SPARSEWAM: RELEVANCE-GUIDED ASYMMETRIC SPARSIFICATION FOR WORLD ACTION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

World Action Models (WAMs) use world-model representations to condition action generation, but compressing these representations can compromise closed-loop robustness. We study where sparsification should be applied, how task relevance should guide token selection, and how world-side compression interacts with action-side acceleration. Our analysis shows that sparsifying exported conditioning representations after dense world-model computation preserves task success substantially better than pruning visual tokens before representation formation, while aggressive diversity-based pruning becomes fragile under distribution shift. Motivated by these observations, we introduce SparseWAM, a training-free framework for relevance-guided asymmetric sparsification. On the world side, SparseWAM follows a relevance-before-redundancy principle: instruction-conditioned scores first restrict the candidate set, after which diversity-based selection enforces a fixed conditioning budget. On the action side, it adapts similarity-based layer skipping to selected Action-DiT denoising steps. At 50% conditioning retention with ActionSkip, SparseWAM achieves 95.00% on LIBERO and 50.66% on LIBERO-Plus with 369.20 ms inference latency, compared with 455.73 ms for the dense model and 378.40 ms for ActionSkip alone. Under a more aggressive 25% budget, semantic guidance improves LIBERO-Plus success from 42.83% to 44.16% with world-side sparsification alone and from 42.50% to 44.22% when combined with ActionSkip, partially recovering the degradation caused by compression. Controlled real-robot shirt-folding trials further show that moderate sparsification retains more successful executions than aggressive joint compression. These results identify compression location and token-selection strategy as important factors in the efficiency–robustness trade-off of WAM inference across both simulated and real-world settings.

1 INTRODUCTION

Vision-Language-Action (VLA) models map observations and instructions to robot actions. World Action Models (WAMs) additionally incorporate world modeling to learn predictive representations for control (Cen et al., 2025; Yuan et al., 2026; Li et al., 2026a). Beyond the cost of generating actions themselves, this architecture introduces an additional interface through which world-model representations condition the action generator. Understanding how this interface can be compressed without destroying control-relevant information is therefore important for efficient WAM inference.

We focus on a representative dual-stream WAM architecture in which a Video DiT encodes the current observation and provides layer-wise conditioning features to an iterative Action DiT (Yuan et al., 2026). Importantly, the world-model branch does not explicitly generate future video frames during inference. This allows us to distinguish two conceptually different forms of sparsification: removing visual tokens before world-model computation, and sparsifying the representations exported from the world branch after they have been densely formed. The distinction matters because the former changes representation formation itself, whereas the latter compresses only the information passed to the action generator. Our experiments reveal a pronounced asymmetry between these two choices. With 50% retention, diversity-based sparsification of exported conditioning tokens preserves 95.75% LIBERO success, compared with 96.75% for the dense baseline. At 25% retention, performance remains 89.85% on LIBERO but decreases from 52.00% to 42.83% on LIBERO-Plus. By contrast, pruning input visual tokens to the same retention ratio yields only 24.90% and 2.66%,

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.

respectively. These results suggest that dense representation formation is substantially more important than dense transmission of the resulting conditioning features. They also reveal a second issue: high in-distribution performance can conceal considerable sensitivity to distribution shift.

This motivates us to ask not only where to sparsify, but also which representations should be retained. Existing token-reduction methods use attention, similarity, semantic relevance, or diversity to construct compact representations (Chen et al., 2024; Zhang et al., 2025b; Alvar et al., 2025), and instruction-conditioned diversity has also been explored in prior work such as CDPruner (Zhang et al., 2025a). Rather than proposing a new generic pruning primitive, we study how these principles behave at the world-to-action interface of WAMs. A key empirical observation is that neither semantic relevance nor diversity alone is sufficient under aggressive compression. We therefore adopt a simple relevance-before-redundancy strategy: instruction-conditioned scores first remove a small fraction of weakly relevant candidates, after which diversity-based selection chooses a compact subset under the same final budget. At 25% retention, this increases LIBERO-Plus success from 42.83% to 44.16%. The gain is deliberately interpreted as a partial recovery under distribution shift rather than a restoration of dense-model robustness. World-side sparsification addresses only one source of inference cost. The action generator itself performs iterative denoising through a deep transformer, suggesting a complementary opportunity along the depth dimension. We adapt the input–output-similarity layer-skipping mechanism of Flash-Unified (Ke et al., 2026a) to selected Action-DiT denoising steps and refer to this component as ActionSkip. Used alone, ActionSkip achieves 96.80% LIBERO success, 51.93% LIBERO-Plus success, and 378.40 ms inference latency. SparseWAM combines this action-side depth reduction with relevance-guided world-to-action sparsification while keeping world-model computation dense. The resulting design is asymmetric: the world branch is compressed along the conditioning-token dimension, whereas the action branch is reduced along network depth to target redundant inference computation during denoising.

We evaluate SparseWAM on LIBERO (Liu et al., 2023), LIBERO-Plus (Fei et al., 2025), and controlled real-robot shirt folding. Our experiments serve two complementary purposes. First, a fixed 25% conditioning budget acts as a stress test for comparing token-selection strategies under aggressive compression. Second, a retention sweep evaluates the accuracy–latency trade-off of the complete system. At 50% retention with ActionSkip, SparseWAM reaches 95.00% LIBERO success and 50.66% LIBERO-Plus success at 369.20 ms, providing additional latency reduction over ActionSkip alone at a measurable accuracy cost. Real-robot trials further examine whether this trade-off persists beyond simulation, showing that moderate compression preserves more successful shirt-folding executions than the aggressive joint configuration. We report successful-episode durations only as execution statistics and do not interpret them as evidence of faster task completion.

