ACCELERATING WORLD ACTION MODELS VIA ACTION-AWARE ASYNCHRONOUS CACHING

Anonymous authors
Paper under double-blind review

ABSTRACT

Multi-view world action models enable robots to predict future interactions from multiple camera views, while introduce substantial inference cost by multi-view processing. Although training-free caching has shown promise in accelerating diffusion and video world models, existing methods largely treat multi-view representations as a unified sequence and fail to exploit the uneven dynamics across different camera views, limiting their efficiency under aggressive acceleration. In this work, we propose **A3Cache**: **Action-Aware Asynchronous Caching**, a training-free acceleration framework for multi-view embodied world action models, which transforms multi-view caching into a asynchronous computation paradigm by adapting computation according to their update demands. Specifically, we introduce **view asynchronous computing** paradigm, which independently evaluates cache validity for different camera views and computes only the active views while predicting the stable ones by cache. We further introduce **world-action contribution-aware token delaying** paradigm within active views, which jointly considers token cache deviation and changes in visual-to-action contribution to prioritize urgent local computation while delaying less urgent tokens. Extensive experiments on multi-view robotic action demonstrate that A3Cache achieves a better efficiency-performance trade-off than existing methods, delivering substantially higher acceleration ratio with minimal performance degradation.

1 INTRODUCTION

World action models take current observations and instructions as input to jointly predict future visual states and robot actions, showing strong potential for embodied intelligence by modeling interaction dynamics Yang et al. (2024); Huang et al. (2026); Bi et al. (2026). For robotic manipulation, however, a single camera often provides incomplete observations due to occlusion and limited field of view. Multi-view observations offer complementary global and local information, which can reduce visual ambiguity and improve robot control Seo et al. (2023). Recent methods Huang et al. (2026); Liao et al. (2026); Bi et al. (2026) therefore model multiple synchronized views to better represent future robot–environment interactions. These advances make multi-view embodied world action models an important direction for embodied prediction and control.

The benefits of multi-view modeling come at a high inference cost. Video world models already require repeated computation over large spatiotemporal features across multiple denoising steps, making iterative generation computationally expensive Wan et al. (2025); Bi et al. (2026); Liao et al. (2026). Multi-view modeling further increases the amount of visual information processed at each step and often introduces additional cross-view interaction cost Bi et al. (2026); Liao et al. (2026). As a result, multi-view world action inference becomes substantially more expensive than its single-view counterpart. Meanwhile, these models also contain substantial computational redundancy. In addition to the temporal and layer-wise redundancy commonly observed in diffusion models Liu et al. (2025); Bu et al. (2026), multi-view inputs introduce extra redundancy because motion and scene changes are often concentrated in only a subset of camera views or local spatial regions, while large portions of the remaining views change little. As illustrated in Figure 1 (a), the update demand in multi-view embodied inference is often uneven: action-related changes are often concentrated in only a few local regions or views, while portions of the remaining observations stay relatively stable. This combination of high cost and structured redundancy makes multi-view world action models a natural target for training-free inference acceleration.

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.

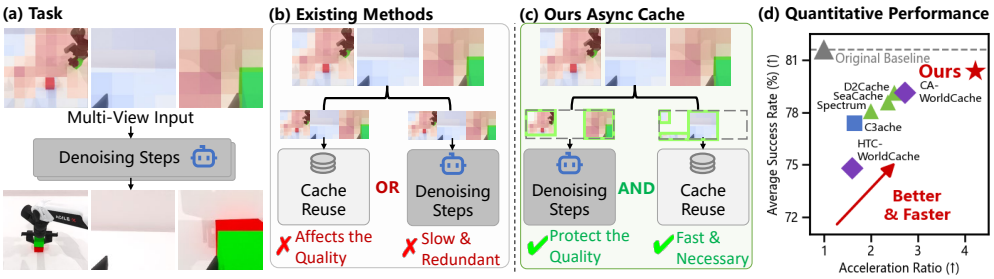


Figure 1: **Motivation and Effectiveness of A3Cache.** (a) Uneven update demand. Action-related changes are concentrated in only a few views and local regions, while other visual content remains stable. (b) Uniform caching. Existing methods either reuse all cached features or recompute them synchronously, causing quality degradation or redundant computation. (c) Asynchronous selective caching. A3Cache computes only high-risk views and urgent local tokens, while delaying stable views and regions through cache reuse. (d) Quality–efficiency trade-off. A3Cache achieves substantially higher acceleration while preserving stronger downstream task success.

Existing training-free acceleration methods mainly reuse intermediate features across denoising steps based on feature similarity, token dynamics, or block-level changes Liu et al. (2025); Bu et al. (2026); Feng et al. (2026); Nawaz et al. (2026). Although these methods effectively reduce repeated computation, they do not explicitly model the view structure of multi-view embodied world action models. In these models, motion can be highly uneven across cameras: an active wrist view may change rapidly, while a global overview remains largely stable. Moreover, even within an active view, not all local visual tokens are equally urgent for current-step action generation. Without modeling this structure, conservative cache decisions introduce unnecessary computation in stable views, whereas aggressive reuse can skip action-relevant local changes and degrade downstream task performance. This limitation makes it difficult for existing methods to maintain reliable generation under high acceleration ratios.

To address this problem, we propose **A3Cache**, a training-free acceleration framework for multi-view embodied world action models. A3Cache reformulates synchronized multi-view inference as hierarchical asynchronous computation, where computation is allocated according to the update demands of individual views and local visual tokens. As illustrated in Figure 1(b,c), this design decouples the computation schedules across camera views and further differentiates local computation within active views. Consequently, stable views and low-urgency regions are delayed through cache reuse, while computation is concentrated on active views and action-relevant visual features.

Specifically, we first introduce **View Asynchronous Computing**, which independently evaluates cache validity for different camera views and computes only the cache-invalid views, preventing local changes in one view from repeatedly triggering unnecessary updates in the others. We further propose **World-Action contribution-aware token delaying** in active views, which jointly considers token cache deviation and changes in visual-to-action contribution to estimate the urgency of local computation. By prioritizing urgent tokens while delaying less urgent ones, A3Cache further reduces redundant visual computation without disrupting action-relevant feature evolution. Figure 1(c) gives an comparison between our asynchronous selective strategy and existing uniform caching strategies.

In summary, our contributions are as follows:

- We propose **A3Cache**: Action-Aware Asynchronous Caching, a training-free framework for accelerating multi-view world action models, which transforms multi-view caching into a asynchronous computation paradigm by adapting computation according to their update demands, enabling higher-ratio acceleration while preserving embodied task performance.
- We introduce **View Asynchronous Computing**, which independently evaluates cache validity for different camera views and computes only the active views, allowing computing of stable views to be skiped instead of being repeatedly triggered by risk of other views.
- We propose **World-Action Contribution-aware Token Delaying**, which jointly considers token cache deviation and visual-to-action contribution deviation to prioritize urgent local computation, reducing redundant cost in active views while preserving action-relevant visual information.

