changes more dynamically than the fixed views and must remain aligned with them. We therefore exclude the wrist view from success judgments and use it only to detect world-model failures. The fixed views are used to assess task progress.

D.2. VLM Evaluation Procedure

For VLM-based evaluation, we use GPT-4o as the judge. Each rollout is sampled at one frame per second and split into two segments. We score the segments independently using GPT-4o, providing 15 sampled frames per segment along with the rubric and the task description. The final per-rollout score is the maximum across the two segments. To reduce evaluation cost, we omit scoring the second segment when the policy is already judged successful in the first segment.

D.3. VLM Prompt Templates

Table 7 presents the rubric used to compute the score. Tables 9 and 8 present the prompts used for the vision-language model evaluator.

D.4. Ablation on WorldArena Evaluation Strategy

We ablate the two design choices of WorldArena's VLM-based evaluation (Section D.1; Table 10). Replacing the rubric with a binary success-rate baseline discards the fine-grained scoring of task progress and world-model errors. Including the wrist view in success judgments—rather than restricting it to world-model failure detection—incurs a larger drop, indicating that the wrist view's higher artifact density biases the judge when used for success scoring.

E. Implementation Details

E.1. Model Architecture

We build on the Wan2.1-T2V-1.3B architecture (Wan et al., 2025) with only minimal modifications. First, we introduce a frame-level causal attention mask, where tokens within the same frame attend bidirectionally, while tokens across frames are restricted to attending only to previous frames. Second, we replace the text embedding layer with an action embedding layer implemented as a two-layer MLP. The resulting action embeddings are consumed by the cross-attention layers in a frame-wise manner. Specifically, because Wan represents every four video frames as a single latent token, we embed action chunks of size four and apply cross-attention between each latent token and the corresponding action embedding. Third, we extend the time embedding mechanism to support frame-wise diffusion timesteps under diffusion forcing. Specifically, each frame is assigned its own diffusion timestep, and the resulting frame-wise embeddings are projected to the adaLN modulation parameters used in each transformer block. These frame-level modulation parameters are then repeated across

Evaluation Rubric Used in the VLM Prompt

Evaluation Criteria

Target object: The object specified in the task instruction.

Score 5: Success

Definition: The policy successfully completes the task as instructed.

- The target object is correctly manipulated to reach the intended goal state.
- The outcome is clearly successful and verifiable.
- Success should not be judged from a single frame.
- Evaluators must check adjacent frames to confirm that the goal state is genuinely achieved and stably maintained.

To determine success, the robot must be clearly observed interacting with the target object.

Score 4: Near success OR World Model failure during target-object contact/interaction

Definition: The policy nearly completes the task but does not fully succeed, OR a world model failure occurs after the policy has made contact with (or begun manipulating) the target object.

- The policy correctly reaches and interacts with the target object, but the final goal is not achieved.

Score 3: World Model failure upon target-object contact/interaction

Definition: A world model failure occurs at the moment the policy begins interacting with the target object.

- The world model fails immediately when the robot makes contact with the target object or begins manipulation.

Score 2: Attempted execution near target

Definition: The policy attempts to solve the task and moves near the target object.

- The robot moves near the target object with task-directed behavior.

Score 1: World Model failure during approach before target-object interaction

Definition: A world model failure occurs while moving toward the target object.

- The robot is near the target object in the fixed view, but the target object disappears or is not visible in the wrist view.
- Evaluation is unreliable due to this fixed-view vs. wrist-view inconsistency.

Score 0: Failure (All other cases)

Definition: Any case not covered by Scores 5, 4, 3, 2, or 1.

- The policy exhibits movements irrelevant to the task instruction.
- The robot moves independently of the target object in the fixed view.
- The robot leaves the scene in the fixed view.
- World model failures that occur while the robot is not meaningfully interacting with or approaching the target object.

Table 7. Evaluation rubric used in the VLM prompt.

all spatial tokens within the corresponding frame. Because our modifications to the original DiT architecture are minimal, the proposed design can be easily transferred to other DiT-based video generation models.

E.2. Training Details

Detailed training configurations are provided in Table 11 and Table 12.

E.3. Inference and Rollout Protocol

We describe the inference-time rollout protocol used for closed-loop policy evaluation, including denoising steps, frame generation procedure, action execution loop, and segment-wise evaluation strategy.

F. Additional Open-loop Results

F.1. Qualitative Trajectories

Figure 6 presents a qualitative comparison of open-loop video generation on RoboArena for Ground Truth, Ctrl-World, Diffusion Forcing, and STEP FORCING. Each method generates a 301-frame rollout, from which we show 11 frames sampled uniformly at 30-frame intervals. STEP FORCING best preserves action-conditioned motion and view consistency across the fixed and wrist cameras. In particular, it more accurately tracks the robot behavior in the wrist view, which is often difficult to model. These results are notable because STEP FORCING achieves this behavior with only 4-step denoising,

Prompt Template Used for VLM Evaluation (Wrist View Used Only for Error Identification)

You are evaluating a robot policy rollout video. The frames below are extracted from the video in chronological order (first frame to last frame).

Task Instruction: {instruction}

{rubric}

Multiview frames are provided, corresponding to a robot policy rollout in the world model. Each frame consists of a fixed left view (upper right view), a fixed right view (upper left view), and a wrist view (bottom left).

