WAM-CACHE: STALENESS-BOUNDED KV REUSE FOR EFFICIENT WORLD ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

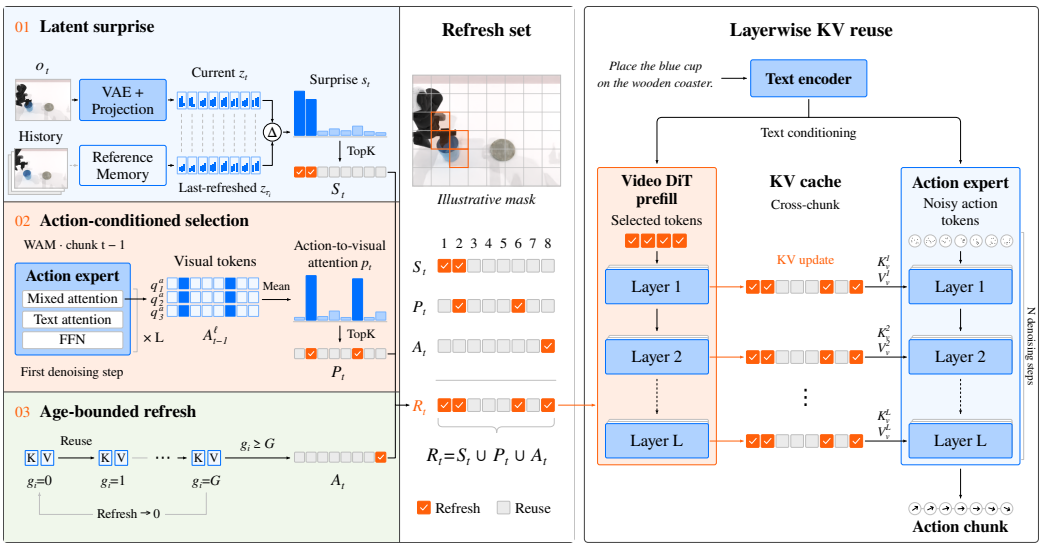
World Action Models (WAMs) enable generalist robot manipulation by conditioning an action expert on representations from a pretrained video Diffusion Transformer (DiT). In closed-loop control, the video DiT runs at every chunk to encode the current observation into layerwise key–value (KV) pairs that the action expert queries. This prefill dominates the per-chunk computational cost, yet existing training-free accelerations leave it fully dense. We present **WAM-Cache**, a training-free framework that retains layerwise key–value representations across chunks and recomputes only a sparse *refresh set* of tokens. Crucially, we find that the intuitive heuristic of refreshing visually drifted tokens plateaus far below the dense baseline, even with an oracle predicting ground-truth KV drift. Downstream action accuracy is instead governed by where the action expert attends, not by what moved. WAM-Cache therefore selects the refresh set by uniting the action expert's cross-attention with visual latent surprise, complemented by a strict age bound that suppresses compounding error. On Fast-WAM, WAM-Cache cuts video DiT prefill FLOPs by 32–42% across RoboTwin 2.0, LIBERO, and real-world experiments, while staying within 0.7–1.8 percentage points of the dense policy in simulation and 2.5 points on a real robot.

1 INTRODUCTION

World Action Models (WAMs) have emerged as a promising paradigm for generalist robot manipulation (Wang et al., 2026a; Ye et al., 2026b; Yuan et al., 2026; Li et al., 2026b). Unlike vision-language-action (VLA) policies that map observations to actions through a vision-language backbone (Black et al., 2024; Brohan et al., 2023; Kim et al., 2024), a WAM pairs a Diffusion Transformer (DiT) (Peebles & Xie, 2023) pretrained for video generation (Wan et al., 2025; NVIDIA et al., 2025) with an action expert that conditions on its representations. This design grounds control in physical dynamics learned from video, substantially enhancing generalization across unseen scenes and objects (Du et al., 2023; Wu et al., 2023; Hu et al., 2024; Zhu et al., 2025; Ye et al., 2026b). Yet, this capability comes at a steep computational cost. When operating in closed loop, a WAM repeatedly observes the environment and replans action chunks, requiring the video DiT to run at every control cycle. This backbone accounts for the vast majority of a WAM's parameters and compute.

Prior efforts to mitigate this cost generally fall into two regimes: retraining-based and training-free approaches. Retraining-based methods distill the model (Akbari et al., 2026), compress the backbone (Li et al., 2026a; Zhang et al., 2026b), or simplify action decoding (Li et al., 2026f; Ma et al., 2026). Training-free approaches, by contrast, focus primarily on caching intermediate denoiser representations across steps or chunks (Ma et al., 2023; Zou et al., 2024; Zhao et al., 2026b; Zeng et al., 2026). Reducing the video DiT's forward pass, the largest cost term, has so far required retraining a backbone with billions of parameters. To our knowledge, existing training-free methods leave this bottleneck entirely untouched. This forward pass—the *video DiT prefill*—encodes the current observation into layerwise key–value (KV) pairs, which the action expert queries while denoising an action chunk. As action experts adopt fewer denoising steps to cut latency (Wang et al., 2024; Dong et al., 2026; Chen et al., 2026b), the prefill increasingly dominates the per-chunk computational cost. On Fast-WAM, for example, the prefill accounts for 50% of DiT FLOPs at the default ten action-denoising steps and surges to 83.5% at two steps, with virtually no loss in task success (Appendix D).

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.



(a) Refresh-set selection (b) WAM inference

Figure 1: **The WAM-Cache inference pipeline.** (a) **Sparse refresh selection at chunk** t: three complementary signals construct the refresh set $\mathcal{R}_t$. Latent surprise measures feature drift from a token's reference embedding; action-conditioned selection identifies tokens heavily attended to by the action expert in the preceding chunk; and an age counter forces recomputation after G consecutive reuses. (b) **Sparse prefill and mixed-cache execution**: the video DiT executes a sparse prefill exclusively over $\mathcal{R}_t$, updating layerwise keys and values in-place. The action expert then denoises the next action chunk over the resulting mixed cache, querying fresh representations for $\mathcal{R}_t$ alongside retained keys and values for unselected tokens.

In large language model (LLM) inference, such prefill redundancy is routinely eliminated by KV caching (Pope et al., 2023; Zhang et al., 2023; Li et al., 2024; Mao et al., 2026): prompt representations are computed once, and subsequent decoding steps simply attend to the cached states. That reuse is exact because the prompt never changes, whereas a WAM ingests a fresh observation at every chunk. However, most of that observation remains static. Manipulation is local: from one chunk to the next, motion is largely restricted to the end-effector and target object, rendering most of the visual field redundant. Despite this spatial locality, the prefill recomputes every visual token at every layer. Crucially, the action expert interacts with the video DiT solely through its key–value states, making KV reuse architecturally transparent. The core challenge is therefore to decide which tokens can stay cached across chunks and which must be recomputed.

To this end, we introduce **WAM-Cache** (Figure 1), a training-free caching framework that retains layerwise key–value representations across inference chunks, recomputing only a sparse *refresh set* of visual tokens at each chunk. The entire design hinges on choosing this refresh set, yet the intuitive rule of refreshing what has changed fails in this setting. Refreshing by change alone causes a steep drop in RoboTwin 2.0 (Chen et al., 2025) success relative to the dense model (Figure 2). Even an oracle with access to the ground-truth key–value drift—refreshing the tokens that drifted most—improves on this rule only marginally (Section 5.3). This performance ceiling reveals that the downstream impact of a stale KV pair is dictated by how the action expert consumes the cache. To a first approximation, the action depends on each cached pair in proportion to the attention allocated to it: attention gauges downstream sensitivity to stale features, whereas drift measures only local representation error. WAM-Cache therefore guides token refresh with two content signals: one tracking feature drift, and the other tracking downstream policy sensitivity. *Latent surprise* approximates drift by measuring how far a token's input embedding has shifted since its last refresh. *Action-conditioned selection* refreshes the tokens most heavily attended to by the action expert in the preceding chunk—leveraging cross-attention maps that are already computed, at zero extra cost. These two signals select largely disjoint token subsets, which WAM-Cache combines via a simple union. However, because both signals rely on rank-based selection, an unselected token can repeatedly miss the cutoff while its cached error steadily compounds, thereby preserving a clear gap to the

dense model. An *age bound* therefore forces a refresh for any token reused across a fixed number of consecutive chunks. This caps the staleness of every pair regardless of rank, nearly closing that gap (Figure 2). Consequently, the prefill runs exclusively on this sparse refresh set, allowing the action expert to operate over the mixed cache without any modification. As a result, WAM-Cache requires no retraining and composes seamlessly with existing orthogonal speedups, including diffusion step reduction, model distillation, and denoiser caching.