Extensive experiments across multi-view embodied manipulation tasks demonstrate that A3Cache achieves a strong quality–efficiency trade-off while significantly improving inference efficiency. As summarized in Figure 1(d), our method achieves the best balance between acceleration and task success among compared methods. Notably, A3Cache maintains over 98% of the original performance under over $4\times$ acceleration, showing advantages especially at high-acceleration rate.

2 Related Work

Multi-view embodied world models. World models learn predictive dynamics of the environment for planning and control Hafner et al. (2020; 2023); Hu et al. (2023); Wang et al. (2024); Zhao et al. (2025), while recent generative world models extend this idea by directly predicting future visual observations Yang et al. (2024); Agarwal et al. (2025). In robotic manipulation, multiple cameras provide complementary observations under occlusion and limited fields of view, motivating world models that jointly learn from multiple viewpoints. Multi-View Masked World Models Seo et al. (2023) learn multi-view representations and dynamics for visual manipulation. More recent video-based approaches further model future robotic interactions from multiple views: EnerVerse Huang et al. (2026) represents embodied environments with multi-view videos, Ctrl-World Guo et al. (2026) supports controllable multi-view prediction for robot manipulation, and Genie Envisioner Liao et al. (2026) learns visual dynamics and actions within a video-generative framework. Motus Bi et al. (2026) further unifies video generation, understanding, and action prediction for embodied control with multi-view observations. These models provide richer observations and stronger world representations for robotic tasks, but also introduce substantial computation from large video backbones, iterative generation, and multi-view processing.

Efficient inference for diffusion and world action models. Diffusion models require repeated network evaluations over multiple denoising steps, motivating training-free methods that reuse or approximate intermediate features to reduce redundant computation. TeaCache Liu et al. (2025) determines feature reuse from timestep-modulated feature differences, while FasterCache Lv et al. (2024) exploits feature and classifier-free-guidance redundancy in video diffusion models. ToCa Zou et al. (2025) further performs token-wise feature caching to account for non-uniform token dynamics, and DiCache Bu et al. (2026) uses a shallow-layer probe to adaptively determine cache reuse. Recent methods further improve caching at different granularities, including block-level and sample-adaptive reuse Cui et al. (2026). D2Cache Liu et al. (2026) improves feature approximation through second-order residual-delta prediction, while SeaCache Chung et al. (2026) introduces spectral-evolution-aware cache scheduling to better estimate redundancy along the denoising trajectory. Spectrum Han et al. (2026) instead models latent feature trajectories with spectral bases to enable longer-range feature forecasting. Beyond general diffusion generation, Heterogeneous WorldCache Feng et al. (2026) models non-uniform token dynamics in diffusion world models, while Content-Aware WorldCache Nawaz et al. (2026) introduces motion-aware cache decisions and feature approximation for video world models. Despite substantial efficiency gains, these methods mainly model redundancy along temporal, or block dimensions, without explicitly considering the asynchronous dynamics of different camera views in multi-view world action models.

Beyond video generation, world action models introduce additional cost from iterative world prediction and action generation. Fast-WAM Yuan et al. (2026) removes explicit future video generation at inference while retaining video modeling during training, whereas AHA-WAM Cai et al. (2026) decouples a low-frequency world planner from high-frequency action execution through a dual-DiT architecture and asynchronous world–action modeling. However, these approaches modify the model architecture or training formulation and therefore require dedicated training or model adaptation, making them less directly applicable to pretrained WAMs and introducing additional data and training costs. In contrast, C^3ache Zhao et al. (2026) provides training-free acceleration by exploiting redundancy across consecutive WAM inference chunks through residual reuse. These works demonstrate the value of WAM-specific efficiency designs, but primarily focus on future-generation frequency, world–action execution frequency, or cross-chunk redundancy, leaving view-specific computational redundancy within synchronized multi-view inference and its relation to action generation largely unexplored.

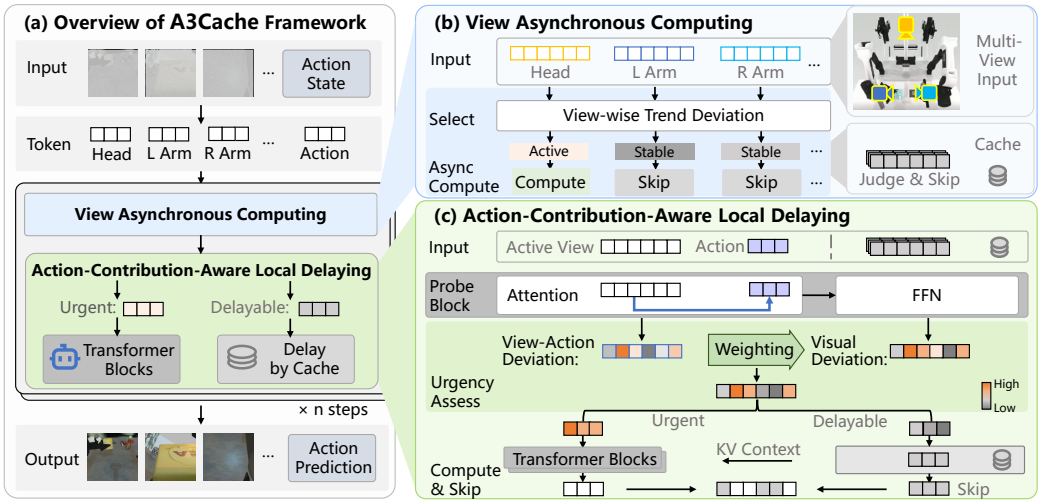


Figure 2: **Overview of A3Cache.** (a) A3Cache preserves the original model architecture and modifies only the training-free inference schedule. (b) It performs **view asynchronous computing** to independently determine the computation demand of each camera view, decoupling synchronized multi-view processing into efficient asynchronous computation across views. (c) Within active views, **World–Action contribution-aware token delaying** further prioritizes urgent local tokens for computation while delaying less urgent ones through cache reuse.

3 METHOD

We first formulate the inference process of multi-view world action models and then present **A3Cache** as a training-free acceleration framework for this model class. Specifically, Section 3.1 introduces the iterative multi-view World-Action inference process, Section 3.2 presents view asynchronous computing, and Section 3.3 presents World-Action contribution-aware token delaying.

