# SWAM: SELF-SUPERVISED WORLD ACTION MODELING VIA ASYMMETRIC DENOISING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

We revisit denoising in world action models from a self-supervised representation learning perspective, shifting emphasis away from video generation. Rather than relying on costly, error-prone future-video rollouts, robot policies need robust visual features that support action prediction from current observations. We therefore introduce the Self-Supervised World Action Model (SWAM), which trains action-relevant robust visual features through asymmetric denoising. SWAM uses Denoising Autoencoding (DAE) on current observations with a noise schedule less aggressive than that of video generation, providing auxiliary self-supervision for the visual backbone. For future observations, Masked Autoencoding (MAE) motivates extreme-noise denoising that limits access to the target and enables single-pass visual inference. SWAM achieves success rates of 99.1% on LIBERO and 83.4% on LIBERO-Plus. On LIBERO-Plus, it exceeds Fast-WAM and Fast-WAM-Joint by 31.9 and 15.0 percentage points, respectively, with comparable inference-path latency to Fast-WAM and over 50% lower latency than Fast-WAM-Joint. These results support asymmetric denoising as an effective self-supervised objective for robot policy learning.

## 1 INTRODUCTION

Robot policies must act on current visual observations while anticipating how interactions may change the scene. Vision-language-action (VLA) policies commonly learn a direct observation-to-action mapping from demonstrations, often drawing on pretrained image-language models (Black et al., 2025; Kim et al., 2025a; Zitkovich et al., 2023; Kim et al., 2025b). World action models (WAMs) additionally learn visual dynamics alongside actions (Ye et al., 2026; Yuan et al., 2026; Zhu et al., 2025; Li et al., 2026). Future observations in the same demonstrations provide dense supervision for motion and scene changes (Li et al., 2025), while pretrained video backbones can supply spatiotemporal priors (Hu et al., 2025; Kim et al., 2026).

Existing WAMs differ in what they generate or retain from future prediction when producing actions. As shown in Figure 1, Joint-WAMs iteratively denoise future video and actions, providing an explicit imagined future but incurring repeated visual computation and exposure to prediction errors (Ye et al., 2026). Fast-WAM retains video co-training while omitting future slots at inference, reducing latency but removing an explicit future-token interface for action prediction (Yuan et al., 2026). Recent one-pass WAMs retain future-aware features without video rollout (Huang et al., 2026; Zhao et al., 2026). These approaches change how future information reaches the policy, yet their visual denoising objectives remain closely tied to future-video generation. This leaves a further question: *How should denoising be designed when its primary purpose is to learn world representations for robot policies?*

The answer depends on an asymmetry in the policy's information. Current observations are available during both training and deployment, whereas future frames are available only as training targets. The policy must act from the observed scene while using future frames only as training supervision. This differs from requiring photorealistic future pixels at inference. Figure 1 contrasts three training and inference paths: Joint-WAM retains iterative future denoising, Fast-WAM removes future slots at inference, and SWAM retains future-position features through a single visual pass. The comparison motivates different noise conditions for current and future observations.

We therefore introduce the Self-Supervised World Action Model (SWAM), which treats visual denoising as self-supervised training for action-relevant features. For the observed current scene,



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

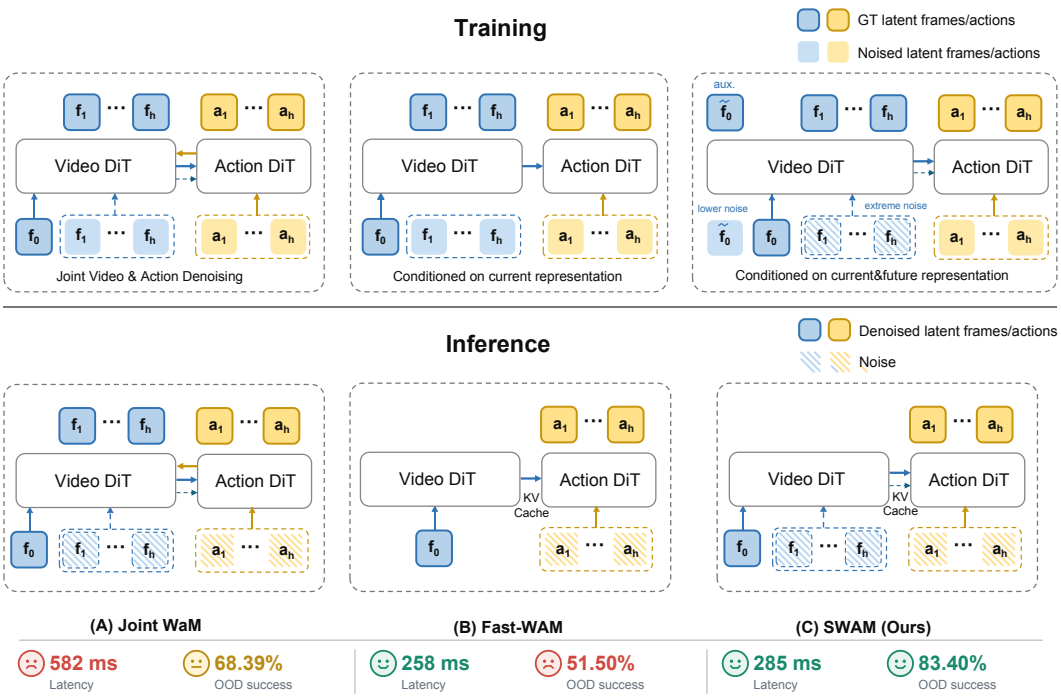



Figure 1: **Training and inference information flow with robustness and latency.** SWAM trains with auxiliary denoising of current observations and extreme-noise denoising of future observations, then uses one visual pass with pure-noise future slots for action prediction. The bottom strip reports LIBERO-Plus success rates and inference latency.

denoising autoencoding (Chen et al., 2025; Vincent et al., 2008) motivates recovering clean latents from perturbed inputs for learning robust representations. A noise schedule less aggressive than that of video generation retains more information from the observed scene while providing an auxiliary denoising signal to the shared visual backbone. This auxiliary current path is used only during training (Figure 1).

For the unobserved future, masked autoencoding (He et al., 2022; Wei et al., 2023) motivates learning from severely limited target information. SWAM applies extreme continuous noise to future latents, reducing access to the future target and bringing training inputs closer to the pure-noise future slots used at inference. Clean current latents and pure-noise future slots receive one visual forward pass. The resulting features condition iterative action denoising without rolling out or decoding future video (Figure 1).

The results suggest that adapting denoising to the asymmetry of temporal information improves robustness while retaining efficient action generation. SWAM achieves success rates of 99.1% on LIBERO and 83.4% on LIBERO-Plus, outperforming Fast-WAM and Fast-WAM-Joint on both benchmarks. We also report RoboTwin and real-world evaluation. Besides, the inference latency of SWAM is comparable to Fast-WAM and over 50% lower than Fast-WAM-Joint (Figure 1).