On Fast-WAM, WAM-Cache cuts video DiT prefill FLOPs by 41.6% on RoboTwin 2.0 at a cost of 1.8 points in success, whereas token-reduction baselines at a matched budget lose 4 to 38 points. The prefill's CUDA latency drops by $1.23\times$ at the native token count and by $1.67\times$ at a $9\times$ larger visual context. WAM-Cache also removes 32.3% of the prefill FLOPs on LIBERO (Liu et al., 2023) within 0.7 points of the dense model. On two real-world manipulation tasks with an AIRBOT Play arm, it removes 40.2% of the prefill FLOPs while achieving success comparable to the dense model, 82.5% against 85.0%.

Our contributions are summarized as follows:

- We propose **WAM-Cache**, to the best of our knowledge, the first training-free key–value reuse framework for the video DiT prefill of World Action Models (Section 4).
- We show that cache refresh must follow policy attention rather than visual drift: change-based rules plateau 25 points below the dense model even with an oracle, while an age bound strictly caps staleness to nearly close the gap (Section 5.3).
- Across RoboTwin 2.0, LIBERO, and real-world robot experiments, WAM-Cache reduces prefill FLOPs by 32–42% while staying close to the dense model (Section 5).

2 Related Work

World Action Models. Generative video models have been used in robotics for trajectory planning (Du et al., 2023; Zhou et al., 2024; Bharadhwaj et al., 2024; Zhen et al., 2024), policy pretraining (Wu et al., 2023; Cheang et al., 2024; Hu et al., 2024), and data generation or policy evaluation (Jang et al., 2025; Team et al., 2026a), and now serve as policy backbones (Zhu et al., 2025; Li et al., 2025; Liang et al., 2025; Pai et al., 2025; Kim et al., 2026; Liao et al., 2025; Cen et al., 2025). Recent WAMs (Lyu et al., 2025; Chen et al., 2026a) mostly pair video DiTs with action decoders (Li et al., 2026b; Zhang et al., 2026a; Bi et al., 2025; Zhou et al., 2026; Team et al., 2026b; Wang et al., 2026b; Team et al., 2026c). Most of them generate future video before or alongside the actions at inference. Fast-WAM (Yuan et al., 2026) and GigaWorld-Policy (Ye et al., 2026a) show that this future imagination can be dropped with little loss in task success, which cuts inference to a single video DiT prefill that encodes the current observation into layerwise key–value pairs for the action expert. This prefill is now the dominant per-chunk cost, and to the best of our knowledge, WAM-Cache is the first method to make it sparse.

Efficient Policy Inference and Token Reuse. Retraining approaches distill (Akbari et al., 2026) or compress (Li et al., 2026a;f; Ma et al., 2026) the model, reduce denoising steps (Wang et al., 2024; Dong et al., 2026; Chen et al., 2026b; Zhang et al., 2026a), or make future imagination cheaper or adaptive (Zhang et al., 2026b; Li et al., 2026d; Zhao et al., 2026a; Tang et al., 2026; Yan et al., 2026; Sun et al., 2026), and asynchronous schemes overlap inference with execution (Guo & Liu, 2026; Li et al., 2026e;c). Training-free caching reuses denoiser features across denoising steps (Ma et al., 2023; Zou et al., 2024; Feng et al., 2026) or inference chunks (Zeng et al., 2026; Zhao et al., 2026b), but the WAM video prefill is a single pass with no denoising trajectory to reuse. Token reduction prunes (Chen et al., 2024) or merges (Bolya et al., 2022) tokens within a frame, while Eventful Transformer (Dutson et al., 2023) and VLA-Cache (Xu et al., 2025; Wu et al., 2026a;b; Luo et al., 2026) reuse static tokens across frames, selecting them by change or by attention inside the unified backbone, with no bound on how long a token is reused. In a WAM, however, visual prefill and action decoding are handled by decoupled modules. WAM-Cache therefore guides token refresh by downstream policy attention rather than visual drift, while enforcing an age bound that strictly caps staleness.

3 PRELIMINARIES

World Action Models. A world action model (WAM) pairs a video DiT with an action-expert DiT for language-conditioned robot control. The model maps an observation o_t and a text instruction c to an action chunk, and the robot executes a prefix of the chunk before observing again. We call one such cycle an *inference chunk* and index it by t. A VAE encodes o_t into a latent, and a patch-embedding projection maps the latent to S_v visual tokens of width d, $\mathbf{z}_t \in \mathbb{R}^{S_v \times d}$. We write $\mathbf{z}_t = \mathrm{Enc}(o_t)$ for this composition. Throughout, a subscript t marks a quantity computed at chunk t, and $[i]$ indexes token i. The video DiT processes $\mathbf{z}_t$ in a single forward pass. At each of its N denoising steps the action expert cross-attends to the prefill's keys and values to predict the action chunk.

Key–Value Caching in DiTs. A Diffusion Transformer (DiT) (Peebles & Xie, 2023) stacks L modules over a sequence of tokens. Module l applies a self-attention layer $\mathcal{F}^l_{\mathrm{SA}}$, a cross-attention layer $\mathcal{F}^l_{\mathrm{CA}}$ that attends to keys and values projected from the text embeddings, and a feed-forward layer $\mathcal{F}^l_{\mathrm{MLP}}$, each with a residual connection. Self-attention at layer l of the video DiT projects its input $\mathbf{H}^l_t$ into queries $\mathbf{Q}^l_t = \mathbf{H}^l_t \mathbf{W}^l_Q$, keys $\mathbf{K}^l_t = \mathbf{H}^l_t \mathbf{W}^l_K$, and values $\mathbf{V}^l_t = \mathbf{H}^l_t \mathbf{W}^l_V$, and each query attends over all S_v keys and values. A dense prefill thus produces S_v key–value pairs at each of the L layers. Key–value caching stores these pairs so that a later pass can reuse them instead of recomputing them.

4 METHOD

WAM-Cache accelerates inference by retaining layerwise key–value representations across chunks, recomputing only a sparse *refresh set* of visual tokens at each chunk (illustrated in Figure 1). In what follows, we first formalize the cache state, the per-chunk execution procedure, and the resulting computational savings (Section 4.1), and then detail the three complementary signals that construct the refresh set (Sections 4.2–4.4).

4.1 OVERVIEW: STALENESS-BOUNDED KV REUSE

Per-token cache state. Across consecutive inference chunks, WAM-Cache maintains the layerwise key–value representations alongside lightweight tracking metadata for each visual token $i \in \{1, \ldots, S_v\}$: (i) the cached key–value pairs $\{(\mathbf{K}^l[i], \mathbf{V}^l[i])\}_{l=1}^{L}$, (ii) a reference embedding $\mathbf{z}_{\tau_i}[i] \in \mathbb{R}^d$ stored at its most recent refresh chunk τ_i, (iii) an age counter $g_i \in \mathbb{N}$ tracking consecutive chunks of reuse, and (iv) the action expert's attention mass $\mathbf{p}_t[i]$ from the preceding chunk (Section 4.3). At episode onset ($t = 0$), WAM-Cache executes a standard dense prefill, initializing $\tau_i = 0$ and $g_i = 0$ for all tokens. In subsequent chunks, these scalar tracking variables are updated in place without logging historical states.

Per-chunk procedure. For each subsequent chunk $t > 0$, WAM-Cache extracts patch embeddings $\mathbf{z}_t = \mathrm{Enc}(o_t)$ and constructs a sparse *refresh set*

$$\mathcal{R}_t \;=\; \mathcal{S}_t \cup \mathcal{P}_t \cup \mathcal{A}_t \;\subseteq\; \{1, \ldots, S_v\}, \tag{1}$$

formed by the union of a latent-surprise set $\mathcal{S}_t$ (Section 4.2), an action-attention set $\mathcal{P}_t$ (Section 4.3), and an age set $\mathcal{A}_t$ enforcing periodic renewal (Section 4.4). The video DiT prefill then executes exclusively over the tokens in $\mathcal{R}_t$. At each block l, the hidden states $\mathbf{H}^l_t[\mathcal{R}_t]$ are recomputed, and their linear projections overwrite the retained cache in place, while unselected tokens maintain their previously stored representations:

$$\mathbf{K}^l_t[i] = \begin{cases} \mathbf{H}^l_t[i]\, \mathbf{W}^l_K, & i \in \mathcal{R}_t, \\ \mathbf{K}^l_{\tau_i}[i], & i \notin \mathcal{R}_t, \end{cases} \qquad \text{and likewise for } \mathbf{V}^l_t. \tag{2}$$

Meanwhile, queries are formed exclusively for refreshed tokens ($i \in \mathcal{R}_t$) and attend over all S_v keys and values, allowing newly recomputed features to integrate both updated dynamics and static context. Following the pass, WAM-Cache updates the refresh timestamps $\tau_i \leftarrow t$, increments the age counters via Eq. (8), and records the action expert's cross-attention for chunk $t + 1$. Because no token can be reused for more than G consecutive chunks, this mechanism guarantees *staleness-bounded* reuse. Algorithm 1 in Appendix A summarizes the per-chunk procedure.