3.1 FORMULATION AND MOTIVATION

We consider a representative class of multi-view embodied world action models that take synchronized camera observations and a language instruction as input, and jointly predict future visual states and robot actions through iterative diffusion or flow-based inference.

Closed-loop embodied inference. At each control round t, the model receives the current multi-view observations $\mathbf{O}_t = \{I_t^v\}_{v=1}^V$ from V cameras together with a task instruction $\mathbf{c}$. The world action model $\mathcal{F}_\theta$ predicts both future multi-view observations and an action chunk:

$$(\hat{\mathbf{O}}_{t:t+H_v}, \hat{\mathbf{A}}_{t:t+H_a}) = \mathcal{F}_\theta(\mathbf{O}_t, \mathbf{c}), \tag{1}$$

where $\hat{\mathbf{O}}_{t:t+H_v}$ denotes the predicted visual future and $\hat{\mathbf{A}}_{t:t+H_a} = \{\hat{\mathbf{a}}_t, \ldots, \hat{\mathbf{a}}_{t+H_a}\}$ denotes the predicted action sequence. The predicted actions are then executed in the environment, after which the updated observations are fed back to the model for the next control round. This process is repeated until the task is completed or the episode terminates.

Multi-view World-Action generation. For each inference round, the observations from different cameras are encoded into view-specific tokens $\mathbf{Z}^v \in \mathbb{R}^{L_v^{(v)} \times d}$. Action variables are represented by action tokens $\mathbf{A} \in \mathbb{R}^{L_a \times d}$, while the language instruction provides conditioning tokens $\mathbf{C}$. At denoising step s, these representations are jointly processed by a stack of B Transformer blocks:

$$\mathbf{X}_s^0 = [\mathbf{Z}_s^1; \ldots; \mathbf{Z}_s^V; \mathbf{A}_s; \mathbf{C}], \qquad \mathbf{X}_s^l = \mathcal{B}_l(\mathbf{X}_s^{l-1}, s), \quad l = 1, \ldots, B. \tag{2}$$

Joint attention enables information exchange among different visual views and action representations, while the following feed-forward layers update their features. The same Transformer blocks

are repeatedly evaluated over multiple denoising steps $s = S, \ldots, 1$, after which the visual and action representations are decoded into future observations and executable actions.

Multi-view processing substantially increases the visual computation and cross-view interaction cost of this iterative inference procedure. At the same time, the evolution of different views is often highly uneven, and only a subset of local visual features needs timely current-step updates. These properties motivate the two-stage asynchronous strategy proposed below.

3.2 View Asynchronous Computing

Multi-view observations are synchronized when captured, but their feature evolution during iterative generation is often uneven. In embodied manipulation, a wrist camera can undergo rapid changes around the manipulated object while the global view remains largely stable, and the active camera may further change as the interaction progresses. Existing training-free caching typically uses a shared cache state, such that noticeable change from one view can trigger redundant computation of the remaining stable views. We therefore decouple cache validation and compute across views, allowing each camera to follow its own update schedule.

View cache validation. We partition the visual representation according to its camera source and apply the cache estimator independently to each view. Let $\boldsymbol{H}_{s,v}$ and $\widetilde{\boldsymbol{H}}_{s,v}$ denote the current and cache-predicted features of view v at denoising step s, respectively. We compute a scalar deviation $\delta^V_{s,v}$ using the cache distance $\mathcal{D}(\cdot,\cdot)$, and maintain an accumulated deviation $a_{s,v}$:

$$\delta^V_{s,v} = \mathcal{D}(\boldsymbol{H}_{s,v}, \widetilde{\boldsymbol{H}}_{s,v}), \qquad a_{s,v} \leftarrow a_{s,v} + \delta^V_{s,v}, \qquad m_{s,v} = \mathbf{1}[a_{s,v} < \tau]. \tag{3}$$

Here, $\delta^V_{s,v}$ measures the current deviation of view v, $a_{s,v}$ accumulates its deviation since the last compute, τ is the threshold, and $m_{s,v}$ indicates whether the cache of the view remains valid.

Asynchronous view computing. At each denoising step, views with $m_{s,v} = 0$ are treated as active views and are computed, while cache-valid views remain delayed and reuse their predicted features. When an active view is computed, only its own accumulated deviation is reset, leaving the cache states of the other views unchanged. The computed and delayed features are then restored to their original positions in the multi-view token sequence, so the remaining inference procedure preserves the original model structure. In this way, different views naturally follow different compute schedules, removing unnecessary synchronization across heterogeneous camera dynamics.

3.3 World-Action Contribution-Aware Token Delay

A view cache miss indicates that a camera view has accumulated sufficient deviation to require current-step computation, but it does not imply that every visual token in that view must be computed immediately. Embodied interactions are spatially localized, and different local changes can affect current-step action generation with different urgency. Therefore, once an active view is identified, we further distinguish the local tokens within that view according to their current-step compute urgency, allowing less urgent updates to be delayed.

Token-level visual deviation. For visual token j in active view v, we first measure the token-level cache deviation $d^V_{s,v,j}$ using the same cache estimator used for view validation. We then accumulate this token-level deviation across delayed steps:

$$e^V_{s,v,j} \leftarrow e^V_{s,v,j} + d^V_{s,v,j}, \tag{4}$$

where $e^V_{s,v,j}$ records the accumulated visual deviation of token j since its last computation. The accumulator is reset only when the token is computed, so repeatedly delayed changes continue to increase their compute priority instead of remaining delayed indefinitely.

Visual-to-action contribution deviation. Visual deviation alone tells us how much a token has changed, but not whether its role in current-step action generation is also changing. We therefore use the first joint-attention layer in an active view to measure the contribution of each visual token to the

Table 1: Performance comparison on Motus Bi et al. (2026). We report task success rate (SR), average SR change (ΔSR), average inference time, TFlops, and acceleration ratio (Acc. Ratio). The best results are represented in bold, and the second-best results are indicated with an underline.