Evaluation Method:

1. Use the fixed views (the two upper views) as the primary reference.
Determine whether the robot interacts with the target object based on the fixed views.
2. Use the wrist view only to identify world model errors (e.g., object disappearance or inconsistency).
Do NOT use the wrist view to infer task success.

Provide:

1. A score (0 to {max_score}) according to the rubric above
2. A brief explanation of your reasoning

Table 8. Prompt template used when the wrist view is used only to identify world model errors, but not to determine task success.

Prompt Template Used for VLM Evaluation (Wrist View Used for Full Evaluation)

You are evaluating a robot policy rollout video. The frames below are extracted from the video in chronological order (first frame to last frame).

Task Instruction: {instruction}

{rubric}

Multiview frames are provided, corresponding to a robot policy rollout in the world model. Each frame consists of a fixed left view (upper right view), a fixed right view (upper left view), and a wrist view (bottom left).

Evaluation Method:

1. Use ALL three views (fixed left, fixed right, and wrist) for evaluation.
All views contribute equally to determining task success and robot behavior.

Provide:

1. A score (0 to {max_score}) according to the rubric above
2. A brief explanation of your reasoning

Table 9. Prompt template used when the wrist view is used as a full evaluation signal.

while still producing more coherent long-horizon predictions than the compared baselines.

F.2. Speed–Performance Trade-off

In Figure 7, we present additional speed–quality comparisons with Ctrl-World (Guo et al., 2025a), PersistWorld (Bardhan et al., 2026), and Diffusion Forcing (Chen et al., 2024a). We vary the number of denoising steps at inference time and report wrist-view FVD, wrist-view MSE, and throughput (FPS), measured on an NVIDIA H200 GPU.

G. Additional Closed-loop Results

G.1. Qualitative Results

We provide qualitative examples of world-model rollouts together with the corresponding VLM annotations for each case. As shown in Figure 8, these examples make it easier to interpret how the evaluator scores different rollout behaviors, including success, near success, partial progress, and irrelevant behavior. We further analyze both world-model failures and VLM evaluation failures in the following examples.

Table 10. Ablation on rank correlation with the RoboArena leaderboard (Spearman ρ). Numbers in parentheses indicate the drop relative to the full WORLDARENA score.

Method / Ablation	**Spearman ρ**
WORLDARENA score	0.970 (0.000)
w/ wrist-view success judgments	0.862 (-0.108)
Success-rate baseline	0.922 (-0.048)

Table 11. Training details for Diffusion Forcing.

Configuration	**Setting**
Training data	
Input resolution	368×640
Sequence length	45 frames
Optimization	
Optimizer	AdamW
Learning rate	1×10^{-5}
Betas	$(0.9, 0.99)$
ϵ	1×10^{-8}
LR schedule	Constant with 1k-step warmup
Gradient clipping	1.0
Batch size	8
Training steps	160k
Precision	Mixed precision (FP16)
EMA decay	0.9999
Diffusion objective	
Training formulation	Rectified flow
Prediction target	Velocity (v)
Timestep schedule	Cosine (shift $= 0.125$)
Loss weighting	Sigmoid (bias $= -1.0$)

G.2. Failure Case Analysis

World Model Failure Cases Figure 9 shows qualitative examples of world-model failures that are correctly identified by the VLM evaluator. We observe that such failures occur primarily during object interaction. Before contact, the generated scene is typically stable, but after the robot begins manipulating the object, the object may disintegrate, morph into unrealistic shapes, or become visually inconsistent. This suggests that contact-rich object dynamics remain a key limitation of the current world model. We expect these failures to be mitigated by scaling to larger-capacity world models and improving interaction modeling.

VLM Failure Cases Figure 10 shows qualitative examples of VLM evaluation failures. While the VLM evaluator generally captures the relative trends across policies, we find that it tends to assign more lenient scores than human judgment. In a manual comparison over approximately 100 sampled rollouts, the average VLM score was about one point higher than the corresponding human score. The examples in Figure 10 illustrate this behavior: GPT-4o occasionally assigns success or near-success scores to rollouts where the scene remains essentially unchanged, hallucinating task completion despite visual evidence to the contrary.

