DUALWAM: DUAL-SYSTEM WORLD ACTION MODELS FOR ASYNCHRONOUS GLOBAL PLANNING AND LOCAL REFINEMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

World Action Models (WAMs) jointly generate robot actions and predict future world states, transferring priors from video pretraining to robot control. However, future visual prediction is computationally expensive, so existing WAMs often rely on long action chunks to amortize inference cost across control steps, at the cost of closed-loop responsiveness. We present DualWAM, a dual-system WAM that preserves broader-horizon world-action generation while enabling high-frequency closed-loop action updates by decoupling global planning and local refinement. System 2 periodically performs high-noise bidirectional denoising over a broader world-action chunk to establish a global plan, while wrist-only System 1 extracts a temporally aligned short window from the intermediate denoising state and completes low-noise refinement using the latest wrist observations, which provide action-aligned cues about local geometry, motion, and contact during interaction. The two systems operate asynchronously along a shared denoising trajectory: each global plan is reused across multiple local updates, while System 1 repeatedly incorporates fresh interaction feedback. Across zero-shot manipulation tasks on Franka and Galbot, DualWAM improves success over the strongest evaluated baseline by 4.5 percentage points on average, while achieving a 16.6× critical-path speedup. Further studies show that role-matched egocentric and UMI data improve success by 14 percentage points, and that the decoupled design naturally supports edge–cloud deployment with substantially lower communication overhead than the baseline.

1 INTRODUCTION

World Action Models (WAMs) offer a promising framework for robot manipulation by jointly predicting future world states and robot actions. By adapting pretrained video generators to control, WAMs can transfer semantic and physical priors learned from large-scale video data to robot policies (Ye et al., 2026; Yuan et al., 2026; Kim et al., 2026). Recent WAMs have demonstrated encouraging zero-shot generalization to tasks and interactions outside the robot training distribution (Ye et al., 2026; Feng et al., 2026).

A central challenge is that future visual prediction is computationally expensive. Existing WAMs therefore often rely on long action chunks to amortize inference cost across control steps, at the cost of closed-loop responsiveness. Executing a longer action chunk between model updates reduces how frequently new interaction feedback can affect subsequent actions. Shortening the chunk also comes with drawbacks: it requires more frequent world-action generation and encourages myopic next-chunk prediction, which can favor local appearance shortcuts over broader temporal dynamics (Xu et al., 2026). This creates a tension between efficient global prediction and responsive closed-loop control.

Existing approaches mitigate this tension by reducing, compressing, or asynchronously decoding future predictions (Yuan et al., 2026; Li et al., 2026b; Zhang et al., 2026c; Lyu et al., 2026; Cai et al., 2026; Guo et al., 2026). However, they do not jointly specialize the denoising stage, visual context, and update frequency within a shared world-action generation process.

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.

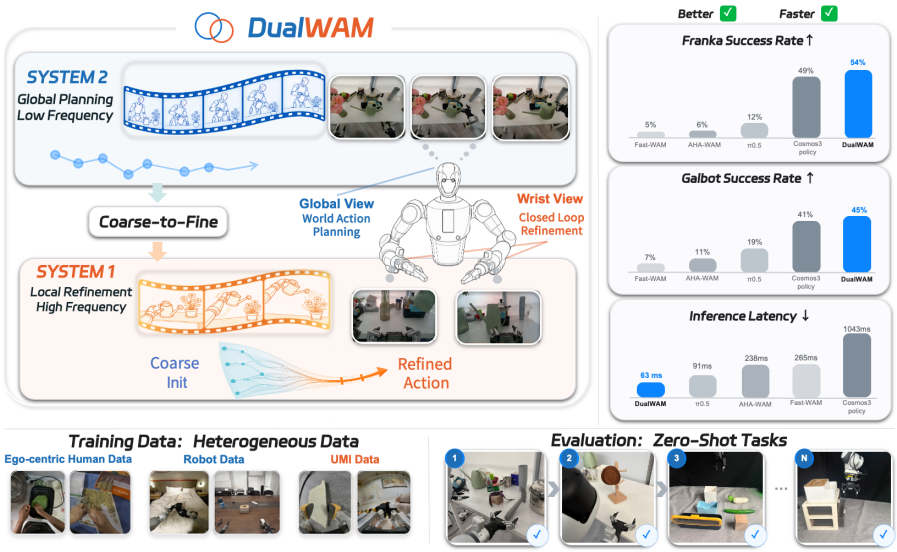

Figure 1: **Overview.** DualWAM couples low-frequency global world-action planning with high-frequency wrist-centric refinement along a shared denoising trajectory, enabling more responsive and efficient zero-shot manipulation.

We introduce DualWAM, a dual-system WAM that addresses this tension by decomposing world-action generation into two complementary stages: global planning and local refinement. Global planning maintains coherent structure over a broader world-action trajectory and operates at a lower frequency, while local refinement focuses on short action windows and incorporates the latest interaction feedback at every control update. Concretely, System 2 periodically applies bidirectional denoising to a broader world-action chunk, progressing from high noise to an intermediate threshold to establish the global structure of the trajectory. At each local control update, System 1 extracts a temporally aligned portion of this intermediate world-action trajectory and continues denoising it over a short local window using the latest wrist observations. The two systems operate asynchronously: each System 2 update serves multiple local control updates, while System 1 continuously incorporates fresh interaction feedback without requiring the complete trajectory to be regenerated.

This decomposition specializes the systems along three aligned dimensions. *Temporally*, System 2 operates at a lower update frequency, while System 1 reacts at every local update. *Generatively*, System 2 organizes a broader world-action state at high noise, whereas System 1 completes low-noise refinement over short local windows. *Perceptually*, System 2 uses multi-view observations, while System 1 relies exclusively on wrist observations that naturally provide action-aligned cues about local geometry, motion, and interaction. These role-specific observation spaces also provide a natural interface for auxiliary embodied data: egocentric videos align with global prediction, while wrist-centric Universal Manipulation Interface (UMI) demonstrations align with local refinement (Chi et al., 2024).

We evaluate DualWAM on zero-shot manipulation tasks using both Franka and Galbot robots, without task-specific finetuning. DualWAM improves success over the strongest evaluated baseline by 4.5 percentage points on average while achieving a 16.6× critical-path speedup. Further studies show that role-matched egocentric and UMI data improve success by 14 percentage points, while the decoupled design reduces average downlink traffic by a factor of 104.6 under edge–cloud execution. Our contributions are threefold. First, we introduce a dual-system WAM that assigns global planning and local refinement to complementary stages of a shared denoising trajectory. Second, we specialize these stages in update frequency, denoising regime, and visual observation space to combine coherent global generation with high-frequency closed-loop correction. Third, we validate the design through controlled ablations and zero-shot evaluation across Franka and Galbot, complemented by auxiliary-data and edge–cloud studies.