Method	Success Rate (%) ↑			ΔSR ↑	Avg. Time (s) ↓	TFlops ↓	Acc. Ratio↑
	Clean	Rand.	Avg.				
Original (Aligned)	87.56	76.24	81.90	0.00 (100.0%)	1.31	3.00	1.00×
D2Cache [CVPR'26]	84.42	<u>73.54</u>	78.98	-2.92 (96.4%)	0.82	1.20	2.50×
SeaCache [CVPR'26]	84.20	72.92	78.56	-3.34 (95.9%)	0.89	1.27	2.40×
Spectrum [CVPR'26]	83.34	72.78	78.06	-3.84 (95.3%)	0.97	1.50	2.00×
HTC-WorldCache [ICML'26]	80.28	69.30	74.79	-7.11 (91.3%)	0.83	1.87	1.60×
CA-WorldCache [ECCV'26]	<u>84.88</u>	73.36	<u>79.32</u>	<u>-2.59 (96.8%)</u>	<u>0.59</u>	<u>1.10</u>	<u>2.73×</u>
C3ache [arXiv'26]	83.16	71.58	77.37	-4.53 (94.5%)	1.08	1.82	1.65×
Ours	**85.94**	**74.76**	**80.35**	**-1.55 (98.1%)**	**0.38**	**0.71**	**4.23 ×**

action tokens. Let $\boldsymbol{c}^A_{s,v,j}$ denote the attention-output contribution from visual token j to the action-token updates, aggregated over action queries and attention heads. We compare this contribution with its trajectory-based prediction $\widetilde{\boldsymbol{c}}^A_{s,v,j}$:

$$d^A_{s,v,j} = \frac{\left\|\boldsymbol{c}^A_{s,v,j} - \widetilde{\boldsymbol{c}}^A_{s,v,j}\right\|_2}{\left\|\boldsymbol{c}^A_{s,v,j}\right\|_2 + \epsilon}, \qquad \boldsymbol{c}^A_{s,v,j} = \mathrm{Softmax}_j\left(\frac{\boldsymbol{q}^A_{s,i}\boldsymbol{k}^\top_{s,v,j}}{\sqrt{d}}\right)\boldsymbol{v}_{s,v,j}. \tag{5}$$

Here, $d^A_{s,v,j}$ measures the change in the token's contribution to action generation at the current step, providing a complementary signal to token-level visual deviation.

World-Action contribution-aware token delay. We combine the accumulated visual deviation with the visual-to-action contribution deviation to determine which token updates are urgent. After normalizing the two quantities, denoted by $\bar{e}^V_{s,v,j}$ and $\bar{d}^A_{s,v,j}$, we compute the token urgency:

$$u_{s,v,j} = \bar{e}^V_{s,v,j}\left(1 + \bar{d}^A_{s,v,j}\right), \qquad \mathcal{T}_{s,v} = \mathrm{TopK}\Big(\{u_{s,v,j}\}_{j=1}^{L_v^{(v)}}, \lceil \rho L_v^{(v)} \rceil\Big), \tag{6}$$

where ρ is the token compute ratio, and $\mathcal{T}_{s,v}$ contains the tokens selected for current-step computation. Visual deviation remains the basis of token staleness, while action-related changes prioritize which stale features are more urgent for current-step computation.

Delayed local computation. For the subsequent Transformer blocks, only tokens in $\mathcal{T}_{s,v}$ are computed for each active view, while the remaining visual tokens are delayed and reuse their cached features. The accumulator $e^V_{s,v,j}$ is reset only when the corresponding token is computed. The active and delayed features are merged at their original sequence positions after each update, preserving the original input organization and cross-token context of the pretrained model. This token-level delay further reduces the cost of already active views while retaining timely updates for the local visual features that matter most for action generation.

4 Experiments

Firstly, we introduce the experimental settings and implementation details in Section 4.1. Section 4.2 compares A3Cache with representative training-free acceleration methods on multi-view embodied manipulation tasks. Section 4.3 analyzes the contributions of our components.

4.1 Experimental Settings

Benchmark and evaluation setup. We evaluate A3Cache on Motus Bi et al. (2026), a multi-view embodied world action model, using RoboTwin 2.0 Chen et al. (2025) under the main simulation setting of the original model. The benchmark contains 50 robotic manipulation tasks with both Clean and Randomized settings, where the latter introduces additional visual and scene variations. We use the publicly released Motus checkpoint and keep its original multi-view World–Action inference

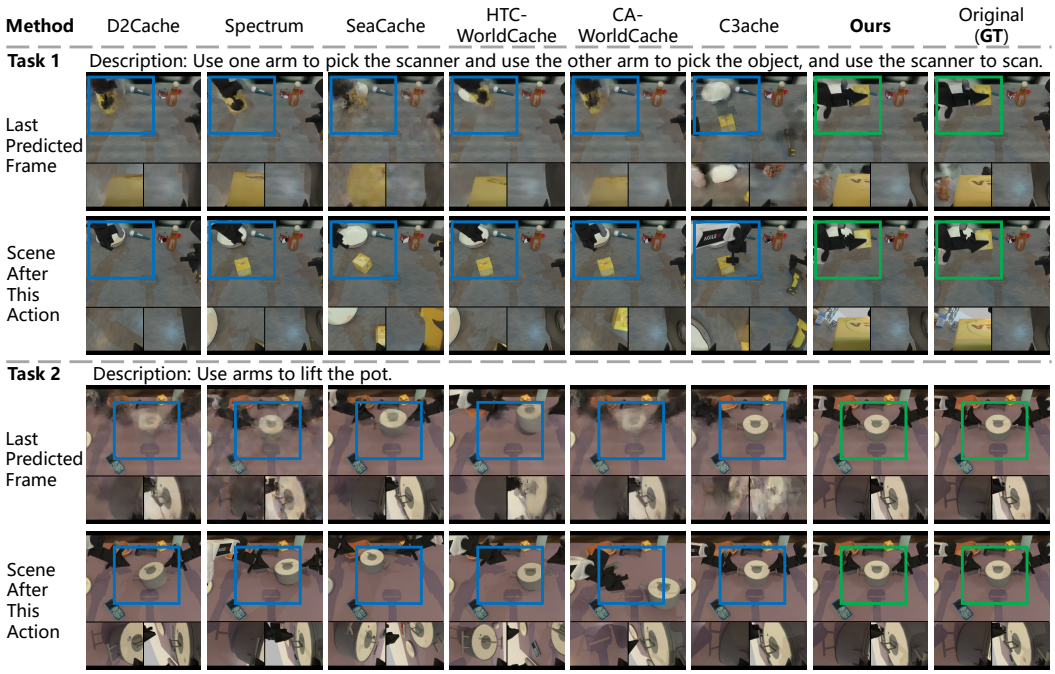


Figure 3: **Qualitative comparison on RoboTwin 2.0 Chen et al. (2025).** We show representative execution frames from the original model, competing acceleration methods, and A3Cache. Our method better preserves the execution process and final task outcome under aggressive acceleration.