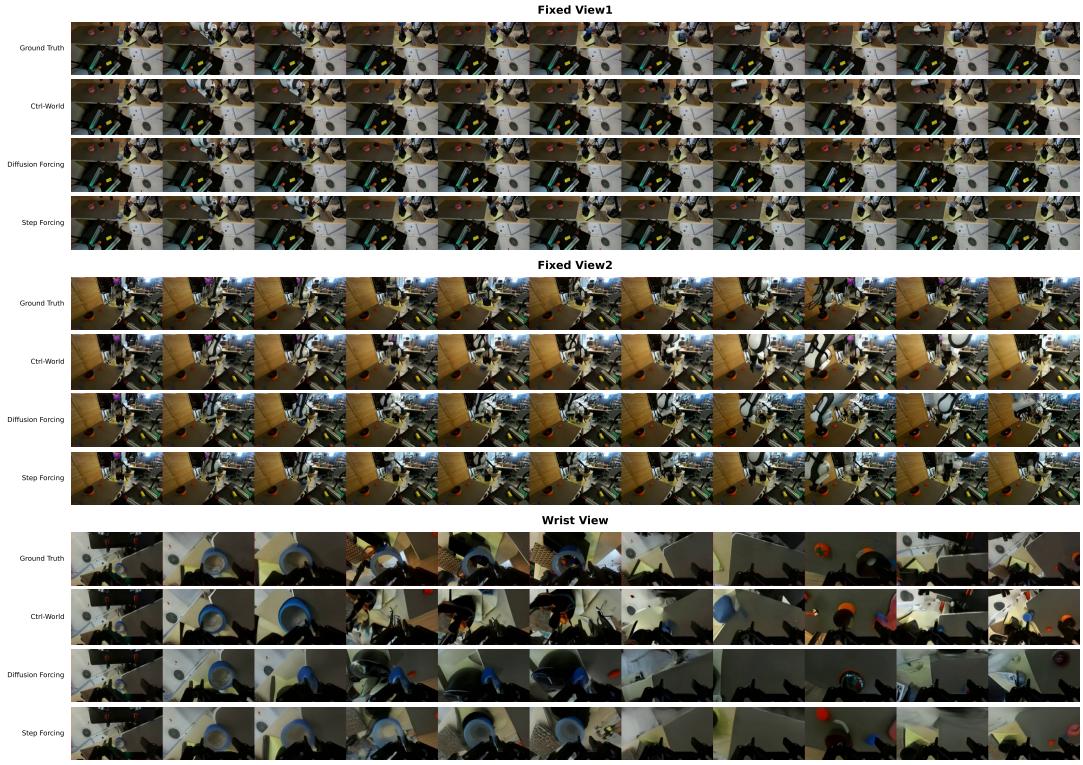
Extending WORLDARENA to Extreme Environments We investigate whether WORLDARENA could be applied to extreme environments (i.e., construction sites or airplane cabins), where real-world robot evaluation at scale is infeasible. From 175 initial observations from RoboArena snapshots, we generate eight synthetic environments (airplane cabin, spacecraft interior, operating room, nuclear facility, underwater station, disaster site, construction site, mine tunnel) utilizing an image editing model and retain 746 valid initial conditions after manual filtering. As shown in Figure 12, WORLDARENA can be robustly applied in extreme environments without physical access or complex asset generation for simulation. Moreover, the correlation between WORLDARENA on extreme environments and RoboArena leaderboard is still mainly retained ($r = 0.970$), implying that WORLDARENA could be utilized for reliable and safe policy evaluation before physical deployment to these environments.

Table 12. Training details for STEP FORCING.

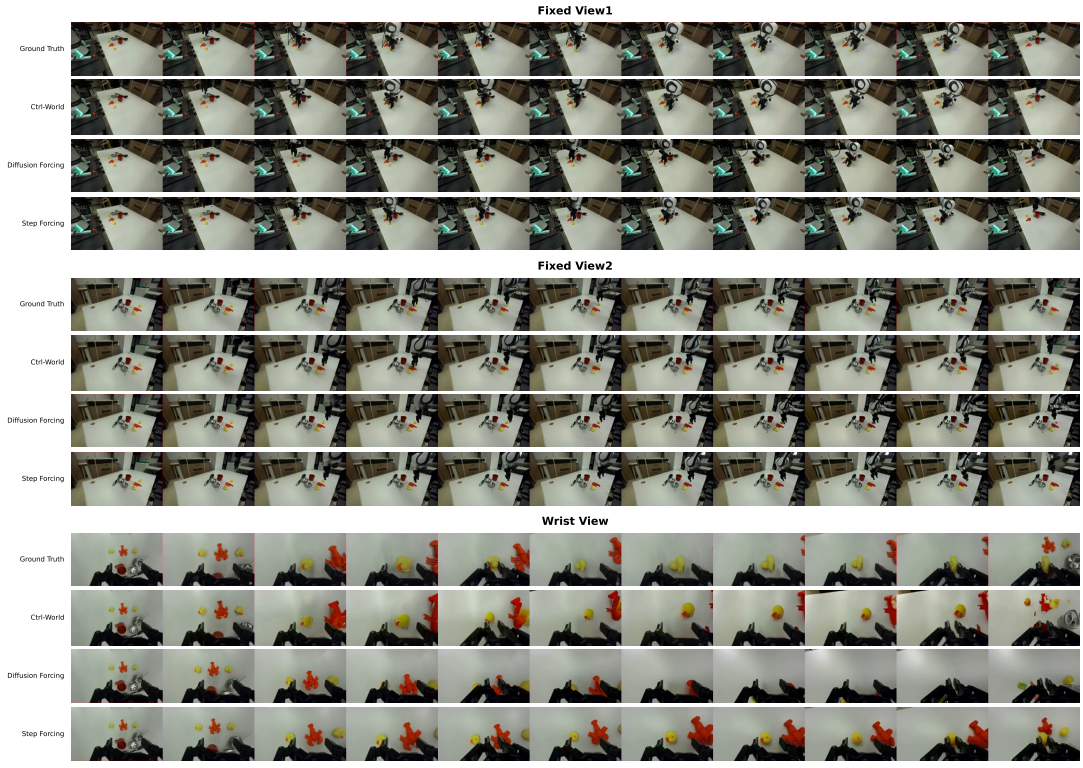
Configuration	Setting
Training data	
Input resolution	368×640
Sequence length	45 frames
Optimization	
Optimizer	AdamW
Learning rate	1×10^{-5}
Betas	$(0.9, 0.99)$
ϵ	1×10^{-8}
Weight decay	1×10^{-3}
LR schedule	Constant with 1k-step warmup
Gradient clipping	1.0
Batch size	8
Training steps	160k
Precision	Mixed precision (FP16)
EMA decay	0.9999
Diffusion objective	
Training formulation	Rectified flow
Prediction target	Velocity (v)
Timestep schedule	Cosine (shift $= 0.125$)
Loss weighting	Sigmoid (bias $= -1.0$)