2 Related Work

World Action Models. World Action Models (WAMs) couple action generation with predictive modeling of future world states, either as an explicit planning interface or as an auxiliary representation for policy learning (Wang et al., 2026; Zhang et al., 2026b). Cascaded approaches first predict future videos, keyframes, or latent representations and subsequently derive actions from these predictions (Du et al., 2023; Zhou et al., 2024; Liang et al., 2024; Bharadhwaj et al., 2024; Zhang & Du, 2026; Hu et al., 2024; Liang et al., 2025; Pai et al., 2025; Zheng et al., 2026). Joint WAMs instead learn future-state and action prediction within a unified autoregressive, diffusion, or flow-based generative model (Wu et al., 2023; Cheang et al., 2024; Zhu et al., 2025; Kim et al., 2026; Ye et al., 2026; Li et al., 2026a). Recent WAMs improve closed-loop control or reduce inference cost through causal generation, structured predictive states, truncated denoising, compact predictive branches, and image-, latent-, or action-centric decoding (Lyu et al., 2026; Pai et al., 2025; Li et al., 2026b; Zhang et al., 2026c; Tian et al., 2026; Yuan et al., 2026). Despite these advances, most WAMs do not explicitly assign global planning and local correction to generators with different contexts, denoising regimes, observations, and update frequencies. DualWAM introduces this specialization while retaining joint visual-action generation in both systems.

Asynchronous and Role-Specialized Robot Policies. Action chunking and diffusion-based policies provide temporally coherent multi-step control, while online chunk refinement, latent predictive representations, and asynchronous execution improve responsiveness or reduce inference overhead (Zhao et al., 2023; Chi et al., 2023; Black et al., 2025; Sendai et al., 2025; Guo & Guo, 2026; Vanjani et al., 2026; Bai et al., 2026). ABot-M0.5 introduces intermediate latent actions and a dual-level architecture for unified mobility and manipulation (Chen et al., 2026). AHA-WAM uses a low-frequency video DiT to provide reusable context to a high-frequency action DiT (Cai et al., 2026). In contrast, both systems in DualWAM remain WAMs along a shared denoising trajectory, allowing wrist-view dynamics to guide local action refinement. LingBot-VA 2.0 asynchronously predicts future visual latents while re-grounding subsequent predictions on the latest observation (Zhang et al., 2026a), and X-WAM uses asynchronous noise sampling to complete action denoising earlier than multi-view RGB-D prediction (Guo et al., 2026). HarmoWAM coordinates predictive and reactive action experts using world-model features, whereas DSWAM combines a vision-language subtask planner with a WAM executor (Feng et al., 2026; Zhu et al., 2026). DualWAM instead decomposes world-action generation itself: System 2 establishes an overall plan, while wrist-based System 1 repeatedly refines its temporally aligned local portions using fresh observations.

Stage-Specialized Video Generation. Video diffusion models have explored combining full-sequence generation with causal or autoregressive prediction. Diffusion Forcing assigns different noise levels across tokens, Self-Forcing reduces exposure bias through model-generated contexts, and Flex-Forcing unifies bidirectional and autoregressive generation through noise-dependent attention patterns (Chen et al., 2024; Huang et al., 2025; Ma et al., 2026). These methods specialize generation within a single video model. DualWAM adapts the broader principle to robot control by assigning the high- and low-noise portions of a shared world-action trajectory to two WAMs with different temporal and perceptual roles.

3 Method

3.1 Overview and Problem Formulation

At control step t, the robot receives a language instruction l, proprioceptive state q_t, global scene observations o_t^g, and wrist observations o_t^w. Here, o_t^g denotes the head-mounted or external views that capture the overall workspace, whereas o_t^w denotes the wrist-mounted views used for local interaction. Given this context, DualWAM generates a continuous stream of robot actions by assigning complementary portions of one world-action flow trajectory to two WAMs.

Let the clean global world-action target over H_2 video frames be

$$X_0 = [Z_0^g, Z_0^w, A_0], \qquad X_\sigma = (1-\sigma)X_0 + \sigma\epsilon, \quad \epsilon \sim \mathcal{N}(0, I), \tag{1}$$

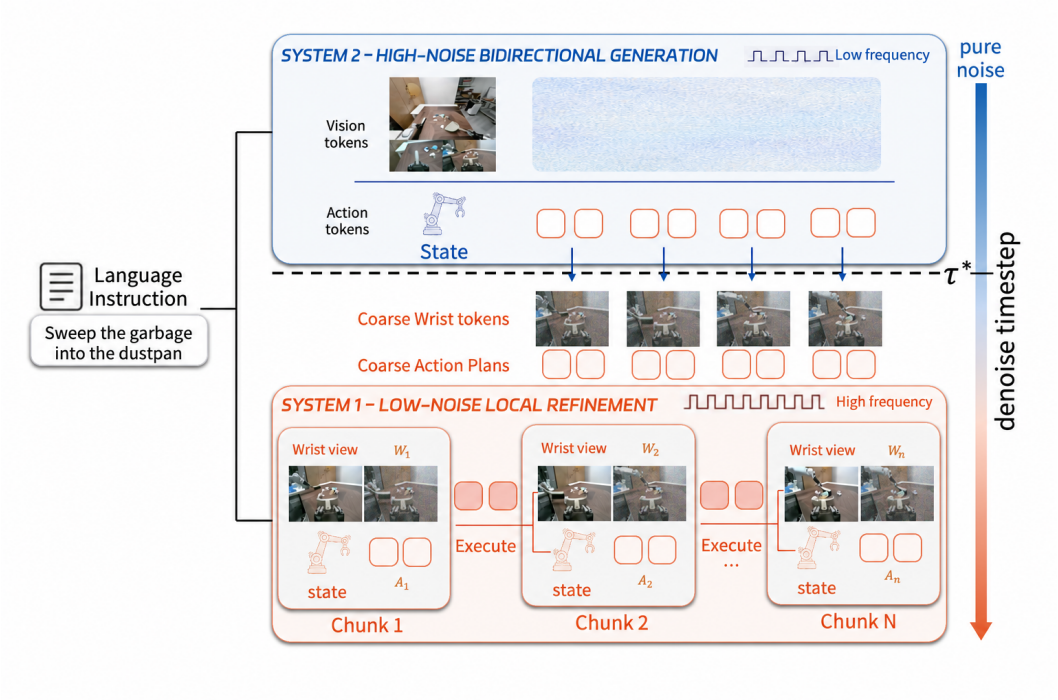

Figure 2: **DualWAM pipeline.** System 2 denoises a complete global world-action state through the high-noise interval. At threshold τ, temporally aligned wrist-action windows are passed to compact System 1, which completes low-noise local refinement using fresh wrist observations.

Table 1: **Complementary roles in DualWAM.** The two WAMs specialize along aligned temporal, generative, and perceptual dimensions.