Prefill cost. With S_c text tokens and feed-forward width d_f, a dense prefill of one video DiT module over S_v visual tokens costs, in FLOPs,

$$C_{\text{dense}} \approx \underbrace{8S_v d^2 + 4S_v^2 d}_{\mathcal{F}_{\text{SA}}} + \underbrace{4S_v d^2 + 4S_v S_c d}_{\mathcal{F}_{\text{CA}}} + \underbrace{4S_v d\, d_f}_{\mathcal{F}_{\text{MLP}}} \tag{3}$$

Let $\rho_t = |\mathcal{R}_t|/S_v$ be the recompute ratio of chunk t, and let $\bar{\rho}$ be its average over the chunks of an episode after the first. Since only the tokens in $\mathcal{R}_t$ form queries and every term of Eq. (3) is linear in the number of query tokens, the per-module cost and the saving are

$$C(\rho_t) = \rho_t\, C_{\text{dense}}, \qquad \Delta C(\rho_t) = (1-\rho_t)\, C_{\text{dense}}. \tag{4}$$

The saving is therefore set entirely by how $\mathcal{R}_t$ is chosen. Appendix B derives both equations, shows that the selection signals add negligible overhead, and bounds $\bar{\rho}$.

The next three subsections build the refresh rule step by step. We start from the intuitive rule of refreshing what changed, and each later signal addresses a failure of the preceding configuration. Figure 2 reports this cumulative construction.

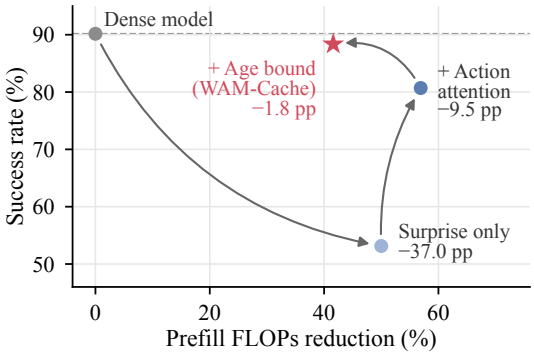


Figure 2: Sequential ablation of caching signals on clean RoboTwin 2.0. The first two configurations run without the age bound.

4.2 Latent Surprise

We begin with the most direct refresh criterion: recompute a token when its input has changed. Target objects shift, or the robot arm traverses previously static visual regions. Even a sequence of minor displacements can separate the current observation from its cached representation.

Surprise. For each token i, we compare its current embedding with its reference $\mathbf{z}_{\tau_i}[i]$ and score the normalized distance

$$\mathbf{s}_t[i] \;=\; \frac{\left\|\mathbf{z}_t[i] - \mathbf{z}_{\tau_i}[i]\right\|_2}{\frac{1}{2}\left(\|\mathbf{z}_t[i]\|_2 + \|\mathbf{z}_{\tau_i}[i]\|_2\right) + \epsilon}, \tag{5}$$

where ϵ prevents division by zero. The latent-surprise set $\mathcal{S}_t = \text{TopK}(\mathbf{s}_t, b_s)$ keeps the highest-scoring tokens under a budget b_s, where $\text{TopK}(\mathbf{s}, b)$ returns the indices of the $\max\{1, \text{round}(bS_v)\}$ largest entries of $\mathbf{s}$. The reference is updated only when its token is recomputed, so the score measures displacement from the observation that the retained pair represents, not from the preceding frame.

Surprise alone is insufficient. On clean RoboTwin 2.0, this rule reaches 53.1% success without the age bound, 37.0 points below the dense model (Figure 2). We attribute this performance gap to regions that change little in latent space yet matter for contact or alignment. Latent displacement does not capture how heavily the action expert relies on a token, so a change-only ranking assigns these regions low priority.

4.3 Action-Conditioned Selection

The first failure concerns regions that stay visually stable while the policy depends on them. Near a grasp, for example, the target object barely moves, yet its exact position determines the next action. The action expert's attention indicates which visual tokens its preceding prediction relied on.

Action attention. Let $\mathbf{A}^l_{t-1} \in \mathbb{R}^{S_a \times S_v}$ be the head-averaged cross-attention from the S_a action queries to the visual tokens at action-expert layer $l \in \{1, \ldots, L_a\}$, recorded at the first action-denoising step of chunk $t-1$. The attention mass on token i is averaged over layers and queries,

$$\mathbf{p}_t[i] \;=\; \frac{1}{L_a S_a} \sum_{l=1}^{L_a} \sum_{m=1}^{S_a} \mathbf{A}^l_{t-1}[m, i], \tag{6}$$

and the action-attention set $\mathcal{P}_t = \mathrm{TopK}(\mathbf{p}_t, b_a)$ keeps the highest-scoring tokens under a budget b_a. The score depends only on the previous chunk, so it is available before the current prefill at no extra cost.

The union $\mathcal{S}_t \cup \mathcal{P}_t$ thus simultaneously captures dynamic physical changes and policy-critical static regions, although inherited attention can lag a transition between manipulation phases. The joint selection raises success to 80.7%, 27.6 points above surprise alone but still 9.5 points below the dense model. We therefore look for a third component that depends on neither content score.

4.4 Age-Bounded Refresh

The second failure arises from repeated reuse. Both content signals are self-referential: each decides which tokens to refresh from state that earlier refresh decisions produced. A refreshed query attends over retained keys and values from unrefreshed tokens, and the subsequent attention score is computed from a partially refreshed context. Because both signals rely on rank-based selection, an unselected token can repeatedly miss the cutoff while its cached error steadily compounds. Neither a low surprise score nor a weak attention weight justifies retaining a token indefinitely.

Age bound. The age set collects every token whose counter has reached the bound G,

$$\mathcal{A}_t \;=\; \{\, i : g_i \geq G \,\}, \tag{7}$$

and these tokens are refreshed regardless of their scores. We write $G = 0$ for the rule disabled, in which case $\mathcal{A}_t = \varnothing$. After the pass, the counters update as

$$g_i \;\leftarrow\; \begin{cases} 0, & i \in \mathcal{R}_t, \\ g_i + 1, & i \notin \mathcal{R}_t. \end{cases} \tag{8}$$

The age bound operates only on state already stored in the cache and adds no forward computation. Yet it nearly closes the gap to the dense model, leaving 1.8 points, while still removing 41.6% of the prefill FLOPs. These forced refreshes target only tokens that neither content score has selected for G chunks and raise $\bar{\rho}$ by at most $1/(G+1)$ (Appendix B). However, the age bound limits how long a pair can stay stale, not how large its error can grow.

5 Experiments

To demonstrate the effectiveness of WAM-Cache, we evaluate our method in both simulation and real-world environments. In the simulation environment, we evaluate WAM-Cache on Fast-WAM (Yuan et al., 2026), using the LIBERO benchmark (Liu et al., 2023) and the RoboTwin 2.0 benchmark (Chen et al., 2025). In the real-world environment, we deploy Fast-WAM with WAM-Cache on an AIRBOT Play arm for two manipulation tasks.

5.1 Implementation Details

We apply WAM-Cache to Fast-WAM (Yuan et al., 2026), which pairs the Wan2.2-5B video DiT (Wan et al., 2025) with a 1B-parameter action-expert DiT and encodes only the current observation at test time (architecture details in Appendix C). All models run in bfloat16 on a single NVIDIA RTX PRO 6000 Blackwell GPU.

Action-denoising steps. We use $N = 2$ action-denoising steps by default and follow Fast-WAM in all other inference settings. For the dense model, two steps stay within 1.5 points of ten on RoboTwin 2.0, whereas one step loses more than 20, and at both step counts the video DiT prefill is the largest per-chunk cost. We also evaluate WAM-Cache separately at Fast-WAM's original $N = 10$, where it stays within 1.1 points of the dense model (Appendix D).

Hyperparameters. We fix $G = 2$ and report two budget settings per benchmark: a default one, to whose prefill FLOPs the baselines are calibrated, and a larger one that approaches the dense model. The exact values of (b_a, b_s) are listed with each table, and Figure 4 shows how they are chosen.

5.2 Simulation Experiments

Benchmarks. RoboTwin 2.0 contains 50 bimanual tasks, each evaluated under clean and randomized scenes, and LIBERO contains four single-arm suites (Spatial, Object, Goal, Long) of ten tasks

Table 1: Results on RoboTwin 2.0 (50 tasks, 100 episodes per task). FLOPs is the per-chunk video DiT prefill cost, and ΔFLOPs and ΔSR are relative to the dense prefill. Each baseline is calibrated to the prefill FLOPs of the first WAM-Cache row of each condition. The two WAM-Cache rows use $(b_a, b_s) = (0.4, 0.1)$ and $(0.6, 0.1)$ under clean observations and $(0.4, 0.2)$ and $(0.4, 0.3)$ under randomization.