Table 13. Segment-wise VLM evaluation examples.

Instruction	Segment	Score	Reasoning
`pick up the lemon and place it on the yellow plate`	0	5	The robot successfully picks up the lemon and places it on the yellow plate. The sequence of frames shows the robot interacting with the lemon, lifting it, and then placing it on the intended target, the yellow plate. The task is completed as instructed, and the outcome is clearly successful and verifiable.
`put the mustard in the red square`	0	0	The robot interacts with the green ball instead of the mustard, which is the target object specified in the task instruction. The mustard remains untouched throughout the frames, indicating a failure to complete the task as instructed.
	1	2	The robot moves near the mustard bottle and appears to attempt interaction, but it does not successfully manipulate the mustard into the red square. The frames show the robot's gripper approaching the mustard, but there is no clear evidence of successful manipulation or placement in the red square.



(a) Example trajectory 1.



(b) Example trajectory 2.

Figure 6. Open-loop video generation comparison. For each example, we show 11 frames sampled at equal intervals from a 301-frame rollout for Ground Truth, Ctrl-World, Diffusion Forcing, and STEP FORCING across three camera views: Fixed View 1, Fixed View 2, and Wrist View.

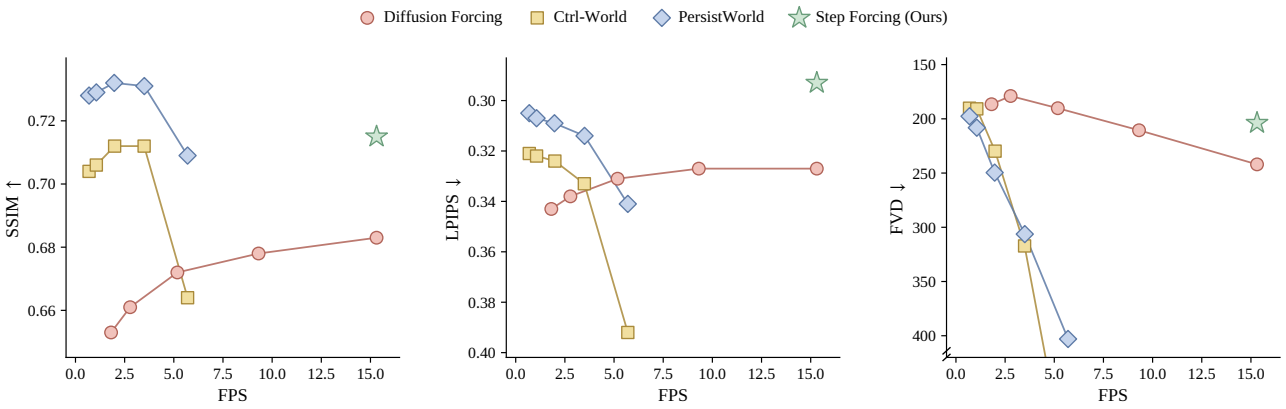
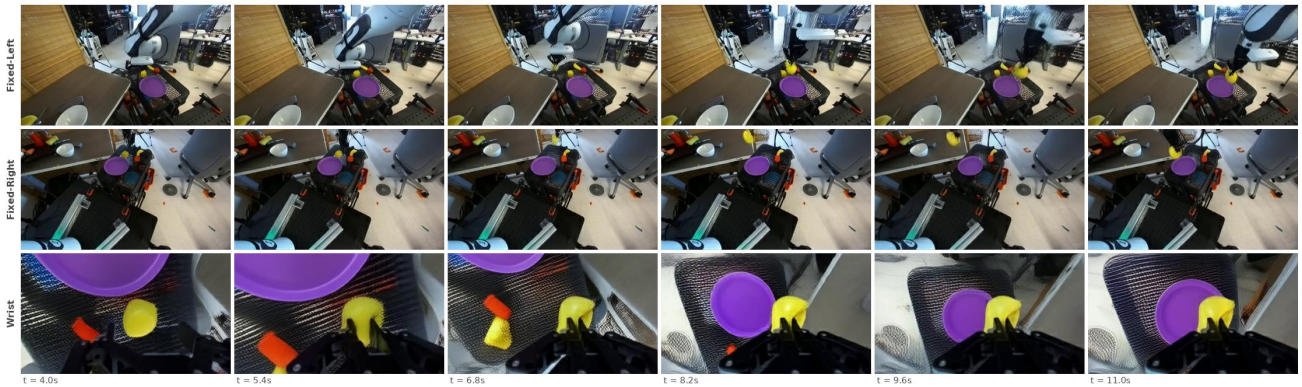


Figure 7. Long-horizon action-conditioned video generation on RoboArena. Quality (SSIM, LPIPS, FVD) vs. FPS. Baseline markers sweep denoising steps $\in \{50, 32, 16, 8, 4\}$. STEP FORCING achieves the best LPIPS at the highest FPS (15.3), and matches the baselines that run substantially slower on SSIM and FVD.