	System 2	System 1
Role	Global planning	Local refinement
State	$[Z^g, Z^w, A]$	$[Z^w, A]$
Horizon (video frames)	H_2	$H_1 < H_2$
Noise interval	$[\tau, 1]$	$[0, \tau)$
Denoising scope	Global state	Aligned local window
Update rate	Low	High
Observation	Global + wrist	Wrist only

where Z_0^g and Z_0^w are future global-view and wrist-view video latents, A_0 is the temporally aligned action sequence, and $\sigma \in [0, 1]$ is the flow noise level, with $\sigma = 1$ denoting pure noise and $\sigma = 0$ the clean sample. The target rectified-flow velocity is $V = \epsilon - X_0$.

At inference, we select a handoff noise level τ. System 2 denoises the complete global state from $\sigma = 1$ to τ, while System 1 continues from τ to 0 over short local windows. For the k-th execution window, the handoff is

$$U_\tau^k = \mathcal{C}_{s_k}(X_\tau; H_1) = [Z_\tau^{w,k}, A_\tau^k], \tag{2}$$

Here, k indexes local updates; s_k is the video-frame offset from the current global trajectory's start, determined by execution progress. The crop selects wrist latents covering H_1 frames and temporally corresponding actions. At inference, X_τ is generated by System 2 and its selected variables are further denoised by System 1.

3.2 Role-Specialized World Action Models

System 2: global planning. System 2 processes the complete multi-view world-action state over horizon H_2, which provides broader temporal and visual context than a local System 1 window. At high noise levels, the visual and action trajectories remain largely undetermined. Bidirectional attention allows variables across the full prediction horizon to constrain one another, enabling System 2 to form an overall motion plan from global scene context and the available wrist observations. The resulting partially denoised trajectory X_τ is a coarse but coherent joint hypothesis over future visual states and actions; it is not independently decoded into the final executable action sequence.

System 1: short-horizon local refinement. System 1 is a compact wrist-centric WAM that refines one temporally aligned window U_τ^k at a time. At each local update, it combines this window with the latest wrist observations and robot state, then completes low-noise refinement. Because the wrist cameras are rigidly attached to the end effector, motion in the wrist-view video provides an action-aligned visual trace of executed end-effector motion. The same close-range view reveals geometry, occlusion, and contact cues that can be difficult to resolve from a wide scene view. These properties make wrist observations a natural visual input for local refinement, and System 1 therefore operates exclusively on the wrist stream. Within each window, causal temporal attention restricts every position to the current and preceding temporal positions while retaining parallel denoising across the window. As execution advances, newly acquired wrist observations trigger successive local updates, yielding high-frequency closed-loop correction. During training, the first window remains aligned. Later windows receive randomly time-shifted denoising inputs but retain supervision aligned with the current execution window. Shift probability increases with execution progress to model temporal misalignment between global predictions and actual execution.

System 1 is trained throughout the low-noise interval up to τ_{train}, permitting handoff at τ within this range. The denoising objectives are

$$\mathcal{L}_i = \mathbb{E}\left[\sum_{m\in\mathcal{M}_i} \lambda_m^{(i)} \left\| V_m^{(i)} - \hat{V}_m^{(i)}(S_\sigma^{(i)}, \sigma, c_i)\right\|_2^2\right], \quad \begin{array}{ll} \mathcal{I}_2 = [\tau_{\text{train}}, 1], & \mathcal{M}_2 = \{g, w, a\}, \\ \mathcal{I}_1 = (0, \tau_{\text{train}}], & \mathcal{M}_1 = \{w, a\}, \end{array} \tag{3}$$

Here, g, w, a denote global video, wrist video, and actions; c_i contains each system's language, observations, and robot state. For System 2, $(S_\sigma^{(2)}, V^{(2)}) = (X_\sigma, \epsilon - X_0)$. For System 1, $S_\sigma^{(1)}$ is temporally augmented, while velocity supervision $V^{(1)}$ remains aligned with the current window. Modality losses have weights $\lambda_m^{(i)}$. The expectation covers data, Gaussian noise, windows, temporal offsets, and $\sigma \sim p_i(\sigma)$ supported on $\mathcal{I}_i$.

3.3 Asynchronous Closed-Loop Execution

System 2 and System 1 execute independently at their respective update rates. As shown in Figure 3, System 2 periodically updates a partially denoised global world-action trajectory in the background, while System 1 refines the temporally aligned window corresponding to the current executed-action index. Windows whose execution times have elapsed are skipped, preventing stale actions from entering the control stream. This schedule removes high-noise generation from the online action-generation path while preserving alignment between the global state and local execution.

In our inference configuration, System 1 completes the remaining low-noise interval with a single denoising forward pass. This reduces the number of online denoising forward passes from four to one. Table 2 reports measured cumulative latency as deployment is progressively optimized from the original single-GPU baseline to the final three-GPU implementation. Asynchronous execution, parallel classifier-free guidance (Ho & Salimans, 2022), implementation optimizations, and one-step local sampling reduce latency from 2854 to 63 ms. Appendix A.1 provides the implementation details.

4 Experiments

Our experiments address four questions: **(1)** How does DualWAM compare with existing policies on zero-shot real-robot manipulation? **(2)** Does separating global planning and local refinement

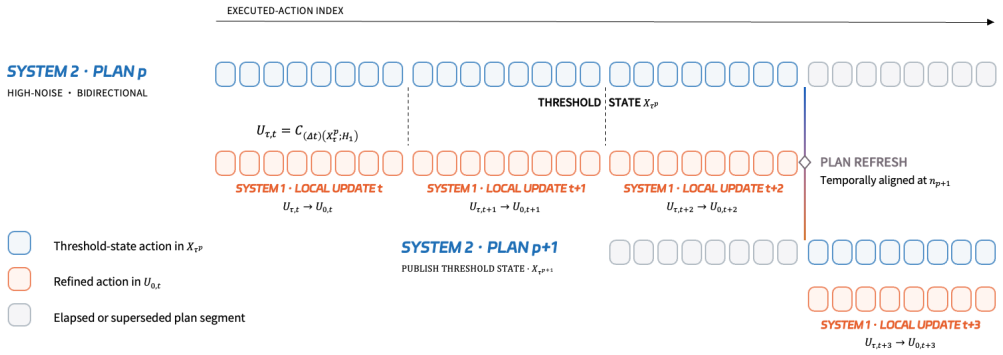

Figure 3: **Asynchronous closed-loop execution.** System 2 updates a partially denoised global world-action trajectory in the background, while System 1 repeatedly refines the window aligned with the current executed-action index.

Optimization	Latency	Cumulative speedup
Original single-GPU 4-step baseline	2854 ms	1.0x
System-level		
+ CFG Parallelism	1679 ms	1.7x
+ Asynchronous Execution	501 ms	5.7x
Implementation-level		
+ Kernel and Scheduling Optimization	423 ms	6.8x
+ Torch Compile + CUDA Graphs	280 ms	10.2x
+ Mixed-Precision Quantization	143 ms	19.9x
Model-level		
+ One-Step Sampling	**63 ms**	**45.3x**

Table 2: **Measured cumulative inference speedups.** Each row includes all preceding optimizations.

into two WAMs improve task performance and inference latency? **(3)** How does asynchronous execution affect closed-loop efficiency, and can the same decomposition support communication-efficient edge–cloud deployment? **(4)** Can role-matched egocentric and UMI data further improve the two systems?