Our contributions are three-fold:

- We revisit WAM denoising from a self-supervised representation learning perspective, grounded in the policy's access to current observations but not future frames.
- We introduce SWAM, which trains current-observation and future-position features through asymmetric denoising and uses them for one-pass visual inference.
- We validate the self-supervised representation learning can facilitate manipulation performance on both simulation and real-world settings, analyze its denoising objectives, and measure the latency of its inference path.

## 2 Related Work

**World action models.** Vision-language-action policies such as $\pi_0$ generate actions from observations and instructions without an explicit future-video objective (Black et al., 2025). World action models instead couple action learning to visual dynamics (Zhu et al., 2025; Li et al., 2025): DreamZero predicts future video and actions, while Fast-WAM shows that video supervision can improve a policy that skips future generation at deployment (Ye et al., 2026; Yuan et al., 2026). Recent methods preserve an inference-time future interface without iterative video rollout. ForeWAM computes future keys and values in one visual pass, RIFT uses learned anticipation tokens, and Faster-WAM selectively fuses future representations into the action expert (Huang et al., 2026; Zhang et al., 2026a; Zhao et al., 2026). These works motivate efficient access to future-position features. Our focus is how visual denoising should train those features when the policy observes the present but cannot observe the future.

**Action-facing visual supervision.** The quality of generated video need not determine the utility of its hidden states for control (Hu et al., 2025; Yuan et al., 2026; Zhang et al., 2026b). AGRA addresses this mismatch by aligning world-model features with an external visual encoder, while X-WAM introduces multi-view depth prediction to provide geometric supervision (Qiu et al., 2026; Guo et al., 2026). SWAM instead derives both visual targets from the robot demonstration: a perturbed current latent is denoised as an auxiliary task, and an extremely noisy future latent is denoised with access to current context. The visual targets are self-supervised; the action objective still uses demonstrated actions.

**Denoising and masked autoencoding.** Denoising autoencoders learn representations by recovering clean inputs from corruptions (Vincent et al., 2008). Masked autoencoders learn from missing visual content, and DiffMAE connects masked reconstruction to diffusion denoising (He et al., 2022; Wei et al., 2023). SWAM adapts these information-control principles to the policy's temporal asymmetry: relatively less aggressive noise on current observations and extreme noise on future observations. Its future tokens remain in the visual transformer as continuous noisy latents, so the method does not use MAE's token dropping or sparse-encoder computation.

## 3 Asymmetric Denoising for World Action Models

SWAM uses visual denoising to train the representations consumed by a robot policy. Its two visual objectives reflect an asymmetry in the policy's information: the current observation is available when acting, while the future observation is not.

### 3.1 Problem formulation

At time $t$, a policy observes a current image observation $o_t$, robot state $p_t$, and instruction $\ell$, and predicts an action sequence $a_{t:t+H-1}$. The policy models

$$p_\theta(a_{t:t+H-1} \mid o_t, p_t, \ell). \tag{1}$$

World action models use future visual prediction as an intermediate learning signal for this policy. In a generate-then-act formulation, the current context is mapped to an imagined future $\hat{o}_{>t}$ before the action sequence is produced; joint WAMs instead denoise future visual and action variables together (Yuan et al., 2026). Both exploit future observations $o_{>t}$ from training demonstrations, even though those observations are unavailable when the robot acts. This creates a distinction between *learning from* future observations and *generating* them as an input to action prediction.

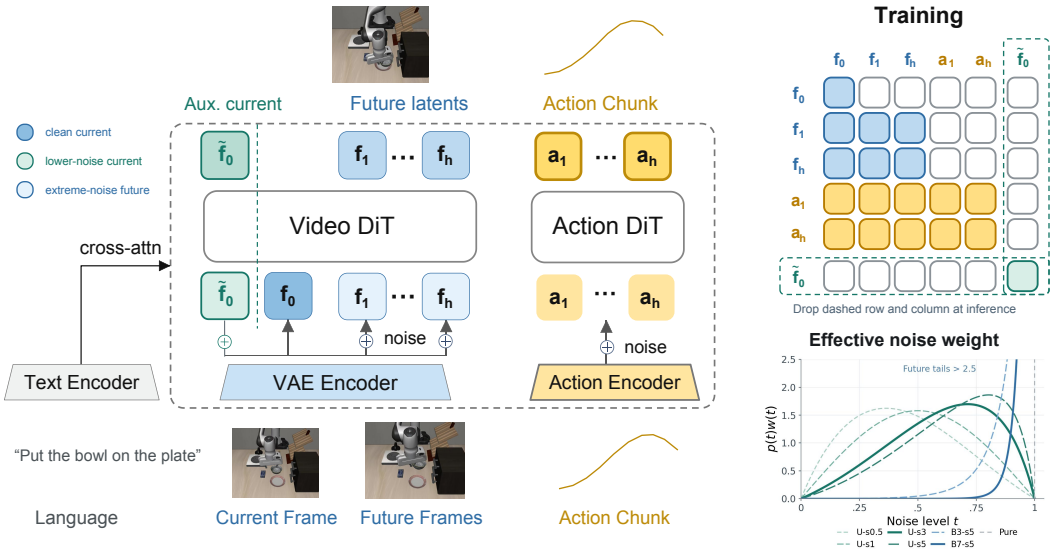
We distinguish current-observation features $z_t^C$ and future-position features $z_t^F$, both used to condition action prediction. At deployment, both must be constructed from the available context and a stochastic latent input $\xi$, without access to $o_{>t}$:

$$(z_t^C, z_t^F) = g_\theta(o_t, p_t, \ell; \xi), \qquad p_\theta(a_{t:t+H-1} \mid z_t^C, z_t^F, p_t, \ell). \tag{2}$$

The future frames in the demonstrations provide self-supervised targets for learning $z_t^F$, while the current observation can also supervise $z_t^C$. This formulation asks the visual model to provide useful context for action prediction without requiring an explicit future-frame rollout at inference.

### 3.2 Model Architecture

Following Fast-WAM (Yuan et al., 2026), SWAM combines a pretrained Video DiT with an action expert. A frozen visual autoencoder $E$ maps observations to latent tokens; let $c = E(o_t)$ and $f = E(o_{>t})$ denote the current and future latents. Language and robot state provide conditioning. During training, the main path processes clean current-observation tokens $C$, corrupted future-observation tokens $F$, and corrupted action tokens $A$. Figure 2 summarizes the token paths and their use at inference.




**(a) SWAM architecture.** **(b) Attention and noise.**

Figure 2: **SWAM architecture and visulizations of attention mask and effective noise weighting.** The main path combines clean current-observation tokens, noisy future-observation tokens, and noisy action tokens. An attention-isolated copy of the current-observation tokens is denoised during training with shared visual weights. The noise panel plots the effective weight $p(\tau)w(\tau)$: the density of sampled noise levels times their loss weight.