Instruction: "put the yellow block onto plate"

policy: paligemma_fast_droid

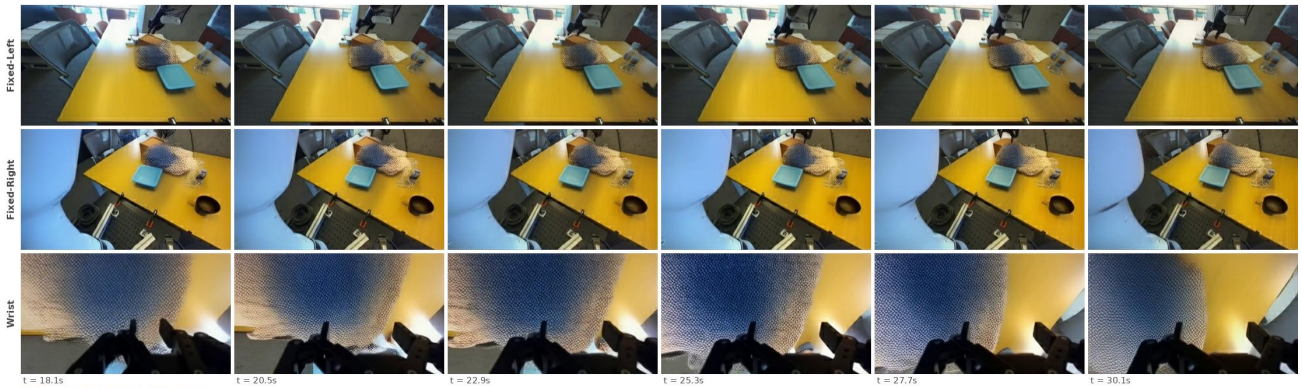


VLM score: 5 (Success)

VLM: "The robot successfully picks up the yellow block and places it onto the purple plate. The task is completed as instructed, with the yellow block clearly manipulated to reach the intended goal state. The outcome is stable and verifiable across multiple frames."

Instruction: "put the towel in the blue plate"

policy: paligemma_binning_droid

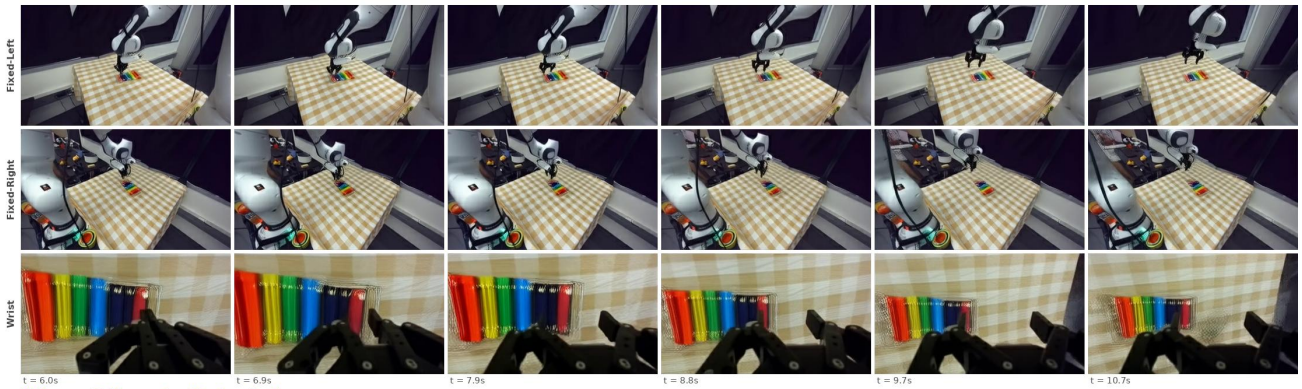


VLM score: 4 (Near success)

VLM: "The robot successfully interacts with the towel and moves it towards the blue plate. However, the towel does not end up inside the blue plate, indicating that the final goal is not fully achieved. The robot's actions are directed towards the task, but the outcome is a near success."