4.1 Experimental Setup

Tasks and metrics. We evaluate zero-shot manipulation on 20 unseen real-robot tasks, with ten tasks on a Galbot G1 and ten on a Franka FR3 following the DROID setup (Khazatsky et al., 2024). Each policy or ablation variant is evaluated for ten trials per task, yielding 100 real-robot rollouts per embodiment. The cross-policy comparison covers both embodiments, while the architecture ablation is conducted on Franka. The edge–cloud and auxiliary-data studies are conducted on Galbot. We define a task as zero-shot when its target objects and evaluation scene do not appear in the robot training tasks and the complete primitive-skill–object–environment combination is absent from training. No task-specific demonstrations or finetuning are used. We report task progress, success rate (SR), critical-path latency, and mean task-completion time over successful rollouts. Task progress is a percentage score based on observable task-specific milestones; SR counts only rollouts that fully complete the task. Completion time measures physical execution time only and is computed as the number of executed control steps divided by the control frequency; policy inference latency is excluded. For latency evaluation, Cosmos3-Nano-Policy uses the same system and implementation optimizations as DualWAM in Table 2, such as multi-GPU classifier-free guidance and quantization; other baselines use their official inference scripts and default sampling configurations on NVIDIA GeForce RTX 4090 GPUs. The complete prompts and progress criteria are provided in the appendix.

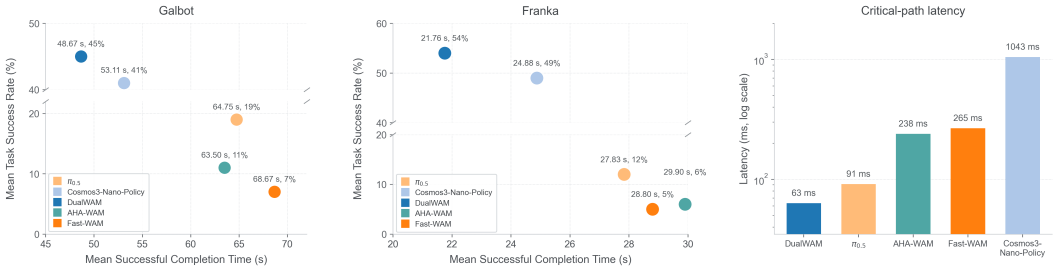

Figure 4: **Zero-shot real-robot performance and critical-path latency.** Left and center: full-task success rate versus mean completion time over successful rollouts on Galbot and Franka, respectively. Completion time is the number of executed control steps divided by the control frequency. Right: critical-path latency on a logarithmic scale. DualWAM attains the highest success rate on both embodiments while providing the lowest latency and completion time.

Implementation details. For the Galbot experiments, we initialize System 2 from Cosmos3-Nano and System 1 from Cosmos3-Edge (NVIDIA, 2026), reinitializing the action-related parameters of both models to accommodate the Galbot action representation. For the Franka experiments, we initialize System 2 directly from Cosmos3-Nano-Policy. System 1 follows the same initialization strategy as in the Galbot setting. Both systems are trained for 10k iterations with a global batch size of 1,024 on eight nodes, each equipped with eight NVIDIA H200 GPUs. Complete implementation details and hyperparameter settings are provided in Appendix A.

Auxiliary data. The heterogeneous-data study uses two self-collected corpora in addition to robot trajectories. The egocentric corpus contains approximately 1,800 hours of human interaction video and is used to train System 2. The wrist-centric UMI corpus contains approximately 1,200 hours of demonstrations and is used to train System 1.

Baselines. We compare four baseline families. $\pi_{0.5}$ is an open-world VLA trained through heterogeneous co-training (Physical Intelligence et al., 2025). Cosmos3-Nano-Policy post-trains the Cosmos3-Nano omnimodal world model into an action policy (NVIDIA, 2026). Fast-WAM retains video co-training but removes future prediction at inference (Yuan et al., 2026). AHA-WAM pairs low-frequency video generation with high-frequency action prediction asynchronously (Cai et al., 2026). For the Franka experiments, we use the publicly released $\pi_{0.5}$ and Cosmos3-Nano-Policy checkpoints trained on DROID, and train Fast-WAM and AHA-WAM using their official scripts and configurations. For the Galbot experiments, we retrain every baseline on the same data with the same batch size, initializing from each method's publicly released bimanual checkpoint when available. We train $\pi_{0.5}$ for 50k optimization steps and all other baselines for 10k steps.

4.2 How Does DualWAM Compare with Existing Policies?

We first assess DualWAM's zero-shot task performance and execution efficiency across Franka and Galbot. Figure 4 shows a consistent advantage across both embodiments. DualWAM achieves the highest success rates, reaching 54% on Franka and 45% on Galbot, outperforming Cosmos3-Nano-Policy by 5 and 4 percentage points, respectively.

Beyond task success, DualWAM also completes successful episodes faster. Mean physical completion time decreases from 24.88 s to 21.76 s on Franka and from 53.11 s to 48.67 s on Galbot. These gains are consistent with DualWAM's combination of global planning and responsive local refinement. Frequent local updates allow execution deviations to be corrected before they accumulate into task failures or require substantial corrective motion, supporting both higher success rates and shorter physical execution times.

At the same time, DualWAM achieves a critical-path latency of only 63 ms, the lowest among all evaluated policies. Compared with 1043 ms for Cosmos3-Nano-Policy, this corresponds to a 16.6× critical-path speedup. Together, these results show that DualWAM improves zero-shot task suc-

cess while enabling more direct closed-loop execution with substantially lower critical-path latency. Detailed per-task results are provided in Figures 7 and 9.

4.3 Ablation of dual-system decomposition and local world-action modeling

We ablate whether global planning and local refinement should be separated and whether the local refiner should retain explicit future prediction. *System 2 only* and *System 1 only* use Cosmos3-Nano-Policy and Cosmos3-Edge-Policy, respectively, as standalone all-view WAMs. The three coupled variants share the same System 1 backbone, wrist observations, action representation, and training data, differing only in future visual modeling: *Action-only DiT S1* predicts only actions, *Fast-WAM S1* retains video–action co-training but removes future-video prediction at inference (Yuan et al., 2026), while DualWAM jointly refines future wrist dynamics and actions from System 2's aligned intermediate trajectory.

Table 3: **Ablation of dual-system decomposition and local world-action modeling on Franka.** DualWAM achieves the highest task progress and success rate at only 63 ms critical-path latency. Performance drops substantially when either the dual-system decomposition or local world-action modeling is removed, highlighting the importance of both global–local specialization and WAM-based local refinement.