An attention mask determines which token groups can exchange information. Queries from current-observation tokens read only $C$; those from future-observation tokens read $C$ and $F$; action-token queries read $C$, $F$, and $A$. Visual queries do not read action tokens. Thus, unlike Fast-WAM, the action expert can use future-position features, while, unlike the jointly denoised Fast-WAM-Joint variant (Yuan et al., 2026), those features are formed without reading candidate actions. An auxiliary copy of the current latents shares the visual patch embedding and Video DiT weights with the main path. It attends only within its own branch, with attention blocked in both directions between the auxiliary and main paths. The auxiliary tokens consequently train the deployed visual weights without entering the action path.

### 3.3 Asymmetric Denoising

We use flow matching for visual and action targets (Lipman et al., 2023). Given a clean target $x$ and Gaussian noise $\epsilon \sim \mathcal{N}(0, I)$, we construct

$$x_\tau = (1-\tau)x + \tau\epsilon, \qquad v^\star = \epsilon - x, \qquad \tau \in [0,1], \tag{3}$$

and predict the velocity $v^\star$ with mean-squared error. The endpoint $\tau = 0$ is clean data and $\tau = 1$ is pure noise. We control the sampled noise level by transforming a raw time $u$ as

$$\tau = \phi_s(u) = \frac{su}{1 + (s-1)u}, \tag{4}$$

where larger $s$ places more samples at high noise. Sampling and loss weighting jointly determine the emphasis of each objective. Let $p_j(\tau)$ be the density of the sampled effective noise time for branch $j \in \{C, F, A\}$. For the current-observation and action objectives, we use the inherited scheduler-weight shape

$$\widetilde{w}(\tau) = e^{-2(\tau-1/2)^2} - e^{-1/2}, \qquad w_C(\tau) = \frac{\widetilde{w}(\tau)}{Z_C}, \quad w_A(\tau) = \frac{\widetilde{w}(\tau)}{Z_A}, \tag{5}$$

where $Z_C$ and $Z_A$ normalize the sampled current-observation and action weights to approximately unit mean; Appendix B gives their exact computation. The *effective noise weight* shown in Figure 2 is $p_j(\tau)w_j(\tau)$: it measures how strongly each noise interval contributes to the expected objective for a fixed prediction error.

**Denoising future latents.** The future observations are training targets but unavailable at deployment. Inspired by masked autoencoding (He et al., 2022), we restrict how much of the future target the model can read, encouraging it to use the observed current scene. We sample $u_F \sim \mathrm{Beta}(\alpha, 1)$ with $\alpha \geq 1$, set $\tau_F = \phi_{s_F}(u_F)$ with $s_F > s_C$, and form $f_{\tau_F} = (1 - \tau_F)f + \tau_F\epsilon_F$ using independent $\epsilon_F \sim \mathcal{N}(0, I)$. At $\alpha = 1$, $\mathrm{Beta}(1, 1) = \mathcal{U}(0, 1)$, recovering the shifted-Uniform future-observation sampling of our Fast-WAM-derived baseline. Increasing $\alpha$ concentrates samples near the pure-noise input used at inference. Strong corruption limits direct access to the future target and brings the training input closer to the pure-noise future slots used at inference. We use uniform future-observation loss weighting, $w_F(\tau_F) = 1$. This choice preserves supervision at the high-noise endpoint: the scheduler-shaped weight used for the current-observation and action objectives instead has $\widetilde{w}(1) = 0$. The future-latent velocity loss is

$$\mathcal{L}_F = \mathbb{E}\Big[w_F(\tau_F)\left\|v_\theta^F(f_{\tau_F}, \tau_F; c, \ell, p_t) - (\epsilon_F - f)\right\|_2^2\Big], \tag{6}$$

where $v_\theta^F$ predicts the velocity of future latents conditioned on the clean current observation. MAE hides image patches and reconstructs their pixels from visible patches; here every future latent position remains present, but its content is continuously obscured by Gaussian noise. We predict a flow velocity at sampled noise levels rather than reconstructing only removed patches. The clean current-observation and corrupted future-observation tokens share the Video DiT, and the resulting future-position features are used for action prediction without decoding a future image.

**Denoising the current observation.** The future-latent objective learns from observations that will be unavailable when the policy acts; the complementary current-observation objective uses the image the policy actually receives as an additional self-supervised target. We therefore feed clean current-observation latents to the main action-conditioning path and train an isolated, corrupted copy as an auxiliary objective. Denoising autoencoders (DAE) learn representations by recovering uncorrupted inputs from noisy views, and diffusion denoising has likewise been used for self-supervised visual representation learning (Vincent et al., 2008; Chen et al., 2025). We adapt this principle with flow matching and a noise schedule biased toward cleaner inputs than a generation-oriented schedule. In particular, the smaller shift $s_C < s_F$ retains more information from the observed scene in the corrupted copy. We sample $u_C \sim \mathcal{U}(0, 1)$, set $\tau_C = \phi_{s_C}(u_C)$, and form $c_{\tau_C} = (1 - \tau_C)c + \tau_C\epsilon_C$ with $\epsilon_C \sim \mathcal{N}(0, I)$. An auxiliary pass through the shared Video DiT predicts the velocity from this corrupted copy:

$$\mathcal{L}_C = \mathbb{E}\Big[w_C(\tau_C)\left\|v_\theta^C(c_{\tau_C}, \tau_C; \ell, p_t) - (\epsilon_C - c)\right\|_2^2\Big], \tag{7}$$

where $v_\theta^C$ denotes the auxiliary velocity prediction. Its gradients update the visual weights shared with the deployed policy. No auxiliary token is kept clean or removed, and the branch is discarded at inference.

**Training objective.** Let $a = a_{t:t+H-1}$ be the demonstrated action sequence. We sample $u_A \sim \mathcal{U}(0, 1)$ independently of $u_C$ and $u_F$, set $\tau_A = \phi_{s_A}(u_A)$, and form $a_{\tau_A} = (1 - \tau_A)a + \tau_A\epsilon_A$, with $\epsilon_A \sim \mathcal{N}(0, I)$. The action expert predicts its velocity while attending to current-observation and future-position features from the main path:

$$\mathcal{L}_A = \mathbb{E}\Big[w_A(\tau_A)\left\|v_\theta^A(a_{\tau_A}, \tau_A; C, F, \ell, p_t) - (\epsilon_A - a)\right\|_2^2\Big]. \tag{8}$$

The full objective is

$$\mathcal{L} = \mathcal{L}_A + \mathcal{L}_F + \mathcal{L}_C. \tag{9}$$

The expectations are over demonstrations, independently sampled noise, and noise times; each squared norm is averaged over its target elements. The two visual terms use targets from demonstration frames without additional labels, while the action term uses demonstrated actions. Appendix B specifies the schedule parameters and loss weights.

**One-pass visual inference.** At each replan, the visual autoencoder encodes the current observation, while future slots are initialized with Gaussian noise at $\tau_F = 1$. The Video DiT processes current-observation and future-position tokens once, producing layerwise visual keys and values. The action expert reuses these fixed features throughout iterative action denoising. The policy executes part of the resulting action sequence and replans from a new observation. The future-position tokens are neither iteratively updated nor decoded into images. One pass therefore refers to visual computation; action generation remains iterative.

