SYSTEMATIC MULTI-LEVEL ACCELERATION FOR WORLD ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

World Action Models (WAMs) are emerging as a promising alternative to Vision-Language-Action (VLA) models. Existing VLA acceleration methods primarily reduce VLM computation and are not directly applicable to diffusion-based WAMs. Moreover, WAM acceleration methods typically target redundancy along only a single computational axis, overlooking substantial redundancy across the full pipeline, resulting in either limited speedups or drops in success rate. To address these limitations, we propose **MARS**, a training-free multi-level acceleration framework that systematically exploits distinct sources of redundancy at three computational levels: (1) across Transformer blocks, we selectively skip redundant attention components; (2) across diffusion denoising steps, we perform action-aware routing to adaptively skip diffusion steps; (3) across consecutive action chunks, we reuse future visual tokens, thereby skipping the VAE encoding. These mechanisms are synergistically integrated in a plug-and-play manner. Extensive experiments across multiple benchmarks and base models demonstrate that our method achieves a superior efficiency-performance trade-off, delivering up to a $4.00\times$/$3.94\times$ speedup in LIBERO/LIBERO-PRO, $2.10\times$ in RoboCasa, and consistent speedups on real-world tasks, while maintaining comparable success rates.

1 INTRODUCTION

World Action Models (WAMs) have recently emerged as an important foundation-model paradigm for embodied intelligence (Guo et al., 2024; Zhu et al., 2025; Shen et al., 2025; Kim et al., 2026; Ye et al., 2026b). Unlike Vision-Language-Action (VLA) models that directly predict actions (Kim et al., 2024; Intelligence et al., 2025), WAMs jointly generate future world states and robot actions within a unified generative process, allowing them to leverage physical priors and interaction dynamics acquired from large-scale video pretraining (Wu et al., 2024; Agarwal et al., 2025). This capability, however, comes with substantial computational cost. Processing high-dimensional visual states through video encoding and multi-step generation makes each policy query considerably more expensive than those of conventional action models. The resulting inference latency often forces robots to use longer action chunks and lower replanning frequencies, limiting the practical use of WAMs for high-frequency closed-loop control.

A straightforward solution is to directly apply existing training-free diffusion or policy acceleration techniques to WAMs (Ma et al., 2024; Bu et al., 2026; Zhao et al., 2026; Zeng et al., 2026). However, existing methods typically target redundancy along only a single computational axis, inevitably facing an efficiency-performance trade-off ceiling. On the one hand, conservative diffusion caching preserves control fidelity but offers only limited latency reductions. On the other hand, aggressive feature reuse across consecutive action chunks substantially cuts computation, yet introduces cumulative approximation errors that severely degrade long-horizon task success. Crucially, the computational bottleneck in WAM inference is fundamentally dynamic: once the dominant DiT backbone computation is substantially compressed, the upstream visual encoding immediately emerges as the new primary latency bottleneck. Consequently, isolated, single-stage optimizations are insufficient for sustained end-to-end acceleration, calling for a holistic, multi-level framework.

This motivates us to ask two questions: where does computational redundancy arise, and which redundant computations can be safely removed? We systematically profile the complete WAM pipeline illustrated in Figure 1(a) and obtain three key analyses, as shown in Figure 2. First, within a

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.

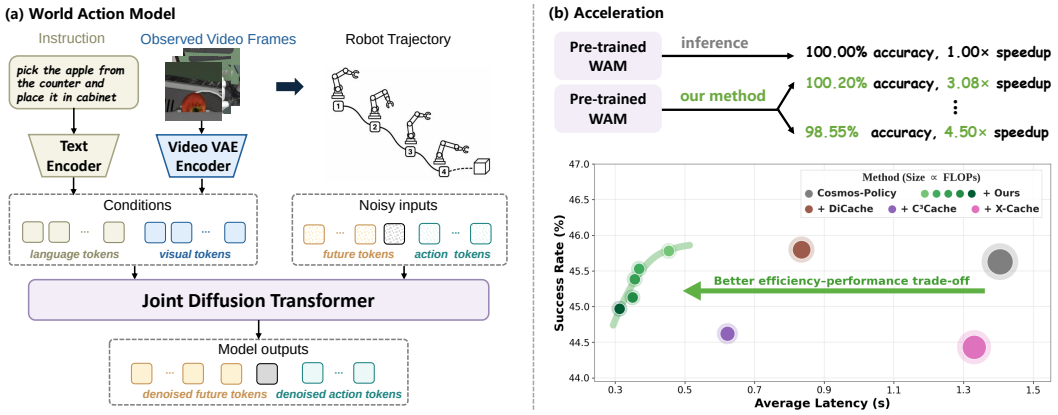

Figure 1: **WAM inference and acceleration overview.** (a) The overall pipeline of a World Action Model. (b) Efficiency-performance comparison in LIBERO-PRO (Zhou et al., 2025), where our method substantially reduces latency and FLOPs with a better efficiency–performance trade-off.

single Transformer, the importance of different attention components is highly non-uniform: many cross-attention branches and self-attention heads have only a small effect on the output, while a smaller subset remains critical for action prediction. Second, throughout diffusion denoising, the sampler input changes substantially, whereas the predicted clean states and DiT residuals remain comparatively stable, indicating that much of the repeated backbone computation can be approximated; meanwhile, different manipulation stages exhibit different sensitivities to such approximation. Third, across consecutive policy queries, the future visual states predicted at the previous query can be reused at the next query, but prediction mismatches make unconditional reuse unreliable.

Together, these analyses reveal a key property of WAM acceleration: redundancy is abundant but non-uniform, and removing computation at different locations carries different risks to control accuracy. We therefore argue that effective training-free acceleration should not rely on a single computation-reuse rule. Instead, redundant computation should be selectively removed throughout the inference pipeline according to its actual importance to action prediction, while reuse that may introduce temporally accumulated errors should be verified online. In other words, the key to WAM acceleration is not simply to remove more computation, but to remove the right computation at the right locations in a manner matched to control sensitivity.

Based on these observations, we propose MARS, a training-free and plug-and-play framework for accelerating WAM inference. MARS reduces redundant computation at three complementary levels. First, at the Transformer level, we use calibration statistics to skip low-impact cross-attention branches and prune low-importance self-attention heads while preserving components that are important for action prediction. Second, at the diffusion level, we use the provisional action available after the first denoising step to estimate the motion granularity of the current query. For coarse-grained motions that better tolerate approximation, we directly adopt one-step generation; for fine-grained stages such as grasping and placement, we retain the original multi-step solver while reusing stable DiT residuals to reduce backbone computation. Finally, across consecutive policy queries, we treat the WAM's own predicted future visual tokens as a speculative representation of the next observation. We compare predicted and observed visual changes using shallow VAE features and reuse the future tokens when they are sufficiently consistent, thereby skipping the remaining visual encoding computation.

We systematically evaluate MARS on multiple benchmarks. While maintaining task performance close to that of the original policy, MARS achieves up to 4.00× end-to-end speedup in LIBERO (Liu et al., 2023), 3.94× in LIBERO-PRO (Zhou et al., 2025), and 2.10× in RoboCasa (Nasiriany et al., 2024), together with substantial reductions in FLOPs. By adjusting the acceleration strength at these three levels, MARS supports flexible efficiency-performance trade-offs, as shown in Figure 1(b). Its applicable components also provide consistent acceleration gains across different WAM and VLA backbones, suggesting that the identified redundancy is not specific to a particular architecture.

Our contributions are summarized as follows:

- We systematically profile WAM inference and reveal three properties: redundancy spans multiple stages, safe removal is stage-dependent, and acceleration shifts the computational bottleneck.

- We introduce MARS, a training-free, plug-and-play framework that jointly exploits redundancy in Transformer computation, diffusion denoising, and cross-query visual processing.
- Extensive experiments across benchmarks, base models, hardware platforms and real-world tasks demonstrate that MARS achieves a substantially improved efficiency–performance trade-off.

2 Related Work

World Action Models. World Action Models (WAMs), an emerging class of embodied foundation models, explicitly model how the environment evolves under candidate actions. They either jointly model future states and actions (Guo et al., 2024; Zhu et al., 2025; Shen et al., 2025; Kim et al., 2026; Ye et al., 2026a) or predict future states and derive actions from them (Du et al., 2023; Pai et al., 2025; Feng et al., 2025). From a generative modeling perspective, existing WAMs are primarily instantiated using autoregressive (Wu et al., 2024; Zhao et al., 2025a; Cen et al., 2025) or diffusion-based formulations. We focus on diffusion-based WAMs. Unified World Models (Zhu et al., 2025) jointly model videos and actions through modality-aware diffusion. Cosmos Policy (Kim et al., 2026) adapts a pretrained video diffusion backbone (Agarwal et al., 2025) to jointly generate future images, actions, proprioceptive states, and values. DreamZero (Ye et al., 2026b) demonstrates that large video-pretrained WAMs can serve as zero-shot policies with strong transfer to unseen motions and embodiments. These works demonstrate the effectiveness of future prediction for action generation, but their video-generation backbones introduce substantial deployment costs.