Variant	Task Progress ↑	SR ↑	Critical-path Latency (ms) ↓
System 2 only	76.6	49	1043
System 1 only	41.0	9	246
System 2 + Action-only DiT S1	34.0	7	59
System 2 + Fast-WAM S1	47.3	14	62
DualWAM (ours)	81.6	54	63

Compared with *System 2 only*, DualWAM improves task progress from 76.6 to 81.6 and success from 49% to 54%, while reducing latency from 1043 to 63 ms, a 16.6× speedup. *System 1 only* achieves 41.0 task progress and 9% success. Together, these results support combining low-frequency global planning with high-frequency local refinement in two specialized systems.

With System 2 fixed, Action-only DiT and Fast-WAM achieve only 7% and 14% success, respectively, at similar latency to DualWAM. The gain from Action-only DiT to Fast-WAM suggests that video–action co-training provides useful visual-dynamics priors, while the larger gap to DualWAM highlights the additional benefit of retaining explicit future prediction at inference. Jointly refining wrist dynamics and actions grounds local corrections in their expected interaction outcomes, which may be particularly important for zero-shot generalization.

4.4 Asynchronous Execution and Edge–Cloud Deployment

We evaluate whether decoupling the update schedules improves closed-loop execution and provides an efficient edge–cloud interface. The synchronous and asynchronous variants use identical models and denoising schedules; they differ only in whether each local action chunk waits for a fresh System 2 state.

Table 4: **Synchronous versus asynchronous execution on Galbot.** With the same models and denoising schedule, synchronous execution requires a fresh System 2 intermediate state for every System 1 action chunk, whereas asynchronous execution reuses the latest valid state while System 2 runs in the background. Asynchronous execution reduces critical-path latency from 850 ms to 63 ms (13.5×) while improving task progress and success rate.

Execution	Avg. Progress ↑	SR ↑	Critical-path Latency (ms) ↓
Synchronous	73.3	39	850
Asynchronous (ours)	80.6	45	63

Table 5: **Edge–cloud communication on Galbot.** Both methods synchronize at the same rate of 0.469 Hz. DualWAM reduces the downlink payload per synchronization and average downlink bandwidth by 104.6×, while reducing the 10 GbE payload-transfer bound by 18.0× compared with AHA-WAM.

Method	Sync Rate (Hz)	Uplink / Sync ↓	Downlink / Sync ↓	Avg. Bandwidth (Up/Down) ↓	10GbE Transfer Bound ↓
AHA-WAM (Cai et al., 2026)	0.469	0.737 MB	44.2 MB	0.346 / 20.7 MB/s	36.0 ms
DualWAM	0.469	2.07 MB	0.423 MB	0.972 / 0.198 MB/s	2.00 ms

As shown in Table 4, asynchronous execution raises success from 39% to 45% while reducing critical-path latency from 850 to 63 ms, a 13.5× speedup. Background System 2 execution enables responsive local updates, consistent with higher success. Reusing global states may also help stabilize global intent. Table 5 shows a complementary deployment benefit: at the same synchronization rate, the compact wrist-latent and action handoff reduces downlink traffic by 104.6× and the 10 GbE transfer bound by 18.0× relative to AHA-WAM.

4.5 Can DualWAM Benefit from Role-Matched Auxiliary Data?

The specialized observation interfaces also provide natural entry points for heterogeneous data. We augment System 2 with egocentric videos and System 1 with wrist-centric UMI demonstrations, independently and jointly.

Table 6: **Role-matched auxiliary-data ablation on Galbot.** All variants are trained with robot trajectories. When auxiliary data are used, egocentric videos are added only to System 2 and UMI demonstrations only to System 1, each mixed with robot data at a 1:1 ratio. Both sources improve performance individually, and their combination achieves the best result.

Training Data		Performance	
System 2	System 1	Task Progress ↑	SR ↑
Robot	Robot	80.6	45
Robot + Ego	Robot	86.0	54
Robot	Robot + UMI	84.8	51
Robot + Ego	Robot + UMI	**89.6**	**59**

Table 6 shows complementary gains from both sources. Egocentric and UMI data independently raise success to 54% and 51%, while their combination reaches 59%, 14 percentage points above robot-only training. Thus, the global–local specialization can absorb role-aligned data without modifying the architecture.

5 Discussion and Limitations

DualWAM is most useful when manipulation requires both global task organization and frequent local correction. Its advantage may narrow for purely reactive tasks, when one camera captures both global context and contact geometry, or when System 2 fails to establish a useful world-action state. A two-WAM system also costs more to train than a compact policy-only baseline, and deployment depends on System 2 running sparsely enough to preserve the local control rate. Future work should therefore improve training efficiency and robustness to stale or inaccurate global states.

6 Conclusion

We presented DualWAM, which assigns global planning and wrist-centric local refinement to complementary intervals of a shared denoising trajectory. Across zero-shot tasks on Franka and Galbot, this asynchronous dual-WAM design improves success and latency while supporting role-aligned auxiliary data and communication-efficient edge–cloud execution.

AI USE STATEMENT

Generative AI tools were used for language editing and translation, literature retrieval, figure refinement, and feedback on the organization of experimental comparisons. The research questions, method design, implementation decisions, and real-robot experiments were conducted and validated by the authors. All AI-assisted content was manually reviewed, and the authors take full responsibility for the final manuscript.

REPRODUCIBILITY STATEMENT

The main paper and appendix describe the model architecture, training and asynchronous inference procedures, implementation settings, evaluation protocol, task definitions, and task-progress criteria. The appendix further provides per-task results, and the supplementary video provides additional qualitative real-robot rollouts. Anonymous inference code and deployment instructions are available at `https://anonymous.4open.science/r/dualwam-inference-B1D5/`.

REFERENCES

Shuanghao Bai, Jing Lyu, Wanqi Zhou, Zhe Li, Dakai Wang, Lei Xing, Xiaoguang Zhao, Pengwei Wang, Zhongyuan Wang, Cheng Chi, et al. Latent reasoning vla: Latent thinking and prediction for vision-language-action models. *arXiv preprint arXiv:2602.01166*, 2026.

Homanga Bharadhwaj, Debidatta Dwibedi, Abhinav Gupta, Shubham Tulsiani, Carl Doersch, Ted Xiao, Dhruv Shah, Fei Xia, Dorsa Sadigh, and Sean Kirmani. Gen2act: Human video generation in novel scenarios enables generalizable robot manipulation. *arXiv preprint arXiv:2409.16283*, 2024.

Kevin Black, Manuel Y. Galliker, and Sergey Levine. Real-time execution of action chunking flow policies. *arXiv preprint arXiv:2506.07339*, 2025.

Jisong Cai, Long Ling, Shiwei Chu, Zhongshan Liu, Jiayue Kang, Zhixuan Liang, Wenjie Xu, Yinan Mao, Weinan Zhang, Xiaokang Yang, et al. Aha-wam: Asynchronous horizon-adaptive world-action modeling with observation-guided context routing. *arXiv preprint arXiv:2606.09811*, 2026.