# 4 Experiments

We evaluate SWAM on Standard LIBERO, LIBERO-Plus, RoboTwin 2.0, and four real-world manipulation tasks. We also measure inference latency and examine how noise sampling for current and future latents affects performance. All ablation points use complete evaluations.

## 4.1 Implementation Details

The visual expert is initialized from Wan2.2-TI2V-5B (Wan Team, 2025). We use 1,712 training episodes from the four LIBERO suites. Unless otherwise stated, training and inference hyperparameters follow Fast-WAM (Yuan et al., 2026). Additional implementation details are provided in Appendix A.

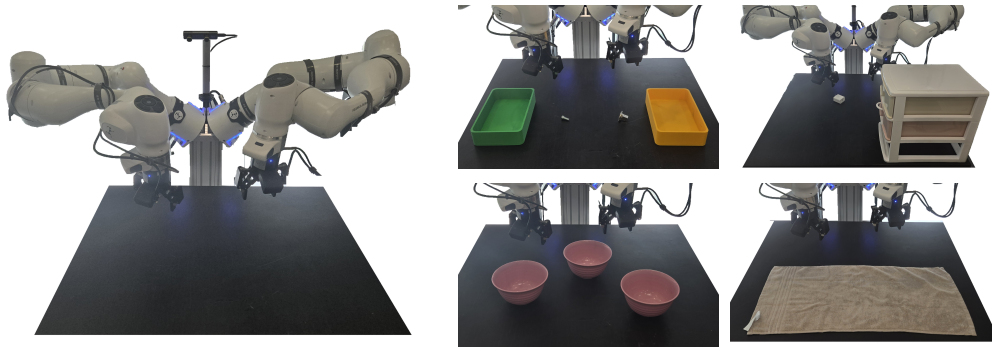
## 4.2 Experimental Setup

**Standard LIBERO.** We evaluate on the Spatial, Object, Goal, and LIBERO-10 suites, each containing ten tasks (Liu et al., 2023). The policy is tested on 50 supplied initial states per task, yielding 2,000 closed-loop episodes. We report success rates for each suite and across all 40 tasks.

**LIBERO-Plus.** We evaluate the same LIBERO-trained checkpoints under seven perturbation categories: camera viewpoint, robot initialization, language, lighting, background, sensor noise, and object layout (Fei et al., 2026). We report category success rates and the micro-average over all cases.

**RoboTwin 2.0.** We train SWAM on clean-only dataset for 30k step and evaluate all 50 bimanual tasks under clean and randomized conditions (Chen et al., 2026) to show in-domain and out-of-domain performance. The training recipe is the same as that of Fast-WAM.

**Real-world experiments.** We conduct real-world experiments on a Franka Research 3 Duo platform across four manipulation tasks: Bimanual Sorting (T1), Open Drawer and Place Charger (T2), Stack Three Bowls (T3), and Fold Towel (T4). These tasks cover bimanual coordination, articulated-object interaction, multi-object stacking, and deformable-object manipulation. Each task contributes 100 trajectories, and all tasks are combined for joint training. We report success rates over 50 trials per task under multiple initial configurations; detailed robot, gripper, camera, and observation configurations are provided in Appendix D. Figure 3 shows the platform and task setups.



(a) Platform (b) Real-world tasks

Figure 3: **Real-world platform and task overview.** (a) Our dual-arm platform. (b) Real-world tasks, shown from left to right, top to bottom: bimanual sorting (T1), opening a drawer and placing a charger inside (T2), stacking three bowls (T3), and folding a towel (T4).

### 4.3 Main Results

**Results on LIBERO and LIBERO-Plus.** As shown in Tables 1 and 2, SWAM achieves a success rate of 99.1% on LIBERO, exceeding Fast-WAM and Fast-WAM-Joint by 1.5 and 0.7 percentage points, respectively. On LIBERO-Plus, SWAM reaches 83.40%, outperforming Fast-WAM at 51.50% and Fast-WAM-Joint at 68.39%. These results show improved robustness to distribution shifts while maintaining strong in-domain performance.

**Results on RoboTwin 2.0.** Table 3 shows that SWAM achieves success rates of 80.7% under Clean conditions and 15.2% under Randomized conditions, compared with 72.9% and 2.7% for Fast-WAM. Its average success rate is 48.0%, exceeding Fast-WAM's 37.8% and ranking highest among the compared methods. The improvements across both conditions demonstrate stronger bimanual manipulation performance and robustness to scene variations.

**Real-world performance.** We compare SWAM with $\pi_0$ (Black et al., 2025) and FastWAM (Yuan et al., 2026) on T1–T4. Success rate is the percentage of evaluation trials that successfully complete a task; Table 4 reports per-task rates and their unweighted mean. SWAM achieves the highest success rate on all four tasks, with a mean of 76.50%, compared with 37.50% for $\pi_0$ and 59.00% for FastWAM, corresponding to a 17.5 percentage point improvement over FastWAM.

**Inference efficiency.** Figure 1 shows that SWAM takes 284.8 ms per action chunk, compared with 258.4 ms for Fast-WAM and 582.4 ms for Fast-WAM-Joint in the matched inference-path comparison. SWAM reduces latency by 51.1% relative to Fast-WAM-Joint while remaining close to Fast-WAM, retaining efficient action generation without iterative future-video rollout.

Table 1: Success rates on Standard LIBERO, in percent. Emb. PT denotes additional embodied-data pretraining outside LIBERO. The final three rows are our evaluations; the other rows report published results.

| Method | Emb. PT | Spatial | Object | Goal | LIBERO-10 | Overall |
|---|---|---|---|---|---|---|
| $\pi_0$ | ✓ | 96.8 | 98.8 | 95.8 | 85.2 | 94.2 |
| OpenVLA-OFT | ✓ | 97.6 | 98.4 | 97.9 | 94.5 | 97.1 |
| $\pi_{0.5}$ | ✓ | 98.8 | 98.2 | 98.0 | 92.4 | 96.9 |
| LingBot-VA | ✓ | 98.5 | 99.6 | 97.2 | 98.5 | 98.5 |
| ABot-M0 | ✓ | 98.8 | 99.8 | 99.0 | 96.6 | <u>98.6</u> |
| Motus | ✓ | 96.8 | 99.8 | 96.6 | 97.6 | 97.7 |
| Fast-WAM | × | 98.2 | 100.0 | 97.0 | 95.2 | 97.6 |
| Fast-WAM-Joint | × | 98.4 | 99.6 | 97.8 | 97.8 | 98.4 |
| **SWAM** | × | 98.8 | 99.8 | 98.8 | 99.0 | **99.1** |

Table 2: Success rates on LIBERO and LIBERO-Plus, in percent. Emb. PT denotes additional embodied-data pretraining. Best results are shown in bold, and second-best results are shown in underlined.