Efficient World Action Models. Recently, several works have explored reducing the inference cost of WAMs. Fast-WAM (Yuan et al., 2026) and GigaWorld-Policy (Ye et al., 2026a) retain video co-training but remove explicit future generation at test time. Efficient-WAM (Li et al., 2026a) reduces future-imagination cost through a compact video expert, low-resolution future latents, and asymmetric video-action denoising. Flash-WAM (Akbari et al., 2026) instead adopts modality-aware consistency distillation to compress both video and action denoising to a single step. Despite achieving substantial speedups, these approaches require dedicated training or architectural changes. Among training-free approaches, C^3ache (Zhao et al., 2026) and X-Cache (Zeng et al., 2026) exploit cross-chunk similarity to reuse intermediate residuals, with X-Cache further employing structure- and action-aware fingerprints for block-level cache matching. Dream-Tac (Lou et al., 2026) instead applies a fixed cache schedule to selected denoising steps within each action chunk. These methods primarily exploit redundancy at a single computational level, whereas we jointly address multi-level redundancy in attention computation, diffusion denoising, and observation encoding.

Training-Free Acceleration for Diffusion and VLA Models. Training-free diffusion acceleration commonly reduces redundancy across denoising steps and network components. DeepCache (Ma et al., 2024) reuses intermediate features with predefined schedules, while TeaCache (Liu et al., 2025) and DiCache (Bu et al., 2026) estimate feature changes online. At a finer granularity, ToCa (Zou et al., 2024), PAB (Zhao et al., 2025b), and DiTFastAttn (Yuan et al., 2024) reduce token- or attention-level computation. WorldCache (Nawaz et al., 2026) further introduces motion- and saliency-aware reuse for video world models. However, these methods are primarily optimized for perceptual fidelity rather than control accuracy. For robot policies, BAC (Ji et al., 2025) exploits block-wise redundancy, while DP-Cache (Zhou et al., 2026), V-AEFusion (Zhou et al., 2026), and ActionCache (Oi et al., 2026) accelerate action generation through caching, computation overlap, or warm-starting. VLA-Cache (Xu et al., 2025) and EfficientVLA (Yang et al., 2025) reduce visual computation through temporal reuse or token selection, whereas SP-VLA (Li et al., 2026c), SpecPrune-VLA (Wang et al., 2026), and ADP (Pei et al., 2026) allocate computation according to action dynamics or manipulation stages. Nevertheless, these VLA acceleration methods largely focus on the VLM backbone, leaving the multi-level redundancy in WAM inference underexplored.

3 Method

3.1 Preliminaries & Problem Formulation

World action models jointly model environment dynamics and robot actions within a unified generative framework. As shown in Fig. 1, Cosmos Policy takes multi-view observations $\mathcal{I}_q = \{I_q^c\}_{c=1}^{C}$,

robot proprioception p_q, and an instruction ℓ as input at each control query q, and predicts

$$\left(A_q, \hat{z}_q^{\text{future}}, \hat{p}_q^{\text{future}}, \hat{V}_q\right) = \mathcal{F}_\theta(\mathcal{I}_q, p_q, \ell), \tag{1}$$

where A_q is the action chunk, while the other outputs are visual latents, robot states, and value. Observations are VAE-encoded, and the instruction is T5-XXL-encoded (Raffel et al., 2020) and injected via cross-attention. After executing A_q, the policy is re-queried with updated observations.

3.2 Profiling and Diagnosing WAM Redundancy

We systematically analyze the pipeline and identify three distinct sources of redundancy, as shown in Figure 2. All diagnostics are performed in the LIBERO (Liu et al., 2023) simulation environment.

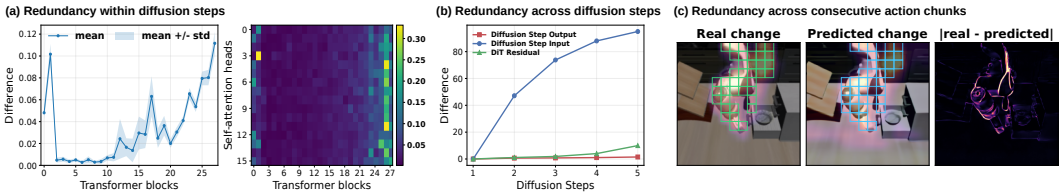

Figure 2: **Redundancy across multiple levels of WAM inference. (a)** Many cross-attention blocks (the left plot) and self-attention heads (the right plot) induce only small relative output changes, indicating Transformer-level redundancy. **(b)** The diffusion step input changes substantially across denoising steps, whereas the diffusion step output and DiT residual remain much more stable, revealing strong diffusion-level redundancy. **(c)** The alignment between change regions in real VAE shallow features and predicted latents reveals action-level redundancy across chunks.

Analysis 1: Transformer-Level Redundancy: Non-Uniform Attention. We observe substantial variation in the importance of different cross-attention blocks and self-attention heads. For the cross-attention branch in block b, we measure its normalized residual update, while for self-attention head h in block b, we measure the output deviation caused by removing that head:

$$s_b^{\text{cross}} = \mathbb{E}_{q,k}\left[\frac{\left\|y_{q,k}^{b,\text{CA}} - x_{q,k}^{b,\text{CA}}\right\|_1}{\left\|x_{q,k}^{b,\text{CA}}\right\|_1 + \varepsilon}\right], \qquad s_{b,h}^{\text{self}} = \mathbb{E}_{q,k}\left[\frac{\left\|y_{q,k}^{b} - y_{q,k}^{b,-h}\right\|_1}{\left\|x_{q,k}^{b}\right\|_1 + \varepsilon}\right]. \tag{2}$$

Here, $x_{q,k}^{b,\text{CA}}$ and $y_{q,k}^{b,\text{CA}}$ denote the input and output of the cross-attention branch in block b, respectively. For self-attention, $x_{q,k}^{b}$ and $y_{q,k}^{b}$ denote the block input and output, respectively, and $y_{q,k}^{b,-h}$ denotes the block output when self-attention head h is removed. Figure 2(a) shows that many intermediate cross-attention branches produce near-identity updates, while removing a subset of self-attention heads causes only limited output deviation. In contrast, certain blocks and heads remain highly influential. In particular, cross-attention blocks in the early and late stages exhibit larger importance scores and are therefore less suitable for skipping. This heterogeneous importance motivates selective component-wise attention skipping rather than uniform pruning.

Analysis 2: Diffusion-Level Redundancy: Residual Stability. Let $x_{q,k}$ and $y_{q,k}$ denote the input and output of the Diffusion Transformer (DiT) backbone at denoising step k during the q-th action-chunk generation. We define the DiT backbone residual as

$$R_{q,k} = y_{q,k} - x_{q,k}. \tag{3}$$

We measure the relative L1 difference between each denoising step and the first step for the diffusion step input, output, and DiT residual. As shown in Figure 2(b), the sampler states evolve substantially along the denoising trajectory, whereas the predicted latents and DiT backbone residuals remain comparatively stable. The stable diffusion step outputs motivate direct one-step denoising, while the small variation of the DiT residuals suggests that the residual computed at the first denoising step can be reused to approximate subsequent steps. The latter retains the original multi-step solver and better preserves fine-grained task performance. These analyses motivate two acceleration modes: direct one-step prediction and residual-reuse multi-step sampling.

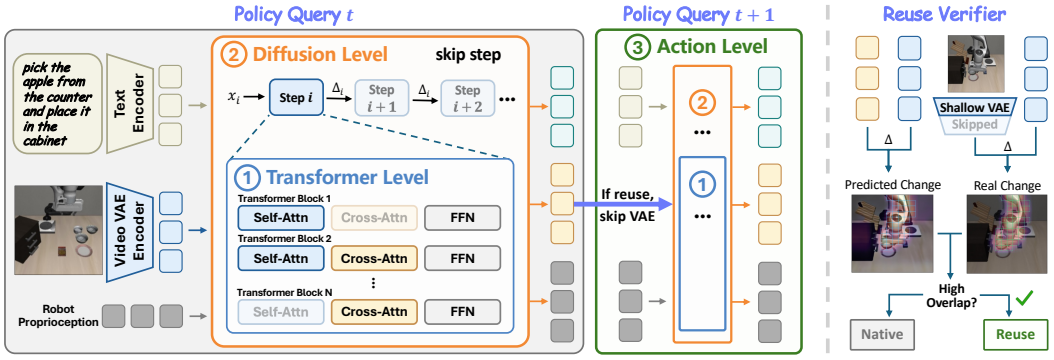

Figure 3: **The framework of MARS.** We provide a training-free acceleration method for WAMs. At the Transformer level, it skips redundant cross-attention and self-attention. At the diffusion level, it skips redundant diffusion steps via an action-aware router. At the action level, it verifies and reuses predicted future visual tokens when reliable, avoiding redundant visual encoding.

Analysis 3: Action-Level Redundancy: Speculative Visual Dynamics. Since the future visual latents predicted at query $q-1$ correspond to the scene after executing the predicted action chunk, they can provide a speculative representation of the observation at query q. However, directly reusing these latents is unsafe when the predicted scene deviates from the actual observation. As illustrated in Figure 2(c), strong spatial alignment between the predicted latent-change regions and the actual shallow-feature change regions suggests that the predicted scene evolution is consistent with the observed change. This reveals cross-chunk visual redundancy and provides a lightweight signal for determining when the previously predicted visual tokens can be safely reused at the current query.

3.3 The MARS Framework

Motivated by these analyses, we propose MARS, a training-free framework that reduces redundant computation at the Transformer, diffusion, and action levels. As illustrated in Figure 3, MARS comprises three corresponding mechanisms: Selective Attention Skipping, Action-Aware Step Skipping, and Shallow-Verify Visual Token Reuse, detailed below.