Chi-Lam Cheang, Guangzeng Chen, Ya Jing, Tao Kong, Hang Li, Yifeng Li, Yuxiao Liu, Hongtao Wu, Jiafeng Xu, Yichu Yang, et al. Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation. *arXiv preprint arXiv:2410.06158*, 2024.

Boyuan Chen, Diego Marti Monso, Yilun Du, Max Simchowitz, Russ Tedrake, and Vincent Sitzmann. Diffusion forcing: Next-token prediction meets full-sequence diffusion. *arXiv preprint arXiv:2407.01392*, 2024.

Ronghan Chen, Yandan Yang, Zuojin Tang, Dongjie Huo, Tong Lin, et al. ABot-M0.5: Unified mobility-and-manipulation world action model. *arXiv preprint arXiv:2607.00678*, 2026.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

Yilun Du, Mengjiao Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Joshua B. Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *arXiv preprint arXiv:2302.00111*, 2023.

Qiuxuan Feng, Jiale Yu, Jiaming Liu, Yueru Jia, Zhuangzhe Wu, Hao Chen, Zezhong Qian, Shuo Gu, Peng Jia, Siwei Ma, and Shanghang Zhang. Harmowam: Harmonizing generalizable and precise manipulation via adaptive world action models. *arXiv preprint arXiv:2605.10942*, 2026.

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.

Tiecheng Guo and Meng Guo. Action ControlNet: A lightweight delay-aware adapter for smooth asynchronous control in vision-language-action models. *arXiv preprint arXiv:2606.25985*, 2026.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

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. *arXiv preprint arXiv:2412.14803*, 2024.

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

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

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.

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*, 2026a.

Ziang Li, Dongzhou Cheng, Yibin Wang, Shiyue Wang, Xiaoyang Xu, Lingxuan Weng, Juan Wang, and Jiaqi Wang. Light-wam: Efficient world action models with state-fusion action decoding. *arXiv preprint arXiv:2606.08242*, 2026b.

Junbang Liang, Ruoshi Liu, Ege Ozguroglu, Sruthi Sudhakar, Achal Dave, Pavel Tokmakov, Shuran Song, and Carl Vondrick. Dreamitate: Real-world visuomotor policy learning via video generation. *arXiv preprint arXiv:2406.16862*, 2024.

Junbang Liang, Pavel Tokmakov, Ruoshi Liu, Sruthi Sudhakar, Paarth Shah, Rares Ambrus, and Carl Vondrick. Video generators are robot policies. *arXiv preprint arXiv:2508.00795*, 2025.

Jiangran Lyu, Kai Liu, Xuheng Zhang, Haoran Liao, Yusen Feng, Wenxuan Zhu, Tingrui Shen, Jiayi Chen, Jiazhao Zhang, Yifei Dong, et al. Lda-1b: Scaling latent dynamics action model via universal embodied data ingestion. *arXiv preprint arXiv:2602.12215*, 2026.

Xinyin Ma, Julius Berner, Chao Liu, Arash Vahdat, Weili Nie, and Xinchao Wang. Flexforcing: Towards a unified autoregressive and bidirectional video diffusion model. *arXiv preprint arXiv:2607.03509*, 2026.

NVIDIA. Cosmos 3: Omnimodal world models for physical AI. *arXiv preprint arXiv:2606.02800*, 2026.

Jonas Pai, Liam Achenbach, Victoriano Montesinos, Benedek Forrai, Oier Mees, and Elvis Nava. mimic-video: Video-action models for generalizable robot control beyond VLAs. *arXiv preprint arXiv:2512.15692*, 2025.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. *arXiv preprint arXiv:2504.16054*, 2025.

Kohei Sendai, Maxime Alvarez, Tatsuya Matsushima, Yutaka Matsuo, and Yusuke Iwasawa. Leave no observation behind: Real-time correction for VLA action chunks. *arXiv preprint arXiv:2509.23224*, 2025.

Yuxuan Tian, Yurun Jin, Bin Yu, Yukun Shi, Hao Wu, Chi Harold Liu, Kai Chen, and Cong Huang. Starry: Spatial-temporal action-centric world modeling for robotic manipulation. *arXiv preprint arXiv:2604.26848*, 2026.

Pankhuri Vanjani, Zhuoyue Li, Jakub Suliga, Moritz Reuss, Gianluca Geraci, Xinkai Jiang, and Rudolf Lioutikov. DAM-VLA: Decoupled asynchronous multimodal vision language action model. *arXiv preprint arXiv:2606.12105*, 2026.

Siyin Wang, Junhao Shi, Zhaoyang Fu, Xinzhe He, Feihong Liu, Chenchen Yang, Yikang Zhou, Zhaoye Fei, Jingjing Gong, Jinlan Fu, Mike Zheng Shou, Xuanjing Huang, Xipeng Qiu, and Yu-Gang Jiang. World action models: The next frontier in embodied AI. *arXiv preprint arXiv:2605.12090*, 2026.

Hongtao Wu, Ya Jing, Chi-Lam Cheang, Guangzeng Chen, Jiafeng Xu, Xinghang Li, Minghuan Liu, Hang Li, and Tao Kong. Unleashing large-scale video generative pre-training for visual robot manipulation. *arXiv preprint arXiv:2312.13139*, 2023.

Gangwei Xu, Qihang Zhang, Jiaming Zhou, Xing Zhu, Yujun Shen, Xin Yang, and Yinghao Xu. Next forcing: Causal world modeling with multi-chunk prediction, 2026. URL `https://arxiv.org/abs/2606.11187`.

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

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.

Qihang Zhang, Lin Li, Luyao Zhang, Shuai Yang, Yiming Luo, et al. Native video-action pretraining for generalizable robot control. *arXiv preprint arXiv:2607.08639*, 2026a.

Xiangcheng Zhang and Yilun Du. World action planner: Generalizable decision-making with action-conditioned world models. *arXiv preprint arXiv:2607.27599*, 2026.

Xiaoxiong Zhang, Xiong Zeng, and Wei Zhang. From world models to world action models: A concise tutorial for robotics. *arXiv preprint arXiv:2607.00836*, 2026b.

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*, 2026c.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

Yixin Zheng, Jiangran Lyu, Yifan Zhang, Jiayi Chen, Mi Yan, Yuntian Deng, Xuesong Shi, Xiaoguang Zhao, Yizhou Wang, Zhizheng Zhang, et al. Emerging extrinsic dexterity in cluttered scenes via dynamics-aware policy learning. *arXiv preprint arXiv:2603.09882*, 2026.

Siyuan Zhou, Yilun Du, Jiaben Chen, Yandong Li, Dit-Yan Yeung, and Chuang Gan. Robodreamer: Learning compositional world models for robot imagination. *arXiv preprint arXiv:2404.12377*, 2024.

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. *arXiv preprint arXiv:2504.02792*, 2025.

Jian Zhu, Jianjun Zhang, Taiyi Su, Tianbin Liu, Zhangyuan Wang, Kai Xie, Zitai Huang, Chong Ma, Youzhang He, Tianjian Wang, et al. Dswam: A dual-system world action foundation model for fine-grained robot manipulation. *arXiv preprint arXiv:2607.04927*, 2026.