Method	Clean				Randomized			
	FLOPs (T) ↓	ΔFLOPs (%)	SR (%) ↑	ΔSR (pp)	FLOPs (T) ↓	ΔFLOPs (%)	SR (%) ↑	ΔSR (pp)
Fast-WAM (dense prefill)	1.053	0.0	90.16	0.0	1.053	0.0	89.66	0.0
+ FastV (Chen et al., 2024)	0.618	−41.3	62.72	−27.4	0.642	−39.0	60.18	−29.5
+ ToMe (Bolya et al., 2022)	0.622	−41.0	77.92	−12.2	0.639	−39.3	74.14	−15.5
+ Eventful (Dutson et al., 2023)	0.623	−40.8	52.94	−37.2	0.641	−39.2	51.36	−38.3
+ VLA-Cache (Xu et al., 2025)	0.623	−40.8	86.16	−4.0	0.649	−38.4	84.26	−5.4
+ WAM-Cache (ours)	**0.615**	**−41.6**	**88.36**	**−1.8**	**0.639**	**−39.3**	**88.32**	**−1.3**
+ WAM-Cache (ours), larger budget	**0.756**	**−28.2**	**90.08**	**−0.1**	**0.663**	**−37.0**	**89.56**	**−0.1**

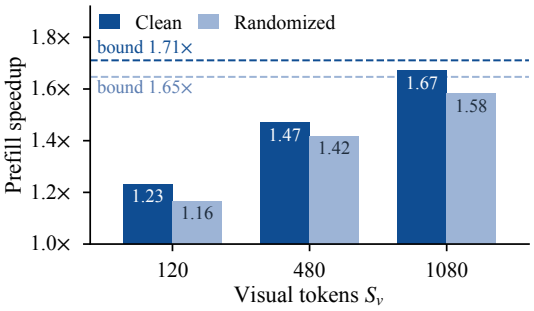


Figure 3: Prefill speedup in CUDA latency of the default clean and randomized configurations. Dashed lines mark the bound $1/\bar{\rho}$.

Selection rule	SR (%) ↑	$\bar{\rho}$ ↓
Random	26.88	0.571
Pixel difference	60.40	0.625
Oracle K/V drift	65.34	0.599
Surprise only	60.62	0.615
Action attention only	81.46	0.643
Attention ∪ surprise (ours)	**88.36**	**0.584**

Table 3: Ablation of refresh rules on clean RoboTwin 2.0 (content budget 0.5, $G = 2$).

each. We use the released Fast-WAM checkpoints and the Fast-WAM evaluation protocol. Episode counts are listed with each table, and execution details are given in Appendix C.

Baselines. We compare against four training-free token-reduction methods, adapted to the same video DiT prefill at a matched FLOPs budget. VLA-Cache (Xu et al., 2025) and Eventful Transformer (Dutson et al., 2023) reuse tokens across frames, while FastV (Chen et al., 2024) and ToMe (Bolya et al., 2022) prune or merge tokens within a frame. VLA-Cache is designed for VLA backbones, so we adapt it to the WAM video DiT prefill. Tokens whose image patches stay static between replans reuse their cached keys and values, except those with high video-to-text attention. Its threshold is calibrated to match the recompute ratio $\bar{\rho}$ of WAM-Cache (Appendix C).

Metrics. We report success rate (SR), prefill FLOPs averaged over all chunks after the first, and CUDA latency of the same stage under CUDA-graph replay, following prior VLM and VLA acceleration work (Chen et al., 2024; Xu et al., 2025).

Main results. Table 1 reports RoboTwin 2.0. At a matched FLOPs budget, WAM-Cache removes 41.6% and 39.3% of the prefill FLOPs under clean and randomized scenes and stays within 1.8 and 1.3 points of the dense prefill, whereas every baseline loses at least 4.0 points. VLA-Cache, the strongest baseline, also combines a change signal with attention, but its attention comes from the language tokens rather than the action expert, and its reuse is unbounded. With a larger budget, WAM-Cache comes within 0.1 points of the dense prefill under both conditions while still removing 28.2% and 37.0% of the prefill FLOPs. Table 2 shows that the same rules transfer to LIBERO with only b_a raised: at a 32.3% reduction WAM-Cache averages 95.85% against 96.50% for the dense prefill and outperforms every baseline at a comparable budget, and at a 26.1% reduction it matches the dense average. On LIBERO the wrist view fills half of the frame, so the policy attends to more tokens per chunk. Most of the remaining loss falls on Long, whose multi-phase tasks expose the lag of inherited attention at phase transitions (Section 4.3).

Table 2: Results on LIBERO (50 episodes per task, 500 per suite). Each baseline is calibrated to the prefill FLOPs of the first WAM-Cache row. The two WAM-Cache rows use $(b_a, b_s) = (0.5, 0.1)$ and $(0.6, 0.1)$.

	SR (%) ↑					FLOPs	ΔFLOPs
Method	Spatial	Object	Goal	Long	Avg.	(T) ↓	(%)
Fast-WAM (dense prefill)	96.80	99.40	96.00	93.80	96.50	0.859	0.0
+ FastV (Chen et al., 2024)	73.20	93.60	75.40	73.00	78.80	0.588	−31.5
+ ToMe (Bolya et al., 2022)	94.20	97.60	92.20	83.80	91.95	0.587	−31.7
+ Eventful (Dutson et al., 2023)	95.40	97.60	94.20	87.40	93.65	0.599	−30.2
+ VLA-Cache (Xu et al., 2025)	96.00	98.40	94.80	90.20	94.85	0.588	−31.5
+ WAM-Cache (ours)	**96.20**	**99.20**	**97.60**	**90.40**	**95.85**	**0.582**	**−32.3**
+ WAM-Cache (ours), larger budget	**96.80**	**99.60**	**95.60**	**94.00**	**96.50**	**0.635**	**−26.1**

CUDA latency. Figure 3 reports the prefill speedup of the first clean and randomized configurations of Table 1 at the native $S_v = 120$ and at $4\times$ and $9\times$ larger visual contexts. No checkpoint exists at the larger sizes, so we replay the refresh ratios measured at $S_v = 120$ and measure cost only. By Eq. (4) the speedup is bounded by $1/\bar{\rho}$. Under clean observations WAM-Cache speeds up the prefill by $1.23\times$, $1.47\times$, and $1.67\times$ at $S_v = 120$, 480, and 1080, reaching 98% of the $1.71\times$ bound at the largest size, and the randomized configuration reaches $1.58\times$, 96% of its bound. At $S_v = 120$ only about 70 tokens are refreshed per chunk, so weight loading and graph replay take a fixed share of each layer's time. The gap to the bound therefore closes as the visual context grows, which is the regime in which the prefill dominates. As WAMs move toward higher resolutions, more camera views, and longer observation histories, the realized speedup of WAM-Cache should increasingly approach its FLOPs bound.

5.3 Ablation Study

Table 3 compares six refresh rules on clean RoboTwin 2.0 under the same content budget, so the rules differ only in *which* tokens they refresh. The single-signal rules spend the whole budget on one score, $b_s = 0.5$ or $b_a = 0.5$, and WAM-Cache splits it as $(b_a, b_s) = (0.4, 0.1)$. The K/V-drift oracle refreshes the tokens whose cached keys and values drift most from a dense prefill of the new frame. It is an upper bound for any rule that predicts cache drift, not a practical method.

Which tokens are refreshed matters more than how many. At a matched recompute ratio, random selection reaches only 26.9%. Every informed rule does far better.

Change-based rules plateau. Three rules select tokens by change: pixel difference (the change signal of VLA-Cache), latent surprise, and the K/V-drift oracle. All three stay between 60.4% and 65.3%. Even ground-truth knowledge of cache drift leaves a 25-point gap to the dense model. Drift is therefore a weak proxy for how much a stale token harms the action (Section 4.2).

Action attention recovers most of the gap. Action attention alone reaches 81.5%. Shifting 0.1 of the budget to surprise raises success to 88.4%, since the two signals select largely disjoint tokens. The union is also the cheapest informed rule, with $\bar{\rho} = 0.584$ against 0.643 and 0.615. The surprise set changes from chunk to chunk, so it resets many counters before the age bound forces a refresh. Tokens selected by both signals are also counted only once.

Age bounding breaks self-referential error accumulation. Without the age bound, content signals select by rank, so low-priority tokens can escape refresh indefinitely. Their error compounds and leaves a 9.5-point gap to the dense model. Setting $G = 2$ bounds the staleness of every token. It recovers 7.7 points of success and brings the policy within 1.8 points of the dense prefill (Figure 2).