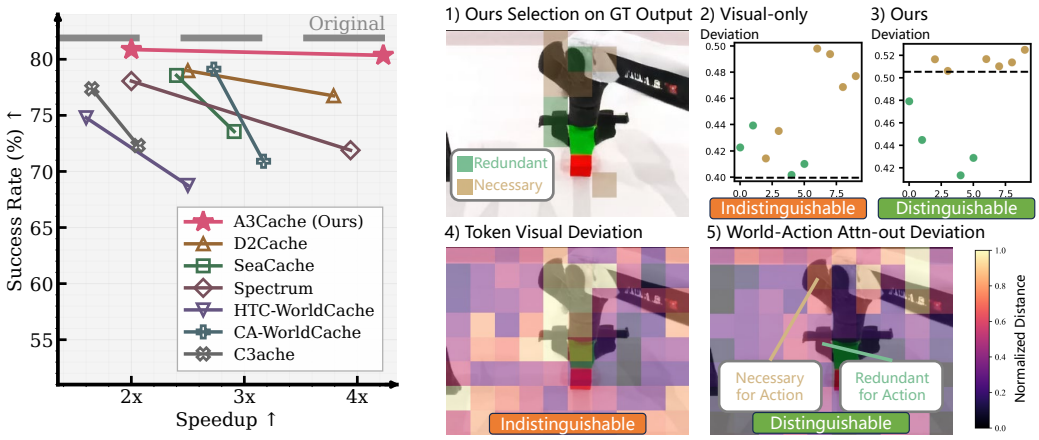
pipeline unchanged, applying acceleration only to the iterative Diffusion Transformer computation. Following the original RoboTwin protocol, we report the task success rate (**SR**) under both settings and their average (**Avg.**). We additionally report the success-rate change relative to unaccelerated Motus (Δ**SR**), inference time, total TFlops, and acceleration ratio.

Comparison Methods. We compare A3Cache with six representative training-free acceleration methods covering different levels of model specialization. D2Cache Liu et al. (2026), Spectrum Han et al. (2026), and SeaCache Chung et al. (2026) are designed for general DiT-based video generation models, while HTC-WorldCache Feng et al. (2026) and CA-WorldCache Nawaz et al. (2026) target iterative video world models. We further include C^{3}ache Zhao et al. (2026), which is specifically developed for repeated inference in world action models. For the main comparison, we configure each method according to its original design and select a practical operating point that provides a competitive trade-off between acceleration and task performance. For the acceleration comparison, we further vary the corresponding cache thresholds to obtain different acceleration ratio levels.

Implementation Details We apply A3Cache directly at inference time without retraining or fine-tuning. For view-level asynchronous computation, we adopt the SeaCache-style spectral deviation Chung et al. (2026) to estimate the cache validity of each camera view and adjust the view-deviation threshold to control the overall acceleration ratio. Within each active view, World–Action contribution-aware token delaying retains the top 75% visual tokens according to the combined visual and action-contribution deviation, while the remaining tokens reuse cached computation. Following common practice in training-free diffusion acceleration methods Chung et al. (2026); Liu et al. (2026), we preserve full computation at the final denoising step and apply cache skipping and local delaying only to intermediate steps to maintain final-output quality. All experiments are conducted on an NVIDIA RTX 4090 GPU with 48 GB memory.

4.2 Main Quantitative Results

Quantitative comparison. Table 1 compares A3Cache with representative training-free acceleration methods on Motus over RoboTwin 2.0 Chen et al. (2025). A3Cache achieves the strongest



(a) Success rate under multiple Acc. Ratio (b) Effect of World–Action contribution-aware token delaying.

Figure 4: (a) Quality–efficiency comparison of different acceleration methods under multiple acceleration ratios. A3Cache remains consistently above the methods and preserves most of the original performance at high ratio. (b) Effect of World–Action contribution-aware token delaying. 1) Token examples of A3Cache on the target output. 2, 3) Compared with the visual-only criterion, our method better separates tokens that are necessary for the current action from those that can be delayed. 4) Token visual deviation captures visual changes but cannot reliably distinguish their relevance to action prediction. 5) World–Action attention-output deviation provides contribution to action complementarily for retaining action-relevant tokens while delaying redundant ones.

Table 2: **Ablation of the main components of A3Cache.** We remove each proposed component from the full method and compare the variants under approximately matched acceleration ratios.

Method	Success Rate (%) ↑			ΔSR ↑	Avg. Time (s) ↓	TFlops ↓	Acc. Ratio ↑
	Clean	Rand.	Avg.				
Original (Aligned)	87.56	76.24	81.90	0.00 (100.0%)	1.31	3.00	1.00×
w/o view asynchronous compute	82.92	71.20	77.06	-4.84 (94.1%)	0.41	0.75	4.00×
w/o contribution-aware token delay	84.26	73.38	78.82	-3.08 (96.2%)	0.45	0.77	3.90×
Full A3Cache	**85.94**	**74.76**	**80.35**	**-1.55 (98.1%)**	**0.38**	**0.71**	**4.23 ×**

efficiency–performance trade-off, delivering the highest acceleration ratio while preserving over 98% of the original task success rate. Figure 4 (a) further shows that A3Cache remains consistently competitive across different acceleration levels, with a smaller performance drop under aggressive acceleration. This benefit comes from its hierarchical allocation of computation: stable views are asynchronously reused, while computation within active views is concentrated on action-relevant local regions, reducing redundancy without excessively discarding important visual updates.

Qualitative Comparison Figure 3 presents qualitative comparisons of manipulation execution results. We visualize representative execution frames from the same task under the original Motus model, representative accelerated baselines, and A3Cache. Compared with existing methods, A3Cache better preserves the original execution trajectory and interaction outcome, especially in manipulation stages requiring precise object contact or viewpoint-sensitive updates. These qualitative results are consistent with the quantitative evaluation and show that our selective compute strategy maintains more reliable closed-loop behavior under high acceleration.

4.3 Additional Analysis

We further analyze A3Cache through component ablations studies, including the two main components, and the two token-level signals that affecting computation allocation.

Table 3: **Ablation of token delaying selection signals.** We compare token visual deviation, World–Action contribution deviation, and their combination under a comparable token-computation budget.

Method	Success Rate (%) ↑			ΔSR ↑	Avg. Time (s) ↓	TFlops ↓	Acc. Ratio ↑
	Clean	Rand.	Avg.				
Original (Aligned)	87.56	76.24	81.90	0.00 (100.0%)	1.31	3.00	1.00×
w/o cache deviation	81.38	71.46	76.42	-5.49 (93.3%)	0.38	0.71	4.23×
w/o action contribution deviation	85.06	73.98	79.52	-2.38 (97.1%)	0.37	0.71	4.23×
w/ both signals (Ours)	**85.94**	**74.76**	**80.35**	**-1.55 (98.1%)**	0.38	0.71	4.23 ×

Ablation of the Main Components. Table 2 evaluates A3Cache by removing each proposed component from the full method. To ensure a fair comparison, the variants are adjusted to operate at comparable acceleration ratios. Removing either View-Asynchronous Computing or World–Action Contribution-Aware Token Delaying consistently reduces task success, while the full model achieves the best performance at a similar efficiency level. This confirms that the two components address different sources of redundancy. View-Asynchronous Computing avoids unnecessary synchronized updates when only a subset of camera views changes significantly, and token delaying further removes redundant local computation within active views. Their combination preserves the computation required by dynamically changing views while reducing unnecessary updates.