A Implementation Details

Table 7 summarizes the model, training, and deployment settings used in our experiments. Both systems are trained on complementary intervals of the same rectified-flow timestep axis. At deployment, the partially denoised trajectory produced by System 2 is divided into temporally aligned windows that System 1 continues denoising.

Table 7: **Implementation settings.**

Setting	Value
Global horizon H_2	32 frames
Local horizon H_1	8 frames
Training low-noise upper bound τ_{train}	0.7
System 2 refresh interval	32 control frames
Optimizer	AdamW
Learning rate	2×10^{-4}
Training precision	bfloat16
Training iterations	10k
Global batch size	1,024
Training hardware	8 nodes $\times$ 8 NVIDIA H200 GPUs
Deployment hardware	3 NVIDIA GeForce RTX 4090 GPUs

A.1 Inference Implementation

The latency optimizations in Table 2 are applied cumulatively. For classifier-free guidance, the conditional and unconditional System 2 branches run in parallel on separate GPUs. The runtime uses cuDNN matrix multiplication, fused attention and MLP epilogues, resident GPU intermediates, and batched stage transitions to reduce kernel launches, synchronization, and host–device transfers. Fixed text keys and values and the 28-layer generation stack are captured in an end-to-end CUDA graph. Finally, the System 1 decoder applies W8A8 quantization to numerically stable projection layers while retaining BF16 for accuracy-sensitive projections. After the single-GPU baseline, all configurations use three NVIDIA GeForce RTX 4090 GPUs.

Logging. Every rollout stores planner-view frames, local wrist frames, the world-action trajectory at the noise threshold, completed wrist-action windows, executed actions, global-trajectory age, denoising steps, and per-module latency. These logs distinguish failures caused by poor global organization, stale global trajectories, and local execution errors.

B Galbot Training-Data Coverage

Our self-collected Galbot robot corpus contains 43,058 annotated manipulation subtasks spanning 74 skill labels. Figure 5 shows a long-tailed distribution: common skills provide broad supervision for reusable manipulation primitives, while the remaining labels increase the diversity of interaction patterns observed during training.

C Zero-Shot Evaluation Details

We evaluate ten unseen tasks on each embodiment and report Franka and Galbot separately, following DreamZero (Ye et al., 2026). Each rollout receives a normalized task-progress score in $[0, 1]$ from observable, task-specific milestones. Full completion is assigned 1.0 and counted as a success; incomplete rollouts receive the partial credit specified below. For multi-object tasks, credit accumulates across completed subgoals and is capped at 1.0. We average rollout scores to obtain task progress, whereas SR is the fraction of rollouts with a score of 1.0. This distinction rewards meaningful partial execution without conflating it with full-task completion.

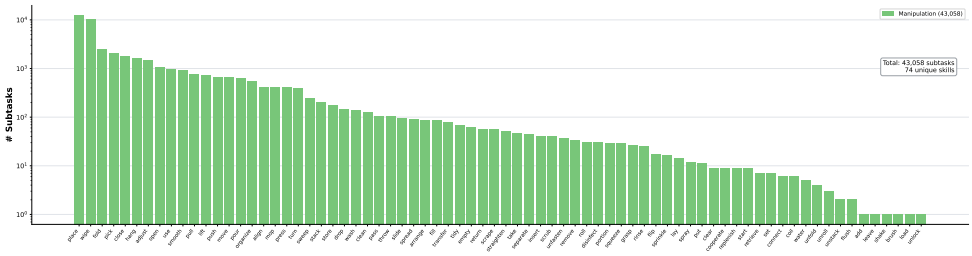

Figure 5: **Skill distribution in the self-collected Galbot robot corpus.** Bars show the number of annotated subtasks for each of 74 manipulation skill labels on a logarithmic scale, totaling 43,058 subtasks.

C.1 Franka Tasks

Figure 6 shows the initial configurations used for the ten Franka tasks. They cover object discovery, relational matching, tool use, articulated-object interaction, and spatial reorientation. Table 8 gives the exact language prompt and the observable milestones used for task-progress scoring.

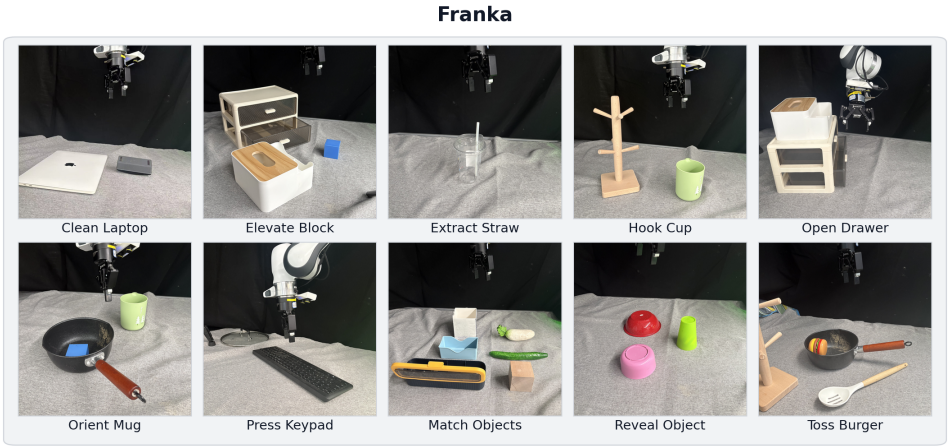

Figure 6: **Franka zero-shot task gallery.** Initial configurations for the ten evaluation tasks.

Table 8: **Franka task prompts and progress criteria.** Partial scores are assigned only when full completion is not reached.

Task Name	Prompt	Task-Progress Criteria
Reveal Object	Reveal the object under the cup.	0.33: touch the correct cup; 0.67: move it; 1.0: reveal the hidden object.
Match Objects	Match the objects with their corresponding boxes based on length.	+0.16: pick an object without placing it; +0.16: place each object in its correct box; 1.0: match all objects.
Elevate Block	Elevate the block to the highest platform.	0.33: pick the block; 0.67: place it on a platform; 1.0: place it on the highest platform.
Extract Straw	Extract the straw from the cup.	0.33: touch the straw; 0.67: move it; 1.0: fully extract it from the cup.
Toss Burger	Toss the burger in the pan.	0.33: grasp the pan handle; 0.67: move the burger; 1.0: toss the burger in the pan.
Orient Mug	Orient the mug with its handle pointing toward the pot.	0.33: touch the mug; 0.67: rotate it; 1.0: orient the handle toward the pot.
Open Drawer	Open the top drawer.	0.33: touch the drawer; 0.67: open it halfway; 1.0: fully open it.
Hook Cup	Hook the cup onto the rack.	0.25: touch the cup; 0.50: pick it up; 0.75: approach the rack; 1.0: hook the cup securely.
Press Keypad	Press the number pad and Enter on the keyboard.	0.50: press the keyboard; 1.0: press the requested keys correctly.
Clean Laptop	Clean the laptop with a brush.	0.33: pick up the brush; 0.67: bring it to the laptop; 1.0: brush the laptop surface.