3.3.1 Selective Attention Skipping

Based on the attention importance analyzed in Section 3.2, we selectively remove low-impact attention at the Transformer level while preserving components that are critical to action generation. We rank cross-attention blocks and self-attention heads using the calibration scores defined in Equation 2. Given computational budgets N_{cross} and N_{head}, the skip sets are constructed as

$$\mathcal{B}_{\text{skip}}^{\text{cross}} = \text{BottomK}\left(\{s_b^{\text{cross}}\}_{b=1}^{L}, N_{\text{cross}}\right), \quad \mathcal{H}_{\text{skip}}^{\text{self}} = \text{BottomK}\left(\{s_{b,h}^{\text{self}}\}_{b,h}, N_{\text{head}}\right), \tag{4}$$

where $\text{BottomK}(\cdot, N)$ returns the N components with the smallest importance scores. For each block b, we bypass the entire cross-attention residual branch when it belongs to the skip set:

$$\hat{y}_{q,k}^{b,\text{CA}} = \begin{cases} x_{q,k}^{b,\text{CA}}, & b \in \mathcal{B}_{\text{skip}}^{\text{cross}}, \\ x_{q,k}^{b,\text{CA}} + \text{CA}_b\left(x_{q,k}^{b,\text{CA}}, c\right), & \text{otherwise}, \end{cases} \tag{5}$$

where c denotes the language conditioning. For self-attention, we structurally prune the selected heads before attention aggregation:

$$\widetilde{\text{MSA}}_b(X) = \sum_{h:\,(b,h)\notin\mathcal{H}_{\text{skip}}^{\text{self}}} \text{head}_{b,h}(X) W_{O,b}^{(h)}, \tag{6}$$

where $W_{O,b}^{(h)}$ denotes the output-projection slice associated with head h. Overall, these component-wise strategies eliminate low-impact attention computation.

3.3.2 Action-Aware Step Skipping

As discussed in Section 3.2, the early denoising stages already yield stable clean predictions and DiT residuals. However, the reliability of aggressive step reduction depends on the motion granularity of

the current query. Coarse-grained motions can generally tolerate direct one-step generation, whereas fine-grained manipulation stages benefit from retaining the original multi-step sampling trajectory. We introduce **Action-Aware Step Skipping**, which dynamically routes each query between direct one-step generation and residual-reuse multi-step sampling.

Residual-Reuse Step Skip. Prior cache-based diffusion acceleration methods typically reuse intermediate features across consecutive denoising steps (Ma et al., 2024; Bu et al., 2026). To preserve perceptual quality in video generation, these methods often introduce elaborate strategies to determine when to cache and how cached features are reused. In WAMs, however, future visual prediction primarily serves as a dynamics prior for action generation, and the ultimate objective is reliable control. This allows the caching strategy within the denoising process to be simplified more aggressively. Since the DiT residual effectively captures the feature transformation induced by the backbone, we cache the residual from the first denoising step and skip the subsequent DiT computations. At each remaining denoising step, we reuse the residual propagated from the preceding step to approximate the current DiT output:

$$\hat{y}_{q,k} = x_{q,k} + (\hat{y}_{q,k-1} - x_{q,k-1}), \qquad k = 2, \dots, K. \tag{7}$$

Following the original sampling schedule with these approximated backbone outputs produces the residual-reuse action chunk, denoted by A_q^{RR}.

Action-Aware Routing. The provisional action chunk $A_q^{(1)}$ is already available after the first denoising step and therefore provides a natural signal for estimating the motion granularity of the current query. For $A_q^{(1)} \in \mathbb{R}^{H\times 7}$, where H denotes the action-chunk horizon, we compute the mean magnitude of its translational components as

$$m_q = \frac{1}{H} \sum_{i=1}^{H} \left\| A_{q,i,1:3}^{(1)} \right\|_2 . \tag{8}$$

Large-motion chunks typically correspond to coarse reaching or transport and can be handled reliably by direct one-step generation. In contrast, small-motion chunks occur more frequently during approach, grasping, placement, and fine-adjustment phases (Wang et al., 2026; Li et al., 2026c), where residual-reuse multi-step sampling provides more stable predictions. Our method therefore selects the final action chunk as

$$A_q = \begin{cases} A_q^{(1)}, & m_q > \tau_{\mathrm{act}}, \\ A_q^{\mathrm{RR}}, & m_q \le \tau_{\mathrm{act}}, \end{cases} \tag{9}$$

where τ_{act} controls the trade-off between efficiency and performance. Unlike prior action-aware acceleration methods that infer motion phases from action history (Wang et al., 2026; Li et al., 2026c), our router directly uses the provisional action chunk of the current query, allowing the acceleration strategy to adapt immediately to changes in motion granularity.

3.3.3 Shallow-Verify Visual Token Reuse

After applying the methods in Sections 3.3.2 and 3.3.1, visual encoding becomes a dominant source of inference latency. Unlike conventional policies, WAMs already predict future visual tokens at each query. The future visual tokens predicted at query $q-1$ are temporally aligned with the observation at query q and can therefore serve as a speculative visual representation. Motivated by Analysis 3 in Section 3.2, we further reduce this overhead with **Shallow-Verify Visual Token Reuse**. Instead of unconditionally trusting the predicted future, we execute a shallow probe of the current VAE encoder and compare the observed shallow-feature changes with the visual changes predicted by the previous query. When the two change patterns are sufficiently consistent, we reuse the predicted future visual tokens and skip the remaining VAE encoder layers. Specifically, let $E_{\le r}$ denote the VAE probe truncated at shallow stage r. For view c, the shallow features of the current and previous observations are

$$f_q^c = E_{\le r}(I_q^c), \qquad f_{q-1}^c = E_{\le r}(I_{q-1}^c). \tag{10}$$

We obtain the actual spatial change map by averaging the channel-wise feature difference:

$$d_{q,\mathrm{real}}^c(u) = \frac{1}{C_r} \sum_{j=1}^{C_r} \left| f_q^c(u,j) - f_{q-1}^c(u,j) \right|, \tag{11}$$

where u indexes spatial positions and C_r is the number of feature channels. The predicted change map is computed from the previous observation latent z^c_{q-1} and its predicted future latent $\hat{z}^{c,\text{future}}_{q-1}$:

$$d^c_{q,\text{pred}}(v) = \frac{1}{C_z} \sum_{j=1}^{C_z} \left| \hat{z}^{c,\text{future}}_{q-1}(v,j) - z^c_{q-1}(v,j) \right|, \tag{12}$$

where v indexes latent spatial positions and C_z denotes the number of channels in the visual latent. For a fraction ρ of spatial positions, we define the support of the largest changes as

$$\mathcal{P}_\rho(d) = \text{TopK}\left(d,\, \rho|\Omega|\right), \tag{13}$$

where Ω denotes the spatial domain. The verification score for view c is then defined as the intersection-over-union v^c_q of the spatially aligned actual and predicted change regions. Because different views exhibit different motion patterns and levels of prediction difficulty, we assign a view-specific verification threshold τ^c_{VAE} to each view. For each view c, we set $g^{\text{VAE},c}_q = 1$ when $v^c_q \geq \tau^c_{\text{VAE}}$; otherwise, $g^{\text{VAE},c}_q = 0$. The visual condition for view c is then given by

$$z^{\text{obs},c}_q = \begin{cases} \hat{z}^{c,\text{future}}_{q-1}, & g^{\text{VAE},c}_q = 1, \\ E_{>r}(f^c_q), & g^{\text{VAE},c}_q = 0. \end{cases} \tag{14}$$

The verifier determines whether the predicted future provides a reliable substitute for the current representation. Once the verification criterion is satisfied, the model reuses the predicted future visual tokens and skips the remaining VAE encoding. To prevent error accumulation, we allow at most N_{VAE} consecutive queries with visual-token reuse, after which full VAE encoding is enforced.

4 Experiments

In this section, we comprehensively evaluate our method from three perspectives: (1) how its efficiency and task performance compare with existing methods; (2) whether the results generalize across task suites, robotic embodiments, base models and hardware platforms; and (3) how much each acceleration component contributes to the overall gains. To this end, we conduct evaluations in both simulation and real-world settings.

4.1 Experimental Setup

Benchmarks. We first evaluate our method in the widely adopted LIBERO (Liu et al., 2023). To further assess generalization, we evaluate in LIBERO-PRO (Zhou et al., 2025), which introduces controlled perturbations across multiple factors. We also evaluate in RoboCasa (Nasiriany et al., 2024), comprising 24 kitchen manipulation tasks with unseen object instances and scene styles.

Base Models. We mainly use Cosmos Policy (Kim et al., 2026) as the base model. We further evaluate our method on additional WAM models (*i.e.*, Fast-WAM (Yuan et al., 2026) and LingBot-VA (Li et al., 2026b)) and on a representative VLA model, $\pi_{0.5}$ (Intelligence et al., 2025).

Baselines. Our baselines include (1) DiCache (Bu et al., 2026), a representative method for video generation models that reuses diffusion residuals; (2) C^{3}ache (Zhao et al., 2026), specifically designed for WAMs by reducing cross-chunk redundancy; (3) X-Cache (Zeng et al., 2026), developed for autoregressive world models; and (4) Falcon (Chen et al., 2025), an acceleration method for diffusion policies. Since Falcon relies on unexecuted actions from the previous prediction as references for partial denoising, it can only be evaluated in RoboCasa. We report two operating points: **Ours (Speed)**, which prioritizes maximum acceleration, and **Ours (Acc.)**, which adopts a more conservative configuration to preserve task success.