The main contributions are summarized as follows:

- A study of where to sparsify world-to-action representations. We compare input-token removal, propagated hidden-token pruning, and exported conditioning sparsification, showing that preserving dense representation formation is critical for closed-loop performance.
- A relevance-guided conditioning strategy for WAMs. We combine instruction-conditioned candidate filtering with diversity-based selection under a fixed conditioning budget, and systematically study candidate-set size, selection order, and robustness under distribution shift across multiple conditioning budgets and evaluation settings.
- Asymmetric world–action sparsification. We combine token-level compression of world-to-action conditioning with action-side depth reduction, quantify the resulting accuracy–latency trade-off against the action-only reference, and validate the behavior in controlled real-robot experiments across both moderate and aggressive compression regimes.

2 Related Work

Efficient Vision-Language-Action Models. Fast-WAM separates the training benefits of video modeling from explicit future generation at inference Yuan et al. (2026), while Efficient-WAM reduces the cost of future imagination through compact video modeling and sparse representations Li et al. (2026a). Concurrent work further explores sparse future conditioning and shallow action modules Zhao et al. (2026); Ma et al. (2026). Together, these studies indicate that efficiency gains can be obtained from multiple parts of a WAM, including future representation, world–action interaction,

and action computation. However, reducing cost at different locations does not necessarily have the same effect on downstream control, since information removed before representation formation cannot be recovered by later modules. SparseWAM therefore focuses on training-free compression of the world-to-action conditioning interface in Fast-WAM, where world-model computation remains dense and only the exported conditioning is sparsified. We additionally study how this compression interacts with action-side skipping, allowing token-level world-side reduction and depth-level action-side reduction to be evaluated within the same closed-loop setting.

Visual Token Pruning. Visual-token pruning exploits similarity, diversity, attention, and task relevance to reduce multimodal inference cost Bolya et al. (2023); Chen et al. (2024); Zhang et al. (2025b); Alvar et al. (2025). Attention-based approaches remove weakly attended tokens, while relevance-guided methods use language or task information to preserve tokens that are more likely to contribute to downstream prediction. Diversity-based methods instead emphasize coverage of distinct visual content and reduce redundancy among retained tokens. Recent VLA methods further incorporate action relevance, temporal structure, or performance-aware token selection Liu et al. (2026); Li et al. (2026b); Jiang et al. (2026). CDPruner combines instruction relevance and diversity through conditional subset selection Zhang et al. (2025a), showing that the two criteria need not be treated independently. SparseWAM differs mainly in where this selection is applied. Rather than pruning the visual sequence before world-representation formation, we apply a hard relevance filter followed by DivPrune at the world-to-action KV interface while keeping world-model representation formation dense. This design isolates conditioning compression from representation construction and allows us to study how compression location, candidate filtering, and selection order affect closed-loop task performance under a fixed final token budget.

Efficient Diffusion Models. Token merging, feature caching, and adaptive computation reduce redundancy in iterative generation Bolya & Hoffman (2023); Ma et al. (2024); Liu et al. (2025); Lyu et al. (2025); Ke et al. (2026b). These methods exploit redundancy across tokens, denoising steps, intermediate features, or network depth, thereby avoiding repeated computation that contributes little to the final output. Such redundancy is present in diffusion-based action generators, where multiple denoising steps repeatedly traverse a deep transformer. ActionSkip adapts the input–output-similarity layer-skipping mechanism of Flash-Unified (Ke et al., 2026a) to selected Action-DiT denoising steps. We evaluate this adaptation both alone and together with compressed world-to-action conditioning. This combination lets us separate the effect of action-side depth reduction from world-side token reduction and measure their joint accuracy–latency trade-off using closed-loop manipulation success rather than generative perceptual quality as the downstream criterion.

3 METHOD

3.1 OVERVIEW

SparseWAM operates on the layer-wise KV interface between the Video DiT and Action DiT in Fast-WAM. In standard Fast-WAM inference, clean observation tokens are encoded by the Video DiT; explicit future video tokens are not instantiated (Yuan et al., 2026). We therefore use conditioning tokens to describe the Video-DiT features exposed to action generation, without implying sampled future frames. The Video-DiT hidden sequence remains dense, and only its exported KV tensors are sparsified. An independent action-side mechanism skips selected Action-DiT blocks during designated denoising steps:

Conditioning KV: token sparsity	Action DiT: depth sparsity.

(1)

The KV selector applies instruction-conditioned filtering before diversity selection. ActionSkip adapts the similarity-based layer-skipping mechanism of Ke et al. (2026a); experiments assess it alone and with compressed conditioning. Figure 1 summarizes the resulting inference pipeline.

3.2 CONDITIONING KV SPARSIFICATION

Consider a Video DiT containing L transformer blocks. Let

$$H_l \in \mathbb{R}^{N \times d} \tag{2}$$

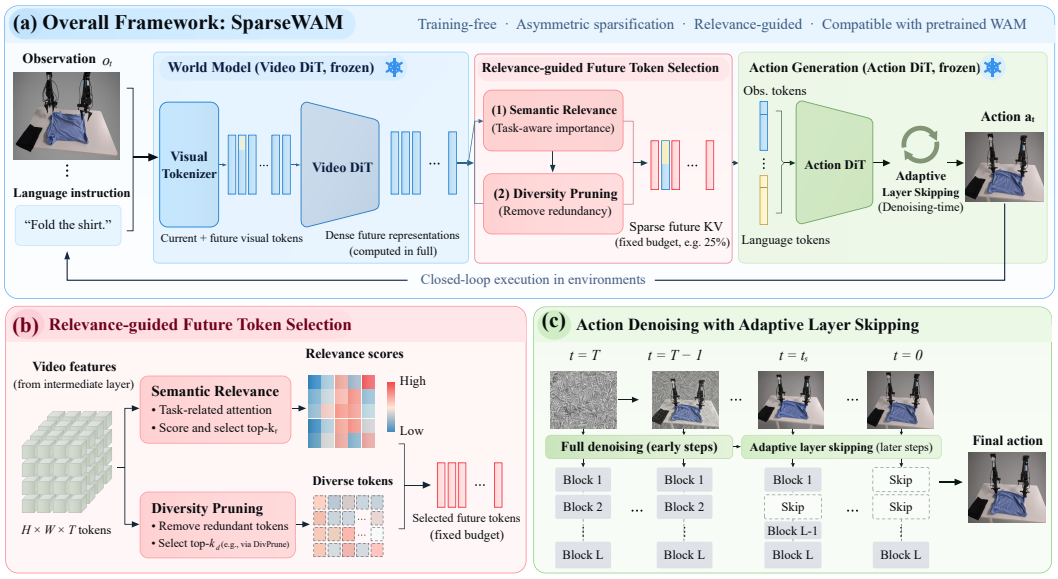

Figure 1: Overview of SparseWAM. (a) Dense Video-DiT computation produces world-model features whose exported conditioning KV is sparsified before action denoising. (b) Instruction-conditioned QK scores restrict the candidate set, and diversity pruning enforces the final KV budget. (c) ActionSkip estimates block similarity at full-depth steps and reuses the selected skip set at designated subsequent steps; final refinement remains full-depth.