Instruction: "Touch the red part of the xylophone."

policy: pi05_droid

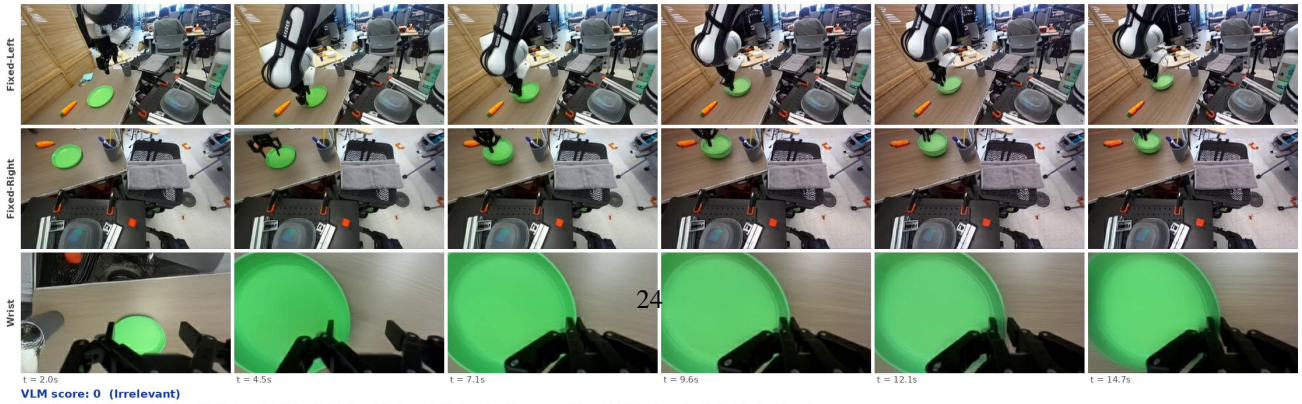


VLM score: 2 (Approach without contact)

VLM: "The robot arm approaches the xylophone and moves near the red part but does not make contact or manipulate it. The task-directed behavior is evident, but no interaction with the target object is observed."

Instruction: "fold the cloth in half"

policy: pi0_droid



VLM score: 0 (Irrelevant)

VLM: "The robot does not interact with the target object, which is the cloth. Instead, it interacts with a green plate, which is irrelevant to the task instruction of folding the cloth in half. The robot's actions are not directed towards the target object, resulting in a failure to complete the task."

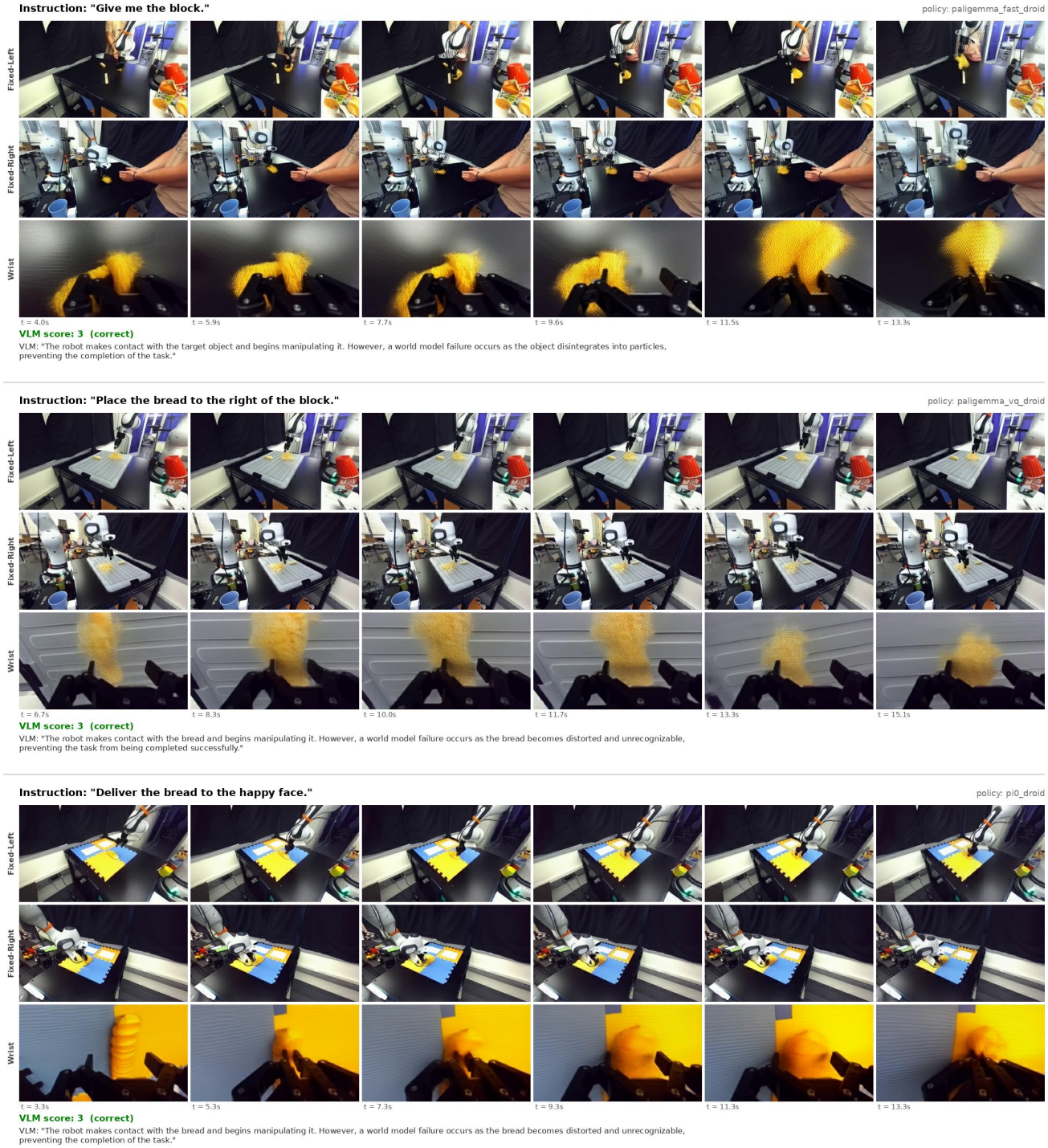


Figure 9. Qualitative examples of world-model failures correctly detected by the VLM evaluator. After the robot makes contact, the manipulated object disintegrates or morphs into unrecognizable artifacts; the task-process-aware rubric assigns its dedicated score (3) to these cases.

