SPARSE-WAM: ACCELERATING WORLD ACTION MODELS VIA ACTION-GUIDED SPARSE IMAGINATION

Anonymous authors
Paper under double-blind review

ABSTRACT

World-action models (WAMs) leverage pretrained video models to improve generalization in robot control by jointly predicting future visual states and actions. This capability comes at a substantial inference cost, as dense future-frame tokens are repeatedly processed during denoising. Prior methods address this by token pruning that prioritizes visual fidelity to reduce denoising costs in video diffusion models. However, these methods do not use action relevance to determine which future-frame tokens to retain during joint denoising in WAMs. In this paper, we propose **Sparse-WAM**, a training-free framework for *action-guided sparse imagination* that selectively processes future-frame tokens to accelerate WAM inference. We observe substantial overlap in the spatial distribution of attention from action tokens to future-frame tokens (action-to-future attention) between consecutive denoising steps, despite continued updates to the future representations. Motivated by this, we develop *Action-Guided Token Selection* to retain frame-specific action-relevant regions together with cross-frame context. However, a naive implementation can incur attention-scoring and token-packing overhead that offsets the computational savings from pruning. We therefore introduce **Pilot**, an efficient engine that reduces sparse inference overhead through lightweight scoring and cross-step reuse of token selections. On LIBERO with FastWAM-Joint and RoboLab-120 with Cosmos 3 Edge, **Sparse-WAM** achieves inference speedups of approximately $2.0\times$ and $1.8\times$, respectively, over dense eager inference on an NVIDIA RTX 4090, while largely preserving task performance.

1 INTRODUCTION

World action models (WAMs) have emerged as a promising paradigm for robotic control. Recent WAMs such as DreamZero (Ye et al., 2026b) and Cosmos 3 (NVIDIA, 2026) jointly denoise future visual states and actions, leveraging spatiotemporal priors to improve generalization and robustness (Zhang et al., 2026). However, explicitly generating high-resolution future frames introduces a large number of visual tokens into each denoising step. Since the cost of each denoising step grows significantly with the number of processed tokens (Anagnostidis et al., 2025), repeatedly processing these future-frame tokens increases inference latency and limits control frequency (Guo et al., 2024).

To reduce this repeated computation, existing methods either remove imagination at inference or compress future-frame tokens. The former uses future imagination during training but omits explicit future prediction during action generation (Yuan et al., 2026; Li et al., 2026b; Zhao et al., 2026a). The latter retains future prediction with fewer or lower-fidelity visual tokens (Chen et al., 2026; Li et al., 2026a), but still devotes computation to regions with limited relevance to actions.

Token pruning offers a fine-grained means of reducing the cost of processing dense imagination tokens by selectively retaining a subset for computation. This paradigm has been explored in VLAs to retain useful information from observed inputs (Xu et al., 2025; Wang et al., 2026) and in video generation to preserve visual quality (Zou et al., 2025; Feng et al., 2026). In WAMs, however, future-frame tokens are initialized from noise and jointly denoised with action tokens, making action-relevant regions difficult to determine in advance.

To understand how action-relevant regions evolve during denoising, we examine attention from action tokens to future-frame tokens (see subsection 3.2). We observe that, despite continued changes in future representations, the regions attended to by action tokens exhibit substantial spatial overlap

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.

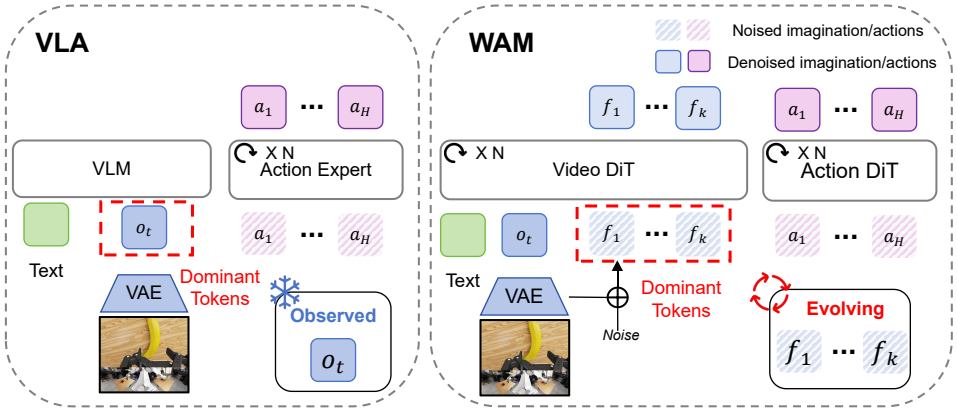


Figure 1: Representative architectures of VLA and WAM. In WAM, noisy imagination tokens constitute the majority of the visual-action sequence and evolve throughout denoising.

between consecutive denoising steps. This consistency offers an opportunity to reuse token selections across steps, but does not imply that the relevant regions remain unchanged throughout denoising. These observations motivate using action-to-future attention to select future regions for sparse computation and reusing the selections across denoising steps. However, attention scoring and token reorganization introduce additional overhead that can offset the computational savings. The core challenge is therefore to identify useful future regions while keeping the cost of selection and sparse execution low.

To address this challenge, we introduce **Sparse-WAM**, a training-free framework for action-guided sparse imagination in WAMs. It uses action-to-future attention to retain frame-specific regions together with shared spatial context. An efficient execution engine combines lightweight scoring with cross-step selection reuse to reduce the cost of joint visual–action inference.

We summarize our contributions as follows:

- We reveal substantial spatial overlap in action-to-future attention between consecutive denoising steps, despite continued changes in future representations. This finding motivates reusing action-guided token selections during joint denoising.
- We propose **Sparse-WAM**, which uses temporally aligned action-to-future attention to select frame-specific regions together with shared spatial context. It concentrates Transformer computation on selected future tokens while retaining all observation and action tokens.
- We develop **Pilot**, an efficient engine for online token selection and sparse execution. It combines lightweight attention profiling with cross-step reuse of selected positions and packing metadata, reducing the overhead of action-guided sparse inference.
- We evaluate Sparse-WAM on three WAMs across LIBERO (Liu et al., 2023), RoboLab-120 (Yang et al., 2026), LIBERO-Plus (Fei et al., 2025), and real-world robotic tasks. On LIBERO, Sparse-WAM achieves a $1.98\times$ speedup over dense eager inference with a 0.30 percentage-point decrease in average task success.

2 Related Work

Efficient World Action Models. Repeated denoising of future-frame tokens makes diffusion-based WAM inference computationally expensive (Ye et al., 2026b; Li et al., 2025; Bi et al., 2025). One line of work retains future prediction during training but generates actions directly from learned world representations at inference (Yuan et al., 2026; Ye et al., 2026a; Li et al., 2026b;c). Another preserves inference-time future modeling while reducing its cost through compact latent subgoals, lower-resolution representations, asymmetric denoising, or feature reuse (Chen et al., 2026; Li et al., 2026a; Zhao et al., 2026a). Our method retains joint visual–action denoising and selectively allocates future-frame token computation according to action relevance, without additional model training.

Token-Level Caching and Pruning. Token caching and pruning reduce inference cost through feature reuse and token removal. VLA methods use observation similarity and task or action cues (Pei et al., 2025; Liu et al., 2025; Ma et al., 2026), as exemplified by VLA-Cache and SpecPrune-VLA (Xu et al., 2025; Wang et al., 2026). In video generation and diffusion world models, selective computation aims to preserve generation quality (Liu et al., 2026; Zhang et al., 2025a). ToCa uses token importance for caching (Zou et al., 2025), while WorldCache exploits trajectory predictability for reuse and extrapolation (Feng et al., 2026). However, observed-input relevance and visual fidelity do not directly determine which evolving future regions support action prediction. Our method therefore uses action-to-future attention to guide future-frame token selection and reuses selected positions across denoising steps.

3 Preliminaries and Motivation

3.1 World Action Models

As shown in Figure 1, WAMs with a shared Transformer sequence jointly denoise future-frame tokens $\mathbf{z}_{\mathrm{v}}$ and an H-step action chunk $\mathbf{a}_{1:H}$ (Ye et al., 2026b; NVIDIA, 2026). Conditioned on the current observation, task instruction, and robot state, successive denoising steps refine the same predicted future and action chunk. During this process, action tokens attend to future-frame tokens, allowing information from the predicted future to inform action updates. Reducing future-frame computation therefore calls for understanding how action tokens access this information.

Action-to-Future Attention. Attention from action queries to future-frame keys provides a token-level view of this interaction. For a predicted future comprising F latent frames with N_s tokens per frame, let $\mathbf{z}_{f,j}$ denote the future-frame token at spatial position j in frame f, and $\mathbf{a}_i$ the i-th action token. At Transformer layer ℓ and denoising step τ (starting from $\tau = 0$), let $A^{(\tau)}_{\ell,h}(i, f, j)$ denote the attention weight from the query of action token $\mathbf{a}_i$ to the key of future-frame token $\mathbf{z}_{f,j}$ in head h. These weights retain the normalization over all keys visible to each action query. We define action-to-future attention by averaging these weights across heads:

$$U^{(\tau)}_{\ell}(i, f, j) = \frac{1}{N_h} \sum_{h=1}^{N_h} A^{(\tau)}_{\ell,h}(i, f, j), \tag{1}$$

where N_h is the number of attention heads, $i = 1, \ldots, H$ indexes action tokens, $f = 1, \ldots, F$ indexes future frames, and $j = 1, \ldots, N_s$ indexes spatial tokens within each frame. A larger value indicates stronger attention from action token i to future-frame token (f, j). We use this attention as a low-cost proxy for token relevance when allocating sparse computation. We measure cross-frame and cross-step overlap by summing the smaller of the two normalized attention values at each spatial position (see Appendix C.4).