Figure 7 disaggregates the aggregate result in the main paper. DualWAM matches or exceeds the strongest baseline across all ten tasks, with particularly large gains on block elevation, straw extraction, and drawer opening, where maintaining a global objective while correcting local interactions is important.

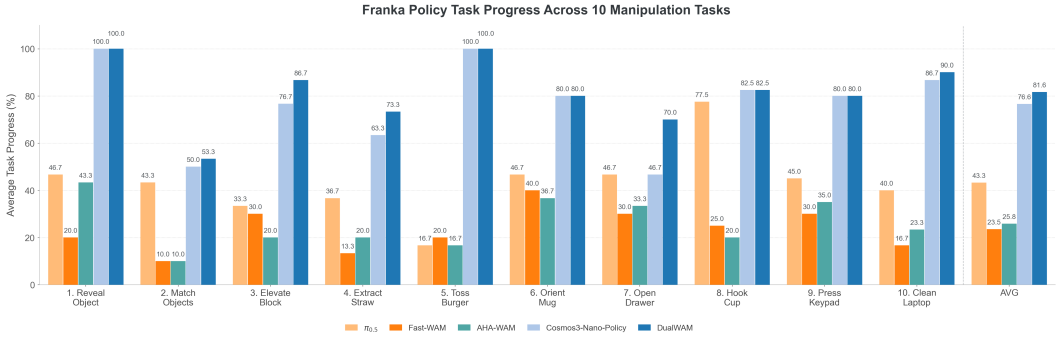

Figure 7: **Per-task progress on Franka.** Mean normalized progress for each policy on the ten zero-shot tasks.

C.2 GALBOT TASKS

The Galbot suite in Figure 8 emphasizes bimanual and contact-rich behaviors, including tool use, pouring, sweeping, untying, and articulated-object manipulation. The corresponding prompts and scoring milestones are listed in Table 9.

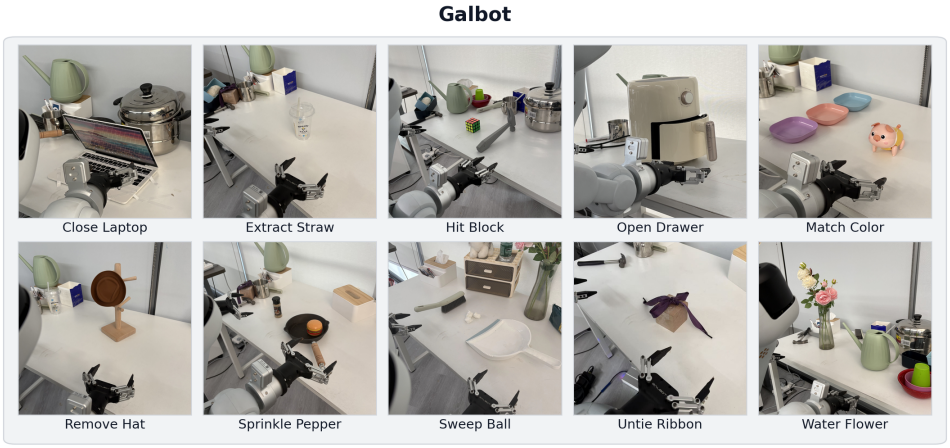

Figure 8: **Galbot zero-shot task gallery.** Initial configurations for the ten evaluation tasks.

Figure 9 shows that DualWAM's improvement is distributed across diverse interaction types rather than being driven by a single task. The largest margins appear on color matching, ribbon untying, and sweeping, which benefit from repeated local correction over a longer task structure.

Table 9: **Galbot task prompts and progress criteria.** Milestones reflect increasing completion of the commanded interaction.

Task Name	Prompt	Task-Progress Criteria
Water Flower	Water the flower.	0.33: touch the kettle handle; 0.67: lift the kettle; 1.0: position it over the flower and perform a pouring motion.
Hit Block	Hit the block with the hammer.	0.33: touch the hammer; 0.67: lift it; 1.0: strike the block.
Remove Hat	Remove the hat from the hook.	0.33: touch the hat; 0.67: grasp it; 1.0: remove it from the hook.
Match Color	Put the toy pig into the corresponding plate based on its color.	0.25: touch the toy; 0.50: pick it up; 0.75: place it on an incorrect plate; 1.0: place it on the color-matched plate.
Extract Straw	Extract the straw from the cup.	0.25: touch the straw; 0.50: lift it; 0.75: move it without successful extraction; 1.0: fully extract it.
Sprinkle Pepper	Sprinkle pepper on the burger.	0.25: touch the shaker; 0.50: grasp it; 0.75: move it over the burger; 1.0: tilt and sprinkle.
Close Laptop	Close the laptop.	0.25: touch the lid; 0.50: initiate closing; 0.75: partially close it; 1.0: close it fully.
Untie Ribbon	Untie the ribbon on the gift.	0.20: touch with one hand; 0.40: touch with both hands; 0.60: grasp with one hand; 0.80: grasp with both; 1.0: pull the ribbon open.
Open Drawer	Open the lower drawer of the cooker.	0.25: touch the handle; 0.50: grasp it; 0.75: open halfway; 1.0: fully open the drawer.
Sweep Ball	Sweep the paper balls into the dustpan.	0.20: touch the brush; 0.40: lift it; 0.60/0.80/1.0: sweep one/two/three balls into the dustpan.

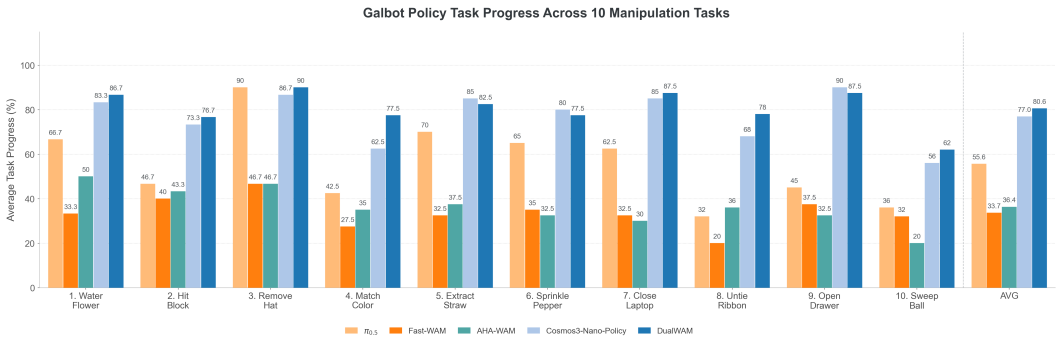

Figure 9: **Per-task progress on Galbot.** Mean normalized progress for each policy on the ten zero-shot tasks.