Hyperparameter sensitivity. Figure 4 sweeps the age bound G and the attention budget b_a on clean RoboTwin 2.0. A larger G keeps tokens cached longer, which saves FLOPs but lets error accumulate. Moving from $G = 1$ to $G = 2$ saves 10.2 points of FLOPs for only 0.6 points of success. Relative to $G = 2$, every larger bound costs more than a quarter of a point of success per point of FLOPs saved, so we use $G = 2$. Success rises steeply with b_a up to 0.4 and then saturates. At $b_a = 0.6$, WAM-Cache comes within 0.1 points of the dense prefill while still removing 28.2%

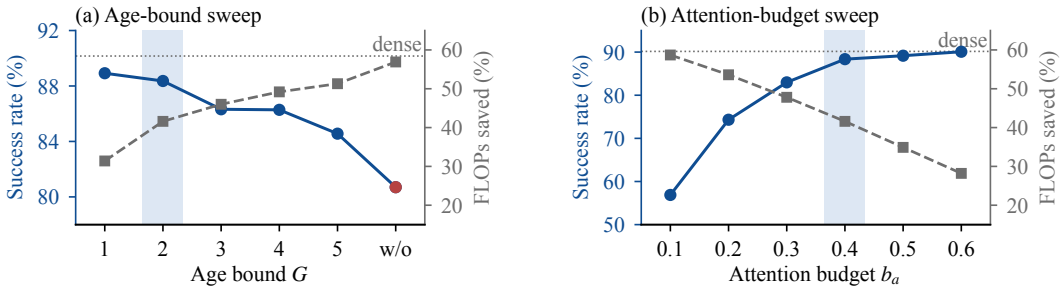


Figure 4: Hyperparameter sweeps on clean RoboTwin 2.0. (a) Age bound G at $(b_a, b_s) = (0.4, 0.1)$, where *w/o* runs without the bound. (b) Attention budget b_a at $b_s = 0.1$ and $G = 2$. Blue circles (left axis) are the success rate and gray squares (right axis) the prefill FLOPs saved. The dotted line is the dense prefill (90.16%), and the shaded band marks the configuration we use.

Table 4: Real-world results on the AIRBOT Play arm (20 trials per task).

	SR (%) ↑			FLOPs	ΔFLOPs
Method	Carrot in Bowl	Stack Cubes	Avg.	(T) ↓	(%)
Fast-WAM (dense prefill)	90.0	80.0	85.0	1.123	0.0
+ WAM-Cache (ours)	**90.0**	**75.0**	**82.5**	**0.672**	**–40.2**

of the prefill FLOPs. The surprise budget b_s only needs to cover moving tokens, so we keep it small. Under randomization, where more tokens move, raising b_s from 0.2 to 0.3 gains 1.2 points of success for 2.3 points of FLOPs (Table 1). We keep $G = 2$ on every benchmark and on the real robot without retuning.

5.4 Real-World Experiments

We further evaluate WAM-Cache on two real-world tasks: placing a carrot in a bowl, which requires grasping an irregular object, and stacking a red cube on a blue cube, which requires precise alignment under contact. We collect 100 demonstrations per task via teleoperation on the AIRBOT Play platform and train Fast-WAM on them for 30k steps. We then apply WAM-Cache with the default clean budgets $(b_a, b_s) = (0.4, 0.1)$ and $G = 2$ without retuning. Observations come from an external camera and a wrist camera. Object placements are randomized in both the demonstrations and the 20 evaluation trials per task, and each trial is scored as a binary success without human intervention. As shown in Table 4, WAM-Cache removes 40.2% of the prefill FLOPs, matches the dense model on carrot placement, and loses a single trial on cube stacking. The budgets chosen in simulation therefore work on the real robot without retuning (setup and rollouts in Appendix E).

6 Conclusion

In this work, we presented **WAM-Cache**, to the best of our knowledge, the first training-free key–value caching framework that targets the dominant computational bottleneck of World Action Models: the video DiT prefill. We show that visual change or representation drift alone is an insufficient refresh signal: even an oracle with access to ground-truth KV drift plateaus far below the dense policy. Instead, downstream action accuracy is governed by *where the policy attends* rather than *what moved*. WAM-Cache therefore combines the action expert's cross-attention with visual latent surprise and adds an age bound on every token, which keeps KV reuse staleness-bounded without retraining. Across RoboTwin 2.0, LIBERO, and real-world manipulation on an AIRBOT Play arm, WAM-Cache cuts video DiT prefill FLOPs by 32–42% while staying within 0.7–1.8 points of the dense policy in simulation and 2.5 points on a real robot. These FLOPs savings yield CUDA-latency speedups that grow as the visual context scales. However, WAM-Cache still leaves room for improvement. At small visual token counts, the measured speedup trails the FLOPs reduction because fixed GPU execution and kernel-launch overheads take a constant share of each layer. This gap narrows as the number of visual tokens grows, which is the regime in which the prefill dominates.

AI USE STATEMENT

We used large language models (LLMs) only to improve the writing of this paper, such as polishing grammar, wording, and readability of text drafted by the authors. LLMs were not used to generate research ideas, design or run experiments, implement methods, analyze or interpret results, or produce any data, figures, or mathematical claims. All AI-edited text was reviewed and revised by the authors, who take full responsibility for the content of this paper.

REFERENCES

Arman Akbari, Ci Zhang, Arash Akbari, Lin Zhao, Yixiao Chen, Weiwei Chen, Xuan Zhang, Geng Yuan, and Yanzhi Wang. Flash-WAM: Modality-Aware Distillation for World Action Models. *arXiv preprint arXiv:2606.05254*, 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.

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.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token Merging: Your ViT But Faster. *arXiv preprint arXiv:2210.09461*, 2022.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, Krzysztof Choromanski, Tianli Ding, Danny Driess, Avinava Dubey, Chelsea Finn, Pete Florence, Chuyuan Fu, Montse Gonzalez Arenas, Keerthana Gopalakrishnan, Kehang Han, Karol Hausman, Alexander Herzog, Jasmine Hsu, Brian Ichter, Alex Irpan, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Lisa Lee, Tsang-Wei Edward Lee, Sergey Levine, Yao Lu, Henryk Michalewski, Igor Mordatch, Karl Pertsch, Kanishka Rao, Krista Reymann, Michael Ryoo, Grecia Salazar, Pannag Sanketi, Pierre Sermanet, Jaspiar Singh, Anikait Singh, Radu Soricut, Huong Tran, Vincent Vanhoucke, Quan Vuong, Ayzaan Wahid, Stefan Welker, Paul Wohlhart, Jialin Wu, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-2: Vision-Language-Action Models Transfer Web Knowledge to Robotic Control. *arXiv preprint arXiv:2307.15818*, 2023.

Jun Cen, Chaohui Yu, Hangjie Yuan, Yuming Jiang, Siteng Huang, Jiayan Guo, Xin Li, Yibing Song, Hao Luo, Fan Wang, Deli Zhao, and Hao Chen. WorldVLA: Towards autoregressive action world model. *arXiv preprint arXiv:2506.21539*, 2025.

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

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*, 2026a.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An Image is Worth 1/2 Tokens After Layer 2: Plug-and-Play Inference Acceleration for Large Vision-Language Models. *arXiv preprint arXiv:2403.06764*, 2024.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

Yuran Chen, Xinye Cai, Zhonglin Gong, and Yang Huang. High-Fidelity One-Step Generative Visuomotor Policy via Recursive Correction, Frequency Consistency, and Contrastive Flow Matching. *arXiv preprint arXiv:2607.03865*, 2026b.

Ju Dong, Liding Zhang, Lei Zhang, Yu Fu, Kaixin Bai, Zoltan-Csaba Marton, Zhenshan Bing, Zhaopeng Chen, Alois Christian Knoll, and Jianwei Zhang. From Flow to One Step: Real-Time Multi-Modal Trajectory Policies via Implicit Maximum Likelihood Estimation-based Distribution Distillation. *arXiv preprint arXiv:2603.09415*, 2026.

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.

Matthew Dutson, Yin Li, and Mohit Gupta. Eventful transformers: Leveraging temporal redundancy in vision transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE, 2023. arXiv:2308.13494.

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. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2603.06331.

Yuanchun Guo and Bingyan Liu. Reflex: Real-Time VLA Control through Streaming Inference. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2607.14695.

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.

Joel Jang, Seonghyeon Ye, Zongyu Lin, Jiannan Xiang, Johan Bjorck, Yu Fang, Fengyuan Hu, Spencer Huang, Kaushil Kundalia, Yen-Chen Lin, Loic Magne, Ajay Mandlekar, Avnish Narayan, You Liang Tan, Guanzhi Wang, Jing Wang, Qi Wang, Yinzhen Xu, Xiaohui Zeng, Kaiyuan Zheng, Ruijie Zheng, Ming-Yu Liu, Luke Zettlemoyer, Dieter Fox, Jan Kautz, Scott Reed, Yuke Zhu, and Linxi Fan. DreamGen: Unlocking generalization in robot learning through video world models. *arXiv preprint arXiv:2505.12705*, 2025.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. OpenVLA: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 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.