3.2 Observations

This section examines how action-to-future attention is distributed across future frames and spatial positions, and how these patterns vary across Transformer layers and denoising steps. We conduct this analysis on Cosmos3-Edge and Cosmos3-Nano using RoboLab (Yang et al., 2026).

Action–Frame Alignment and Spatial Variation. We identify two complementary patterns in action-to-future attention. First, Figure 2 (Insight 1(a)) shows an approximately diagonal band in the frame-level attention map: each future frame predominantly receives attention from a contiguous group of roughly H/F action queries. Queries near group boundaries also attend to neighboring frames, indicating that the alignment extends across temporal group boundaries. Second, Figure 2 (Insight 1(b)) shows that attention hotspots shift spatially across future frames. These patterns motivate scoring each future frame using its temporally associated action queries and selecting spatial regions separately across frames.

Cross-Frame Context Consistency. We further compare cross-frame attention overlap with and without hotspot regions. As shown in Figure 2 (Insight 2), overlap increases after hotspot exclusion,

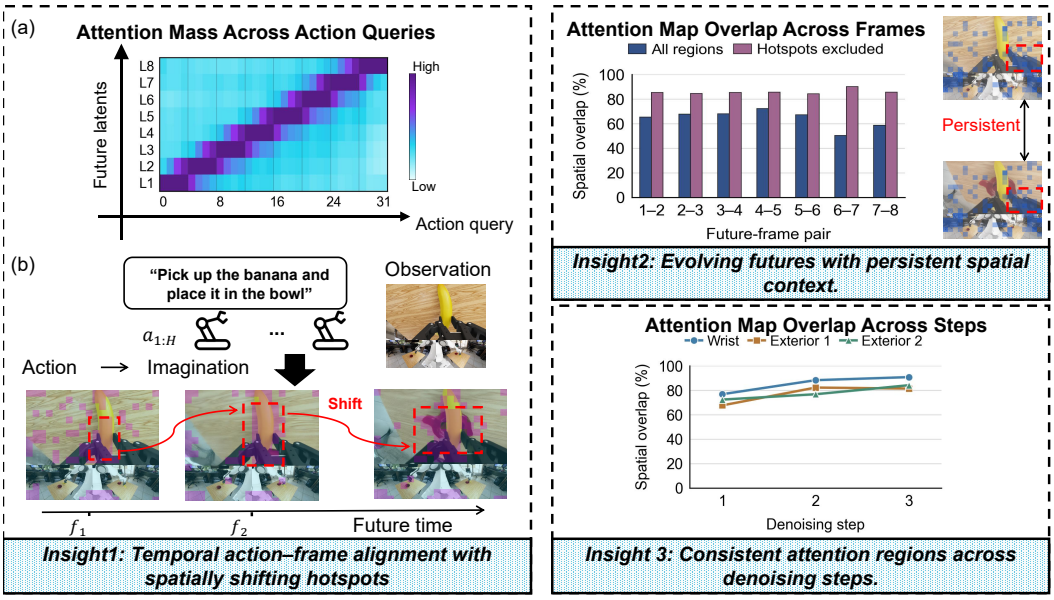


Figure 2: Insight 1: (a) Action queries exhibit temporal alignment with future frames, with attention extending to neighboring frames near temporal group boundaries. (b) Action-relevant hotspots shift spatially across future frames. Insight 2: Excluding hotspots yields higher attention overlap between adjacent frames. Insight 3: Attention maps exhibit substantial spatial overlap between consecutive denoising steps in all three camera views.

indicating greater consistency in the spatial distribution of the remaining attention. The visualizations also show attention to contextual regions, including parts of the gripper outside the object-contact area. Together, these observations suggest that persistent spatial context complements frame-specific hotspots, motivating shared anchor positions alongside per-frame token selection.

Cross-Step Attention Consistency. We next examine how spatial attention distributions change across denoising steps that refine the same predicted future. As shown in Figure 2 (Insight 3), the distributions exhibit substantial overlap between consecutive steps in each of the three camera views. Thus, future representations continue to evolve while their spatial attention distributions remain relatively consistent. This observation motivates reusing token selections across denoising steps to reduce repeated selection overhead. Appendix C.4 defines the overlap measure and provides cross-step and cross-frame comparisons with attention visualizations.

4 Methodology

As illustrated in Figure 3, Sparse-WAM uses action-to-future attention to allocate computation within imagined futures. In subsection 4.1, we combine frame-specific core tokens with shared spatial anchors to retain changing attention hotspots and persistent context. In subsection 4.2, we introduce **Pilot**, an engine that combines lightweight attention profiling with cross-step reuse to execute these selections efficiently.

4.1 Online Action-Guided Token Selection

As discussed in Figure 2, the token selection needs to follow the hotspots of each future frame while retaining contextual regions shared across frames. We address these complementary needs with K_c frame-specific core tokens and K_s shared spatial anchors per frame, where $K_c + K_s \leq N_s$. Selection uses attention from the dense conditional forward at a full denoising step τ_d. For brevity, we write $U_\ell(i, f, j) = U_\ell^{(\tau_d)}(i, f, j)$ below.

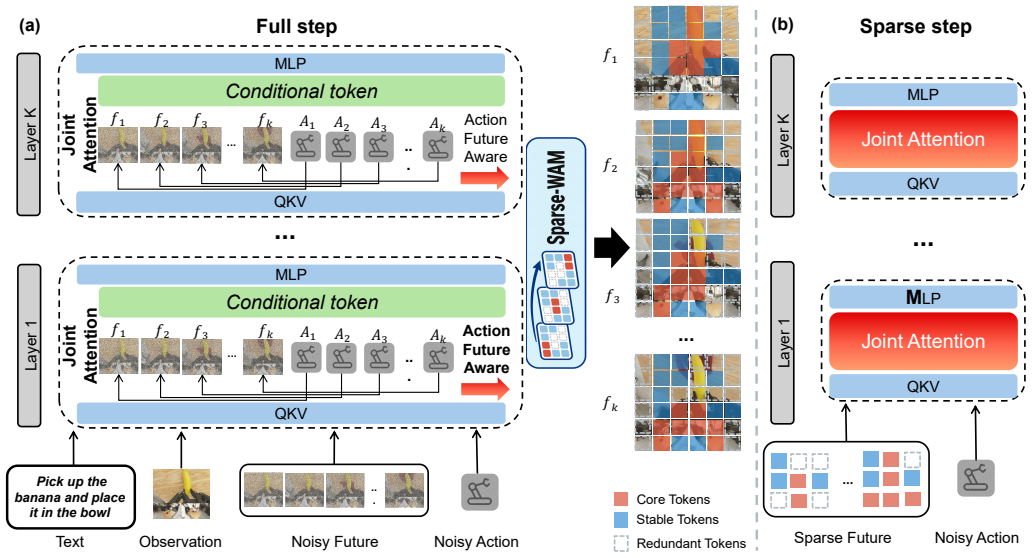


Figure 3: Overview of Sparse-WAM. Future-frame token positions are selected using action-to-future attention and reused during sparse denoising.

Scoring Action-Relevant Regions. The temporal alignment in Figure 2 (Insight 1(a)) suggests that each future frame should be scored using its associated action queries. Assuming that H is divisible by F, we partition the H queries into F consecutive groups of size H/F, denoting the group for frame f by $\mathcal{I}_f$. The spatial score at layer ℓ is

$$S_\ell(f,j) = \sum_{i \in \mathcal{I}_f} \alpha_{f,i} U_\ell(i,f,j), \tag{2}$$

where the nonnegative weights $\alpha_{f,i}$ sum to one. Central queries receive larger weights because queries near group boundaries also attend to neighboring frames.

To identify layers whose attention provides localized signals for token selection, we assign each layer a score $Q_\ell = R_\ell(1-E_\ell)$. Here, R_ℓ measures frame-aligned attention mass, and E_ℓ is the normalized spatial entropy averaged across frames (Zhang et al., 2025b). We select the K_{layer} highest-scoring layers, forming $\mathcal{L}_{\text{core}}$, and aggregate their spatial scores:

$$V(f,j) = \frac{\sum_{\ell \in \mathcal{L}_{\text{core}}} Q_\ell S_\ell(f,j)}{\sum_{\ell \in \mathcal{L}_{\text{core}}} Q_\ell}. \tag{3}$$

Appendix A.1 provides the query-group definition and layer-quality calculations.

Frame-Specific Core Tokens. Since attention hotspots shift across future frames, each frame selects its own core positions using $V(f,j)$. We first identify positions with high core scores in at least one future frame, then select frame-specific tokens from this common candidate pool.

Let $\mathcal{J} = \{1,\dots,N_s\}$ denote all spatial positions. The candidate pool contains $N_s - K_s$ positions, preserving capacity for the shared anchors. We construct the pool and select core positions as

$$\mathcal{P} = \text{TopK}_{j \in \mathcal{J}} \left(\max_f V(f,j),\ N_s - K_s \right), \tag{4}$$

$$\mathcal{C}_f = \text{TopK}_{j \in \mathcal{P}} \left(V(f,j), K_c \right), \tag{5}$$

where TopK returns the indices of the largest scores. The maximum across frames determines the common candidate pool, while each frame's own scores determine its core selection, allowing the retained positions to follow spatially shifting hotspots.

Shared Spatial Anchor Tokens. The increased cross-frame attention overlap after hotspot exclusion (Figure 2, Insight 2) suggests that contextual regions receive more consistent attention across future frames. We therefore complement frame-specific core tokens with shared spatial anchors.