Implementation Details. Results reported in the tables are averaged over three random seeds to ensure statistical robustness. All main experiments are conducted on NVIDIA A40 GPUs.

4.2 Main Results

Table 1 compares our method with Cosmos Policy and representative training-free acceleration baselines. In LIBERO, the original model achieves an average success rate of 98.05% with a

Table 1: **Comparison with representative training-free acceleration methods on the LIBERO (Liu et al., 2023) and LIBERO-PRO (Zhou et al., 2025) benchmarks.** All acceleration methods are adapted to the Cosmos Policy (Kim et al., 2026) architecture for a fair comparison.

LIBERO	Success Rate (%, ↑) / Speedup (↑)				Success Rate (↑)	Speedup (↑)	Latency (↓)	FLOPs (↓)
Method	**Goal**	**Object**	**Spatial**	**Long**				
Cosmos Policy	98.20 / 1.00	99.87 / 1.00	97.47 / 1.00	96.67 / 1.00	98.05%	1.00×	1.4001 s	100.00%
+ DiCache	98.07 / 1.70	99.87 / 1.66	97.80 / 1.67	97.27 / 1.72	**98.25%**	1.69×	0.8273 s	48.12%
+ C^{3}ache*	97.80 / 2.19	99.00 / 2.16	97.53 / 2.25	94.47 / 2.30	97.20%	2.24×	0.6256 s	33.18%
+ X-Cache*	97.87 / 1.11	99.67 / 1.17	97.47 / 1.12	97.00 / 1.21	98.00%	1.17×	1.2017 s	80.20%
+ Ours (Speed)	96.73 / 3.89	99.07 / 4.08	97.60 / 3.87	96.27 / 4.07	97.42%	**4.00×**	**0.3498 s**	**18.39%**
+ Ours (Acc.)	98.47 / 3.06	99.60 / 3.29	97.80 / 3.10	96.60 / 3.19	98.12%	3.17×	0.4414 s	19.19%
LIBERO-PRO	**Success Rate (%, ↑) / Speedup (↑)**				**Success Rate (↑)**	**Speedup (↑)**	**Latency (↓)**	**FLOPs (↓)**
Method	**Goal**	**Object**	**Spatial**	**Long**				
Cosmos Policy	43.67 / 1.00	46.13 / 1.00	47.40 / 1.00	45.33 / 1.00	45.63%	1.00×	1.4044 s	100.00%
+ DiCache	44.13 / 1.67	46.27 / 1.66	47.73 / 1.63	45.07 / 1.73	**45.80%**	1.68×	0.8347 s	50.32%
+ C^{3}ache*	42.80 / 2.22	45.53 / 2.20	47.53 / 2.28	42.60 / 2.30	44.62%	2.26×	0.6222 s	32.86%
+ X-Cache*	43.73 / 1.17	46.20 / 1.22	47.80 / 1.17	44.33 / 1.26	45.52%	1.22×	1.1549 s	75.55%
+ Ours (Speed)	44.20 / 3.91	45.67 / 3.86	47.33 / 3.96	44.93 / 3.99	45.53%	**3.94×**	**0.3564 s**	**18.80%**
+ Ours (Acc.)	43.67 / 3.05	47.00 / 3.08	47.60 / 3.06	44.60 / 3.11	45.72%	3.08×	0.4558 s	19.05%

* denotes our reimplementation due to unavailable official code.

latency of 1.40 s. DiCache preserves strong task performance through a carefully designed caching strategy, but provides relatively limited gains in inference speed and reductions in computational cost (FLOPs). In contrast, the cross-chunk acceleration methods C^{3}ache and X-Cache still offer limited speedups, while suffering performance degradation. These results show that exploiting redundancy at a single level either provides limited acceleration or incurs noticeable performance degradation. We jointly exploit redundancy across multiple computational levels, yielding a substantially better efficiency-performance trade-off. **Ours (Speed)** achieves a 4.00× speedup while preserving comparable task success, whereas **Ours (Acc.)** provides a more conservative 3.17× acceleration while slightly exceeding the original performance. Notably, our method reduces FLOPs to only 19% of the original model, demonstrating substantial computational savings. The acceleration gains are consistent across all four task suites and remain robust on the more challenging LIBERO-PRO benchmark, where our method achieves up to a 3.94× speedup under distribution shifts. We further evaluate our method in RoboCasa to examine its transferability to a different benchmark, where it achieves the best acceleration and task performance, as shown in Table 2. Additional experimental analyses, including evaluations across different hardware settings and random seeds, are provided in Appendices B, C, and D. Qualitative results and analyses are provided in Appendix G.

Table 2: **Evaluation on the RoboCasa (Nasiriany et al., 2024) benchmark.**

Method	S.R. (↑)	Speedup (↑)	Latency (↓)	FLOPs (↓)
Cosmos Policy	66.69%	1.00×	2.2958 s	100.00%
+ DiCache	66.47%	1.50×	1.5305 s	49.99%
+ C^{3}ache*	65.00%	1.81×	1.2653 s	33.64%
+ X-Cache*	66.37%	1.20×	1.9202 s	72.86%
+ Falcon	66.47%	1.21×	1.8968 s	74.56%
+ Ours	**66.94%**	**2.10×**	**1.0911 s**	**22.50%**

To further evaluate our method with different WAM and VLA base models, we apply its compatible components (*i.e.*, Transformer-level and diffusion-level acceleration) to Fast-WAM (Yuan et al., 2026), LingBot-VA (Li et al., 2026b), and the VLA model $\pi_{0.5}$ (Intelligence et al., 2025) in LIBERO using publicly released checkpoints. The results are reported in Table 3. Our method achieves substantial speedups on both WAM backbones while preserving task performance. The speedup on $\pi_{0.5}$ is more modest, as a substantial fraction of its latency lies in the VLM backbone. Nevertheless, our plug-and-play design is orthogonal to existing VLA acceleration techniques. Despite their distinct architectures and inference pipelines, these results demonstrate that our method is not specific to Cosmos Policy and can generalize across different WAM and VLA backbones.

Table 3: **Our method with additional WAM and VLA base models in LIBERO (Liu et al., 2023).** LingBot-VA can only be evaluated in Long, as only the Long checkpoint is publicly available.

LIBERO	Success Rate (%, ↑) / Speedup (↑)				Success Rate (↑)	Speedup (↑)	Latency (↓)
Method	**Goal**	**Object**	**Spatial**	**Long**			
Fast-WAM	95.87 / 1.00	99.40 / 1.00	97.13 / 1.00	93.93 / 1.00	**96.58%**	1.00×	0.5365 s
+ Ours	96.07 / 3.43	99.00 / 3.25	96.93 / 3.43	93.20 / 2.93	96.30%	**3.17×**	**0.1693 s**
LingBot-VA	–	–	–	97.40 / 1.00	97.40%	1.00×	13.4105 s
+ Ours	–	–	–	98.13 / 2.88	**98.13%**	**2.88×**	**4.6558 s**
$\pi_{0.5}$	97.80 / 1.00	98.67 / 1.00	98.40 / 1.00	92.93 / 1.00	**96.95%**	1.00×	0.1462 s
+ Ours	97.87 / 1.29	99.07 / 1.30	98.47 / 1.33	92.27 / 1.24	96.92%	**1.28×**	**0.1145 s**

Table 4: **Ablation study of the three acceleration components of our method on the LIBERO-PRO benchmark (Zhou et al., 2025).**

LIBERO-PRO	Success Rate (%, ↑) / Speedup (↑)				Success Rate (↑)	Speedup (↑)	Latency (↓)	FLOPs (↓)
Method	Goal	Object	Spatial	Long				
Cosmos Policy	43.67 / 1.00	46.13 / 1.00	47.40 / 1.00	45.33 / 1.00	45.63%	1.00×	1.4044 s	100.00%
Ours (Speed)	44.20 / 3.91	45.67 / 3.86	47.33 / 3.96	44.93 / 3.99	45.53%	**3.94×**	**0.3564 s**	**18.80%**
w/o Action-Aware Step Skipping	43.67 / 1.18	44.67 / 1.18	47.00 / 1.18	44.47 / 1.21	44.95%	1.19×	1.1803 s	92.81%
w/o Attention Skipping	43.80 / 3.64	45.67 / 3.59	47.80 / 3.69	43.93 / 3.69	45.30%	3.66×	0.3841 s	20.19%
w/o Visual Token Reuse	43.60 / 3.07	46.87 / 3.01	47.53 / 3.08	44.67 / 3.07	**45.67%**	3.06×	0.4595 s	19.05%

Table 5: **Sensitivity Analysis of the VAE Feature Layer in LIBERO-PRO.** Layer-1 achieves the best task performance and inference efficiency among the evaluated feature layers.