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.

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*, 2026b.

Peize Li, Ruimeng Zhang, Ru Zhang, Cong Huang, Kai Chen, and Shanghang Zhang. FBFM: A Training-Free Asynchronous Feedback Mechanism for Flow-Matching in World-Action Models Execution. *arXiv preprint arXiv:2607.29235*, 2026c.

Shuang Li, Yihuai Gao, Dorsa Sadigh, and Shuran Song. Unified video action model. *arXiv preprint arXiv:2503.00200*, 2025.

Xiang Li, Yupeng Zheng, Songen Gu, Huailiang Ma, Feng Yu, Yuhang Zheng, Xian Nie, Shanshuai Yuan, Yujie Zang, Weize Li, Shuai Tian, Moyang Liu, Ya-Qin Zhang, and Wenchao Ding. Latent Action as Intention Enables Efficient Future Imagination for World Action Models. *arXiv preprint arXiv:2608.24882*, 2026d.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM knows what you are looking for before generation. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-0722.

Zekai Li, Jiaming Tang, and Zhijian Liu. FlashVLA: Streaming Action Decoding for Fast and Asynchronous VLA Inference. *arXiv preprint arXiv:2608.27384*, 2026e.

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*, 2026f.

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.

Yue Liao, Pengfei Zhou, Siyuan Huang, Donglin Yang, Shengcong Chen, Yuxin Jiang, Yue Hu, Jingbin Cai, Si Liu, Jianlan Luo, Liliang Chen, Shuicheng Yan, Maoqing Yao, and Guanghui Ren. Genie Envisioner: A unified world foundation platform for robotic manipulation. *arXiv preprint arXiv:2508.05635*, 2025.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. *arXiv preprint arXiv:2306.03310*, 2023.

Qi Luo, Shuaijun Liu, Hao Zhao, Kunlin Li, Xiaobo Wang, Ningxing Su, Dongsheng Wang, and Yun Chen. The Gate, Not the Cache: Gate Provenance Bounds the Closed-Loop Reliability of Training-Free VLA Token Skipping. *arXiv preprint arXiv:2608.00391*, 2026.

Jiangran Lyu, Ziming Li, Xuesong Shi, Chaoyi Xu, Yizhou Wang, and He Wang. DyWA: Dynamics-adaptive world action model for generalizable non-prehensile manipulation. *arXiv preprint arXiv:2503.16806*, 2025.

Liheng Ma, Rui Heng Yang, Zhanguang Zhang, Mateo Clemente, Ziwen Hu, Tongtong Cao, and Yingxue Zhang. Faster-WAM: Do World Action Models Need Deep Action Modules? *arXiv preprint arXiv:2608.02365*, 2026.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. DeepCache: Accelerating Diffusion Models for Free. *arXiv preprint arXiv:2312.00858*, 2023.

Weian Mao, Xi Lin, Wei Huang, Yuxin Xie, Tianfu Fu, Bohan Zhuang, Song Han, and Yukang Chen. TriAttention: Efficient Long Reasoning with Trigonometric KV Compression. *arXiv preprint arXiv:2604.04921*, 2026.

NVIDIA, Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, Daniel Dworakowski, Jiaojiao Fan, Michele Fenzi, Francesco Ferroni, Sanja Fidler, Dieter Fox, Songwei Ge, Yunhao Ge, Jinwei Gu, Siddharth Gururani, Ethan He, Jiahui Huang, Jacob Huffman, Pooya Jannaty, Jingyi Jin, Seung Wook Kim, Gergely Klár, Grace Lam, Shiyi Lan, Laura Leal-Taixe, Anqi Li, Zhaoshuo Li, Chen-Hsuan Lin, Tsung-Yi Lin, Huan Ling, Ming-Yu Liu, Xian Liu, Alice Luo, Qianli Ma, Hanzi Mao, Kaichun Mo, Arsalan Mousavian, Seungjun Nah, Sriharsha Niverty, David Page, Despoina Paschalidou, Zeeshan Patel, Lindsey Pavao, Morteza Ramezanali, Fitsum Reda, Xiaowei Ren, Vasanth Rao Naik Sabavat, Ed Schmerling, Stella Shi, Bartosz Stefaniak, Shitao Tang, Lyne Tchapmi, Przemek Tredak, Wei-Cheng Tseng, Jibin Varghese, Hao Wang, Haoxiang Wang, Heng Wang, Ting-Chun Wang, Fangyin Wei, Xinyue Wei, Jay Zhangjie Wu, Jiashu Xu, Wei Yang, Lin Yen-Chen, Xiaohui Zeng, Yu Zeng, Jing Zhang, Qinsheng Zhang, Yuxuan Zhang, Qingqing Zhao, and Artur Zolkowski. Cosmos world foundation model platform for physical AI. *arXiv preprint arXiv:2501.03575*, 2025.

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.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4195–4205. IEEE, 2023.

Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Anselm Levskaya, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. In *Proceedings of Machine Learning and Systems*, volume 5, 2023.

Yirui Sun, Guangyu Zhuge, Keliang Liu, Jie Gu, Shiqin Dai, Xinyu Bing, Zhongxue Gan, and Chunxu Tian. SANTS: A State-Adaptive Scheduler for World Action Models. *arXiv preprint arXiv:2605.27947*, 2026.

Yinzhou Tang, Jingbo Xu, Yu Shang, Zihao Song, Chen Gao, Wei Wu, and Yong Li. Dreaming when Necessary: Advancing World Action Models with Adaptive Multi-Modal Reasoning. *arXiv preprint arXiv:2606.07089*, 2026.

GigaWorld Team, Angyuan Ma, Boyuan Wang, Bohan Li, Chaojun Ni, Guo Li, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, Jingyu Liu, Jiwen Lu, Qiuping Deng, Tingdong Yu, Xuancheng Xu, Xinyu Zhou, Xiuwei Xu, Xinze Chen, Xiaofeng Wang, Xiaoyu Tian, Yang Wang, Yifan Chang, Yukun Zhou, Yun Ye, Zhenyu Wu, Zhanqian Wu, and Zheng Zhu. GigaWorld-1: A Roadmap to Build World Models for Robot Policy Evaluation. *arXiv preprint arXiv:2607.02642*, 2026a.

GigaWorld Team, Angen Ye, Angyuan Ma, Boyuan Wang, Chaojun Ni, Fangzheng Ye, Guan Huang, Guo Li, Guosheng Zhao, Haodong Yan, Hengtao Li, Jiwen Lu, Kai Wang, Mingming Yu, Qitang Hu, Qiuping Deng, Songling Liu, Xiaoyu Tian, Xiaofeng Wang, Xinyu Zhou, Xiuwei Xu, Xinze Chen, Yang Wang, Yejun Zeng, Yifan Chang, Yun Ye, Zhenyu Wu, Zhanqian Wu, and Zheng Zhu. GigaWorld-Policy-0.5: A Faster and Stronger WAM Empowered by AutoResearch. *arXiv preprint arXiv:2607.13960*, 2026b.

Motubrain Team, Chendong Xiang, Fan Bao, Haitian Liu, Hengkai Tan, Hongzhe Bi, James Li, Jiabao Liu, Jingrui Pang, Kiro Jing, Louis Liu, Mengchen Cai, Rongxu Cui, Ruowen Zhao, Runqing Wang, Shuhe Huang, Yao Feng, Yinze Rong, Zeyuan Wang, and Jun Zhu. MotuBrain: An Advanced World Action Model for Robot Control. *arXiv preprint arXiv:2604.27792*, 2026c.

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

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*, 2026a.

Yuran Wang, Siqiao Huang, Mingleyang Li, Chenhao Zhang, Jiaqi Liang, Weiyang Jin, Yue Chen, Xuemin Chi, Donghao Zhou, Qize Yu, Yu-Kai Wang, Yuhan Rui, Shenzhe Yao, Zhen Yuan, Zhenhao Shen, Kefei Zhu, Zijie Zhu, Ning Gao, Xiaowei Chi, Guanqi He, Shanghang Zhang, Hao Dong, Lin Shao, and Hang Zhao. OpenWAM: An open, modular exploration towards systematic world-action model pretraining. *arXiv preprint arXiv:2609.07398*, 2026b.

Zhendong Wang, Zhaoshuo Li, Ajay Mandlekar, Zhenjia Xu, Jiaojiao Fan, Yashraj Narang, Linxi Fan, Yuke Zhu, Yogesh Balaji, Mingyuan Zhou, Ming-Yu Liu, and Yu Zeng. One-Step Diffusion

Policy: Fast Visuomotor Policies via Diffusion Distillation. *arXiv preprint arXiv:2410.21257*, 2024.