Ablation of the Token Selection Signals. Table 3 analyses the signals used to estimate token-level computation urgency. Using either token visual deviation or World–Action contribution deviation alone leads to lower success, while their combination achieves the strongest performance. Visual deviation measures how much a token has changed relative to its cached state, but cannot determine whether that change is important for the current action prediction. In contrast, contribution deviation captures changes in the interaction between visual tokens and the action branch, but does not fully reflect visual evolution. Combining the two signals therefore favors tokens that are both visually changing and action-relevant, leading to a more reliable allocation of the computation budget.

Effect of World–Action Contribution-Aware Token Delaying. Figure 4 (b) provides a analysis of the token-selection behavior at the same inference step. We visualize the token visual deviation and World–Action attention-output deviation, together with the resulting separation between action-necessary and action-redundant regions. Visual deviation highlights regions with strong local changes, but regions with similar visual variation can have different effects on the current action prediction. The World–Action contribution signal provides an additional cue for distinguishing these cases: regions associated with action-induced changes exhibit stronger contribution variation, while visually changing but action-redundant regions remain less prominent. As a result, the combined criterion assigns high computation urgency to action-relevant tokens and delays redundant updates.

5 Conclusion

In this work, we presented **A3Cache**: Action-Aware Asynchronous Caching, a training-free acceleration framework for multi-view embodied world action models that reformulates synchronized multi-view caching as asynchronous computation by adapting computation across views and local tokens according to their update demands, thereby enabling higher-ratio acceleration while preserving embodied task performance. Our results demonstrate that explicitly accounting for asynchronous view dynamics and action-related local changes enables substantially more aggressive cache reuse without proportional performance degradation, by independently scheduling computation across camera views and further prioritizing action-relevant tokens within active views. Across multi-view robotic manipulation tasks, A3Cache consistently improves the efficiency–performance trade-off over existing training-free acceleration methods, preserving over 98% of the original performance under more than $4\times$ acceleration. These results suggest that effective caching for embodied world action models should consider not only feature similarity across denoising steps, but also the heterogeneous dynamics across views and their interaction with action generation.

AI USE STATEMENT

In this work, we used generative AI tools to aid and polish the writing of the manuscript and to support retrieval and discovery, including identifying potentially relevant related work. All AI-assisted writing was reviewed and revised by the authors, and retrieved references and related-work information were manually verified against the original sources. We take full responsibility for the final content of this work, including all text, citations, and artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We provide the implementation details of our proposed method in the experimental section of the main paper. And we include a source code package of our method in the supplementary material.

REFERENCES

Niket Agarwal, Arslan Ali, Maciej Bala, Yogesh Balaji, Erik Barker, Tiffany Cai, Prithvijit Chattopadhyay, Yongxin Chen, Yin Cui, Yifan Ding, et al. Cosmos world foundation model platform for physical ai. *arXiv preprint arXiv:2501.03575*, 2025.

Hongzhe Bi, Hengkai Tan, Shenghao Xie, Zeyuan Wang, Shuhe Huang, Haitian Liu, Ruowen Zhao, Yao Feng, Chendong Xiang, Yinze Rong, et al. Motus: A unified latent action world model. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 35101–35113, 2026.

Jiazi Bu, Pengyang Ling, Yujie Zhou, Yibin Wang, Yuhang Zang, Dahua Lin, and Jiaqi Wang. Dicache: Let diffusion model determine its own cache. *Proceedings of International Conference on Learning Representations*, 2026.

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

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.

Jiwoo Chung, Sangeek Hyun, MinKyu Lee, Byeongju Han, Geonho Cha, Dongyoon Wee, Youngjun Hong, and Jae-Pil Heo. Seacache: Spectral-evolution-aware cache for accelerating diffusion models. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 14283–14294, 2026.

Hanshuai Cui, Zhiqing Tang, Zhifei Xu, Zhi Yao, Wenyi Zeng, and Weijia Jia. Bwcache: Accelerating video diffusion transformers through block-wise caching. In *Proceedings of International Conference on Learning Representations*, volume 2026, pp. 105765–105791, 2026.

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

Yanjiang Guo, Lucy Shi, Jianyu Chen, and Chelsea Finn. Ctrl-world: A controllable generative world model for robot manipulation. In *Proceedings of International Conference on Learning Representations*, volume 2026, pp. 6121–6138, 2026.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *Proceedings of International Conference on Learning Representations*, 2020.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023.

Jiaqi Han, Juntong Shi, Puheng Li, Haotian Ye, Qiushan Guo, and Stefano Ermon. Adaptive spectral feature forecasting for diffusion sampling acceleration. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 43320–43330, 2026.

Anthony Hu, Lloyd Russell, Hudson Yeo, Zak Murez, George Fedoseev, Alex Kendall, Jamie Shotton, and Gianluca Corrado. Gaia-1: A generative world model for autonomous driving. *arXiv preprint arXiv:2309.17080*, 2023.

Siyuan Huang, Liliang Chen, Pengfei Zhou, Shengcong Chen, Yue Liao, Zhengkai Jiang, Yue Hu, Peng Gao, Hongsheng Li, Maoqing Yao, et al. Enerverse: Envisioning embodied future space for robotics manipulation. In *Proceedings of Advances in Neural Information Processing Systems*, volume 38, pp. 37693–37720, 2026.

Yue Liao, Pengfei Zhou, Siyuan Huang, Donglin Yang, Shengcong Chen, Yuxin Jiang, Yue Hu, Si Liu, Jianlan Luo, Liliang Chen, et al. Genie envisioner: A unified world foundation platform for robotic manipulation. In *Proceedings of International Conference on Learning Representations*, pp. 88446–88463, 2026.

Enhuai Liu, Yunke Wang, Changming Sun, and Chang Xu. D2cache: Second-order delta caching for higher video diffusion acceleration. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, pp. 43589–43599, 2026.

Feng Liu, Shiwei Zhang, Xiaofeng Wang, Yujie Wei, Haonan Qiu, Yuzhong Zhao, Yingya Zhang, Qixiang Ye, and Fang Wan. Timestep embedding tells: It's time to cache for video diffusion model. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*, 2025.

Zhengyao Lv, Chenyang Si, Junhao Song, Zhenyu Yang, Yu Qiao, Ziwei Liu, and Kwan-Yee K. Wong. Fastercache: Training-free video diffusion model acceleration with high quality. *arXiv preprint arXiv:2410.19355*, 2024.