To avoid duplicating core positions, anchors are selected from $\mathcal{E}$, the positions not selected as core tokens in any frame. Because all core selections lie within the common pool $\mathcal{P}$ of size $N_s - K_s$, at least K_s positions remain available for anchors. Appendix A.1 provides the formal candidate-set definition and availability guarantee.

To rank these candidates, we aggregate $S_\ell(f, j)$ over all layers using the quality weights Q_ℓ, including signals beyond the layers selected for core localization. Let μ_j and CV_j denote the cross-frame mean and coefficient of variation of this aggregated score at position j. We select positions with strong and consistent attention:

$$\mathcal{S} = \mathrm{TopK}_{j \in \mathcal{E}} \left(\frac{\mu_j}{1 + \mathrm{CV}_j}, K_s \right). \tag{6}$$

The numerator rewards attention strength, while the denominator penalizes cross-frame variation. Each frame retains the positions $\mathcal{C}_f \cup \mathcal{S}$. The two sets are disjoint, giving exactly $K_c + K_s$ tokens per frame and a fixed sequence length for sparse execution. Only anchor positions are shared: their representations remain frame-specific and are updated separately. Appendix C.5 visualizes the selected core tokens and shared anchors.

4.2 Pilot: Online Selection and Sparse Execution

In this section, we implement sparse execution through lightweight attention profiling, cross-step selection reuse, and cached prediction updates. Together, these mechanisms reduce selection and execution overhead while maintaining the sampling updates required by joint visual–action denoising.

Low-Overhead Attention Profiling. At each full step, Pilot obtains the required scores from the dense conditional forward without an additional network forward. The scorer computes only the query–key products between each future frame and its associated action-query group. It reuses the original log-sum-exp normalizers computed over all visible keys, preserving the attention mass used in layer-quality scoring. The resulting probabilities are aggregated directly into $S_\ell(f, j)$, avoiding materialization of the full attention matrix or an additional value-weighted attention output. Appendix A.2 provides the extraction formula and implementation details.

Cross-Step Reuse and Compact Execution. The observed cross-step attention consistency motivates reusing selections within each action chunk. A full step constructs the selection and caches the visual velocity predictions; subsequent sparse steps reuse both. Given the short denoising schedules of the evaluated WAMs, our default configuration uses only the initial step for full computation. Selections and cached predictions are recomputed for each new action chunk. Alternative refresh schedules are evaluated in Appendix C.3.

During sparse steps, the selected future tokens are processed as a compact sequence together with all observation and action tokens. Pilot reuses their original-position indices and compatible packing metadata across Transformer layers and denoising steps. The fixed per-frame token budget maintains a constant sequence length despite different spatial selections across frames.

Sampling Updates for Omitted Regions. Future tokens omitted from Transformer computation still participate in the sampling process. Let τ_d denote the most recent full step and $\mathbf{M}_\mathrm{v}$ the retained-position mask mapped to the full future-latent grid. Pilot combines current predictions at retained positions with cached predictions elsewhere:

$$\widetilde{\mathbf{v}}_\mathrm{v}^{(\tau)} = \mathbf{M}_\mathrm{v} \odot \mathbf{v}_\mathrm{v}^{(\tau)} + (\mathbf{1} - \mathbf{M}_\mathrm{v}) \odot \mathbf{v}_\mathrm{v}^{(\tau_d)}, \qquad \tau > \tau_d, \tag{7}$$

where $\odot$ denotes element-wise multiplication, $\mathbf{v}_\mathrm{v}^{(\tau)}$ contains current predictions scattered to their retained positions in the full grid, and $\mathbf{v}_\mathrm{v}^{(\tau_d)}$ is the cached prediction from the full step. The original sampler uses $\widetilde{\mathbf{v}}_\mathrm{v}^{(\tau)}$ to update all future latents, including omitted regions, while action predictions are recomputed at every step.

Table 1: Performance and inference efficiency on LIBERO.

Method	Success Rate (%)				Avg. SR (%)	Avg. Speedup	FLOPs
	Spatial	Object	Goal	Long			
FastWAM-Joint	99.00%	100.00%	98.20%	97.80%	98.75%	1.00×	100.00%
+ ToCa	98.80%	99.80%	98.60%	98.60%	98.95%	1.09×	69.19%
+ C^{3}ache	98.60%	99.40%	97.60%	97.40%	98.25%	1.30×	75.00%
+ SpecPrune-VLA	97.20%	99.80%	96.80%	92.40%	96.55%	1.43×	73.53%
+ WorldCache	99.60%	99.80%	98.00%	98.60%	99.00%	1.62×	60.00%
+ **Sparse-WAM**	98.80%	99.20%	98.60%	97.20%	98.45%	1.98×	49.65%

Table 2: Performance and inference efficiency on RoboLab-120.

Method	Success Rate (%)			Avg. SR (%)	Avg. Speedup	FLOPs
	Simple	Moderate	Complex			
Cosmos3-Edge	25.60%	23.30%	11.80%	22.90%	1.00×	100.00%
+ ToCa	9.84%	14.10%	1.18%	10.00%	1.26×	65.68%
+ C^{3}ache	9.06%	12.82%	0.59%	9.08%	1.51×	75.14%
+ SpecPrune-VLA	10.31%	14.10%	0.00%	10.08%	1.52×	50.35%
+ WorldCache	21.56%	21.03%	11.18%	19.92%	1.52×	75.14%
+ **Sparse-WAM**	25.00%	25.90%	8.82%	23.00%	1.85×	60.73%
Cosmos3-Nano-Policy	40.63%	35.38%	25.29%	36.75%	1.00×	100.00%
+ ToCa	9.69%	12.82%	0.00%	9.33%	1.33×	65.21%
+ C^{3}ache	16.09%	13.08%	1.18%	13.00%	1.48×	75.00%
+ SpecPrune-VLA	27.50%	23.08%	11.76%	23.83%	1.88×	54.79%
+ WorldCache	36.56%	30.00%	20.00%	32.08%	1.48×	75.00%
+ **Sparse-WAM**	40.16%	32.82%	24.12%	35.50%	1.81×	59.78%

5 Experiments

5.1 Experimental Settings

Benchmarks and Models. We evaluate Sparse-WAM using FastWAM-Joint (Yuan et al., 2026) on LIBERO (Liu et al., 2023) and LIBERO-Plus (Fei et al., 2025), and Cosmos 3 Nano Policy and Cosmos 3 Edge (NVIDIA, 2026) on RoboLab-120 (Yang et al., 2026). LIBERO comprises four manipulation suites, LIBERO-Plus adds seven perturbation categories, and RoboLab-120 contains 120 tasks across three difficulty levels. We also evaluate FastWAM-Joint on real-world object packing, cup stacking, and battery insertion tasks. Figure 4 illustrates representative tasks. Appendix B.1 provides benchmark protocols.

Baselines and Evaluation Setup. We compare against four inference acceleration methods: ToCa (Zou et al., 2025), WorldCache (Feng et al., 2026), SpecPrune-VLA (Wang et al., 2026), and C^{3}ache (Zhao et al., 2026b). All policy inference runs on an NVIDIA RTX 4090, with latency measured using CUDA events. Appendix B.2 provides implementation details, while Appendix B.3 describes the baseline adaptations.

Evaluation Metrics. We report task success rate (SR), inference speedup, and FLOPs as a percentage of the corresponding dense model. Speedups in the main tables compare optimized accelerated inference against dense eager inference and include gains from both the acceleration methods and execution optimizations. Appendix B.4 details the timing protocol and reports on an execution backend ablation and comparisons with matched execution backends.

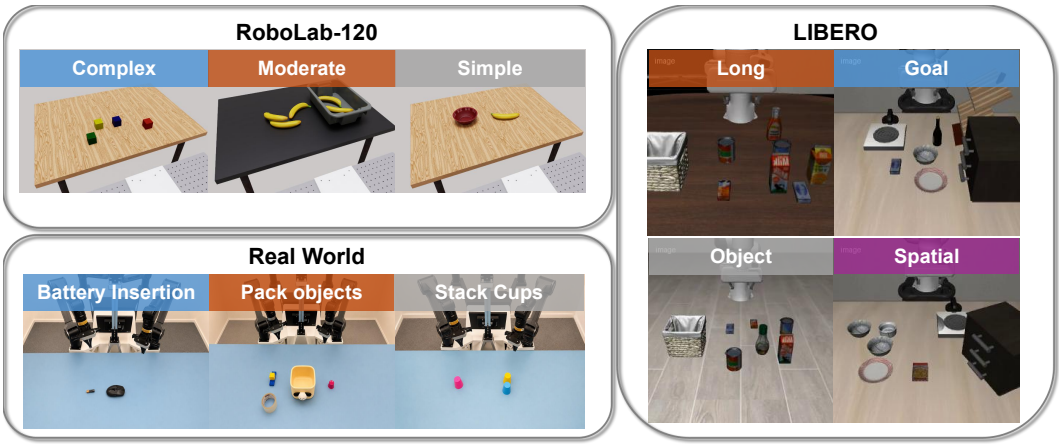


Figure 4: Tasks on LIBERO, RoboLab-120 and Real World.

Table 3: Real-world manipulation performance on AgileX Cobot Magic.