denote the hidden representation at layer l, where N is the number of conditioning tokens and d is the hidden dimension. The model constructs cached world-conditioning key/value features:

$$\widetilde{K}_l = \phi_l^K(H_l), \qquad \widetilde{V}_l = \phi_l^V(H_l), \tag{3}$$

where ϕ_l^K and ϕ_l^V denote the corresponding projections exposed to the Action DiT.

A straightforward strategy would prune H_l directly:

$$H_l \to H_l[S] \to H_{l+1}. \tag{4}$$

However, this removes information from subsequent Video-DiT computation.

Instead, we preserve the complete hidden sequence and sparsify only the cached world-conditioning KV:

$$\widetilde{K}'_l = \widetilde{K}_l[S_l], \qquad \widetilde{V}'_l = \widetilde{V}_l[S_l], \tag{5}$$

where K denotes the actual integer token count for the requested retention ratio ρ:

$$S_l \subseteq \{1, \ldots, N\}, \qquad |S_l| = K, \quad 1 \le K \le N. \tag{6}$$

This separates representation formation from compressed world-to-action conditioning. The retained fraction ρ specifies the KV-token budget, not the fraction of model FLOPs or inference time.

Depth-aware grouped selection. To avoid selecting a separate mask at every layer, we use three fixed Video-DiT groups:

$$\mathcal{G}_1 = \{1, \ldots, 10\}, \quad \mathcal{G}_2 = \{11, \ldots, 20\}, \quad \mathcal{G}_3 = \{21, \ldots, 30\}. \tag{7}$$

For group $\mathcal{G}_g$, the hidden representation at the group's final layer $b_g \in \{10, 20, 30\}$ determines its pruning indices:

$$S_g = \mathcal{P}(H_{b_g}). \tag{8}$$

The same mask is shared by all KV representations within the group:

$$\widetilde{K}'_l = \widetilde{K}_l[S_g], \qquad \widetilde{V}'_l = \widetilde{V}_l[S_g], \qquad l \in \mathcal{G}_g. \tag{9}$$

All KV-only sparsification configurations use this grouped design; the separate propagated hidden-token diagnostic is identified in Sec. 4.2. The grouped masks are an implementation choice whose selection granularity is held fixed in the reported comparisons.

3.3 TRAINING-FREE RELEVANCE BEFORE REDUNDANCY

Diversity-based pruning selects tokens by feature complementarity, which does not explicitly condition the selection criterion on the task instruction. We apply a conservative instruction-conditioned candidate filter before DivPrune (Alvar et al., 2025):

$$\boxed{\text{semantic relevance} \rightarrow \text{redundancy reduction.}} \tag{10}$$

For each Video-DiT group $\mathcal{G}_g$, semantic scoring is performed at the group's final layer b_g. Let $Q^V_{b_g} \in \mathbb{R}^{H\times N\times d_h}$ and $K^T_{b_g} \in \mathbb{R}^{H\times M\times d_h}$ denote the Video-query and text-key projections from the existing cross-attention module. They are used only for scoring and do not modify Video-DiT outputs.

Let M be the number of text positions, $\mathcal{M}$ their valid-token indices, and H the number of attention heads. For conditioning token i, we compute:

$$s^{\text{sem}}_{g,i} = \frac{1}{H|\mathcal{M}|}\sum_{h=1}^{H}\sum_{j\in\mathcal{M}}\left\langle q^{(h)}_{b_g,i}, k^{(h)}_{b_g,j}\right\rangle. \tag{11}$$

This score averages raw QK affinities, rather than softmax-normalized attention probabilities. It serves as an instruction-conditioned ranking heuristic; it is not a calibrated estimate of a token's causal contribution to control. No additional scorer or trainable parameter is introduced.

For requested semantic retention ratio ρ_s, let K_s be the actual candidate count, with $1 \le K \le K_s \le N$. We construct:

$$C^{(g)}_{\text{sem}} = \text{TopK}(\{s^{\text{sem}}_{g,i}\}^N_{i=1}, K_s). \tag{12}$$

We then apply diversity pruning within this candidate set. Here, DivPrune denotes selection within the candidate set with the selected indices mapped back to their original Video-DiT token positions:

$$S_g = \text{DivPrune}(H_{b_g}[C^{(g)}_{\text{sem}}], \rho_d). \tag{13}$$

The diversity stage retains K of the K_s candidates. Its effective retention ratio is therefore:

$$\rho_d = \frac{K}{K_s} \approx \frac{\rho}{\rho_s}, \qquad |S_g| = K. \tag{14}$$

Thus, semantic filtering changes token selection rather than token quantity. For ratios $\rho = 0.25$ and $\rho_s = 0.93$, semantic filtering retains approximately 93% of candidates before diversity selection reaches the 25% target budget. The semantic stage thus excludes approximately 7% of candidates; diversity selection performs the reduction. This treats the semantic score as a conservative exclusion criterion rather than using it to select the entire retained set. The resulting original-token indices are shared within each 10-layer group, while Video hidden states remain dense.

3.4 DYNAMIC ACTION-DIT DEPTH SPARSIFICATION

Conditioning KV sparsification reduces the context read by action attention, while repeated Action-DiT evaluation remains a separate source of inference cost. We adapt the dynamic layer-skipping strategy of Ke et al. (2026a) to this action generator.