LIBERO-PRO	Success Rate (%, ↑) / Speedup (↑)				Success Rate (↑)	Speedup (↑)	Latency (↓)	VAE Skip Rate
Method	Goal	Object	Spatial	Long				
Cosmos Policy	43.67 / 1.00	46.13 / 1.00	47.40 / 1.00	45.33 / 1.00	45.63%	1.00×	1.4043 s	–
Layer-0	44.13 / 3.91	45.07 / 3.86	46.93 / 3.91	44.73 / 3.98	45.22%	3.93×	0.3574 s	56.20%
Layer-1	**44.20 / 3.91**	**45.67 / 3.86**	**47.33 / 3.96**	**44.93 / 3.99**	**45.53%**	**3.94×**	**0.3564 s**	56.25%
Layer-2	43.67 / 3.85	45.27 / 3.79	46.87 / 3.92	45.33 / 3.93	45.28%	3.88×	0.3621 s	56.41%
Layer-3	43.73 / 3.83	45.40 / 3.78	47.20 / 3.90	45.33 / 3.91	45.42%	3.86×	0.3638 s	56.44%

4.3 Ablation

Contribution of Each Acceleration Component. Table 4 evaluates the three components of our method in LIBERO-PRO. Action-Aware Step Skipping provides the largest acceleration gain, indicating substantial redundancy across diffusion denoising steps. Shallow-Verify Visual Token Reuse further contributes considerable speedup by reducing the increasingly dominant VAE encoding cost, while Attention Skipping provides an additional efficiency gain by eliminating redundant computation within the Transformer. Importantly, removing any individual module has only a minor effect on task success, suggesting that these modules exploit complementary sources of redundancy at different computational levels. Their joint application achieves the best overall efficiency-performance trade-off, reaching a 3.94× speedup with negligible performance degradation.

Further Analysis of Shallow-Verify Visual Token Reuse. We further analyze Shallow-Verify Visual Token Reuse by varying the depth of the shallow VAE probe. A shallower probe incurs less verification overhead but provides lower-level visual features, whereas deeper layers capture more abstract visual representations and tend to yield slightly higher skip rates, at the cost of additional encoding computation. The results are reported in Table 5. Overall, Layer-1 achieves the best balance between reuse rate and verification overhead while maintaining a high task success rate, and is therefore adopted as the default shallow probe. Nevertheless, our method maintains comparable task performance and substantial acceleration across different probe depths, indicating that it does not rely on a carefully tuned probe layer to achieve a favorable efficiency-performance trade-off. Additional latency analyses and parameter sensitivity analyses are provided in Appendices E and F.

4.4 Evaluation on the real-world task

We evaluate our method across 40 trials on the real-world pick-and-place task on Franka Panda, using an NVIDIA H20 for Cosmos Policy and an RTX 3090 for $\pi_{0.5}$. As shown in Table 6, our method effectively accelerates inference while preserving the success rate. These results further demonstrate the practical applicability of our method.

Table 6: **Evaluation on the real-world task on a Franka robot.**

Method	S.R. (↑)	Speedup (↑)	Latency (↓)
Cosmos Policy	90.00%	1.00×	0.6949 s
+ Ours	**95.00%**	**2.65×**	**0.2625 s**
$\pi_{0.5}$	25.00%	1.00×	0.1463 s
+ Ours	**25.00%**	**1.21×**	**0.1209 s**

5 Conclusion

In this paper, we introduced MARS, a training-free multi-level acceleration framework for world action models. Our in-depth analysis reveals distinct sources of redundancy across attention components, diffusion denoising, and consecutive action chunks. Based on these analyses, our method skips low-impact cross- and self-attention computation and adaptively routes each query between direct one-step generation and residual-reuse multi-step sampling according to the predicted action. It further conditionally reuses previously predicted future visual latents through online verification. These mechanisms can be systematically integrated into existing backbones in a plug-and-play manner, substantially reducing inference latency across multiple benchmarks and architectures, thereby demonstrating strong generalizability. Overall, our method provides a practical path toward efficient real-world deployment of world action models.

AI USE STATEMENT

Generative AI tools, specifically Large Language Models (LLMs), were used to improve the clarity and linguistic quality of the manuscript and to assist with code implementation. No LLMs were used for research idea generation or experimental design. All AI-assisted code was carefully reviewed and tested by the authors to ensure correctness, and all AI-assisted text was manually checked and revised. The authors take full responsibility for the final manuscript and experimental results.

ETHICS STATEMENT

This work investigates inference acceleration for existing World Action Models. Our evaluation is conducted on standard robotic manipulation benchmarks in simulation and on real-world robotic platforms under controlled experimental settings. The study does not involve human subjects, collection of personal or sensitive information, or the release of new datasets containing such information. Real-world experiments were carried out following standard laboratory safety and research integrity practices. We do not identify ethical concerns specific to the proposed acceleration techniques.

REPRODUCIBILITY STATEMENT

We have taken several measures to support the reproducibility of our results. The backbone models and checkpoints used in this work are publicly available, and the experimental benchmarks, including LIBERO, LIBERO-PRO, and RoboCasa, are standardized and publicly accessible. We will release the implementation code in a public GitHub repository. Additional experimental results are provided in the appendix to further facilitate reproduction and verification of our findings.

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.

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.

Jiazi Bu, Pengyang Ling, Yujie Zhou, Yibin Wang, Yuhang Zang, Dahua Lin, and Jiaqi Wang. DiCache: Let diffusion model determine its own cache. In *ICLR*, 2026.

Jun Cen, Siteng Huang, Yuqian Yuan, Kehan Li, Hangjie Yuan, Chaohui Yu, Bohan Hou, Yuming Jiang, Jiayan Guo, Xin Li, et al. RynnVLA-002: A unified vision-language-action and world model. *arXiv preprint arXiv:2511.17502*, 2025.

Haojun Chen, Minghao Liu, Chengdong Ma, Xiaojian Ma, Zailin Ma, Huimin Wu, Yuanpei Chen, Yifan Zhong, Mingzhi Wang, Qing Li, et al. Falcon: Fast visuomotor policies via partial denoising. In *ICML*, 2025.

Yilun Du, Sherry Yang, Bo Dai, Hanjun Dai, Ofir Nachum, Josh Tenenbaum, Dale Schuurmans, and Pieter Abbeel. Learning universal policies via text-guided video generation. *NeurIPS*, 36: 9156–9172, 2023.

Yao Feng, Hengkai Tan, Xinyi Mao, Chendong Xiang, Guodong Liu, Shuhe Huang, Hang Su, and Jun Zhu. Vidar: Embodied video diffusion model for generalist manipulation. *arXiv preprint arXiv:2507.12898*, 2025.

Yanjiang Guo, Yucheng Hu, Jianke Zhang, Yen-Jen Wang, Xiaoyu Chen, Chaochao Lu, and Jianyu Chen. Prediction with Action: Visual policy learning via joint denoising process. *NeurIPS*, 37: 112386–112410, 2024.

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

Kangye Ji, Yuan Meng, Hanyun Cui, Ye Li, Jianbo Zhou, Shengjia Hua, Lei Chen, and Zhi Wang. Block-wise adaptive caching for accelerating diffusion policy. *arXiv preprint arXiv:2506.13456*, 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. In *ICLR*, 2026.

Jiajun Li, Tiecheng Guo, Yifan Ye, Rongyu Zhang, Xiaowei Chi, Qianpu Sun, Ying Li, Yunfan Lou, Yan Huang, Zhihe Lu, et al. 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, et al. Causal world modeling for robot control. *arXiv preprint arXiv:2601.21998*, 2026b.

Ye Li, Yuan Meng, Zewen Sun, Kangye Ji, Chen Tang, Jiajun Fan, Xinzhu Ma, Shutao Xia, Zhi Wang, and Wenwu Zhu. SP-VLA: A joint model scheduling and token pruning approach for vla model acceleration. In *ICLR*, 2026c.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. *NeurIPS*, 36:44776–44791, 2023.

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 *CVPR*, pp. 7353–7363, 2025.

Yunfan Lou, Yifan Ye, Yankai Fu, Jun Cen, Xiaowei Chi, Yaoxu Lyu, Peidong Jia, Sirui Han, Zhihe Lu, and Shanghang Zhang. Dream-Tac: A unified tactile world action model for contact-rich robot manipulation. *arXiv preprint arXiv:2606.08737*, 2026.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. DeepCache: Accelerating diffusion models for free. In *CVPR*, pp. 15762–15772, 2024.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. RoboCasa: Large-scale simulation of everyday tasks for generalist robots. *arXiv preprint arXiv:2406.02523*, 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 *ECCV*, 2026.

Ryuji Oi, Hikari Otsuka, Kosuke Matsushima, Yuki Ichikawa, Masato Motomura, Tatsuya Kaneko, and Daichi Fujiki. Training-free acceleration for vision-language-action models with action caching and refinement. *arXiv preprint arXiv:2607.06370*, 2026.

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

Xiaohuan Pei, Yuxing Chen, Siyu Xu, Yunke Wang, Yuheng Shi, and Chang Xu. Action-aware dynamic pruning for efficient vision-language-action manipulation. In *ICLR*, 2026.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

Yichao Shen, Fangyun Wei, Zhiying Du, Yaobo Liang, Yan Lu, Jiaolong Yang, Nanning Zheng, and Baining Guo. VideoVLA: Video generators can be generalizable robot manipulators. *NeurIPS*, 38:95597–95621, 2025.

Hanzhen Wang, Jiaming Xu, Yushun Xiang, Jiayi Pan, Yongkang Zhou, Yong-Lu Li, and Guohao Dai. SpecPrune-VLA: Accelerating vision-language-action models via action-aware self-speculative pruning. In *ICML*, 2026.