Umair Nawaz, Ahmed Heakl, Ufaq Khan, Abdelrahman Shaker, Salman Khan, and Fahad Shahbaz Khan. Worldcache: Content-aware caching for accelerated video world models. In *Proceedings of European Conference on Computer Vision*, 2026.

Younggyo Seo, Junsu Kim, Stephen James, Kimin Lee, Jinwoo Shin, and Pieter Abbeel. Multi-view masked world models for visual robotic manipulation. In *Proceedings of International Conference on Machine Learning*, pp. 30613–30632, 2023.

Team Wan, Ang Wang, Baole Ai, Bin Wen, Chaojie Mao, Chen-Wei Xie, Di Chen, Feiwu Yu, Haiming Zhao, Jianxiao Yang, et al. Wan: Open and advanced large-scale video generative models. *arXiv preprint arXiv:2503.20314*, 2025.

Xiaofeng Wang, Zheng Zhu, Guan Huang, Xinze Chen, Jiagang Zhu, and Jiwen Lu. Drivedreamer: Towards real-world-driven world models for autonomous driving. In *Proceedings of European Conference on Computer Vision*, pp. 55–72, 2024.

Sherry Yang, Yilun Du, Seyed Ghasemipour, Jonathan Tompson, Leslie Kaelbling, Dale Schuurmans, and Pieter Abbeel. Learning interactive real-world simulators. In *Proceedings of International Conference on Learning Representations*, 2024.

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.

Guosheng Zhao, Xiaofeng Wang, Zheng Zhu, Xinze Chen, Guan Huang, Xiaoyi Bao, and Xingang Wang. Drivedreamer-2: Llm-enhanced world models for diverse driving video generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025.

Weisen Zhao, Lam Nguyen, Zhicong Lu, and Yuzhang Shang. C3 ache: Accelerating world action models with cross inference chunk cache. *arXiv preprint arXiv:2606.08962*, 2026.

Chang Zou, Xuyang Liu, Ting Liu, Siteng Huang, and Linfeng Zhang. Accelerating diffusion transformers with token-wise feature caching. In *Proceedings of International Conference on Learning Representations*, pp. 75467–75488, 2025.

APPENDIX

We provide additional ablation studies and qualitative analyses to complement the experiments in the main paper. The appendix is organized as follows.

- Section A studies final-step computation choices of our A3Cache.
- Section B analyzes the effect of different acceleration methods on world prediction and the resulting environment states after action execution.
- Section C further compares the generated action trajectories with unaccelerated Motus, providing a fine-grained evaluation of action preservation under acceleration.

A ADDITIONAL ABLATION STUDIES

We further analyze two implementation choices of A3Cache that affect the balance between acceleration and task performance, including the treatment of the final denoising step.

Effect of final-step computation. A3Cache disables acceleration at the final denoising step to preserve the quality of the final World-Action representation. To evaluate this design, we compare the default setting with a variant that continues cache-based acceleration at the final step, while adjusting the remaining cache parameters to obtain a similar overall acceleration ratio. As shown in Table 4, accelerating the final step consistently reduces task performance despite the comparable efficiency. This suggests that approximation errors accumulated during intermediate denoising steps become more critical near the end of the trajectory, where the final representation directly determines the predicted world state and action. Preserving full computation at the last step therefore provides an effective quality safeguard with only limited impact on the overall acceleration ratio.

Table 4: **Ablation of final-step computation.** We compare the default setting that preserves full computation at the final denoising step with a variant that continues acceleration while maintaining a similar overall acceleration ratio.

Method	**Success Rate (%) ↑**			**ΔSR ↑**	**Avg. Time (s) ↓**	**TFlops ↓**	**Acc. Ratio ↑**
	Clean	Rand.	Avg.				
Original (Aligned)	87.56	76.24	81.90	0.00 (100.0%)	1.31	3.00	1.00×
w/o Final-Step Protection	82.78	71.20	76.99	-4.91(94.0%)	0.49	0.78	3.85 ×
Ours	**85.94**	**74.76**	**80.35**	**-1.55 (98.1%)**	**0.38**	**0.71**	**4.23** ×

B WORLD PREDICTION AND EXECUTION ANALYSIS

We further provide a case study on one RoboTwin 2.0 Chen et al. (2025) manipulation episode to examine how different acceleration methods affect both world prediction and action generation under comparable computation budgets. Figure 5 compares the predicted future frames from a key Motus inference call with the environment state observed after executing the corresponding predicted actions. Several comparison methods exhibit visible degradation in the predicted interaction state, which is followed by inaccurate execution and ultimately leads to task failure in this episode. In contrast, A3Cache preserves the key action-related visual changes more faithfully, producing a predicted future state that is closer to the actual successful execution. This result is consistent with our design of preserving computation for dynamically changing views and action-relevant local regions while delaying redundant visual updates.

C ACTION TRAJECTORY ANALYSIS

We further compare the action trajectories generated in the same episode. Figure 6 further compares the trajectories of six representative robot joints throughout the same episode. The accelerated