Let

$$A^{\text{in}}_{t,l}, \qquad A^{\text{out}}_{t,l} \tag{15}$$

denote the input and output representations of Action-DiT block l at denoising step t. We rank candidate blocks using input–output similarity:

$$r_{t,l} = \frac{\langle A^{\text{in}}_{t,l}, A^{\text{out}}_{t,l}\rangle}{\|A^{\text{in}}_{t,l}\|_2\|A^{\text{out}}_{t,l}\|_2}. \tag{16}$$

High similarity identifies skip candidates under this heuristic; preservation of task behavior is evaluated empirically. For the current model, action generation contains ten denoising steps and the

Table 1: Fixed-budget stress test: pruned rows retain 25% of conditioning KV tokens. Speedup is relative to the reproduced dense Fast-WAM baseline.

Method	LIBERO ↑	LIBERO-Plus ↑	Latency (ms) ↓	Speedup ↑
Fast-WAM	96.75	52.00	455.73	1.00×
DivPrune (25% ret.)	89.85	42.83	385.20	1.18×
Semantic → DivPrune (25% ret.)	90.65	44.16	381.54	1.19×
ActionSkip	96.80	51.93	378.40	1.20×
DivPrune (25% ret.) + ActionSkip	90.00	42.50	360.10	1.27×
Semantic → DivPrune (25% ret.) + ActionSkip	90.10	44.22	355.16	1.28×

Action DiT contains 30 layers. We periodically execute the full Action DiT to estimate layer redundancy. At step 1, three layers are selected and skipped at steps 2–3. The skip set is refreshed at step 4 and reused at steps 5–6. Steps 7–10 remain full-depth; similarity is re-estimated at step 7. The two operations act on different dimensions. For a denoising step with skipping enabled:

$$\text{Conditioning KV pruning}: \quad N \to K, \tag{17}$$

$$\text{Action depth pruning}: \quad L_A \to L_A - k. \tag{18}$$

4 Experiments

Experimental Setup. We evaluate SparseWAM on LIBERO and LIBERO-Plus. LIBERO contains Spatial, Object, Goal, and Long suites; LIBERO-Plus introduces Camera, Robot, Language, Lighting, Background, Noise, and Layout shifts. Tables report task success rates, suite-level results where available, and the recorded aggregate scores. Our reproduced Fast-WAM baseline achieves 96.75% on LIBERO and 52.00% on LIBERO-Plus and serves as the dense reference throughout. Unless specified, KV sparsification uses the three-group design and $\rho_s = 0.93$, selected on LIBERO without tuning on LIBERO-Plus. All conclusions concern this Fast-WAM implementation. Latency is measured on a single Alibaba Cloud PPU GPU with BF16 precision, an all-zero 224×448 image, action horizon 32, and 10 denoising steps. Text features are pre-encoded and excluded from timing. CUDA synchronization is applied before and after timed intervals, with 5 warm-up runs and 10 measured runs. VAE encoding, Video prefill, and Action-DiT denoising are profiled separately; Overall latency is measured by a full inference call (Appendix D). These measurements characterize model inference under this protocol, not robot task-completion time.

4.1 Fixed-Budget Token-Selection Stress Test

Table 1 isolates token selection at an aggressive 25% conditioning-KV retention ratio. The final KV budget and grouped design are shared across the pruned configurations; the dense and ActionSkip-only rows provide unpruned reference points. Section 4.5 separately evaluates less aggressive joint configurations. With DivPrune alone, success falls from 96.75% to 89.85% on LIBERO and from 52.00% to 42.83% on LIBERO-Plus. Semantic filtering raises these aggregate scores to 90.65% and 44.16%, gains of 0.80 and 1.33 percentage points at the same KV budget. With ActionSkip, it raises LIBERO-Plus from 42.50% to 44.22%, while LIBERO changes from 90.00% to 90.10%. These comparisons support partial recovery of aggregate performance under aggressive pruning, not equivalence to the dense baseline.

ActionSkip alone retains similar aggregate success to the dense model and reduces latency from 455.73 ms to 378.40 ms. The semantic-guided 25% joint configuration reaches 355.16 ms, an additional 23.24 ms reduction over ActionSkip alone, with lower success on both benchmarks. Thus, the strongest latency reduction in this table comes with a substantial accuracy cost; the selector comparison and the system-level trade-off answer different questions.

4.2 Where and How Aggressively Should Conditioning Be Compressed?

Figure 2 distinguishes compression location from retention level. First, DivPrune achieves 95.75% LIBERO success at 50% KV retention, compared with 89.85% at 25%. Second, at 25% retention, pruning input visual tokens before Video-DiT computation yields 24.90% on LIBERO and 2.66% on LIBERO-Plus, whereas exported-KV pruning yields 89.85% and 42.83%. The latter preserves

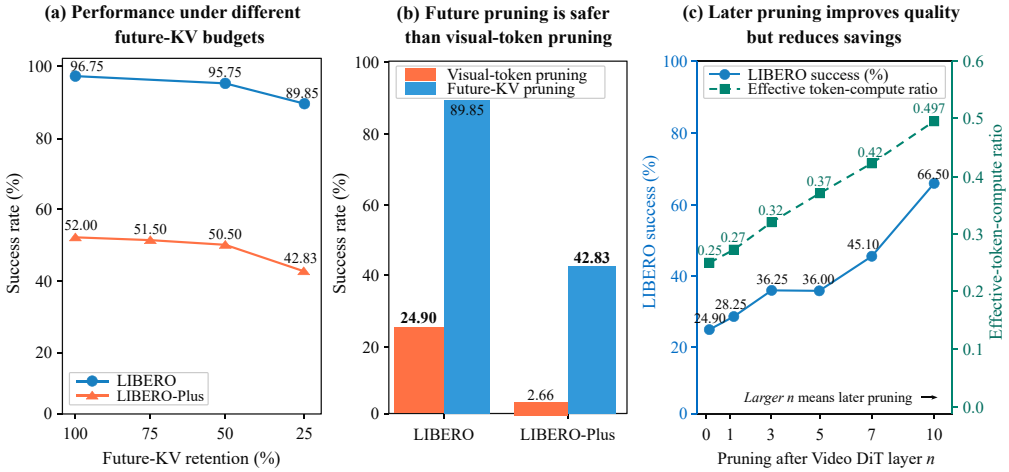

Figure 2: Properties of conditioning-token sparsification. (a) Performance under different conditioning-KV budgets. (b) Exported conditioning-KV pruning versus input-token pruning. (c) Diagnostic hidden-token pruning at different Video-DiT depths; unlike SparseWAM, this sweep propagates the shortened sequence through subsequent Video-DiT computation.