| Method | Emb. PT | Rollout | LIBERO | LIBERO-Plus | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Standard | Camera | Robot | Lang. | Light | Backg. | Noise | Layout | Overall |
| $\pi_0$ | ✓ | — | 94.2 | 13.8 | 6.0 | 58.8 | 85.0 | 81.4 | 79.0 | 68.9 | 53.6 |
| $\pi_0$-Fast | ✓ | — | 85.5 | **65.1** | 21.6 | 61.0 | 73.2 | 73.2 | 74.4 | 68.8 | 61.6 |
| OpenVLA-OFT | ✓ | — | 97.1 | 56.4 | 31.9 | 79.5 | 88.7 | **93.3** | 75.8 | 74.2 | 69.6 |
| ABot-M0 | ✓ | — | <u>98.6</u> | 60.4 | <u>67.9</u> | 86.4 | <u>96.2</u> | <u>91.6</u> | <u>86.4</u> | 82.6 | <u>80.5</u> |
| Fast-WAM | × | × | 97.6 | 16.4 | 44.5 | 68.9 | 78.2 | 53.7 | 37.7 | 60.7 | 51.50 |
| Fast-WAM-Joint | × | ✓ | 98.4 | 42.2 | 62.9 | 91.3 | 94.7 | 50.9 | 59.6 | 80.3 | 68.39 |
| SWAM w/o Curr. loss | × | × | 98.4 | 54.3 | 69.0 | **96.8** | 97.1 | 62.4 | 78.8 | <u>83.5</u> | 77.18 |
| **SWAM** | × | × | **99.1** | <u>62.7</u> | **76.4** | <u>94.3</u> | **97.4** | 80.2 | **91.9** | **84.1** | **83.40** |

Table 3: Success rates on RoboTwin 2.0, in percent. Clean and Randomized denote in-domain and out-of-domain evaluation. Emb. PT denotes additional embodied-data pretraining.

| Method | Emb. PT | Clean2Clean (ID) | Clean2Random (OOD) | Average |
|---|---|---|---|---|
| StarVLA | × | 46.5 | 3.2 | 24.8 |
| AHA-WAM | × | 64.3 | 3.2 | 33.8 |
| ABot-M0 | ✓ | 57.4 | 30.4 | 43.9 |
| X-VLA | ✓ | 68.0 | 20.9 | 44.5 |
| X-WAM | ✓ | 70.0 | 25.8 | 47.9 |
| Fast-WAM | × | 72.9 | 2.7 | 37.8 |
| **SWAM** | × | 80.7 | 15.2 | **48.0** |

Table 4: **Real-world task success rates (%).** Mean is the unweighted average across T1–T4 on our FR3 Duo. The best result in each column is bold.

| Method | T1 | T2 | T3 | T4 | Mean |
|---|---|---|---|---|---|
| $\pi_0$ | 48 | 24 | 64 | 14 | 37.50 |
| FastWAM | 56 | 52 | 82 | 46 | 59.00 |
| SWAM (ours) | **80** | **76** | **96** | **54** | **76.50** |

### 4.4 Ablation Studies

**Future prediction noise.** The future objective should prepare the model to act without clean future observations. Figure 4(a) examines this choice under one-pass inference. Among Future-only runs with shift 5 and uniform loss weighting, Beta$(\alpha, 1)$ sampling at $\alpha = 1$, 3, and 7 yields 75.61%, 75.85%, and 77.18% LIBERO-Plus success. At $\alpha = 7$, the training inputs concentrate near the pure-noise future slots used at deployment, and success improves over the Uniform setting. Always-terminal noise ($t = 1$) yields 73.96%, indicating that a range of highly corrupted inputs is more useful here than training only at the endpoint.

Uniform loss weighting retains Future supervision at the pure-noise endpoint used in deployment. In a separate readout diagnostic, allowing Action to access Future slots corresponds to 77.18% success, versus 62.14% without that access. Together, these results show the utility of high-noise Future training and future-position features for action prediction.

**Current observation noise.** Figure 4(b) tests how strongly to corrupt the observation that remains available at deployment. All four configurations denoise every auxiliary Current token, share the same high-noise Future objective, and use one-pass inference. LIBERO-Plus success rises from 80.32% at shift 0.5 to 82.13% at shift 1 and 83.40% at shift 3, then falls to 81.08% at the generation-oriented shift 5. This intermediate maximum is consistent with balancing a useful denoising signal against retention of information from the observed image. The 6.22-point gain over the Future-only model (Table 2), including gains under background and sensor perturbations, further supports auxiliary current-observation denoising for policy robustness.

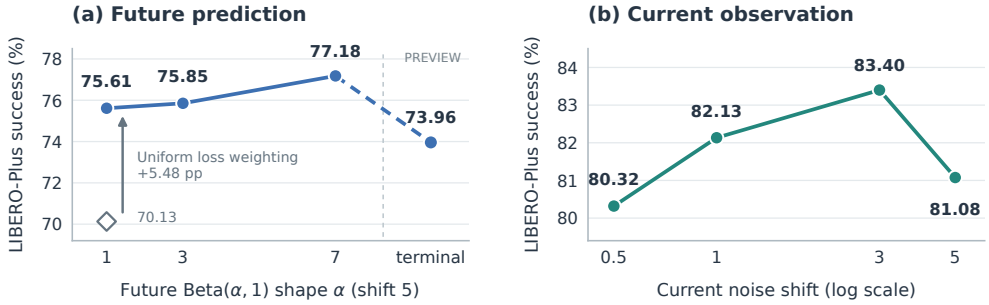



Figure 4: LIBERO-Plus denoising ablations. (a) Future time sampling by Beta$(\alpha, 1)$ shape at shift 5 with uniform loss weighting. Uniform sampling is $\alpha = 1$. The dashed segment connects Beta$(7, 1)$ to terminal Future noise ($t = 1$), the limiting distribution as $\alpha \to \infty$; its horizontal spacing is schematic. (b) Different shift of Current noise scheduler.

## 5 Conclusion

SWAM designs asymmetric visual denoising around the information available to a robot policy. Auxiliary denoising of current observations and high-noise denoising of future observations provide complementary self-supervised training signals. At deployment, one visual pass supplies fixed features for action denoising without future-video rollout. SWAM achieves 83.4% success on LIBERO-Plus and 99.1% on standard LIBERO with inference latency comparable to Fast-WAM and 51.1% lower than Fast-WAM-Joint. These results support asymmetric visual denoising as a useful training strategy for robust and efficient robot policies.

### AI USE STATEMENT

Large Language Models (LLMs) and other generative AI tools were used in this work to assist with research ideation, result interpretation, literature retrieval and discovery, bibliography preparation, evidence organization, and consistency checks. These tools were also used to assist with drafting portions of the manuscript and to polish the writing for clarity, concision, and presentation. All AI-assisted outputs were reviewed and verified by the authors. The authors take full responsibility for the final content of this work, including its text, claims, figures, and other artifacts.

