ADAMEM: ACTION-UTILITY GUIDED MEMORY MANAGEMENT FOR BUDGETED LONG-HORIZON VISION-LANGUAGE-ACTION POLICIES

Anonymous authors
Paper under double-blind review

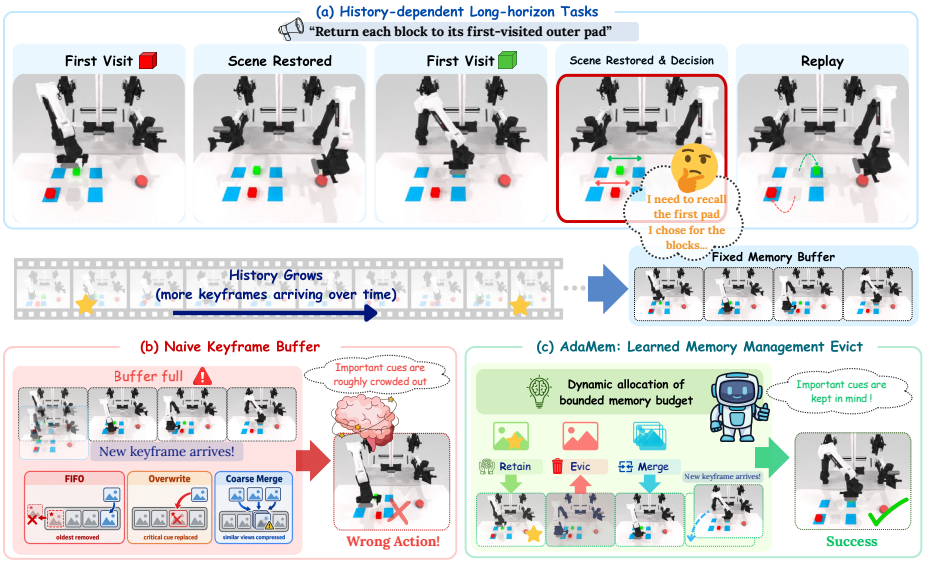


Figure 1: **Motivation for budgeted memory management.** History-dependent tasks require recalling earlier evidence. Fixed-buffer heuristics may discard useful information as keyframes arrive, whereas ADAMEM selects retain, merge, and evict plans using task-conditioned action utility.

ABSTRACT

Long-horizon robotic manipulation requires preserving task-relevant information beyond the current observation, yet VLA policies operate under limited temporal context and deployment resources. Existing event-memory methods mainly focus on memory admission, while post-admission maintenance is often handled by simple replacement heuristics. We introduce ADAMEM, an action-utility guided memory manager that learns to maintain a bounded episodic memory with a frozen VLA backbone. For each incoming event, ADAMEM evaluates feasible retain, merge, and evict operations according to their predicted effect on future action quality, using counterfactual future-action supervision. ADAMEM-SELECT learns budget-aware raw-frame selection, while ADAMEM-FULL further supports compact representations, temporally ordered merging, and gated latent-memory integration. We also provide a finite-candidate regret analysis and evaluate both offline action-proxy quality and closed-loop performance. Across eight RoboTwin-MeM tasks, ADAMEM consistently improves performance under matched memory budgets. At $K = 3$, ADAMEM-FULL achieves 90.1% closed-loop success versus 80.9% for Raw-FIFO. These results show that learned memory maintenance can improve long-horizon VLA execution without modifying the action backbone.

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.

1 INTRODUCTION

Long-horizon robotic manipulation requires preserving task-relevant information beyond the current observation. Such evidence may be transient: an object configuration can determine execution order, an interaction may reveal a hidden target, or a completed action may establish a correspondence that later becomes visually unobservable. As a result, visually similar states can require different actions depending on prior events.

Vision-language-action (VLA) models map visual observations and language instructions to executable actions (Brohan et al., 2023; Kim et al., 2024; Team et al., 2024; Li et al., 2024; Intelligence et al., 2025). Recent memory-augmented VLA policies extend this capability with event tokens, episodic embeddings, or structured visual memories (Koo et al., 2026; Shi et al., 2026; Yang et al., 2026; Zeng et al., 2026). However, existing methods mainly study how memories are represented, admitted, or retrieved, while paying less attention to how a bounded memory should be maintained as new events arrive and task relevance changes over time.

Figure 1 illustrates this challenge. In the shown task, each block must be returned to the first outer pad it previously visited. After the scene is restored, the current observation no longer reveals the original block–pad correspondence. A bounded memory must therefore preserve early but still relevant evidence while accommodating later events, making simple age-based replacement insufficient.

We introduce **AdaMem** (Adaptive Memory Management), a lightweight controller for online bounded episodic memory maintenance. For each incoming event, AdaMem evaluates feasible retain, reject, replace, and merge operations using a learned utility estimator that predicts their effect on future action quality. The VLA backbone remains frozen, and the controller is trained offline with counterfactual future-action supervision.