Method	Success Rate (%)				Latency (ms)	Speedup
	Pack objects	Stack cups	Battery assembly	Average		
FastWAM-Joint	75.00	75.00	83.33	77.78	501	1.00×
+ **Sparse-WAM**	83.33	75.00	66.67	75.00	242	2.08×

5.2 Simulation Results

Results on LIBERO. Table 1 reports performance and inference efficiency across the four LIBERO suites. Speedups in both main tables are measured relative to dense eager inference and reflect the combined effects of each acceleration method and execution optimizations. Sparse-WAM achieves the largest speedup among the evaluated methods (1.98×), reducing FLOPs by 50.35% while attaining 98.45% success, 0.30 percentage points below dense inference. ToCa, WorldCache, and C^3ache also maintain high success rates but achieve smaller speedups, up to 1.62×. SpecPrune-VLA incurs a larger performance drop, particularly on long-horizon tasks.

Results on RoboLab-120. Table 2 shows that Sparse-WAM achieves the highest average success rates among the accelerated variants on both Cosmos backbones: 23.00% on Edge and 35.50% on Nano, compared with 22.90% and 36.75% for dense inference. With execution optimizations enabled, it achieves respective speedups of 1.85× and 1.81× over dense eager inference. WorldCache preserves performance better than ToCa and C^3ache but is slower than Sparse-WAM. SpecPrune-VLA is slightly faster on Nano (1.88×), at a success rate 11.67 percentage points below ours. On Edge, Sparse-WAM outpaces our SpecPrune-VLA adaptation (1.85× versus 1.52×) despite using more FLOPs (60.73% versus 50.35% of dense inference), showing that FLOPs alone do not determine latency. Our implementation reuses token selections and maintains fixed sparse sequence lengths to support compiled execution. Appendix B.4 provides backend ablations and matched-backend comparisons on Cosmos 3 Edge; Appendix C.1 presents LIBERO-Plus robustness results and perturbation examples.

5.3 Real-World Experiments

We evaluate Sparse-WAM on the AgileX Cobot Magic platform, which is equipped with three cameras providing different viewpoints: one primary camera and two wrist-mounted cameras. We fine-tune FastWAM-Joint using our collected data, following the configuration detailed in Appendix B.5.

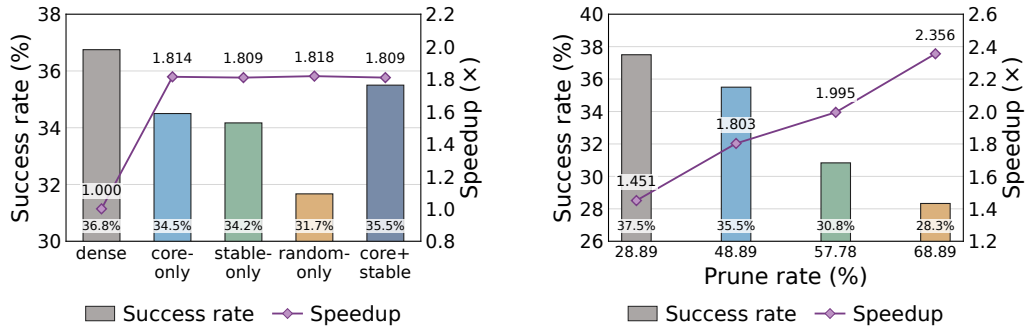
We design three tasks: object packing, cup stacking, and battery insertion, as shown in Figure 4. Table 3 reports the real-world performance of Sparse-WAM. Our method achieves a 2.08× speedup with an average success rate of 75.00%, compared with 77.78% for dense FastWAM-Joint. These

results demonstrate the potential of Sparse-WAM to accelerate real-world WAM inference by reducing redundant computation over imagination tokens while largely preserving average task success.

5.4 Ablation Studies

Component Ablation. Figure 5a compares token selection strategies on Cosmos 3 Nano Policy, with all sparse variants retaining 184 of 360 tokens per future frame. Combining $K_c = 80$ core tokens with $K_s = 104$ stable anchors achieves a 35.5% success rate, outperforming both core-only and stable-only selection. Random selection achieves only 31.7%, compared with 36.8% under dense inference. The combined variant maintains approximately $1.81\times$ speedup, comparable to core-only selection. These results support the complementary roles of frame-specific core tokens and shared stable anchors in preserving action-relevant imagination, with little additional inference overhead.

Pruning Ratio. Figure 5b examines the effect of future-token pruning on Cosmos 3 Nano Policy. We define the future-token pruning ratio as $r = 1 - K_{\text{ret}}/N_s$, where K_{ret} is the number of retained future tokens per frame. Increasing the pruning ratio from 28.89% to 68.89% improves speedup from $1.45\times$ to $2.36\times$, while success rate decreases from 37.5% to 28.3%. These results illustrate the efficiency–control trade-off: more aggressive pruning accelerates inference at the expense of task success. Appendix C.2 further ablates informative-layer aggregation and temporal action weighting.



(a) Core tokens and stable anchors.

(b) Future-token pruning ratio.

Figure 5: Ablation studies of token selection components and future-token pruning ratios.

6 Limitations and Future Work

Future work could broaden the evaluation and deployment of Sparse-WAM in three directions. First, our inference benchmarks were conducted on an NVIDIA RTX 4090; extending evaluation to other GPUs and resource-constrained devices would help assess its efficiency across hardware platforms. Second, we evaluated three WAMs, and further studies could examine its applicability to a broader range of WAM architectures and backbones. Third, the performance degradation observed under environmental perturbations motivates adaptive token-selection mechanisms that adjust to changing conditions to improve robustness while maintaining inference efficiency.

7 Conclusion

In this paper, we presented Sparse-WAM, a training-free method that prunes evolving imagination tokens online during joint visual-action denoising. Evaluations on Cosmos 3 Edge, Cosmos 3 Nano Policy, and FastWAM-Joint across LIBERO, RoboLab-120, and real-world manipulation tasks demonstrate its inference efficiency while largely preserving task success. Sparse-WAM achieves approximately $1.8\times$ speedup on RoboLab-120 and $2.0\times$ on LIBERO. These results indicate that dense computation over evolving imagination tokens in WAMs is partially redundant for action prediction. Sparsifying this computation enables more efficient inference while largely preserving task performance.

AI USE STATEMENT

We used generative AI tools to assist with language editing and provide feedback on the clarity, structure, and technical presentation of the manuscript. All AI-assisted revisions were reviewed and verified by the authors, who take full responsibility for the content, claims, and associated artifacts of this work.

ETHICS STATEMENT

This work investigates sparse imagination to accelerate inference in world-action models. We evaluate our approach on LIBERO, RoboLab-120, and LIBERO-Plus, as well as on the AgileX Cobot Magic dual-arm platform, in accordance with the applicable terms of use. These evaluations do not establish the safety of deploying the evaluated models near humans or in safety-critical settings.

REPRODUCIBILITY STATEMENT

Section 4 and Appendix A describe our method and its implementation. Section 5 presents the experimental setup and main results. Appendix B provides evaluation protocols, implementation settings, baseline adaptations, timing procedures, and real-world fine-tuning configurations. Appendix C reports additional ablations, robustness evaluations, and attention analyses. We plan to release the code for evaluation, sparse inference, and system optimizations upon publication.

REFERENCES

Sotiris Anagnostidis, Gregor Bachmann, Yeongmin Kim, Jonas Kohler, Markos Georgopoulos, Artsiom Sanakoyeu, Yuming Du, Albert Pumarola, Ali K. Thabet, and Edgar Schönfeld. FlexiDiT: Your diffusion transformer can easily generate high-quality samples with less compute. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 28316–28326, 2025. doi: 10.1109/CVPR52734.2025.02637.

Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, Hongyan Zhao, Hanyu Liu, Zhizhong Su, Lei Ma, Hang Su, and Jun Zhu. Motus: A unified latent action world model. *arXiv preprint arXiv:2512.13030*, 2025. URL https://arxiv.org/abs/2512.13030.

Jialei Chen, Kai Wang, Kang Chen, Shuaihang Chen, Feng Gao, Wenhao Tang, Zhiyuan Li, Weilin Liu, Zhuyu Yao, Boxun Li, Yuanbo Xu, and Chao Yu. LaWAM: Latent world action models for efficient dynamics-aware robot policies. *arXiv preprint arXiv:2606.15768*, 2026. URL https://arxiv.org/abs/2606.15768.

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: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025. URL https://arxiv.org/abs/2510.13626.

Weilun Feng, Guoxin Fan, Haotong Qin, Mingqiang Wu, Yuqi Li, Xiangqi Li, Zhulin An, Libo Huang, Dingrui Wang, Longlong Liao, Michele Magno, Yongjun Xu, and Chuanguang Yang. WorldCache: Accelerating world models for free via heterogeneous token caching. *arXiv preprint arXiv:2603.06331*, 2026. URL https://arxiv.org/abs/2603.06331.

Yanjiang Guo, Yucheng Hu, Jianke Zhang, Yen-Jen Wang, Xiaoyu Chen, Chaochao Lu, and Jianyu Chen. Prediction with action: Visual policy learning via joint denoising process. In *Advances in Neural Information Processing Systems*, volume 37, pp. 112386–112410. Curran Associates, Inc., 2024. URL https://papers.nips.cc/paper_files/paper/2024/hash/cbe25fa0e7c7084049276888a09acc8d-Abstract-Conference.html.

Jiajun Li, Tiecheng Guo, Yifan Ye, Rongyu Zhang, Xiaowei Chi, Qianpu Sun, Ying Li, Yunfan Lou, Yan Huang, Zhihe Lu, Meng Guo, and Shanghang Zhang. Efficient-WAM: A 1B-parameter world-action model with low-cost future imagination. *arXiv preprint arXiv:2606.10040*, 2026a. URL https://arxiv.org/abs/2606.10040.