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. In *ICLR*, volume 2024, pp. 10641–10662, 2024.

Siyu Xu, Yunke Wang, Chenghao Xia, Dihao Zhu, Tao Huang, and Chang Xu. VLA-Cache: Efficient vision-language-action manipulation via adaptive token caching. *NeurIPS*, 2025.

Yantai Yang, Yuhao Wang, Zichen Wen, Luo Zhongwei, Chang Zou, Zhipeng Zhang, Chuan Wen, and Linfeng Zhang. EfficientVLA: Training-free acceleration and compression for vision-language-action models. *NeurIPS*, 38:40891–40914, 2025.

Angen Ye, Boyuan Wang, Chaojun Ni, Guan Huang, Guosheng Zhao, Hao Li, Hengtao Li, Jie Li, Jindi Lv, Jingyu Liu, et al. 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, et al. 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.

Zhihang Yuan, Hanling Zhang, Pu Lu, Xuefei Ning, Linfeng Zhang, Tianchen Zhao, Shengen Yan, Guohao Dai, and Yu Wang. DiTFastAttn: Attention compression for diffusion transformer models. *NeurIPS*, 37:1196–1219, 2024.

Yixiao Zeng, Jianlei Zheng, Chaoda Zheng, Shijia Chen, Mingdian Liu, Tongping Liu, Tengwei Luo, Yu Zhang, Boyang Wang, Linkun Xu, et al. X-Cache: Cross-chunk block caching for few-step autoregressive world models inference. *arXiv preprint arXiv:2604.20289*, 2026.

Qingqing Zhao, Yao Lu, Moo Jin Kim, Zipeng Fu, Zhuoyang Zhang, Yecheng Wu, Zhaoshuo Li, Qianli Ma, Song Han, Chelsea Finn, et al. CoT-VLA: Visual chain-of-thought reasoning for vision-language-action models. *arXiv preprint arXiv:2503.22020*, 2025a.

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

Xuanlei Zhao, Xiaolong Jin, Kai Wang, and Yang You. Real-time video generation with pyramid attention broadcast. In *ICLR*, volume 2025, pp. 3296–3319, 2025b.

Kaijun Zhou, Qiwei Chen, Da Peng, Zhiyang Li, Xijun Li, and Jinyu Gu. Characterizing Vision-Language-Action Models across XPUs: Constraints and acceleration for on-robot deployment. *arXiv preprint arXiv:2604.24447*, 2026.

Xueyang Zhou, Yangming Xu, Guiyao Tie, Yongchao Chen, Guowen Zhang, Duanfeng Chu, Pan Zhou, and Lichao Sun. LIBERO-PRO: Towards robust and fair evaluation of vision-language-action models beyond memorization. *arXiv preprint arXiv:2510.03827*, 2025.

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 Experimental Settings

A.1 Simulation and Real-World Environments

We evaluate our method on three simulation benchmarks covering diverse robotic manipulation tasks and generalization settings:

- **LIBERO (Liu et al., 2023):** LIBERO is a widely used benchmark for evaluating generalizable robot manipulation policies. We evaluate on its four task suites, including LIBERO-Spatial, LIBERO-Object, LIBERO-Goal, and LIBERO-Long, which assess spatial reasoning, object interaction, goal-conditioned manipulation, and long-horizon task execution, respectively. In total, we conduct 2,000 evaluation trajectories to assess model robustness.
- **LIBERO-PRO (Zhou et al., 2025):** LIBERO-PRO extends LIBERO with controlled distribution shifts across five core generalization dimensions, including Object, which varies object appearance, color, and scale; Position, which changes object locations within feasible spatial regions; Semantic, which paraphrases language instructions; Task, which modifies task logic and target states; and Environment, which introduces different scene layouts and environments. We conduct 2,000 evaluation trajectories to examine whether acceleration methods remain robust under these challenging distribution shifts.
- **RoboCasa (Nasiriany et al., 2024):** RoboCasa provides diverse household manipulation tasks in realistic kitchen environments. We evaluate 24 tasks using a Franka Emika Panda robot. For each task, the success rate is measured over 50 trials across five evaluation scenes with different floor plans and styles, resulting in 1,200 trials. All evaluation objects are unseen instances, and two of the five scenes for each task additionally use styles not observed during training, providing a challenging test of visual and environmental generalization.
- **Real-World Environment:** We further evaluate our method on real-world robotic manipulation tasks using the Franka platform. These experiments assess whether our method can provide practical inference acceleration while preserving closed-loop performance under real-world visual observations and physical interactions.

A.2 Base Models

We evaluate our method across multiple WAM and VLA architectures to examine its architectural generality:

- **Cosmos Policy (Kim et al., 2026):** Cosmos Policy serves as our primary experimental backbone. It adapts the pretrained Cosmos-Predict2 video diffusion model to jointly generate robot actions, future visual observations, proprioceptive states, and state values within a unified generative process.
- **Fast-WAM (Yuan et al., 2026):** Fast-WAM retains video modeling during training while removing explicit future visual generation at inference time. We use it to evaluate whether the compatible components of our method generalize to WAMs without test-time future imagination.
- **LingBot-VA (Li et al., 2026b):** LingBot-VA is an autoregressive diffusion-based WAM that jointly performs future-frame prediction and policy execution. Its distinct autoregressive inference pipeline provides an additional setting for evaluating our method.
- $\pi_{0.5}$ **(Intelligence et al., 2025):** $\pi_{0.5}$ is a general-purpose VLA model designed for broad open-world manipulation generalization. We apply the compatible acceleration component of our method to examine its transferability beyond WAM architectures.

A.3 Baselines

We compare our method with representative training-free inference acceleration methods. Methods marked with * are our implementations, as official code is unavailable. All baselines are adapted to the same Cosmos Policy backbone for a fair comparison:

- **DiCache (Bu et al., 2026):** DiCache is a training-free diffusion acceleration method that exploits redundancy across consecutive denoising steps. It estimates intermediate feature changes and reuses cached residuals when the variation is sufficiently small.
- **C^{3}ache (Zhao et al., 2026):** C^{3}ache exploits redundancy across consecutive inference chunks in WAMs. It caches and reuses DiT residuals at corresponding denoising steps across action chunks to reduce repeated computation.
- **X-Cache (Zeng et al., 2026):** X-Cache is a training-free cross-chunk caching method that performs fine-grained block-level reuse. It employs structure- and action-aware fingerprints to determine whether individual Transformer blocks should recompute or reuse residuals.
- **Falcon (Chen et al., 2025):** Falcon is a training-free diffusion policy acceleration method that exploits temporal dependencies across control steps. It reuses historical partially denoised actions as the initialization for current denoising, using one-step estimation and a thresholding mechanism to select suitable starting points and reduce sampling steps.

B Analysis of Experimental Results

Table 1 compares our method with representative training-free acceleration methods in LIBERO and LIBERO-PRO. The results reveal three main observations:

(1) Carefully controlled caching preserves accuracy but provides limited acceleration. DiCache preserves task performance well, achieving 98.25% and 45.80% success rates in LIBERO and LIBERO-PRO, respectively, even slightly outperforming the original policy. However, it provides only 1.69× and 1.68× speedups. This suggests that diffusion caching strategies originally designed for video generation are overly conservative for WAMs, leaving substantial computational redundancy unexploited.

(2) Aggressive cross-chunk reuse can impair long-horizon task execution. C^{3}ache and X-Cache exhibit larger performance degradation, particularly on the Long suite, indicating that approximation errors introduced by cross-chunk computation reuse can accumulate over consecutive action chunks. This highlights the sensitivity of WAMs to aggressive computation reuse across temporally dependent action chunks.

(3) Multi-level redundancy elimination yields a better efficiency-performance trade-off. Our method jointly exploits redundancy across Transformer components, denoising steps, and consecutive action chunks. **Ours (Speed)** achieves 4.00×/3.94× speedups in LIBERO/LIBERO-PRO, while **Ours (Acc.)** achieves 3.17×/3.08× acceleration and preserves, and even slightly improves the success rate of the original policy. A similar trend is observed on the RoboCasa benchmark, where our method provides substantial acceleration while maintaining task performance. These results demonstrate that coordinated multi-level acceleration provides a more favorable efficiency-performance trade-off than exploiting a single source of redundancy.

Table 7: **Comparison of our method across different GPUs in LIBERO.**

LIBERO	Success Rate (%, ↑) / Speedup (↑)				Success Rate (↑)	Speedup (↑)	Latency (↓)
Method	Goal	Object	Spatial	Long			
A40							
Cosmos Policy	98.20 / 1.00	99.87 / 1.00	97.47 / 1.00	96.67 / 1.00	98.05%	1.00×	1.4001 s
+ Ours (Speed)	96.73 / 3.89	99.07 / 4.08	97.60 / 3.87	96.27 / 4.07	97.42%	**4.00**×	**0.3498 s**
RTX 3090							
Cosmos Policy	97.67 / 1.00	100.00 / 1.00	97.53 / 1.00	97.40 / 1.00	98.15%	1.00×	1.4181 s
+ Ours (Speed)	97.07 / 3.97	99.53 / 4.16	98.13 / 3.93	96.13 / 4.10	97.72%	**4.06**×	**0.3496 s**
RTX 5090							
Cosmos Policy	98.27 / 1.00	100.00 / 1.00	97.27 / 1.00	97.60 / 1.00	98.28%	1.00×	0.5698 s
+ Ours (Speed)	97.13 / 3.66	99.33 / 3.82	98.00 / 3.63	95.60 / 3.73	97.52%	**3.72**×	**0.1532 s**

C Evaluation Across Hardware Platforms

To assess the hardware generality of our method, we benchmark its inference efficiency on three GPU platforms: NVIDIA GeForce RTX 3090, NVIDIA GeForce RTX 5090, and NVIDIA A40. As shown in Table 7, our method consistently reduces end-to-end latency across all devices on the LIBERO benchmark. Although the absolute latency and observed speedup vary with hardware architecture, memory bandwidth, and kernel efficiency, the acceleration ratio remains consistently around $4\times$. These results demonstrate that our method is not tailored to a specific accelerator and can provide practical efficiency gains across both consumer-grade and datacenter GPUs.