Table 2: Ablation of relevance-guided pruning order under a fixed 25% conditioning-KV budget.

Conditioning-token selection	LIBERO	LIBERO-Plus
Semantic only	74.35	37.85
DivPrune only	89.85	42.83
DivPrune $\rightarrow$ Semantic	88.85	43.49
Semantic $\rightarrow$ DivPrune	90.65	44.16

the full representation-formation process and compresses only the action generator's access to its outputs. These are matched retention ratios, not matched computational savings. Third, delaying propagated hidden-token pruning generally improves LIBERO success while leaving more Video-DiT computation intact. This depth sweep shortens the sequence processed by subsequent Video-DiT layers and is distinct from SparseWAM's KV-only mechanism. Its effective-token-compute ratio is a diagnostic proxy, not a measured latency or FLOPs ratio. Together, these diagnostics motivate preserving representation formation while compressing the action-facing interface.

4.3 Candidate Filtering Strength and Selection Order

We fix the final conditioning-KV budget at 25% and vary the semantic candidate-set size (Fig. 3; Appendix B). The highest LIBERO aggregate score in this sweep is 90.65% at $\rho_s = 0.93$, compared with 89.85% for DivPrune alone; we use this setting without tuning on LIBERO-Plus. The measured LIBERO-Plus maximum occurs at $\rho_s = 0.91$ (44.75%), which is not used to select the default. At $\rho_s = 0.25$, selection becomes semantic-only: LIBERO success falls to 74.35%, and LIBERO-Long falls from 81.20% at $\rho_s = 1.0$ to 43.60%. This supports using semantic relevance as a mild candidate filter while retaining diversity-based selection.

Table 2 compares the two stage orders at the same final KV budget. Semantic$\rightarrow$DivPrune yields 90.65%/44.16% on LIBERO/LIBERO-Plus, compared with 88.85%/43.49% for the evaluated reverse-order configuration. The aggregate gain is not uniform across tasks or shifts. At the default setting, LIBERO-Long decreases from 81.20% to 79.40%, and Camera, Noise, and Layout success also decrease relative to DivPrune alone, while the other four LIBERO-Plus categories improve (Appendix C). Accordingly, the evidence supports improved aggregate selection under this stress test, not general protection of long-horizon or all out-of-distribution behavior.

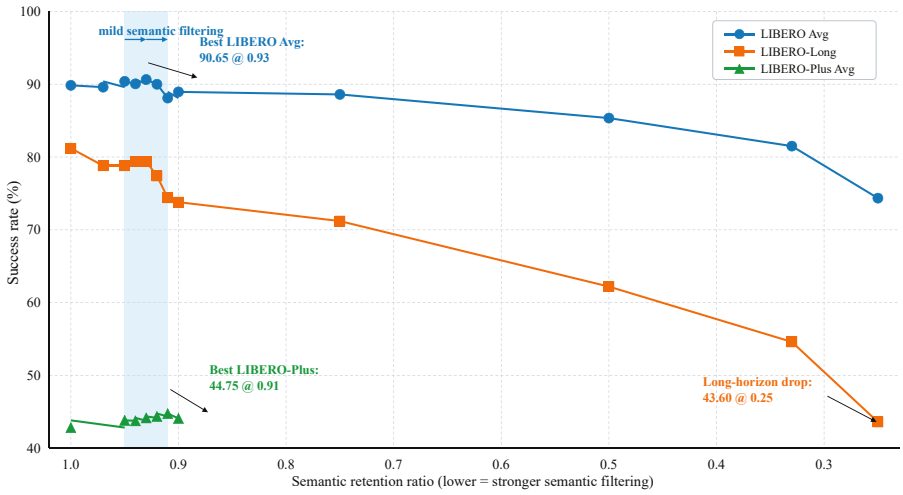

Figure 3: Semantic relevance and diversity exhibit a non-monotonic trade-off under a fixed 25% final conditioning-KV budget. Mild semantic filtering improves average conditioning-token selection, whereas overly aggressive semantic filtering is associated with a degradation on long-horizon tasks under the same final token-retention constraint.

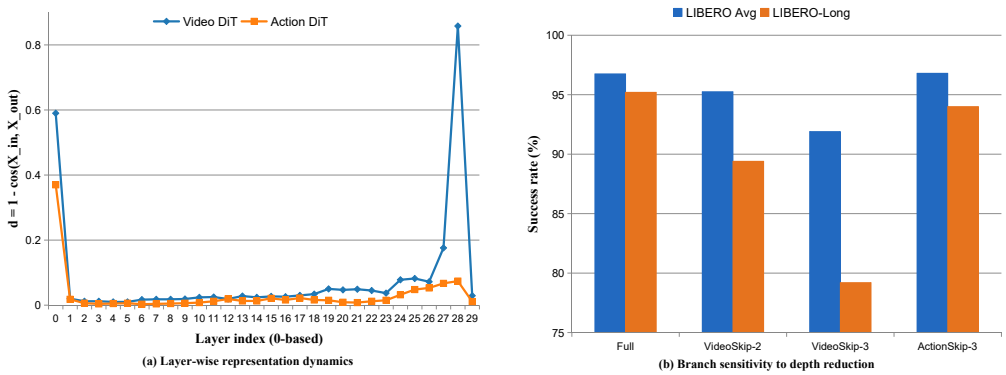

Figure 4: Branch-specific depth-reduction diagnostics. (a) Layer-wise input–output representation change. (b) Task success under Video-DiT terminal-layer truncation and intermittent similarity-based ActionSkip; these interventions are not compute-matched.