Jingyu Li, Zhe Liu, Dongnan Hu, Junjie Wu, Zipei Ma, Wenxiao Wu, Chao Han, Zhihui Hao, Zhikang Liu, Kun Zhan, Jiankang Deng, Xiatian Zhu, and Li Zhang. Metis: A generalizable and efficient world-action model for autonomous driving and urban navigation. *arXiv preprint arXiv:2606.15869*, 2026b. doi: 10.48550/arXiv.2606.15869. URL https://arxiv.org/abs/2606.15869.

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.

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*, 2026c. URL https://arxiv.org/abs/2606.08242.

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.

Ziming Liu, Yifan Yang, Chengruidong Zhang, Yiqi Zhang, Lili Qiu, Yang You, and Yuqing Yang. Region-adaptive sampling for diffusion transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2346–2356, 2026.

Ziyan Liu, Yeqiu Chen, Hongyi Cai, Tao Lin, Shuo Yang, Zheng Liu, and Bo Zhao. VLA-Pruner: Temporal-aware dual-level visual token pruning for efficient vision-language-action inference. *arXiv preprint arXiv:2511.16449*, 2025. URL https://arxiv.org/abs/2511.16449v1.

Shilin Ma, Chubin Zhang, Changyuan Wang, Yuji Wang, Yue Wu, Zixuan Wang, Jingqi Tian, Zheng Zhu, and Yansong Tang. SAFE-Pruner: Semantic attention–guided future-aware token pruning for efficient vision-language-action manipulation. *arXiv preprint arXiv:2605.29662*, 2026. URL https://arxiv.org/abs/2605.29662.

NVIDIA. Cosmos 3: Omnimodal world models for physical AI. *arXiv preprint arXiv:2606.02800*, 2026. URL https://arxiv.org/abs/2606.02800.

Xiaohuan Pei, Yuxing Chen, Siyu Xu, Yunke Wang, Yuheng Shi, and Chang Xu. Action-aware dynamic pruning for efficient vision-language-action manipulation. *arXiv preprint arXiv:2509.22093*, 2025. URL https://arxiv.org/abs/2509.22093.

Hanzhen Wang, Jiaming Xu, Yushun Xiang, Jiayi Pan, Yongkang Zhou, Yong-Lu Li, and Guohao Dai. SpecPrune-VLA: Accelerating vision-language-action models via action-aware self-speculative pruning. In *Proceedings of the International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2509.05614.

Siyu Xu, Yunke Wang, Chenghao Xia, Dihao Zhu, Tao Huang, and Chang Xu. VLA-Cache: Efficient vision-language-action manipulation via adaptive token caching. *arXiv preprint arXiv:2502.02175*, 2025. URL https://arxiv.org/abs/2502.02175.

Xuning Yang, Rishit Dagli, Alex Zook, Hugo Hadfield, Ankit Goyal, Stan Birchfield, Fabio Ramos, and Jonathan Tremblay. RoboLab: A high-fidelity simulation benchmark for analysis of task generalist policies. *arXiv preprint arXiv:2604.09860*, 2026. URL https://arxiv.org/abs/2604.09860.

Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, Jie Li, Jindi Lv, Jingyu Liu, Min Cao, Peng Li, Qiuping Deng, Wenjun Mei, Xiaofeng Wang, Xinze Chen, Xinyu Zhou, Yang Wang, Yifan Chang, Yifan Li, Yukun Zhou, Yun Ye, Zhichao Liu, and Zheng Zhu. GigaWorld-Policy: An efficient action-centered world–action model. *arXiv preprint arXiv:2603.17240*, 2026a. URL https://arxiv.org/abs/2603.17240.

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*, 2026b. 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.

Evelyn Zhang, Jiayi Tang, Xuefei Ning, and Linfeng Zhang. Training-free and hardware-friendly acceleration for diffusion models via similarity-based token pruning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 9878–9886, 2025a. doi: 10.1609/aaai.v39i9.33071.

Zhanguang Zhang, Zhiyuan Li, Behnam Rahmati, Rui Heng Yang, Yintao Ma, Amir Rasouli, Sajjad Pakdamansavoji, Yangzheng Wu, Lingfeng Zhang, Tongtong Cao, Feng Wen, Xinyu Wang, Xingyue Quan, and Yingxue Zhang. Do world action models generalize better than VLAs? A robustness study. *arXiv preprint arXiv:2603.22078*, 2026. URL https://arxiv.org/abs/2603.22078.

Zhisong Zhang, Yan Wang, Xinting Huang, Tianqing Fang, Hongming Zhang, Chenlong Deng, Shuaiyi Li, and Dong Yu. Attention entropy is a key factor: An analysis of parallel context encoding with full-attention-based pre-trained language models. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 9840–9855, 2025b. doi: 10.18653/v1/2025.acl-long.485. URL https://aclanthology.org/2025.acl-long.485/.

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*, 2026a. URL https://arxiv.org/abs/2608.04404.

Weisen Zhao, Lam Nguyen, Zhicong Lu, and Yuzhang Shang. C^{3}ache: Accelerating world action models with cross inference chunk cache. *arXiv preprint arXiv:2606.08962*, 2026b. URL https://arxiv.org/abs/2606.08962.

Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=yYZbZGo4ei.

A Additional Method Details

This section supplements the token selection procedure in subsection 4.1 and the attention profiling mechanism in subsection 4.2. All attention scores below are obtained at a full denoising step τ_d. We omit this step index for brevity and write $U_\ell(i, f, j) = U_\ell^{(\tau_d)}(i, f, j)$.

A.1 Token Selection Details

Frame-Aligned Action Queries. The temporal alignment observed in Figure 2, Insight 1(a), motivates aggregating attention from the action queries associated with each future frame. Assuming that H is divisible by F, we partition the H action queries into F consecutive groups:

$$\mathcal{I}_f = \left\{ (f-1)\frac{H}{F} + 1, \ldots, f\frac{H}{F} \right\}, \qquad f = 1, \ldots, F. \tag{8}$$

The frame-aligned score $S_\ell(f, j)$ is obtained by aggregating $U_\ell(i, f, j)$ over $i \in \mathcal{I}_f$, as defined in Equation 2. The query weights satisfy $\alpha_{f,i} \geq 0$ and $\sum_{i \in \mathcal{I}_f} \alpha_{f,i} = 1$. Queries near the center of each group receive higher weights than those near its boundaries, since boundary queries also attend to neighboring future frames. The groups themselves remain disjoint.

Layer Quality. We assess each layer using both the attention mass assigned to the associated future frames and the spatial concentration of that attention. The frame-aligned attention mass is

$$R_\ell = \frac{1}{F} \sum_{f=1}^{F} \sum_{j=1}^{N_s} S_\ell(f, j). \tag{9}$$

Because $S_\ell(f, j)$ retains the original attention normalization over all visible keys, R_ℓ measures how much attention is assigned to the associated future frames.

To measure spatial concentration, we normalize the scores within each frame:

$$\bar{S}_\ell(f, j) = \frac{S_\ell(f, j)}{\sum_{j'=1}^{N_s} S_\ell(f, j')}. \tag{10}$$

We then compute the normalized spatial entropy (Zhang et al., 2025b):

$$E_\ell = -\frac{1}{F \log N_s} \sum_{f=1}^{F} \sum_{j=1}^{N_s} \bar{S}_\ell(f, j) \log \bar{S}_\ell(f, j), \tag{11}$$

with the convention $0 \log 0 = 0$. Lower entropy indicates more concentrated spatial attention.

The resulting layer-quality score is

$$Q_\ell = R_\ell(1 - E_\ell). \tag{12}$$

This score favors layers that assign substantial attention to the associated future frames and localize that attention within each frame. We select the K_{layer} layers with the highest Q_ℓ values to form $\mathcal{L}_{\text{core}}$. Their scores are aggregated using Equation 3 to obtain $V(f, j)$ for frame-specific core selection.

Anchor Candidates and Availability. The common candidate pool $\mathcal{P}$ and the core positions $\mathcal{C}_f$ are defined in Equation 4 and Equation 5, respectively. Anchors are selected from positions unused by every frame's core selection:

$$\mathcal{E} = \mathcal{J} \setminus \bigcup_{f=1}^{F} \mathcal{C}_f, \qquad \mathcal{J} = \{1, \ldots, N_s\}. \tag{13}$$

Since $\mathcal{C}_f \subseteq \mathcal{P}$ for every frame and $|\mathcal{P}| = N_s - K_s$, we have

$$|\mathcal{E}| = N_s - \left| \bigcup_{f=1}^{F} \mathcal{C}_f \right| \geq N_s - |\mathcal{P}| = K_s. \tag{14}$$

Thus, at least K_s anchor candidates remain available regardless of how core positions vary across frames. The candidate set includes all positions outside $\mathcal{P}$ and any positions inside $\mathcal{P}$ that no frame selects as core.

Shared-Anchor Scores. Core localization uses selected layers to emphasize spatially concentrated attention. For shared anchors, we aggregate scores from all layers using the same quality weights:

$$V_{\text{all}}(f, j) = \frac{\sum_\ell Q_\ell S_\ell(f, j)}{\sum_\ell Q_\ell}. \tag{15}$$

This aggregation includes contextual attention beyond the layers selected for core localization.

For each spatial position j, we compute the mean and coefficient of variation across future frames:

$$\begin{aligned} \mu_j &= \frac{1}{F} \sum_{f=1}^{F} V_{\text{all}}(f, j), \\ \text{CV}_j &= \frac{\text{Std}_f\left[V_{\text{all}}(f, j)\right]}{\mu_j}. \end{aligned} \tag{16}$$

Here, Std_f denotes the standard deviation across future frames. The anchor score $\mu_j/(1 + \text{CV}_j)$ rewards strong attention and penalizes variation across frames. The K_s highest-scoring positions in $\mathcal{E}$ form $\mathcal{S}$, as defined in Equation 6.