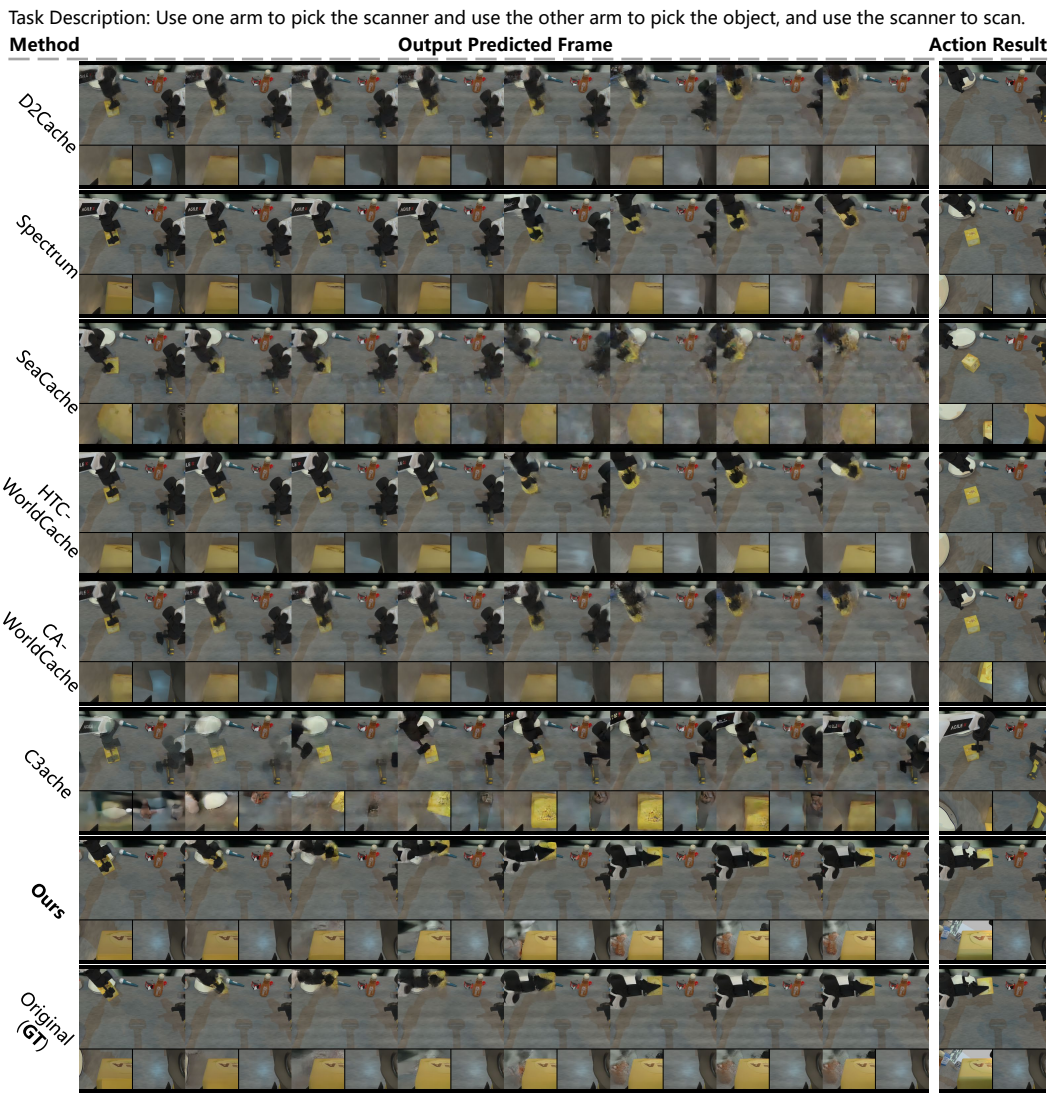


Figure 5: **Qualitative comparison of world prediction and post-action execution under comparable computation budgets.** We show the predicted future frames from a key Motus inference call and the corresponding environment states after executing the predicted actions. Several acceleration baselines exhibit degraded world predictions that lead to inaccurate execution, whereas A3Cache better preserves the action-related scene changes and produces an execution outcome closer to the unaccelerated Motus reference.

methods introduce different degrees of deviation from the unaccelerated Motus trajectory, whereas A3Cache remains substantially closer to the reference across the key action stages. We quantify this difference in Table 5 using action MAE over three common replanning steps and 48 aligned action points. A3Cache achieves the lowest overall MAE among all accelerated methods (0.0131), substantially below the next closest baselines. Similar improvements are observed for both left- and right-arm joints as well as the grippers. These results indicate that the proposed asynchronous view computation and action-aware local delaying preserve not only visual prediction quality but also the action dynamics required for successful execution.

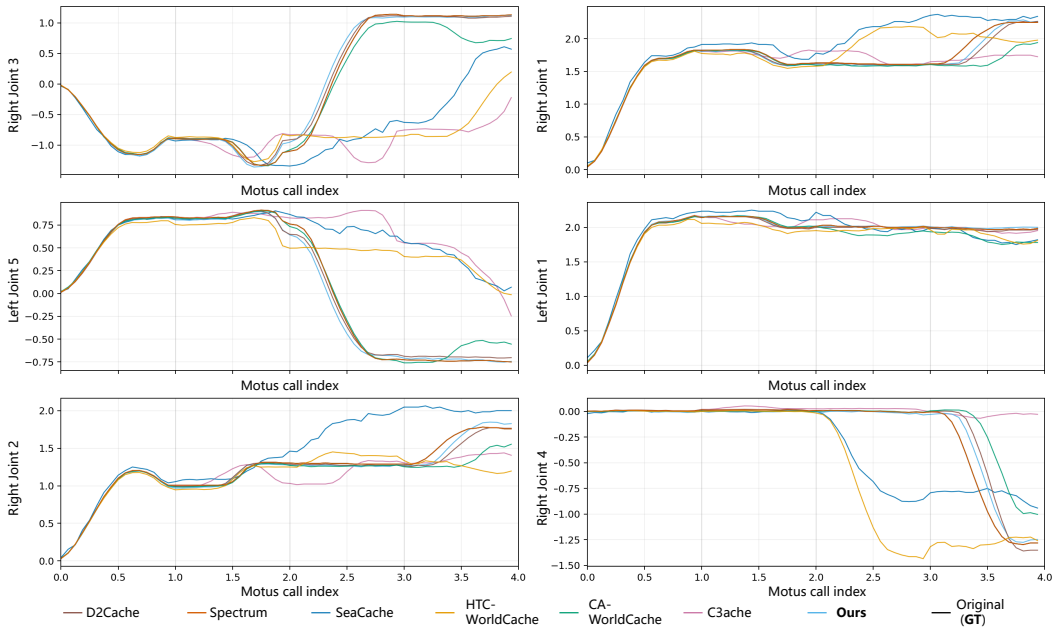


Figure 6: **Action trajectory comparison on the same RoboTwin 2.0 Chen et al. (2025) manipulation episode.** We plot six representative joint trajectories produced by Motus and different acceleration methods over the aligned execution sequence. A3Cache follows the unaccelerated Motus trajectory more closely than the comparison methods, particularly around the key action transitions.

Table 5: **Action deviation from unaccelerated Motus on the qualitative episode.** All methods are aligned over 3 common replanning steps and 48 action points. We report MAE with respect to the action trajectory generated by unaccelerated Motus.

Method	Overall MAE ↓	Left Joints ↓	Left Gripper ↓	Right Joints ↓	Right Gripper ↓
D2Cache	0.0333	0.0440	0.0136	0.0284	0.0193
SeaCache	0.0430	0.0425	0.0150	0.0523	0.0182
Spectrum	<u>0.0307</u>	<u>0.0350</u>	0.0224	<u>0.0299</u>	0.0177
HTC-WorldCache	0.0315	0.0377	<u>0.0097</u>	0.0318	<u>0.0145</u>
CA-WorldCache	0.0563	0.0448	0.0929	0.0549	0.0976
C3ache	0.0853	0.0743	0.1642	0.0686	0.1725
A3Cache (Ours)	**0.0131**	**0.0112**	**0.0100**	**0.0165**	**0.0064**