### ETHICS STATEMENT

This work studies simulated and real-world robot manipulation. Our real-world evaluation covers four tasks on a single dual-arm platform and does not establish safety in uncontrolled environments. Deployment would require collision constraints, human oversight, failure detection, and further evaluation around people, fragile objects, and sensor failures. The training data and benchmarks may underrepresent environments, languages, object appearances, and users. These broader deployment risks remain outside the scope of our evaluation.

### REPRODUCIBILITY STATEMENT

Section 3 describes the model architecture, training objectives, and inference procedure. Section 4 specifies the simulation benchmarks, training settings, evaluation protocols, and trial counts. Appendix A provides architecture and optimization details, while Appendix B gives the noise schedules definitions. Appendix C reports complete per-task RoboTwin results. For the real-world experiments, Section 4.2 specifies the tasks, demonstration counts, joint training protocol, and evaluation budget, and Section 4.3 reports the results; Appendix D documents the robot, grippers, cameras, and observation resolutions.

## REFERENCES

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. $\pi_0$: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.010. URL https://www.roboticsproceedings.org/rss21/p010.html.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, Weiliang Deng, Yubin Guo, Tian Nian, Xuanbing Xie, Qiangyu Chen, Kailun Su, Tianling Xu, Guodong Liu, Mengkang Hu, Huan-ang Gao, Kaixuan Wang, Zhixuan Liang, Yusen Qin, Xiaokang Yang, Ping Luo, and Yao Mu. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. In *Proceedings of the 43rd International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2506.18088.

Xinlei Chen, Zhuang Liu, Saining Xie, and Kaiming He. Deconstructing denoising diffusion models for self-supervised learning. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/8b52c29d1a77e3fcc6c8f90fa0ac0e8e-Abstract-Conference.html.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, Jinlan Fu, Jingjing Gong, and Xipeng Qiu. LIBERO-Plus: A progressive robustness benchmark for visual-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 38574–38583, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Fei_LIBERO-Plus_A_Progressive_Robustness_Benchmark_for_Visual-Language-Action_Models_CVPR_2026_paper.html.

Jun Guo, Qiwei Li, Peiyan Li, Zilong Chen, Nan Sun, Yifei Su, Heyun Wang, Yuan Zhang, Xinghang Li, and Huaping Liu. Unified 4D world action modeling from video priors with asynchronous denoising. *arXiv preprint arXiv:2604.26694*, 2026. URL https://arxiv.org/abs/2604.26694v2.

Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/He_Masked_Autoencoders_Are_Scalable_Vision_Learners_CVPR_2022_paper.html.

Yucheng Hu, Yanjiang Guo, Pengchao Wang, Xiaoyu Chen, Yen-Jen Wang, Jianke Zhang, Koushil Sreenath, Chaochao Lu, and Jianyu Chen. Video prediction policy: A generalist robot policy with predictive visual representations. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 24328–24346. PMLR, 2025. URL https://proceedings.mlr.press/v267/hu25g.html.

Jiakai Huang, Zhongbo Wu, Zheng Zhang, Zihan Wang, Shan You, and Tao Huang. Foresight without seeing: Latent futures for world action models. *arXiv preprint arXiv:2608.11605*, 2026. URL https://arxiv.org/abs/2608.11605.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. In *Proceedings of Robotics: Science and Systems*, 2025a. doi: 10.15607/RSS.2025.XXI.017. URL https://www.roboticsproceedings.org/rss21/p017.html.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 2679–2713. PMLR, 2025b. URL https://proceedings.mlr.press/v270/kim25c.html.

Moo Jin Kim, Yihuai Gao, Tsung-Yi Lin, Yen-Chen Lin, Yunhao Ge, Grace Lam, Percy Liang, Shuran Song, Ming-Yu Liu, Chelsea Finn, and Jinwei Gu. Cosmos policy: Fine-tuning video models for visuomotor control and planning. *arXiv preprint arXiv:2601.16163*, 2026. URL https://arxiv.org/abs/2601.16163.

Lin Li, Qihang Zhang, Yiming Luo, Shuai Yang, Ruilin Wang, Fei Han, Mingrui Yu, Zelin Gao, Nan Xue, Xing Zhu, Yujun Shen, and Yinghao Xu. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026. URL https://arxiv.org/abs/2601.21998.

Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified Video Action Model. In *Proceedings of Robotics: Science and Systems*, 2025. doi: 10.15607/RSS.2025.XXI.074. URL https://www.roboticsproceedings.org/rss21/p074.html.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

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*, volume 36, pp. 44776–44791, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/8c3c666820ea055a77726d66fc7d447f-Abstract-Datasets_and_Benchmarks.html.

Lu Qiu, Yizhuo Li, Yi Chen, Yuying Ge, Yixiao Ge, and Xihui Liu. Making foresight actionable: Repurposing representation alignment in world action models. *arXiv preprint arXiv:2606.12217*, 2026. URL https://arxiv.org/abs/2606.12217.

Pascal Vincent, Hugo Larochelle, Yoshua Bengio, and Pierre-Antoine Manzagol. Extracting and composing robust features with denoising autoencoders. In *Proceedings of the 25th International Conference on Machine Learning*, pp. 1096–1103, 2008. doi: 10.1145/1390156.1390294. URL https://icml.cc/Conferences/2008/papers/592.pdf.

Wan Team. Wan2.2: Open-source video generation models. https://github.com/Wan-Video/Wan2.2, 2025. Official repository, accessed 2026-09-19.

Chen Wei, Karttikeya Mangalam, Po-Yao Huang, Yanghao Li, Haoqi Fan, Hu Xu, Huiyu Wang, Cihang Xie, Alan Yuille, and Christoph Feichtenhofer. Diffusion models as masked autoencoders. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 16284–16294, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Wei_Diffusion_Models_as_Masked_Autoencoders_ICCV_2023_paper.html.

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. *arXiv preprint arXiv:2602.15922*, 2026. URL https://arxiv.org/abs/2602.15922.

Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-WAM: Do world action models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026. URL https://arxiv.org/abs/2603.16666.

Chushan Zhang, Jinguang Tong, Xuesong Li, Yikai Wang, and Hongdong Li. Keep the future, drop the rollout: RIFT for world action models. *arXiv preprint arXiv:2608.11521*, 2026a. URL https://arxiv.org/abs/2608.11521v2.

Yuyang Zhang, Wenyao Zhang, Zekun Qi, He Zhang, Haitao Lin, Jingbo Zhang, Yao Mu, Xiaokang Yang, Wenjun Zeng, and Xin Jin. ImageWAM: Do world action models really need video generation, or just image editing? arXiv preprint arXiv:2606.19531, 2026b. URL https://arxiv.org/abs/2606.19531.

Weiheng Zhao, Haoyi Jiang, Xin Shi, Liu Liu, Fan Huang, Zhizhong Su, Wei Sui, and Xinggang Wang. Faster-WAM: Efficient inference-time future conditioning for robust world action models. *arXiv preprint arXiv:2608.04404*, 2026. URL https://arxiv.org/abs/2608.04404.