4.4 Branch-Specific Depth-Reduction Diagnostics

We measure layer-wise representation change using $d = 1 - \cos(X_{\text{in}}, X_{\text{out}})$. Figure 4(a) shows larger changes at several Video-DiT layers than in the Action DiT under the evaluated setting. This is a descriptive similarity diagnostic; a small representation change alone does not establish that a block is dispensable for control. Truncating the last two Video-DiT layers yields 95.25% LIBERO success; truncating the last three yields 91.90%, including 79.20% on LIBERO-Long. The evaluated ActionSkip configuration retains 96.80% LIBERO and 94.00% LIBERO-Long success. The interventions differ in both selection rule and execution frequency: VideoSkip removes terminal layers, whereas ActionSkip selects blocks by similarity and skips them only at designated denoising steps. This comparison characterizes the tested interventions rather than isolating an intrinsic branch asymmetry at matched compute. It motivates retaining dense Video-DiT computation while evaluating selective action-side depth reduction in the joint system.

4.5 Efficiency–Performance Trade-off

Table 3 evaluates joint configurations across KV budgets, with ActionSkip alone as a strong reference. The 75% joint configuration achieves 96.50% LIBERO and 52.26% LIBERO-Plus success,

Table 3: Accuracy–efficiency trade-off under different conditioning-KV retention ratios. KV-pruned rows use Semantic→DivPrune.

Configuration	LIBERO	LIBERO-Plus	Latency (ms)
Full Fast-WAM	96.75	52.00	455.73
ActionSkip only	96.80	51.93	378.40
75% Sem→Div KV + ActionSkip	96.50	52.26	387.26
50% Sem→Div KV + ActionSkip	95.00	50.66	369.20
25% Sem→Div KV + ActionSkip	90.10	44.22	355.16

Table 4: Controlled real-robot shirt folding over 10 trials per configuration. Completion time is averaged over successful trials and reported only as a behavioral statistic.

KV Ret.	ActionSkip	Success	Time
100%	✗	100%	1:03
75%	✗	100%	1:00
50%	✗	90%	1:03
25%	✗	80%	1:44
25%	✓	60%	1:08

but its measured 387.26 ms latency exceeds ActionSkip alone at 378.40 ms. The 50% joint configuration reaches 369.20 ms, reducing latency by 19.0% relative to dense Fast-WAM. Relative to ActionSkip alone, its additional reduction is 9.20 ms (2.43%), with success decreasing by 1.80 percentage points on LIBERO and 1.27 on LIBERO-Plus. This is an intermediate measured operating point for applications that accept that accuracy cost. Reducing retention from 50% to 25% saves another 14.04 ms but lowers LIBERO/LIBERO-Plus success from 95.00%/50.66% to 90.10%/44.22%. The retention sweep therefore exposes diminishing latency gains alongside larger accuracy losses. The semantic selector is isolated at 25% in Table 1; the 50% joint result does not independently establish its contribution at that budget.

4.6 Real-Robot Evaluation

We evaluate compression in a controlled real-robot shirt-folding setting with ten trials per configuration (Table 4; Appendix E). The dense baseline and 75% KV retention without ActionSkip each succeed in 10/10 trials; 50% and 25% retention without ActionSkip succeed in 9/10 and 8/10. The 25% joint configuration succeeds in 6/10 trials. These small-sample outcomes illustrate the potential execution cost of aggressive joint compression, rather than establishing a statistically resolved difference between configurations. This study does not compare semantic and diversity-only selectors, and it does not evaluate either ActionSkip alone or the 50% joint configuration. Consequently, it cannot establish a real-robot advantage for the semantic stage or validate the deployment trade-off of the 50% joint setting. Completion times are conditional on success and exclude failed trials; they describe successful executions and do not demonstrate reduced physical task-completion time.

5 Conclusion

SparseWAM combines relevance-guided sparsification of the world-to-action KV interface with an adaptation of dynamic action-side layer skipping. On Fast-WAM, preserving dense representation formation is less destructive than early visual-token removal at the evaluated retention ratios, and mild semantic filtering partially recovers aggregate performance at an aggressive KV budget. The joint retention sweep makes its incremental latency savings and accuracy costs explicit relative to ActionSkip alone. Controlled shirt-folding trials further illustrate why model inference efficiency must be assessed alongside execution reliability.

Scope and limitations. The reported evidence covers one WAM implementation, aggregate point estimates without uncertainty intervals across repeated evaluations, and a small controlled real-robot study. Random candidate filtering and matched-latency alternative action schedules are not evaluated; the observed semantic gains therefore do not by themselves identify a causal mechanism or a generally optimal compression policy.

AI USE STATEMENT

Generative AI tools assisted with language editing, organization, and checks of consistency between claims and results. The authors are responsible for the methods, experiments, and manuscript.

ETHICS STATEMENT

This work evaluates manipulation policies in simulation and on physical robot platforms. It does not involve humans or personal data. All real-robot experiments were conducted in laboratory settings.

REPRODUCIBILITY STATEMENT

The main paper specifies the SparseWAM design, the instruction-guided semantic-scoring rule, fixed-budget semantic/diversity pruning, the grouped conditioning-KV strategy, the Action-DiT skipping schedule, and the unified latency protocol. The appendix reports the raw measurements underlying the main conditioning-pruning analyses, semantic-retention sweeps, available per-perturbation LIBERO-Plus results, latency decomposition, and controlled real-robot trial logs.

REFERENCES

Saeed Ranjbar Alvar, Gursimran Singh, Mohammad Akbari, and Yong Zhang. Divprune: Diversity-based visual token pruning for large multimodal models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9392–9401, 2025.

Daniel Bolya and Judy Hoffman. Token merging for fast stable diffusion. In *CVPR Workshop on Efficient Deep Learning for Computer Vision*, 2023.