Hongtao Wu, Ya Jing, Chilam 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.

Yuzhou Wu, Yuxin Zheng, Muchun Niu, Yishan Yang, Tianhao Liu, Hanwen Kang, Jiajian Jing, Linfeng Zhang, and Chuan Wen. Reducing Temporal Redundancy for Efficient Vision-Language-Action Inference. *arXiv preprint arXiv:2607.12287*, 2026a.

Zhijie Wu, Kento Kawaharazuka, and Kei Okada. Neural Introspection Gating for Adaptive KV-Cache Reuse in Vision-Language-Action Models. *arXiv preprint arXiv:2608.10824*, 2026b.

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.

Ge Yan, Jinghao Liu, Yuzhi Fan, Lei Cai, Minwen Liao, Jesse Zhang, and Dieter Fox. Flex-π: A Multi-Stream World-Action Model with Compute Flexibility. *arXiv preprint arXiv:2608.10860*, 2026.

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.

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.

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.

Yixiao Zeng, Jianlei Zheng, Chaoda Zheng, Shijia Chen, Mingdian Liu, Tongping Liu, Tengwei Luo, Yu Zhang, Boyang Wang, Linkun Xu, Siyuan Lu, Bo Tian, and Xianming Liu. X-Cache: Cross-Chunk Block Caching for Few-Step Autoregressive World Models Inference. *arXiv preprint arXiv:2604.20289*, 2026.

Qihang Zhang, Lin Li, Luyao Zhang, Shuai Yang, Yiming Luo, Shuaiting Li, Ruilin Wang, Junke Wang, Jiahao Shao, Gangwei Xu, Jiaming Zhou, et al. Native video-action pretraining for generalizable robot control. *arXiv preprint arXiv:2607.08639*, 2026a.

Yuyang Zhang, Wenyao Zhang, Zekun Qi, He Zhang, Haitao Lin, Jingbo Zhang, Yao Mu, Xiaokang Yang, Wenjun Zeng, and Xin Jin. ImageWAM: Do World Action Models Really Need Video Generation, or Just Image Editing? *arXiv preprint arXiv:2606.19531*, 2026b.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. H2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-1506.

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.

Weisen Zhao, Lam Nguyen, Zhicong Lu, and Yuzhang Shang. C^3ache: Accelerating World Action Models with Cross Inference Chunk Cache. *arXiv preprint arXiv:2606.08962*, 2026b.

Haoyu Zhen, Xiaowen Qiu, Peihao Chen, Jincheng Yang, Xin Yan, Yilun Du, Yining Hong, and Chuang Gan. 3D-VLA: A 3D vision-language-action generative world model. *arXiv preprint arXiv:2403.09631*, 2024.

Jiaming Zhou, Qihang Zhang, Gangwei Xu, Cunxin Fan, Yujie Zhao, Ruilin Wang, Yiming Luo, Shuai Yang, Xing Zhu, Yujun Shen, Junwei Liang, and Yinghao Xu. Zero-WAM: In-context world-action modeling from human videos for open-ended task generalization. *arXiv preprint arXiv:2608.26103*, 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.

Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating Diffusion Transformers with Token-wise Feature Caching. *arXiv preprint arXiv:2410.05317*, 2024.

A WAM-CACHE ALGORITHM

Algorithm 1 gives the per-chunk procedure of Section 4.

Algorithm 1 WAM-Cache: one inference chunk $t > 0$

Require: observation o_t; retained $\{\mathbf{K}^l, \mathbf{V}^l\}_{l=1}^{L}$, references $\{\mathbf{z}_{\tau_i}[i]\}$, counters $\{g_i\}$, previous attention $\{\mathbf{A}_{t-1}^l\}$; budgets b_s, b_a; age bound G

$\mathbf{z}_t \leftarrow \mathrm{Enc}(o_t)$
$\mathcal{S}_t \leftarrow \mathrm{TopK}(\mathbf{s}_t, b_s)$, Eq. (5)
$\mathcal{P}_t \leftarrow \mathrm{TopK}(\mathbf{p}_t, b_a)$, Eq. (6)
$\mathcal{A}_t \leftarrow \{i : g_i \geq G\}$ if $G > 0$, else $\varnothing$, Eq. (7)
$\mathcal{R}_t \leftarrow \mathcal{S}_t \cup \mathcal{P}_t \cup \mathcal{A}_t$, Eq. (1)
for $l = 1$ **to** L **do** ▷ video DiT prefill on $\mathcal{R}_t$
  recompute $\mathbf{H}_t^l[\mathcal{R}_t]$; update $\mathbf{K}^l[\mathcal{R}_t], \mathbf{V}^l[\mathcal{R}_t]$, Eq. (2)
  queries of $\mathcal{R}_t$ attend over full $\mathbf{K}^l, \mathbf{V}^l$ and the cached text KV
end for
run action expert for N denoising steps over prefilled $\{\mathbf{K}^l, \mathbf{V}^l\}$; record $\{\mathbf{A}_t^l\}$ at step 1
$\tau_i \leftarrow t$, $g_i \leftarrow 0$ for $i \in \mathcal{R}_t$; $g_i \leftarrow g_i + 1$ otherwise, Eq. (8)

Ensure: action chunk

B COST ANALYSIS

This appendix derives the cost statements of Section 4.1.

Prefill FLOPs. We count a multiply–accumulate as two FLOPs. Eq. (3) omits normalization and modulation terms of order $S_v d$. The self-attention term covers the query, key, value, and output projections and the two $S_v \times S_v$ attention products. The cross-attention term covers the query and output projections and the two $S_v \times S_c$ products. The MLP term covers its two linear layers. The text keys and values of $\mathcal{F}_{\mathrm{CA}}$ do not appear. They depend only on the instruction, which is constant within an episode, so they are projected once at the first chunk and reused by every later chunk, for the dense model and for WAM-Cache alike.

WAM-Cache forms queries only for the refreshed tokens, as in Eq. (2). Every term of Eq. (3) is therefore scaled by ρ_t, including the two attention products, in which the refreshed queries still attend over all S_v keys. This gives Eq. (4), and the same factor applies to the full prefill of L modules. The relative saving $1 - \rho_t$ therefore does not depend on S_v, d, or d_f and is set by the refresh policy alone.

Selection overhead. Latent surprise costs $\mathcal{O}(S_v d)$ for the per-token distances of Eq. (5). The action-attention score reuses attention maps that the action expert already computes and aggregates them in $\mathcal{O}(L_a S_a S_v)$. The age rule costs $\mathcal{O}(S_v)$, and the two top-k selections cost $\mathcal{O}(S_v \log S_v)$. Each of these is below the $\mathcal{O}(S_v d^2)$ of a single projection in one module, and none is visible in the measured CUDA latency (Section 5.2).

Bounding the recompute ratio. Content-based selection alone gives the per-chunk bound $\rho_t \leq b_s + b_a$, with equality only when $\mathcal{S}_t$ and $\mathcal{P}_t$ are disjoint. The age term can exceed this bound in an individual chunk. In the worst case, synchronized counters force a refresh of every token at once. The amortized contribution of the age term is bounded, however. After any refresh, a token's counter resets, so two forced refreshes of the same token are separated by at least $G + 1$ chunks. The time-averaged ratio therefore satisfies

$$\bar{\rho} \;\leq\; b_s + b_a + \frac{1}{G+1}. \tag{9}$$

The bound tightens to $b_s + b_a + (1 - b_s - b_a)/(G + 1)$ when the content signals concentrate on a stable subset of tokens, since a token refreshed by content never triggers the age bound. The measured steady-state ratio of our default configuration, 0.584, lies below both bounds, 0.83 and 0.67, and so does that of every configuration in this paper (Section 5.2). It can also fall below $b_s + b_a$, since $\mathcal{S}_t$ and $\mathcal{P}_t$ may overlap.

C Implementation Details

Base model. Fast-WAM (Yuan et al., 2026) builds on the Wan2.2-5B video DiT (Wan et al., 2025) and adds a 1B-parameter action-expert DiT, while the T5 text encoder and video VAE remain unchanged. The two branches share attention in a Mixture-of-Transformer structure. The images of all cameras are spatially concatenated into a single frame before VAE encoding, which yields S_v visual tokens (Section 3). On RoboTwin 2.0 the head camera and two wrist cameras form a 384×320 (height $\times$ width) frame with $S_v = 120$, and on LIBERO the third-person and wrist cameras form a 224×448 frame with $S_v = 98$.