D Results Across Random Seeds

To examine the robustness of our results to evaluation randomness, we report the complete results in the LIBERO simulation using three random seeds (195, 196, and 197) in Table 8. The results reported in the main-text tables are averaged over these three runs. Across different seeds, our method exhibits highly consistent performance and acceleration. In particular, Ours (Speed) achieves average success rates of 97.45%, 97.30%, and 97.50%, together with speedups of $4.04\times$, $3.99\times$, and $3.98\times$, respectively. These results demonstrate that the effectiveness of our method is robust to different random seeds, with only minor variations in task success rate and inference efficiency.

E Latency Analysis

Figure 4 shows how the computational bottleneck shifts as the three acceleration modules are progressively introduced in LIBERO. In Cosmos Policy, DiT computation dominates the overall latency. Action-Aware Step Skipping substantially reduces repeated DiT computation, lowering the total measured latency from 1.40 s to 0.47 s. Selective Attention Skipping further reduces Transformer cost by eliminating low-impact attention computation. Consequently, VAE encoding becomes the dominant remaining bottleneck, accounting for 47.7% of the latency in the accelerated pipeline. Finally, Shallow-Verify Visual Token Reuse reduces VAE encoding latency, achieving nearly a $4\times$ end-to-end speedup. This progressive shift in the latency composition validates the effectiveness of multi-level acceleration for WAM inference.

Table 8: **Results in LIBERO with different random seeds.**

LIBERO Method	Success Rate (%, ↑) / Speedup (↑)				Success Rate (↑)	Speedup (↑)	Latency (↓)
	Goal	Object	Spatial	Long			
Seed 195							
Cosmos Policy	98.20 / 1.00	99.80 / 1.00	98.00 / 1.00	96.00 / 1.00	98.00%	1.00×	1.4194 s
+ DiCache	98.20 / 1.69	100.00 / 1.59	98.60 / 1.65	96.80 / 1.60	**98.40%**	1.62×	0.8737 s
+ C^{3}ache∗	98.40 / 2.20	99.00 / 2.18	97.00 / 2.26	95.20 / 2.39	97.40%	2.28×	0.6215 s
+ X-Cache*	97.80 / 1.10	99.40 / 1.17	97.80 / 1.12	95.80 / 1.24	97.70%	1.18×	1.2036 s
+ Ours (Speed)	96.20 / 3.86	99.60 / 4.08	97.80 / 3.86	96.20 / 4.19	97.45%	**4.04**×	**0.3511 s**
+ Ours (Acc.)	99.00 / 3.06	99.40 / 3.29	98.20 / 3.10	96.60 / 3.29	98.30%	3.21×	0.4417 s
Seed 196							
Cosmos Policy	99.00 / 1.00	100.00 / 1.00	97.20 / 1.00	96.60 / 1.00	**98.20%**	1.00×	1.3935 s
+ DiCache	98.40 / 1.71	100.00 / 1.70	97.20 / 1.68	97.20 / 1.79	**98.20%**	1.74×	0.8030 s
+ C^{3}ache∗	98.00 / 2.18	98.80 / 2.16	97.80 / 2.25	93.20 / 2.26	96.95%	2.22×	0.6279 s
+ X-Cache*	98.20 / 1.10	99.80 / 1.17	97.00 / 1.13	97.00 / 1.19	98.00%	1.16×	1.2039 s
+ Ours (Speed)	97.20 / 3.91	98.80 / 4.08	97.60 / 3.89	95.60 / 4.01	97.30%	**3.99**×	**0.3496 s**
+ Ours (Acc.)	98.20 / 3.07	99.40 / 3.32	97.60 / 3.10	95.80 / 3.16	97.75%	3.16×	0.4403 s
Seed 197							
Cosmos Policy	97.40 / 1.00	99.80 / 1.00	97.20 / 1.00	97.40 / 1.00	97.95%	1.00×	1.3873 s
+ DiCache	97.60 / 1.70	99.60 / 1.68	97.60 / 1.66	97.80 / 1.78	98.15%	1.72×	0.8052 s
+ C^{3}ache∗	97.00 / 2.18	99.20 / 2.15	97.80 / 2.24	95.00 / 2.25	97.25%	2.21×	0.6273 s
+ X-Cache*	97.60 / 1.11	99.80 / 1.17	97.60 / 1.12	98.20 / 1.19	98.30%	1.16×	1.1976 s
+ Ours (Speed)	96.80 / 3.90	98.80 / 4.07	97.40 / 3.86	97.00 / 4.02	97.50%	**3.98**×	**0.3486 s**
+ Ours (Acc.)	98.20 / 3.06	100.00 / 3.27	97.60 / 3.09	97.40 / 3.12	**98.30%**	3.14×	0.4423 s

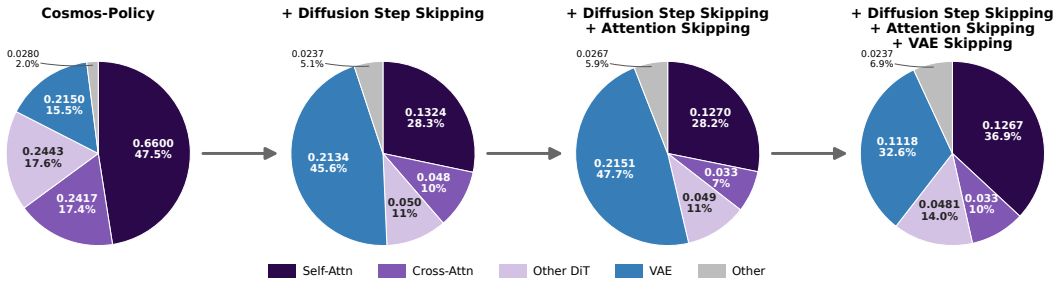

Figure 4: **Analysis of the latency.** Component-wise latency breakdown and the resulting shift in computational bottlenecks as different acceleration components are progressively introduced.

Table 9: **Sensitivity Analysis of the Action Routing Threshold τ_{act} in LIBERO-PRO.** Our method maintains comparable task performance and substantial acceleration across a broad range of action threshold values.

LIBERO-PRO	Success Rate (%, ↑) / Speedup (↑)				Success Rate (↑)	Speedup (↑)	Latency (↓)
Method	Goal	Object	Spatial	Long			
Cosmos Policy	43.67 / 1.00	46.13 / 1.00	47.40 / 1.00	45.33 / 1.00	45.63%	1.00×	1.4043 s
Ours-1 (0.0)	44.00 / 4.01	45.47 / 3.95	46.80 / 4.02	44.27 / 4.10	45.13%	**4.03**×	**0.3483 s**
Ours-2 (0.2)	43.73 / 3.93	45.60 / 3.84	46.80 / 3.95	43.80 / 4.04	44.98%	3.96×	0.3550 s
Ours-3 (0.4)	**44.20 / 3.91**	**45.67 / 3.86**	**47.33 / 3.96**	**44.93 / 3.99**	**45.53%**	3.94×	0.3564 s
Ours-4 (0.6)	43.93 / 3.90	45.47 / 3.86	47.27 / 3.95	44.40 / 3.99	45.27%	3.94×	0.3567 s
Ours-5 (0.8)	43.00 / 3.87	45.60 / 3.82	47.47 / 3.91	44.27 / 3.96	45.08%	3.90×	0.3601 s
Ours-6 (1.0)	43.73 / 3.85	45.53 / 3.78	47.73 / 3.88	43.67 / 3.94	45.17%	3.87×	0.3625 s

F Sensitivity Analysis of the Action Routing Threshold

The action routing threshold τ_{act} controls the trade-off between aggressive one-step generation and the more conservative Residual-Reuse Step Skip. To examine the sensitivity of our method to this hyperparameter, we vary τ_{act} over $\{0.0, 0.2, 0.4, 0.6, 0.8, 1.0\}$ and evaluate both task success rate and end-to-end inference speedup, as reported in Table 9. A smaller threshold routes more action chunks to direct one-step generation, yielding higher acceleration but potentially reducing robustness for fine-grained manipulation, whereas a larger threshold retains Residual-Reuse Step Skip for more queries and therefore favors task performance at the cost of additional computation. Across the tested range of τ_{act}, our method consistently maintains comparable task performance and substantial acceleration, demonstrating that its effectiveness is not sensitive to a particular threshold choice and does not rely on specific hyperparameter tuning.