Each frame retains $\mathcal{C}_f \cup \mathcal{S}$. Because $\mathcal{S} \subseteq \mathcal{E}$, the core and anchor sets are disjoint in every frame. Together with $K_c + K_s \leq N_s$, this construction provides exactly $K_c + K_s$ retained tokens per frame. The shared set $\mathcal{S}$ specifies spatial positions; the corresponding token representations remain frame-specific.

A.2 Attention Profiling in Pilot

Pilot computes the scores required for token selection during the dense conditional forward at each full step. It evaluates only the relevant action-to-future query–key pairs and reuses the original attention normalizers, avoiding an additional network forward and storage of the full attention matrix.

Preserving the Original Attention Normalization. Let $x_{\ell,h}(i, f, j)$ denote the attention logit between action query i and future-frame key (f, j) at layer ℓ and head h. Let $\lambda_{\ell,h}(i)$ denote the log-sum-exp normalizer for the same query, computed over all visible keys in the original attention operation. The corresponding attention probability is therefore $\exp(x_{\ell,h}(i, f, j) - \lambda_{\ell,h}(i))$.

Pilot combines head averaging and weighted query aggregation directly:

$$S_\ell(f, j) = \frac{1}{N_h} \sum_{i \in \mathcal{I}_f} \sum_{h=1}^{N_h} \alpha_{f,i} \exp(x_{\ell,h}(i, f, j) - \lambda_{\ell,h}(i)) \,. \tag{17}$$

This expression recovers the frame-aligned score in Equation 2. Retaining the original normalizers also preserves the attention mass used to compute R_ℓ. Renormalizing only over future-frame keys would discard their attention mass relative to other visible tokens and change the layer-quality scores.

Restricted Computation and Direct Reduction. For each future frame f, the scorer evaluates query–key products only for action queries in $\mathcal{I}_f$. It follows the backbone's query–key normalization, positional encoding, attention scaling, head mapping, and visibility constraints when computing these logits.

The computation is batched across future frames, and the resulting probabilities are reduced directly over heads and action queries to produce $S_\ell(f, j)$. Consequently, the scorer stores only the aggregated spatial scores required for selection, without materializing the full attention matrix or computing an additional value-weighted attention output. Pilot then uses these scores to construct the retained positions for subsequent sparse steps, as described in subsection 4.2.

B Experimental Details

B.1 Benchmarks and Evaluation Protocols

LIBERO. We evaluate on four LIBERO suites (Liu et al., 2023): Spatial, Object, Goal, and Long, which cover spatial relationships, object variations, task goals, and long-horizon manipulation, respectively. Each suite contains 10 tasks. We perform 50 rollouts per task, yielding 500 rollouts per suite and 2,000 rollouts across all 40 tasks for each evaluation configuration.

RoboLab-120. RoboLab-120 (Yang et al., 2026) comprises 120 language-conditioned manipulation tasks covering visual, relational, and procedural competencies. The tasks are divided into three difficulty levels: 64 Simple, 39 Moderate, and 17 Complex tasks. We evaluate all 120 tasks with 10 rollouts per task, resulting in 1,200 rollouts per evaluation configuration.

LIBERO-Plus. We evaluate robustness using a balanced subset of LIBERO-Plus (Fei et al., 2025) constructed from the same 40 base tasks across the four LIBERO suites. The subset covers seven perturbation categories: background textures, camera viewpoints, language instructions, lighting conditions, object layouts, robot initial states, and sensor noise. For each base task and perturbation category, we select two distinct perturbation variants and perform one rollout per variant. This yields 560 rollouts per evaluation configuration, with 80 rollouts per category and 140 rollouts per suite. All compared methods use the same selected variants, initial-state indices, and environment and model seeds.

B.2 Implementation Details

Backbone architectures. Cosmos 3 Nano Policy and Cosmos 3 Edge comprise an understanding tower and a diffusion generation tower. The generation tower jointly denoises future visual and action

tokens within a shared Transformer backbone, conditioned on representations from the understanding tower. We cache and reuse instruction representations across action chunks, while updating visual conditioning with each incoming observation. Both models receive one wrist-camera view and two smaller third-person views. FastWAM-Joint uses a Mixture-of-Transformers (MoT) architecture with separately parameterized video and action diffusion Transformers coupled through shared attention. Language embeddings condition both branches through cross-attention, and visual observations comprise one third-person view and one wrist-camera view. Table 4 summarizes the inference configurations.

Table 4: Backbone configurations used in our simulation experiments.

Configuration	Cosmos 3 Nano Policy	Cosmos 3 Edge	FastWAM-Joint
Transformer layers	36	28	30
Camera views	3	3	2
Future frame token	360	340	98
Action horizon	32	32	32
Future latent frames	8	8	2
Denoising steps	4	4	10
Guidance scale	3.0	3.0	1.0
Noise schedule shift	5.0	5.0	5.0

Parameter Setup. Table 5 summarizes the default configurations of Sparse-WAM for the three backbones. We use fixed token budgets for each backbone, while selecting informative layers and token positions online for each action chunk. In the evaluated schedule, attention statistics are collected from all Transformer layers during the initial dense conditional forward of each action chunk. Core scores aggregate the six layers with the highest quality scores Q_ℓ, whereas stable-anchor scores aggregate all layers.

For temporal aggregation, Table 5 reports the number of aligned queries per frame and the center/outer query weight ratio. Each future latent frame is associated with four action queries in Cosmos 3 and sixteen in FastWAM-Joint. We set the center/outer query weight ratio to 2 for all backbones, assigning twice the weight to queries in the central half of each group compared with those in the outer quarters. After normalization, the weights are $(1, 2, 2, 1)/6$ for Cosmos 3; for FastWAM-Joint, each central query receives weight $1/12$ and each outer query receives weight $1/24$.

Table 5: Default Sparse-WAM configurations.

Parameter	Nano	Edge	Joint
Core tokens K_c	80	80	19
Stable anchors K_s	104	104	12
Retained tokens $K_c + K_s$	184	184	31
Informative layers K_{layer}	6	6	6
Aligned queries per frame	4	4	16
Center/outer query weight ratio	2	2	2
CV penalty coefficient	1.0	1.0	1.0
τ_d	0	0	0

B.3 Comparative methods

ToCa (Zou et al., 2025). ToCa selectively reuses token features across denoising steps based on token importance and caching history. We apply caching only to future visual tokens, while observation and action tokens remain fully computed. Dense steps refresh the feature cache, whereas cached steps reuse future-token attention residuals and selectively update their MLP features. We use dense steps $\{0, 2\}$ for the four-step Cosmos 3 models and $\{0, 4, 9\}$ for ten-step FastWAM-Joint, with all remaining steps using cached computation.

Table 6: Timing analysis on Cosmos 3 Edge. Panel (a) cumulatively enables CUDA Graphs and `torch.compile` for sparse inference, with speedups relative to dense eager inference. Panel (b) enables both optimizations for all methods, including the common dense reference. Latencies are medians, except for C^{3}ache, which reports latency amortized over complete refresh/reuse cycles. SpecPrune-VLA uses a minimum retention budget of 184 future tokens per frame.

Configuration / Method	Latency (ms)	Speedup
(a) Execution-backend ablation		
Dense eager	859.31	1.00×
Sparse-WAM eager	555.55	1.55×
+ CUDA Graph	467.16	1.84×
+ `torch.compile`	464.95	1.85×
(b) Matched-backend comparison		
Dense (optimized)	727.44	1.00×
ToCa	687.03	1.06×
WorldCache	562.10	1.29×
SpecPrune-VLA	576.83	1.26×
C^{3}ache	569.61	1.28×
Sparse-WAM	**464.95**	1.56×

WorldCache (Feng et al., 2026). WorldCache exploits token-trajectory predictability to reuse or extrapolate denoising predictions. We apply its heterogeneous prediction mechanism to both future visual and action outputs, using fixed dense/cache schedules. For Cosmos 3, the first three steps are dense and the final step uses cached predictions. For FastWAM-Joint, the first three and final steps are dense, while the six intermediate steps use cached predictions. Each action chunk maintains its own prediction history.

SpecPrune-VLA (Wang et al., 2026). SpecPrune-VLA combines historical and current visual information to guide token selection and progressive layer-wise pruning. We adapt its observation-based selection by broadcasting the selected spatial positions to all future latent frames, while retaining all observation and action tokens. The initial conditional forward constructs a layer-wise pruning plan, which is reused in subsequent denoising forwards. Selected future tokens continue through the Transformer, while pruned tokens retain their exit-layer hidden states, restored before the output heads to preserve the complete latent layout.

C^{3}ache (Zhao et al., 2026b). C^{3}ache reuses denoising residuals across action chunks at matching denoising steps. We extend this mechanism to the complete joint visual–action Transformer stack, while keeping input embeddings, output heads, and sampling updates active. Fully dense refresh chunks alternate with cache-reuse chunks. In reuse chunks, the first two of four steps use cached residuals for Cosmos 3, and the first five of ten steps use cached residuals for FastWAM-Joint

B.4 Timing Details

We benchmark generation latency per action chunk on Cosmos 3 Edge using an NVIDIA RTX 4090, batch size 1, BF16 precision, and four denoising steps. All configurations use identical recorded inputs and random seeds. After five warm-up runs, we collect 30 measurements per configuration using CUDA events, with GPU synchronization at both timing boundaries. The timed region includes observation encoding, initial profiling, online scoring and token selection, token packing and restoration, cache operations, and sampler updates. Model loading, compilation warm-up, output transfers to the CPU, video decoding, RPC, and simulation are excluded.