Budget choices. Randomization on RoboTwin 2.0 moves distractors and changes appearance, so more tokens are displaced per chunk. A larger surprise budget ($b_s = 0.2$ and 0.3) keeps task-relevant motion from being crowded out of the surprise set. On LIBERO the wrist view fills half of the frame, against a third on RoboTwin 2.0. The gripper and the manipulated object dominate this view, so the policy reads more tokens per chunk. We therefore raise b_a to 0.5 and 0.6 at $b_s = 0.1$.

Baseline adaptation. VLA-Cache (Xu et al., 2025) reuses tokens inside a VLA backbone that both encodes the observation and decodes the action, whereas a WAM separates the two into a video DiT and an action expert. We therefore apply it to the video DiT prefill. Tokens whose image patches stay static between replans reuse their cached keys and values, unless they receive high video-to-text attention. The static threshold is calibrated to match the recompute ratio $\bar{\rho}$ of WAM-Cache.

Evaluation protocol. We run 100 episodes per task under each condition on RoboTwin 2.0 and 50 episodes per task on LIBERO, with success averaged over tasks. Each chunk executes 24 actions before replanning on RoboTwin 2.0 and 10 on LIBERO.

D Denoising Steps and Prefill Share

Step sweep. Table 5 reports the dense Fast-WAM checkpoint for $N \in \{1, 2, 5, 10\}$ action-denoising steps, without caching. RoboTwin 2.0 follows the protocol of Appendix C with a fixed seed, and LIBERO uses the released LIBERO checkpoint. On RoboTwin 2.0, $N = 5$ and $N = 10$ differ by less than half a point. $N = 2$ is 1.3 and 1.5 points below $N = 10$ under clean and randomized scenes, and $N = 1$ loses more than 20 points. On LIBERO the sweep is nearly flat. $N = 2$ is 0.6 points below $N = 10$ on average, and even $N = 1$ loses only 1.4 points, mostly on Long (94.6 to 91.2) and Goal (96.6 to 95.0). The step count the action expert needs therefore depends on the benchmark, and $N = 2$ is the smallest value that holds on both. We attribute the failure of $N = 1$ on RoboTwin 2.0 to the action distribution. A single step moves from noise to action along one straight line. This suffices when the target is nearly unimodal, as in LIBERO's single-arm tasks, but it averages across modes when bimanual tasks admit several valid coordinations. Unlike the step reduction of LingBot-VA 2.0 (Zhang et al., 2026a), ours requires no consistency distillation. Fast-WAM (Yuan et al., 2026) reports $N = 10$. We use $N = 2$ by default and measure every success rate against the dense model at the same N, so the step reduction and the caching are never conflated.

Table 5: Dense Fast-WAM success rate (%) versus the number of action-denoising steps N on RoboTwin 2.0 and LIBERO.

	RoboTwin 2.0			LIBERO				
N	Clean	Rand.	Avg.	Spatial	Object	Goal	Long	Avg.
1	69.72	67.72	68.72	97.4	99.0	95.0	91.2	95.65
2	90.16	89.66	89.91	96.8	99.4	96.0	93.8	96.50
5	91.94	90.80	91.37	96.8	99.6	96.6	94.2	96.80
10	91.46	91.18	91.32	97.2	99.8	96.6	94.6	97.05

WAM-Cache at $N = 10$. Table 6 repeats the RoboTwin 2.0 comparison of Table 1 at Fast-WAM's original $N = 10$ with the same budgets. At the first budget, WAM-Cache removes 41.6% and 39.5% of the prefill FLOPs under clean and randomized scenes and stays within 0.6 and 1.1 points of the dense model. These gaps are smaller than the 1.8 and 1.3 points at $N = 2$. At the larger

budget, WAM-Cache matches the dense model under clean scenes and comes within 0.2 points under randomized scenes. The refresh rule therefore carries over to $N = 10$ without retuning.

Table 6: WAM-Cache on RoboTwin 2.0 at $N = 10$ action-denoising steps (50 tasks $\times$ 100 episodes per cell), with the budget settings of Table 1. ΔSR and ΔFLOPs are relative to the dense prefill at $N = 10$.

Scene	Method	(b_a, b_s)	SR (%) ↑	ΔSR (pp)	ΔFLOPs (%)
Clean	Fast-WAM (dense prefill)	–	91.46	0.0	0.0
	+ WAM-Cache	$(0.4, 0.1)$	90.82	-0.6	-41.6
	+ WAM-Cache, larger budget	$(0.6, 0.1)$	91.46	0.0	-28.2
Randomized	Fast-WAM (dense prefill)	–	91.18	0.0	0.0
	+ WAM-Cache	$(0.4, 0.2)$	90.04	-1.1	-39.5
	+ WAM-Cache, larger budget	$(0.4, 0.3)$	90.94	-0.2	-37.3

Prefill share. Table 7 splits the per-chunk FLOPs at $S_v = 120$ into the video DiT prefill, the N action-expert steps, and the VAE encoding of the observation. The prefill costs 1.053 TFLOPs regardless of N, excluding the text keys and values that are projected once per episode. The action expert costs 0.104 TFLOPs per step. At $N = 2$ the prefill takes 83.5% of the FLOPs of the two DiTs and 57.5% of the per-chunk total including the VAE. At $N = 10$ these shares are 50.2% and 39.5%. The prefill therefore dominates the per-chunk cost at both step counts, and more so at the $N = 2$ we deploy. With the 41.6% prefill reduction of Table 1, the two DiTs drop from 1.262 to 0.824 TFLOPs per chunk at $N = 2$, a 35% reduction, or 24% including the VAE. Compared with dense Fast-WAM at $N = 10$, this is $2.5\times$ fewer DiT FLOPs per chunk.

Table 7: Per-chunk FLOPs (T) of Fast-WAM at $S_v = 120$ and the share taken by the video DiT prefill, with and without the VAE encoding.

	FLOPs (T)			Prefill share	
N	Prefill	Action expert	VAE	DiTs only	incl. VAE
2	1.053	0.209	0.571	83.5%	57.5%
10	1.053	1.044	0.571	50.2%	39.5%

E Real-World Setup and Rollouts

Platform. The AIRBOT Play is a 6-DoF lightweight arm from DISCOVER Robotics with a 1.5 kg rated payload, a 647 mm reach, $\pm$0.1 mm repeatability, and a 3.78 kg arm mass, fitted with a parallel-jaw gripper. Joints 1–3 rotate at up to 180°/s and joints 4–6 at up to 360°/s. Observations come from an external RGB camera facing the workspace and a wrist-mounted RGB camera, synchronized at 20 Hz, and the two views are concatenated into one frame before VAE encoding. Actions are streamed to the arm at 20 Hz.

Tasks and protocol. *Carrot in Bowl* requires grasping a carrot from the table and placing it in a bowl, and succeeds when the carrot rests inside the bowl at the end of the episode. *Stack Cubes* requires placing a red cube on a blue cube, and succeeds when the red cube rests on the blue cube without falling. For each task we collect 100 demonstrations via teleoperation with randomized object placements, train Fast-WAM on them for 30k steps, and evaluate 20 trials with placements drawn from the same ranges. Each trial is scored as a binary success without human intervention.

Rollouts. Figure 5 shows one rollout of each task from the external camera. The arm approaches the object, grasps it, carries it over the target, and releases it, while the rest of the scene stays fixed from frame to frame.

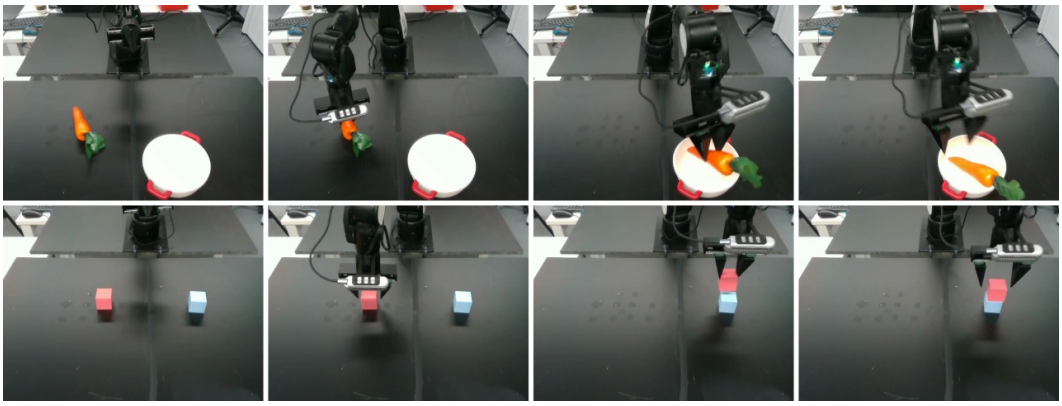


Figure 5: Real-world tasks on the AIRBOT Play arm, viewed from the external camera. Top: picking up a carrot and placing it in a bowl. Bottom: stacking a red cube on a blue cube. Frames run left to right, from the initial scene through grasping and transport to placement.