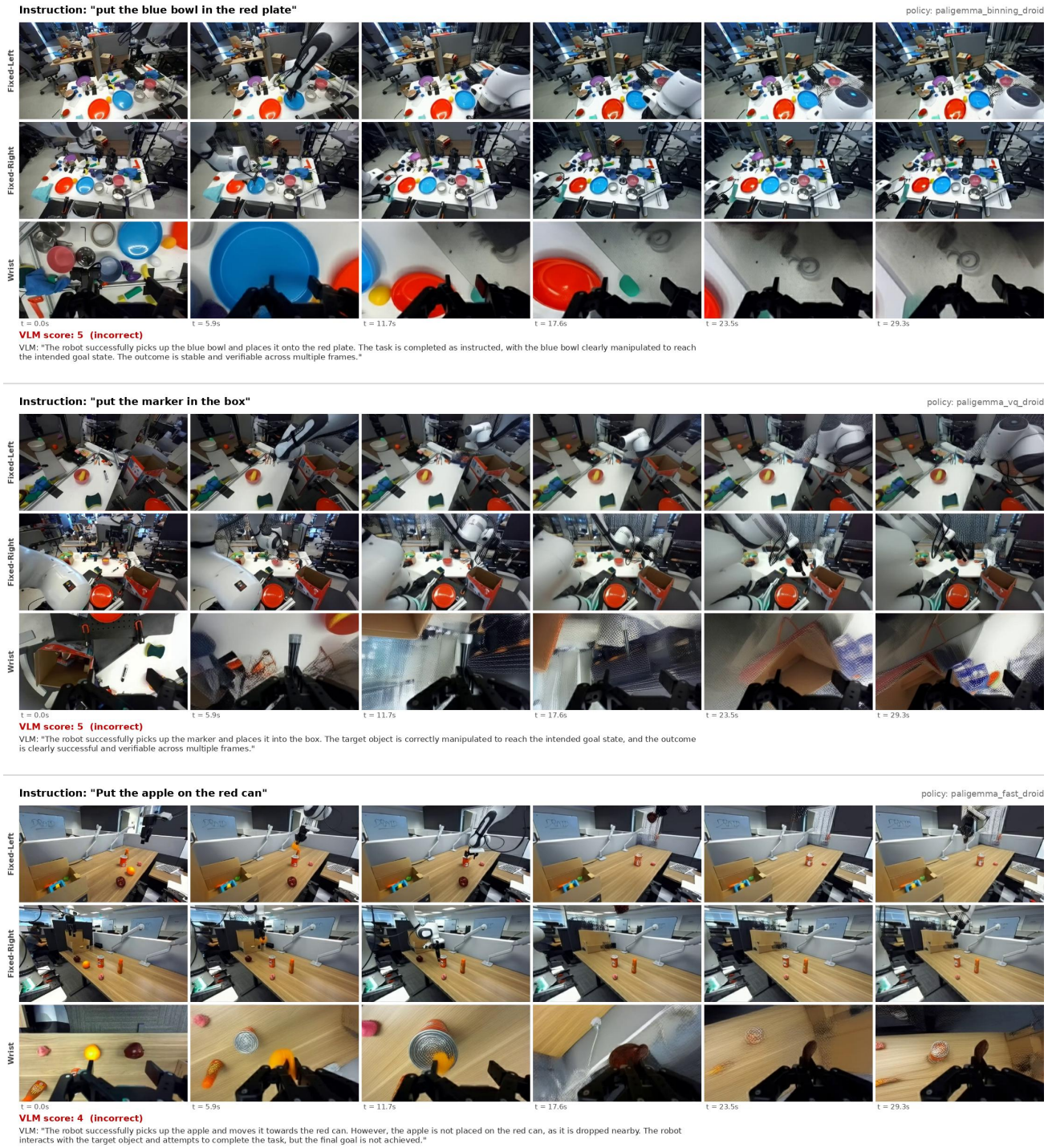


Figure 10. Qualitative examples of VLM evaluation failures. GPT-4o assigns success or near-success scores (5, 5, 4) to rollouts in which the scene remains essentially unchanged, hallucinating task completion that the video contradicts.