Daniel Bolya, Cheng-Yang Fu, Xiaoli Dai, Judy Hoffman, and Christoph Feichtenhofer. Token merging: Your vit but faster. In *International Conference on Learning Representations*, 2023.

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

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 19–35, 2024. doi: 10.1007/978-3-031-73004-7_2.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, Jinlan Fu, Jingjing Gong, and Xipeng Qiu. Libero-plus: In-depth robustness analysis of vision-language-action models. *arXiv preprint arXiv:2510.13626*, 2025.

Titong Jiang, Xuefeng Jiang, Yuan Ma, Xin Wen, Bailin Li, Kun Zhan, Peng Jia, Yahui Liu, Sheng Sun, and Xianpeng Lang. The better you learn, the smarter you prune: Towards efficient vision-language-action models via differentiable token pruning. In *IEEE International Conference on Robotics and Automation*, 2026.

Junlong Ke, Zichen Wen, Boxue Yang, Yantai Yang, Xuyang Liu, Chenfei Liao, Zhaorun Chen, Shaobo Wang, and Linfeng Zhang. Flash-unified: A training-free and task-aware acceleration framework for native unified models. *arXiv preprint arXiv:2603.15271*, 2026a.

Junlong Ke, Zichen Wen, Boxue Yang, Yantai Yang, Xuyang Liu, Chenfei Liao, Zhaorun Chen, Shaobo Wang, and Linfeng Zhang. Flash-Unified: A training-free and task-aware acceleration framework for native unified models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Findings*, pp. 9131–9142, 2026b.

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

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 *International Conference on Learning Representations*, 2026b.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. Libero: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 44776–44791, 2023.

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 the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7353–7363, 2025.

Ziyan Liu, Yeqiu Chen, Hongyi Cai, Tao Lin, Shuo Yang, Zheng Liu, and Bo Zhao. Bridging the semantic-action gap in visual token pruning for efficient VLA inference. *arXiv preprint arXiv:2511.16449*, 2026.

Zhengyao Lyu, 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. In *International Conference on Learning Representations*, 2025.

Liheng Ma, Rui Heng Yang, Zhanguang Zhang, Mateo Clemente, Ziwen Hu, Tongtong Cao, and Yingxue Zhang. Faster-wam: Do world action models need deep action modules? *arXiv preprint arXiv:2608.02365*, 2026.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. DeepCache: Accelerating diffusion models for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15762–15772, 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.

Qizhe Zhang, Mengzhen Liu, Lichen Li, Ming Lu, Yuan Zhang, Junwen Pan, Qi She, and Shanghang Zhang. Beyond attention or similarity: Maximizing conditional diversity for token pruning in MLLMs. *arXiv preprint arXiv:2506.10967*, 2025a. URL https://arxiv.org/abs/2506.10967.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis A. Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, and Shanghang Zhang. Sparsevlm: Visual token sparsification for efficient vision-language model inference. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pp. 74840–74857, 2025b.

Weiheng Zhao, Haoyi Jiang, Xin Shi, Liu Liu, Fan Huang, Zhizhong Su, Wei Sui, and Xinggang Wang. Faster-wam: Efficient inference-time future conditioning for robust world action models. *arXiv preprint arXiv:2608.04404*, 2026.

A Implementation Details and Compression Location

Grouped conditioning-KV sparsification. The Video DiT contains 30 layers and is partitioned into three groups, $\mathcal{G}_1 = \{1, \ldots, 10\}$, $\mathcal{G}_2 = \{11, \ldots, 20\}$, and $\mathcal{G}_3 = \{21, \ldots, 30\}$. Semantic and diversity-based token selection is performed at layers 10, 20, and 30, respectively, and the selected indices are reused by all world-conditioning K/V tensors within the corresponding group. Crucially, the Video-DiT hidden sequence itself remains dense throughout prefill; sparsification is applied only to the cached world-conditioning K/V consumed by the Action DiT.

Instruction-guided scoring. At the end of each group, we reuse the existing cross-attention projections: Video queries are multiplied with text keys, invalid/padding text tokens are masked out, and the resulting affinities are averaged over valid text tokens and attention heads. The resulting scalar score ranks Video tokens for the semantic candidate set. No additional scorer or trainable parameter is used.

Action-side schedule. The Action DiT contains 30 layers and performs 10 denoising steps. Step 1 executes all layers and selects three high-similarity blocks; steps 2–3 skip those blocks. Step 4 refreshes the skip set and steps 5–6 reuse it. Step 7 returns to full depth and re-estimates redundancy, while steps 8–10 remain full depth for final refinement.

A.1 Input Visual Tokens vs. World-to-Action Conditioning

Table 5 reports the raw input-visual pruning results used in Fig. 2(b). Pruning the current visual tokens before Video-DiT computation is substantially more destructive than sparsifying the world-conditioning representation.

Table 5: Direct input-visual token pruning with DivPrune.

Visual-token retention	LIBERO	LIBERO-Plus
75%	95.45	47.61
50%	89.25	34.20
25%	24.90	2.66

A.2 Effect of Pruning Depth

To isolate the effect of where pruning is introduced, we keep all tokens up to Video-DiT layer n and retain 25% of tokens afterward. Table 6 reports the corresponding LIBERO results and the effective-token-compute proxy used in Fig. 2(c); this proxy is not a measured FLOPs or latency ratio.

Table 6: Effect of delaying 25%-retention pruning to deeper Video-DiT layers.

Layer n	Eff. ratio	Spatial	Object	Goal	Long	Avg.
0	0.250	43.00	14.80	35.60	6.20	24.90
1	0.270	44.80	34.20	23.20	10.80	28.25
3	0.320	42.20	53.20	31.20	18.40	36.25
5	0.370	33.40	47.60	40.20	22.80	36.00
7	0.420	50.60	54.40	47.40	28.00	45.10
10	0.497	71.40	81.60	68.20	44.80	66.50

B Semantic-Retention Details

All rows in this section use a fixed final conditioning-KV retention ratio of 25%. The semantic-retention ratio ρ_s changes only the size of the candidate set passed to DivPrune. The nominal second-stage retention is $\rho_d = 0.25/\rho_s$; with integer token counts, the actual ratio is K/K_s, as in Eq. 14. The default $\rho_s = 0.93$ is selected on LIBERO and is not tuned on LIBERO-Plus.