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. URL https://www.roboticsproceedings.org/rss21/p015.html.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, Quan Vuong, Vincent Vanhoucke, Huong Tran, Radu Soricut, Anikait Singh, Jaspiar Singh, Pierre Sermanet, Pannag R. Sanketi, Grecia Salazar, Michael S. Ryoo, Krista Reymann, Kanishka Rao, Karl Pertsch, Igor Mordatch, Henryk Michalewski, Yao Lu, Sergey Levine, Lisa Lee, Tsang-Wei Edward Lee, Isabel Leal, Yuheng Kuang, Dmitry Kalashnikov, Ryan Julian, Nikhil J. Joshi, Alex Irpan, Brian Ichter, Jasmine Hsu, Alexander Herzog, Karol Hausman, Keerthana Gopalakrishnan, Chuyuan Fu, Pete Florence, Chelsea Finn, Kumar Avinava Dubey, Danny Driess, Tianli Ding, Krzysztof Marcin Choromanski, Xi Chen, Yevgen Chebotar, Justice Carbajal, Noah Brown, Anthony Brohan, Montserrat Gonzalez Arenas, and Kehang Han. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 2165–2183. PMLR, 2023. URL https://proceedings.mlr.press/v229/zitkovich23a.html.

## A Implementation and Evaluation Details

**Architecture.** Each $224 \times 224$ camera view encodes to a $7 \times 7$ token grid. The main current observation therefore has 98 tokens, the two future latent times have 196 tokens, and the training-only auxiliary current branch duplicates all 98 current tokens. The visual expert is a 30-layer Wan2.2 TI2V transformer with hidden width 3,072, 24 heads, and a 14,336-dimensional feed-forward block (Wan Team, 2025). The 30-layer Action expert has width 1,024 and feed-forward width 4,096. Language features are cached with a maximum of 128 tokens. An MLP embeds the current 8-D proprioceptive state as an additional context token. The visual autoencoder and language encoder are frozen; the visual and Action transformers and proprioception MLP are trainable.

**Data and optimization.** The processed training manifest contains 1,712 episodes and 277,713 transition starts from LIBERO Spatial, Object, Goal, and LIBERO-10. A sample spans 33 source positions, yields 32 actions, and samples nine video frames that compress to one current and two future latent times. Optimization uses AdamW with $\beta = (0.9, 0.95)$, weight decay 0.01, initial learning rate $10^{-4}$, 5% linear warmup, cosine decay to $10^{-6}$, gradient-norm clipping at 1, BF16, global batch 128, and seed 42.

## B Exact Noise, Attention, and Loss Contracts

The shifted noise schedule for shift $s$ is

$$\phi_s(u) = \frac{su}{1 + (s-1)u}. \tag{10}$$

For shifted Uniform noise, the density in effective time is $p_{U,s}(\tau) = s/[s - (s-1)\tau]^2$. For the Future recipe, $u \sim \mathrm{Beta}(7, 1)$ followed by $s = 5$, giving

$$p_{B7,5}(\tau) = \frac{35\tau^6}{(5 - 4\tau)^8}. \tag{11}$$

The inherited scheduler weight used by the headline all-noisy Current, shift-3 auxiliary objective and the Action objective is

$$\widetilde{w}(\tau) = \exp[-2(\tau - 1/2)^2] - \exp(-1/2), \qquad w_s(\tau) = \frac{\widetilde{w}(\tau)}{Z_s + 10^{-10}}, \quad s \in \{3, 5\}, \tag{12}$$

where $Z_3$ is the scheduler's numerical expectation under shift-three Uniform sampling and $Z_5$ is the mean on its historical 1,000-point shift-five grid. In the headline recipe, auxiliary current uses $w_3$, Action uses $w_5$, and future uses uniform loss weighting with multiplier one. Because $\widetilde{w}(1) = 0$, uniform weighting retains supervision at the pure-noise endpoint.

## C Complete RoboTwin 2.0 Results

Table 5 preserves the official 50-task order and pairs Clean with Randomized success for StarVLA, AHA-WAM, ABot-M0, X-VLA, X-WAM, Fast-WAM, and SWAM, matching Table 3 exactly.

Table 5: Per-task RoboTwin 2.0 success rates (%) in the official 50-task order. C denotes Clean and R denotes Randomized.

| Task | StarVLA | | AHA-WAM | | ABot-M0 | | X-VLA | | X-WAM | | Fast-WAM | | **SWAM** | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | C | R | C | R | C | R | C | R | C | R | C | R | C | R |
| Adjust Bottle | 84 | 0 | **100** | 0 | 99 | **49** | 99 | 33 | 92 | 13 | **100** | 0 | 99 | 17 |
| Beat Block Hammer | 75 | 2 | 52 | 0 | 72 | **33** | 87 | 9 | **92** | 21 | 88 | 0 | 89 | 9 |
| Blocks Ranking RGB | 39 | 2 | 60 | 1 | 61 | 23 | 3 | 4 | 85 | **36** | 86 | 0 | **99** | 24 |
| Blocks Ranking Size | 14 | 0 | 32 | 0 | 33 | 7 | 56 | **25** | 33 | 0 | **57** | 0 | 47 | 1 |
| Click Alarmclock | 60 | 0 | **98** | 17 | 0 | 21 | 62 | 43 | 95 | **61** | 96 | 52 | 87 | 40 |
| Click Bell | 75 | 7 | **100** | 19 | 5 | 11 | **100** | **63** | **100** | 52 | 96 | 11 | 92 | 13 |
| Dump Bin Bigbin | 54 | 4 | 90 | 27 | 89 | **68** | 83 | 41 | 88 | 40 | 95 | 0 | **96** | 14 |