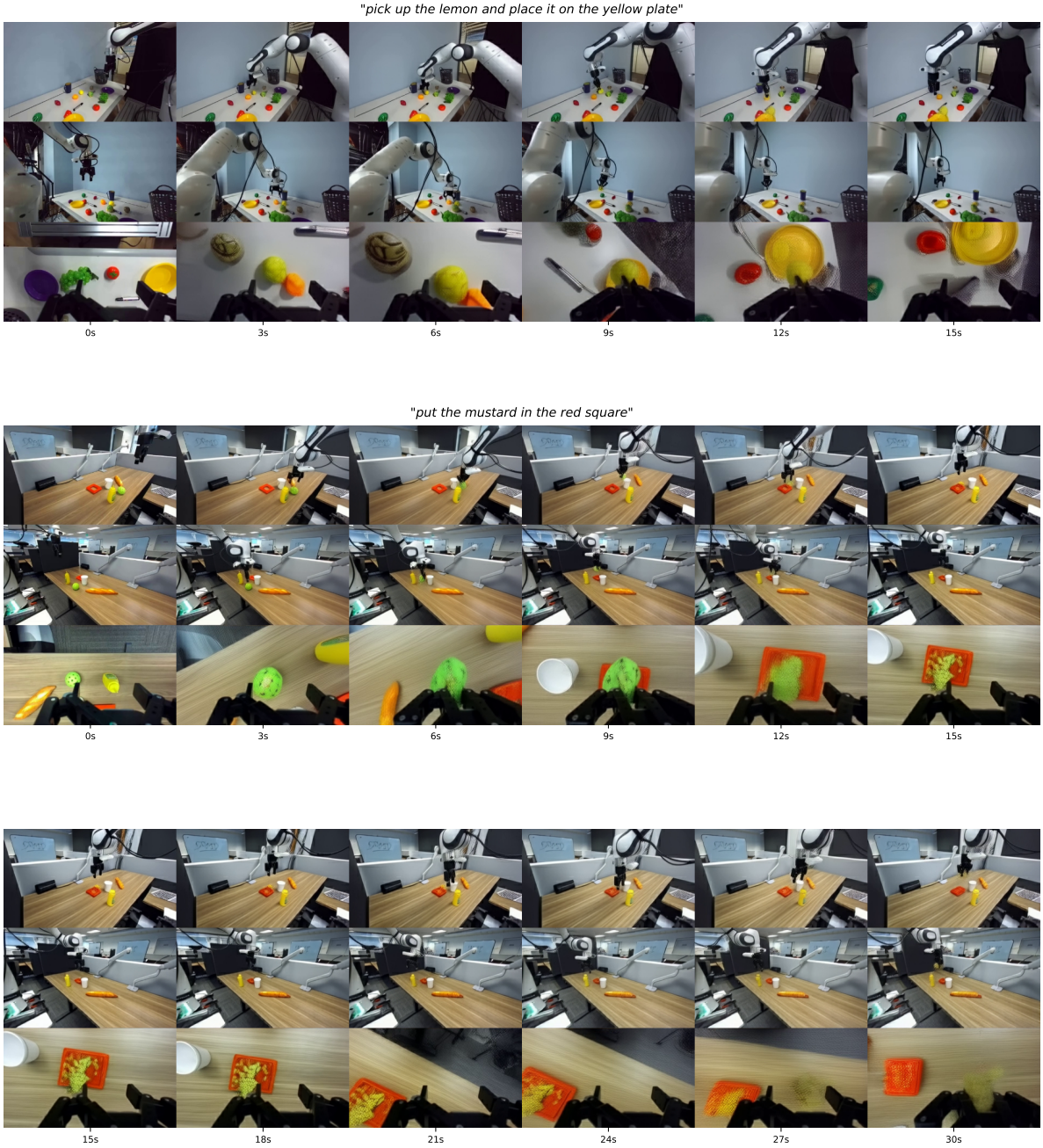


Figure 11. Example rollout of $\pi_{0.5}$ in WorldArena.

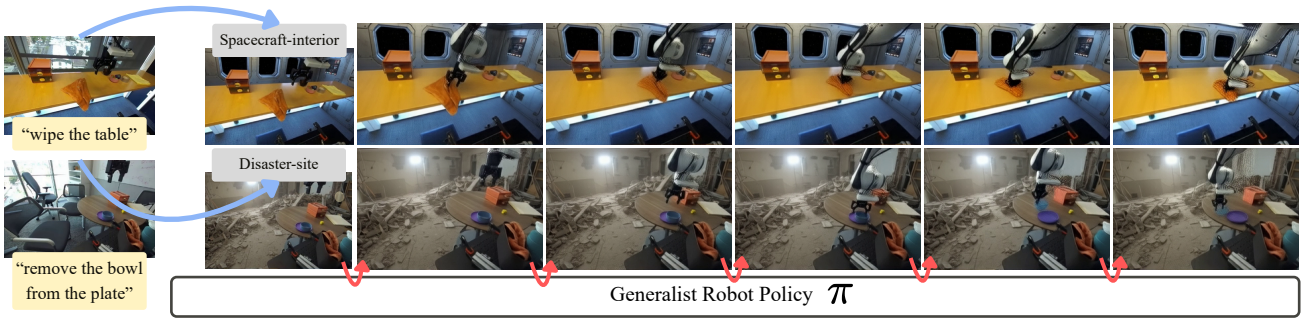


Figure 12. Qualitative examples of WORLDARENA in synthetic extreme environments. We transform real-world robot images into extreme environment scenes (e.g., spacecraft interiors, disaster sites), and conduct closed-loop policy evaluation.