C LIBERO-Plus Per-Perturbation Results

Table 9 reports the per-perturbation values available for the main conditioning-sparsification operating points. We report only configurations for which the full seven-shift breakdown was recorded; aggregate-only settings are not expanded with inferred values.

D Latency Breakdown

All values in Table 10 follow the unified timing protocol described in Sec. 4: a fixed single Alibaba Cloud PPU GPU, BF16 precision, a synthetic 224×448 all-zero image, action horizon 32, 10 denoising steps, pre-encoded text features, CUDA synchronization immediately before and after each timed interval, 5 untimed warm-up runs, and 10 measured runs. Each component is profiled

Table 7: Complete LIBERO semantic-retention sweep under a fixed 25% final conditioning-KV budget.

ρ_s	Spatial	Object	Goal	Long	Avg.
1.00	95.80	93.80	88.60	81.20	89.85
0.97	95.00	95.20	89.40	78.80	89.60
0.95	96.20	96.40	90.20	78.80	90.40
0.94	95.80	95.20	89.80	79.40	90.05
0.93	96.00	97.00	90.20	79.40	90.65
0.92	95.60	96.20	90.80	77.40	90.00
0.91	94.00	96.20	87.80	74.40	88.10
0.90	95.00	96.80	90.20	73.80	88.95
0.75	92.00	97.80	93.40	71.20	88.60
0.50	95.20	95.80	88.20	62.20	85.35
0.33	92.20	96.20	83.00	54.60	81.50
0.25	90.60	89.40	73.80	43.60	74.35

Table 8: Available LIBERO-Plus semantic-retention sweep under the same fixed 25% final conditioning-KV budget. Only measured operating points are reported.

ρ_s	Camera	Robot	Lang.	Light	Backg.	Noise	Layout	Avg.
1.00	19.58	34.85	65.98	67.90	36.76	28.40	53.18	42.83
0.95	20.42	40.40	65.98	70.37	41.62	26.46	50.45	43.82
0.94	20.83	41.41	69.26	69.75	38.92	25.68	48.64	43.76
0.93	19.17	42.42	67.62	70.37	40.00	27.63	50.45	44.16
0.92	17.92	41.41	69.67	70.99	37.84	29.18	51.36	44.36
0.91	19.17	41.92	68.03	70.37	36.22	29.96	55.00	44.75
0.90	17.50	42.42	68.44	69.14	36.76	27.63	54.55	44.09

separately. The Overall column is measured by an additional complete `infer_action` call and is therefore not the arithmetic sum of the three component columns. Only point estimates are reported, so the small timing differences between selectors should not be interpreted as established ordering beyond this protocol.

E Controlled Real-Robot Trial Logs

We use the controlled shirt-folding setting reported in the main paper, with 10 trials per configuration. Table 11 lists each trial outcome and completion time. Times are reported only for successful trials.

Additional garment baseline. For completeness, the dense PyTorch baseline was also evaluated for ten trials on four garment categories. The recorded success rates were 90% for shirts, 100% for skirts, 90% for pants, and 90% for towels, with mean successful episode durations of 1:59, 1:35, 1:13, and 1:34, respectively. We report these results only as supplementary context because they were not collected under the same controlled shirt-only sweep used for the main compression comparison.

Table 9: Per-perturbation LIBERO-Plus success rates for measured operating points. KV-pruned rows with ActionSkip use Semantic→DivPrune.

Configuration	Camera	Robot	Lang.	Light	Backg.	Noise	Layout	Avg.
Full Fast-WAM	26.30	41.90	70.10	84.00	44.90	46.30	58.20	52.00
DivPrune, 25% KV	19.58	34.85	65.98	67.90	36.76	28.40	53.18	42.83
Sem→Div, 25% KV	19.17	42.42	67.62	70.37	40.00	27.63	50.45	44.16
Sem→Div, 75% KV + ActionSkip	27.08	48.48	70.49	82.10	43.24	39.69	63.18	52.26
Sem→Div, 50% KV + ActionSkip	27.50	44.44	69.67	79.63	44.86	34.24	63.18	50.66
Sem→Div, 25% KV + ActionSkip	17.92	44.44	69.26	70.37	37.84	27.63	50.45	44.22

Table 10: Latency decomposition in milliseconds under the unified protocol.

Configuration	VAE Enc.	Video Prefill	Action	Overall
Full Fast-WAM	13.70	33.58	379.66	455.73
DivPrune, 25% KV	13.26	38.92	317.54	385.20
Sem→DivPrune, 25% KV	13.21	41.39	313.88	381.54
ActionSkip only	13.25	33.40	322.13	378.40
DivPrune, 25% KV + ActionSkip	13.16	37.68	293.50	360.10
Sem→DivPrune, 25% KV + ActionSkip	12.94	38.44	291.29	355.16

Table 11: Controlled shirt-folding trial logs. "F" denotes failure.

Configuration	1	2	3	4	5	6	7	8	9	10
100% KV, no skip	0:53	1:04	1:08	1:01	1:02	1:09	1:04	1:00	1:00	1:07
75% KV, no skip	1:02	1:00	0:57	0:59	1:07	0:59	0:57	0:58	0:59	0:59
50% KV, no skip	1:03	1:00	1:02	1:06	1:07	1:05	1:03	1:01	1:04	F
25% KV, no skip	1:07	1:20	2:38	1:11	2:21	F	F	1:05	2:58	1:10
25% KV + ActionSkip	1:17	1:10	0:49	1:27	F	F	F	F	0:58	1:09

Table 12: Aggregate controlled shirt-folding results computed from the trial logs in Table 11. Completion time is averaged over successful trials only.

Configuration	Success	Avg. time
100% KV, no skip	100%	1:03
75% KV, no skip	100%	1:00
50% KV, no skip	90%	1:03
25% KV, no skip	80%	1:44
25% KV + ActionSkip	60%	1:08