In the main experiments, all accelerated methods use `torch.compile` and CUDA Graphs, while the dense reference uses eager execution. For Sparse-WAM, LSE-based attention scoring introduces additional operators, and token selection affects CUDA Graph capture and execution. The runtime costs of scoring and selection are included in the measurements. Table 6 separates the gains from sparse inference and execution optimizations through an execution-backend ablation and a matched-backend comparison.

Under eager execution, Sparse-WAM achieves a $1.55\times$ speedup over dense inference. With both execution optimizations enabled, the speedup reaches $1.85\times$ relative to dense eager inference and $1.56\times$ relative to dense inference with the same optimizations.

B.5 Real World Experiment

Finetuning. We perform full-parameter finetuning on three real-world tasks: placing three objects from the table into a container, assembling the mouse battery, and stacking three cups into a small tower. Training uses eight NVIDIA H800 GPUs with a global batch size of 64. Each prediction produces an action chunk of 32 steps conditioned on a single observation, without observation history. Table 7 summarizes the training configuration and task-specific training budgets.

Table 7: Finetuning configurations for real-world experiments.

Parameter	Value
Finetuning	Full parameters
GPUs	$8\times$ H800
Batch size per GPU	8
Global batch size	64
Gradient accumulation	1
Learning rate	10^{-4}
Learning-rate scheduler	Cosine
Weight decay	0.01
Mixed precision	BF16
Gradient clipping	1.0
Action chunk size	32
Observation history	Single-step
Task	**Training steps**
Place three objects into a container	15,000
Assemble the mouse battery	20,000
Stack three cups into a tower	15,000

Evaluation. Real-world evaluation results are presented in Figure 9. We evaluate dense FastWAM-Joint and Sparse-WAM on object packing, cup stacking, and the battery task. Table 3 reports the task-level success rates, average success rate, and inference latency.

C Extended Experiments

C.1 Robustness Analysis

Results on LIBERO-Plus. Table 8 compares robustness and inference efficiency under seven perturbation categories and Figure 6 illustrates the evaluated perturbations. On the evaluated LIBERO-Plus subset, Sparse-WAM achieved 63.04% success, compared with 70.89% for dense FastWAM-Joint and 51.96% for action-only FastWAM. Generation latency decreased from 431.7 ms to 207.2 ms per chunk relative to dense FastWAM-Joint, corresponding to a 2.08× speedup. Sparse-WAM therefore retained a higher observed success rate than the action-only variant while incurring a 7.85-percentage-point decrease relative to the dense joint model. With 32 actions per chunk, this latency is compatible with asynchronous inference overlapping with action execution at 50 Hz. These results demonstrate a trade-off between preserving robustness through future imagination and reducing inference latency.

C.2 Additional ablations

Informative layer aggregation. We compare informative-layer and all-layer attention aggregation for core token selection on Cosmos3-Nano-Policy. Table 9(a) shows that aggregating attention from informative layers improves the average success rate by 2.25 percentage points, with little change in inference latency. This result supports selecting informative layers to guide core token selection.

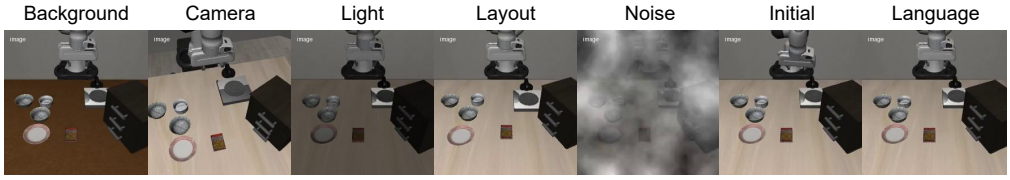


Figure 6: LIBERO-Plus visualization.

Table 8: Success rates (%) and inference efficiency on the evaluated LIBERO-Plus subset.

Method	Success Rate (%)								Latency (ms)
	Camera	Robot	Lang.	Light	Backg.	Noise	Layout	Avg.	
FastWAM-Joint	51.25	51.25	87.50	95.00	66.25	61.25	83.75	70.89	431.7
+ **Sparse-WAM**	31.25	46.25	85.00	92.50	57.50	47.50	81.25	63.04	207.2
FastWAM (action-only)	26.25	41.25	63.75	77.50	57.50	36.25	61.25	51.96	90.1

Table 9: Ablations of informative-layer aggregation and temporal action weighting on Cosmos 3 Nano Policy. Panel (a) compares all-layer and informative-layer aggregation; panel (b) compares uniform and center-weighted action aggregation.

(a) Informative layer selection

Core aggregation	Avg. SR(%)	Score	latency(ms)
All layers	33.25	48.80	1846.8
Informative layers	**35.50**	**51.84**	1857.4

(b) Temporal action weighting

Action weighting	Avg. SR(%)	Score
Uniform	32.92	48.31
Center-weighted	**35.83**	**50.71**

Temporal action weighting. Motivated by the temporal action–frame alignment observed in Figure 2, Insight 1(a), we compare uniform and center-weighted action aggregation for core token scoring. Center-weighted aggregation emphasizes queries near the center of each frame-associated action group, which attend more strongly to the corresponding frame. As shown in Table 9(b), this weighting improves the average success rate from 32.92% to 35.83%, supporting the use of temporal alignment when aggregating action-to-future attention.

C.3 Selection Reuse and Refresh Frequency

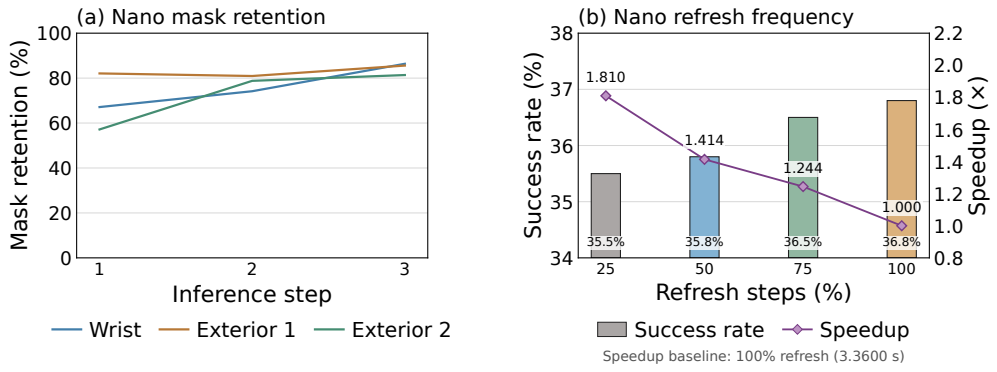


Figure 7: Selection reuse on Cosmos 3 Nano Policy. (a) Mask retention across denoising steps for three camera views. (b) Success rate and speedup under different proportions of full-refresh steps. Speedup is measured relative to full refresh at every denoising step.

We examine the effectiveness of reusing token selections across denoising steps on Cosmos 3 Nano Policy. Figure 7(a) reports mask retention across three camera views, showing substantial retention in later denoising steps. This observation motivates reusing selected spatial positions rather than repeatedly recomputing the selection.

We further vary the proportion of full-refresh steps, which perform dense computation and update token selections, while the remaining steps reuse the selections for sparse inference. As shown in Figure 7(b), a single full-refresh step in the four-step denoising schedule achieves a $1.81\times$ speedup with a 35.5% success rate, compared with 36.8% under full refresh at every step. More frequent refreshes improve success rate but reduce the acceleration benefit. These results support using limited dense scoring and cross-step selection reuse to achieve substantial acceleration with a modest reduction in task success.

C.4 CROSS-STEP AND CROSS-FRAME ATTENTION OVERLAP

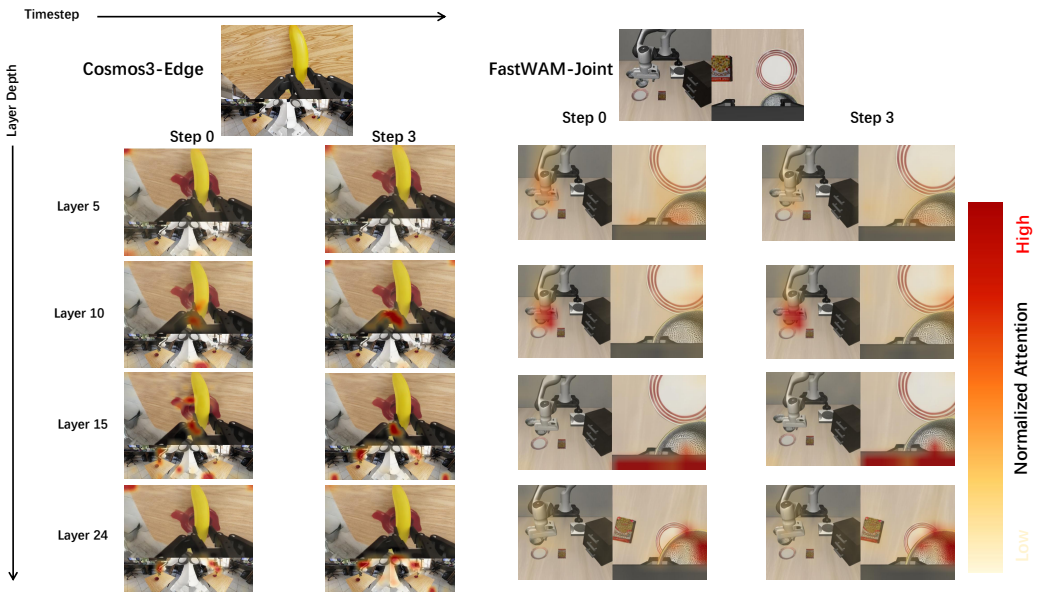


Figure 8: Visualization of action-to-future attention on Cosmos3 Edge and Fastwam-Joint

We examine action-to-future attention on Cosmos 3 Edge and FastWAM-Joint to assess its consistency across denoising steps and future frames. For two normalized spatial attention distributions P and Q, we measure their overlap as

$$\text{Overlap}(P, Q) = \sum_i \min(P_i, Q_i), \tag{18}$$

where higher values indicate greater agreement in the spatial distribution of attention.