*Table 5 continued*

| Task | StarVLA | | AHA-WAM | | ABot-M0 | | X-VLA | | X-WAM | | Fast-WAM | | **SWAM** | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | C | R | C | R | C | R | C | R | C | R | C | R | C | R |
| Grab Roller | 94 | 44 | 98 | 3 | **100** | **91** | **100** | 20 | 95 | 16 | **100** | 0 | **100** | 24 |
| Handover Block | 6 | 0 | 63 | 1 | 16 | 0 | 77 | 1.3 | 21 | **21** | 43 | 0 | **80** | 0 |
| Handover Mic | 66 | 1 | 99 | 3 | 90 | **22** | 98 | 1 | 97 | 8 | 71 | 0 | **100** | 2 |
| Hanging Mug | 1 | 0 | 13 | 0 | 7 | **4** | 19 | 2 | 17 | 0 | 20 | 0 | **43** | 2 |
| Lift Pot | 68 | 0 | 62 | 0 | 51 | **29** | 96 | 15 | 77 | 0 | 88 | 0 | **98** | 0 |
| Move Can Pot | 33 | 0 | 48 | 0 | 42 | 4 | 58 | 0 | 77 | **34** | 80 | 0 | **87** | 4 |
| Move Pillbottle Pad | 40 | 0 | 33 | 0 | 56 | 4 | 66 | 3 | 78 | **26** | 86 | 0 | **94** | 4 |
| Move Playingcard Away | 62 | 0 | 82 | 0 | 78 | **50** | 84 | 5 | 83 | 11 | **96** | 0 | 94 | 0 |
| Move Stapler Pad | 14 | 0 | 5 | 0 | 16 | **4** | **42** | 0 | 26 | 2 | 17 | 0 | 19 | 1 |
| Open Laptop | 22 | 2 | 89 | 9 | 68 | **64** | 85 | 16 | 72 | 6 | 90 | 3 | **94** | 7 |
| Open Microwave | 10 | 1 | 59 | 1 | **67** | **26** | 43 | 5 | 9 | 24 | 18 | 1 | 29 | 8 |
| Pick Diverse Bottles | 43 | 15 | 36 | 0 | 12 | 7 | 50 | **30** | **74** | 26 | 50 | 1 | 58 | 28 |
| Pick Dual Bottles | 60 | 20 | 60 | 1 | 28 | 19 | 41 | 51 | **99** | **55** | 77 | 3 | 71 | 49 |
| Place A2B Left | 24 | 0 | 65 | 1 | 56 | **26** | 53 | 15 | 70 | 18 | 58 | 0 | **82** | 7 |
| Place A2B Right | 25 | 1 | 50 | 0 | 41 | **22** | 49 | 13 | 59 | 21 | 63 | 0 | **81** | 7 |
| Place Bread Basket | 82 | 5 | 54 | 3 | 70 | 47 | 85 | **52** | 82 | 40 | 79 | 1 | **87** | 26 |
| Place Bread Skillet | 79 | 2 | 68 | 0 | 73 | 25 | **89** | 17 | 82 | **37** | 76 | 0 | 86 | 14 |
| Place Burger Fries | 85 | 11 | **96** | 0 | 93 | **55** | 90 | 35 | 93 | 36 | 86 | 1 | 93 | 18 |

*Table 5 continued*

| Task | StarVLA | | AHA-WAM | | ABot-M0 | | X-VLA | | X-WAM | | Fast-WAM | | **SWAM** | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | C | R | C | R | C | R | C | R | C | R | C | R | C | R |
| Place Can Basket | 4 | 0 | 61 | 2 | 38 | 13 | 70 | 9 | 63 | **15** | 63 | 0 | **72** | 7 |
| Place Cans Plasticbox | **95** | 2 | 90 | 1 | 56 | 36 | 93 | **60** | 93 | 36 | 93 | 0 | 93 | 8 |
| Place Container Plate | 87 | 3 | 99 | 11 | **100** | **62** | 94 | 39 | 95 | 52 | **100** | 2 | 99 | 34 |
| Place Dual Shoes | 30 | 2 | 48 | 0 | 28 | **15** | 30 | 12 | 38 | 7 | 61 | 1 | **74** | 5 |
| Place Empty Cup | 85 | 0 | 64 | 2 | 97 | **43** | 90 | 16 | 93 | 36 | 95 | 0 | **99** | 17 |
| Place Fan | 30 | 1 | 58 | 0 | 48 | **10** | 63 | 1 | 47 | 6 | **65** | 0 | 62 | 0 |
| Place Mouse Pad | 11 | 0 | 12 | 0 | 31 | 5 | 53 | 0 | 49 | **6** | 43 | 0 | **61** | 0 |
| Place Object Basket | 4 | 1 | 71 | 0 | 70 | **33** | 81 | 6 | 52 | 20 | 64 | 0 | **82** | 13 |
| Place Object Scale | 20 | 1 | 34 | 0 | 73 | **23** | 76 | 6 | 66 | 14 | 68 | 0 | **85** | 4 |
| Place Object Stand | 66 | 0 | 79 | 0 | 79 | 25 | 91 | **43** | 83 | 28 | **92** | 2 | 89 | 30 |
| Place Phone Stand | 18 | 0 | 61 | 0 | 33 | **26** | 84 | 5 | 35 | 4 | 79 | 0 | **95** | 5 |
| Place Shoe | 46 | 0 | 74 | 5 | 74 | 24 | **83** | **39** | 76 | 31 | 78 | 0 | 76 | 18 |
| Press Stapler | 91 | 10 | 82 | 2 | 78 | 52 | 87 | **73** | 85 | 20 | **93** | 12 | 91 | 21 |
| Put Bottles Dustbin | 11 | 0 | 58 | 1 | 46 | **26** | 47 | 20 | 71 | 21 | 37 | 0 | **78** | 1 |
| Put Object Cabinet | 29 | 0 | 36 | 0 | 41 | **22** | 41 | 21 | 28 | 16 | 48 | 0 | **51** | 9 |
| Rotate QRcode | 11 | 0 | 61 | 0 | 52 | **6** | 81 | 1 | 47 | 2 | **83** | 0 | 78 | 0 |
| Scan Object | 63 | 5 | 51 | 0 | 38 | **17** | 57 | 9 | 32 | 14 | 60 | 0 | **72** | 8 |
| Shake Bottle Horizontally | 98 | 9 | **100** | 15 | 97 | **85** | **100** | 78 | 97 | 66 | **100** | 21 | **100** | 64 |
| Shake Bottle | **100** | 5 | **100** | 14 | 96 | **84** | **100** | 76 | 96 | 67 | **100** | 20 | **100** | 60 |
| Stack Blocks Three | 20 | 1 | 57 | 1 | 39 | 10 | 0 | 0 | 72 | 26 | 75 | 0 | **98** | **28** |
| Stack Blocks Two | 53 | 1 | 89 | 2 | 82 | 42 | 10 | 2 | **97** | **76** | 96 | 0 | **97** | 41 |
| Stack Bowls Three | 46 | 0 | 71 | 0 | 76 | **44** | **77** | 9 | 69 | 38 | 66 | 0 | 75 | 23 |
| Stack Bowls Two | 65 | 0 | **93** | 5 | 92 | 65 | 85 | 15 | **93** | **67** | 92 | 0 | **93** | 34 |
| Stamp Seal | 12 | 0 | 13 | 0 | 45 | **19** | 50 | 2 | **58** | 16 | 42 | 0 | 56 | 5 |
| Turn Switch | 16 | 0 | 42 | 13 | 38 | **20** | 40 | 6 | **68** | 4 | 37 | 3 | 53 | 7 |
| Average (50 tasks) | 46.52 | 3.16 | 64.30 | 3.20 | 57.40 | **30.36** | 68.00 | 20.90 | 70.00 | 25.80 | 72.92 | 2.68 | **80.68** | 15.22 |

## D Real-World Hardware and Observation Details

The platform uses two FR3 manipulators, each with seven degrees of freedom. Both arms terminate in 3D-printed Robotiq grippers. An overhead ZED Mini stereo camera provides a global workspace view, and one RealSense D405 camera is mounted on each wrist for local views near the corresponding end effector. The ZED Mini captures images at its native $1280 \times 720$ resolution, which are processed to $640 \times 480$. All three camera streams provide $640 \times 480$ observations at 30 FPS.