G Qualitative Results and Analyses

To provide a deeper understanding of our proposed framework, we compare Ours (Acc.) with the original Cosmos Policy through qualitative visualizations in both LIBERO and LIBERO-PRO. We analyze 12,000 paired episodes across three random seeds and identify 186 cases in which Ours succeeds while Cosmos Policy fails, compared with 177 cases exhibiting the opposite outcome. Although these changes largely offset each other, Ours still achieves a slight overall improvement in success rate.

Ours more often recovers trajectories involving direct object transport, where Cosmos Policy may contact or grasp the target but fail to deliver it to the desired location. As shown in Figure 5, the qualitative comparisons include representative tasks where Ours succeeds while the original policy fails. We hypothesize that using direct one-step denoising more frequently during coarse-grained manipulation stages can facilitate task completion. This advantage is particularly evident across object pick-and-place tasks in LIBERO-PRO, where Ours yields a net gain of 17 successful episodes over the original Cosmos Policy. In the shown cases, Cosmos Policy prematurely opens the gripper during large motions, whereas Ours maintains a stable grasp until placement.

In contrast, Ours is relatively more sensitive to long-horizon sequential tasks involving drawers or microwaves, as illustrated in Figure 6. These tasks often require more precise grasping, positioning, and interaction, for which aggressive inference acceleration may introduce a slight degradation in control precision. Overall, these results suggest that our method largely preserves the original policy performance under substantial acceleration, with slightly more trajectories benefiting from acceleration than being degraded by it.

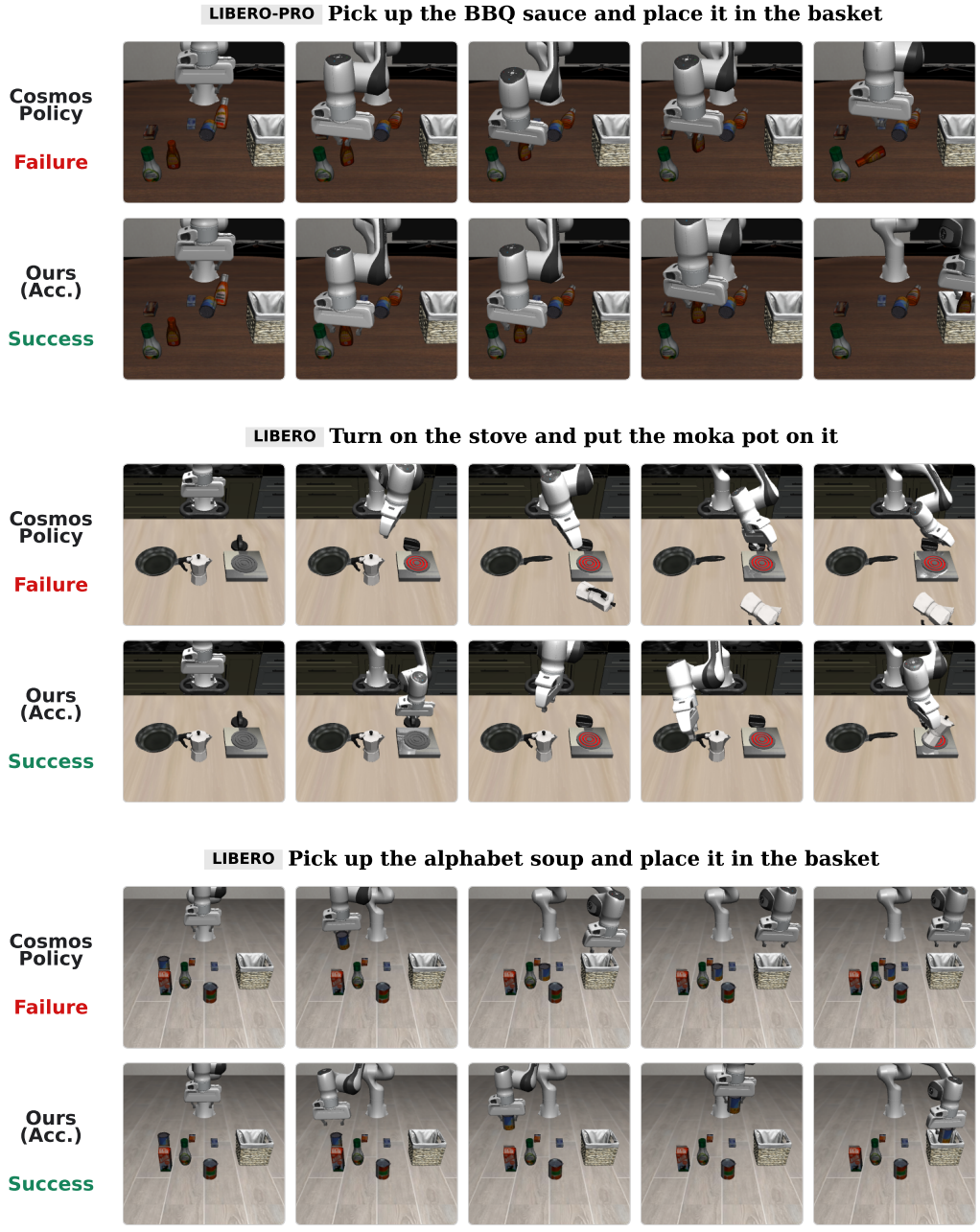

Figure 5: **Qualitative examples where our method improves task completion.** Representative comparisons in LIBERO and LIBERO-PRO where Ours (Acc.) succeeds while the original Cosmos Policy fails.

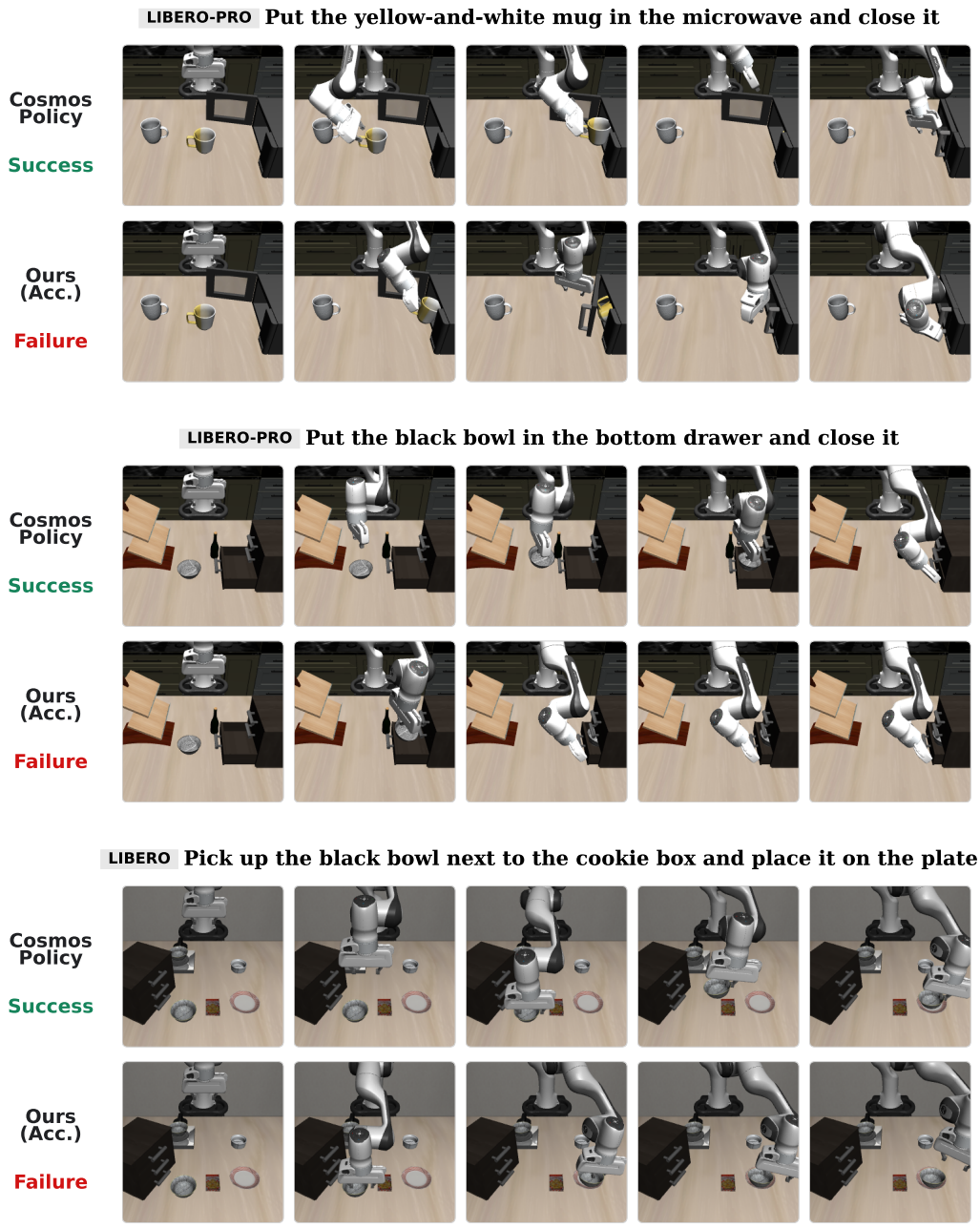

Figure 6: **Qualitative examples of failure cases under acceleration.** Representative comparisons in LIBERO and LIBERO-PRO where the original Cosmos Policy succeeds while Ours (Acc.) fails.