Cross-step consistency. Using the overlap measure in Equation 18, we measured mean overlap between consecutive denoising steps of 81.11% on Cosmos 3 Edge and 97.97% on FastWAM-Joint. Figure 8 provides representative visualizations of action-to-future attention. The measured agreement in spatial attention distributions motivates selection reuse between full steps.

Cross-frame variation. Mean cross-frame overlap is lower, reaching 68.89% on Cosmos 3 Edge and 81.85% on FastWAM-Joint. Within each model, this contrast indicates greater variation in attention across future frames than across adjacent denoising steps. Such variation motivates selecting core tokens independently for each future frame, allowing spatial selections to follow changes in action-relevant regions along the predicted future.

C.5 EXPERIMENT VISUALIZATION

To examine which regions of imagined futures are retained by Sparse-WAM, Figure 10 visualizes token selections on four manipulation tasks using FastWAM-Joint and Cosmos 3 Edge. Red patches denote frame-specific core tokens, while blue patches denote stable anchors at shared spatial positions

across future frames. The visualizations illustrate how Sparse-WAM retains object–gripper interaction regions together with surrounding spatial context, combining frame-specific attention hotspots with shared contextual anchors.

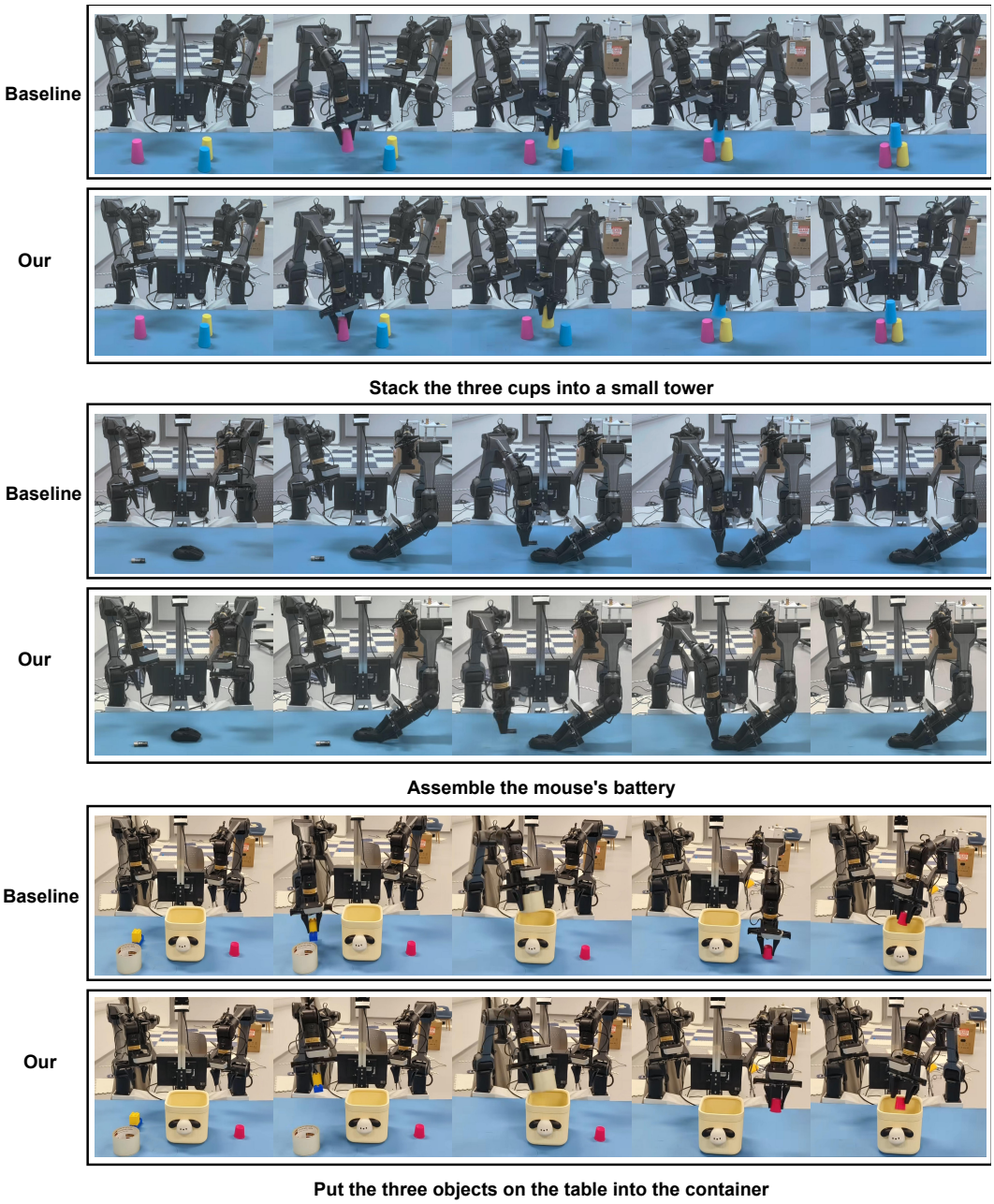


Figure 9: Comparison between baseline and Sparse-WAM in real world.

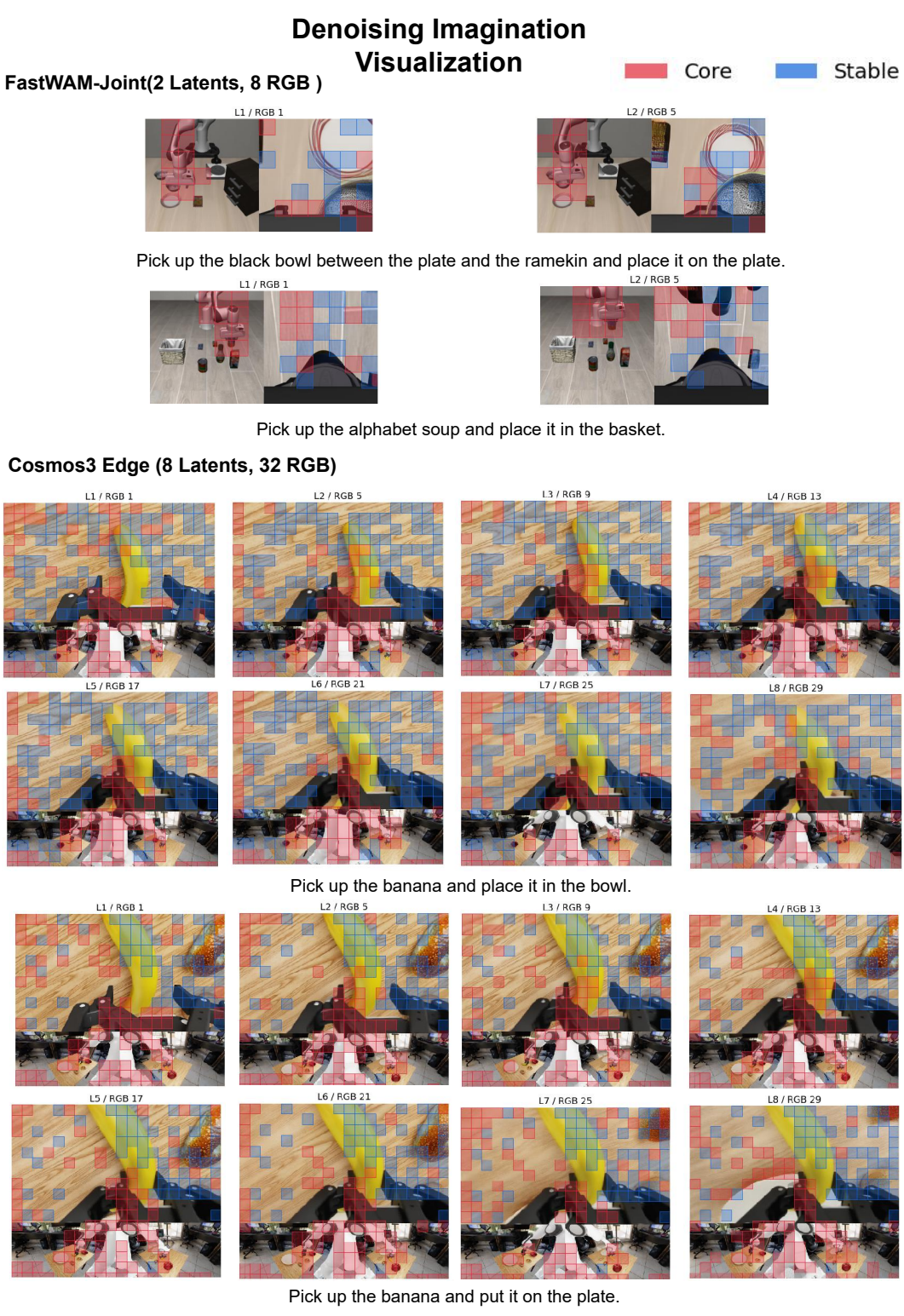


Figure 10: Core-token and stable-anchor selections in imagined future frames from FastWAM-Joint and Cosmos 3 Edge. Red and blue patches denote frame-specific core tokens and shared spatial anchors, respectively.