We instantiate two variants. **AdaMem-Select** performs budget-aware admission and eviction while retaining the original raw-frame memory interface. **AdaMem-Full** further supports compact event representations, temporally ordered pairwise merging, and gated latent-memory integration. Both are evaluated under matched context-token and resident-memory budgets.

For the finite online candidate set, the learned utility estimator admits a regret bound of $2\epsilon + \delta$ under bounded estimation error. Because this guarantee applies only to the proxy objective, we additionally evaluate counterfactual ranking quality and closed-loop manipulation performance.

We also reproduce the released EventVLA B3 checkpoint (Yang et al., 2026) on all eight RoboTwin-MeM tasks with 100 episodes per task, obtaining 647 successes out of 800 episodes (80.875% macro success), compared with 77.5% in the updated official report. This provides a common reference for the budget-matched experiments.

Our contributions are:

- **Action-utility guided memory management.** We formulate bounded memory maintenance as structured selection over candidate memory states and use counterfactual future-action supervision to guide retain, merge, and evict decisions with a frozen VLA backbone.
- **Budget-aware mixed-fidelity memory.** We introduce AdaMem-Select for raw-frame selection and AdaMem-Full for mixed-fidelity memory with compact representations, temporally ordered merging, and gated latent integration, together with a finite-candidate regret bound.
- **Controlled evaluation.** Across eight RoboTwin-MeM tasks, AdaMem improves closed-loop success under matched budgets while reducing context usage, resident memory, and policy latency.

2 RELATED WORKS

2.1 Long-Horizon VLA Policies and History-Dependent Manipulation

Generalist robot policies such as RT-1, RT-2, PaLM-E, Octo, OpenVLA, CogACT, and $\pi_{0.5}$ connect visual observations and language instructions to scalable action generation (Brohan et al., 2023; Driess et al., 2023; Team et al., 2024; Kim et al., 2024; Li et al., 2024; Intelligence et al., 2025).

Long-horizon manipulation further introduces partial observability, because task-relevant evidence may disappear after scene changes and visually similar observations can require different future actions (Chen et al., 2026a; Zeng et al., 2026). Hierarchical policy orchestration and multi-stage planning provide complementary mechanisms for long-horizon execution (Hu et al., 2026; Peng et al., 2026; Huang et al., 2026; Feng et al., 2025; Jung et al., 2025), but do not directly address bounded episodic-memory maintenance.

2.2 Memory-Augmented VLA Policies

Recent memory-augmented robot policies explicitly preserve information beyond the current observation window. HAMLET, μVLA, ReMem-VLA, and VPWEM propagate recurrent, working-memory, or compressed historical representations (Koo et al., 2026; Cherepanov et al., 2026; Li et al., 2026; Lei et al., 2026). MemoryVLA and MEM introduce structured multi-scale memory, while SAM2Act couples a visual foundation model with a dedicated memory architecture for manipulation (Shi et al., 2026; Torne et al., 2026; Fang et al., 2025). EventVLA and KEMO emphasize event-driven or keyframe-based admission, whereas BPP, Keyframe-Chaining, and MemER select, chain, or retrieve task-relevant history (Yang et al., 2026; Zeng et al., 2026; Mark et al., 2026; Chen et al., 2026b; Sridhar et al., 2025). RMBench further evaluates tasks whose correct actions depend on information absent from the current observation (Chen et al., 2026a). Most of these methods focus on representation, admission, or retrieval.

2.3 Bounded and Resource-Aware Memory Management

Learned external memory has been widely studied in sequence modeling and partially observable control. Neural Turing Machines and Differentiable Neural Computers introduced differentiable external memory (Graves et al., 2014; 2016), while MERLIN used predictive representation learning to support memory-based control under partial observability (Wayne et al., 2018). Transformer-XL, Compressive Transformers, Memorizing Transformers, and LongMem extend usable context through recurrence, compression, or retrieval from stored representations (Dai et al., 2019; Rae et al., 2020; Wu et al., 2022; Wang et al., 2023).

Language-agent systems such as MemoryBank, Generative Agents, MemGPT, Mem0, and A-MEM maintain persistent memories through storage, retrieval, summarization, or structured updates (Zhong et al., 2023; Park et al., 2023; Packer et al., 2023; Chhikara et al., 2025; Xu et al., 2025b). Efficiency-oriented methods instead manage limited inference context: H2O retains important KV states, StreamingLLM combines attention sinks with a sliding cache, and VLA-Cache reuses slowly changing visual tokens (Zhang et al., 2023; Xiao et al., 2024; Xu et al., 2025a). Their objectives mainly concern language memory or inference efficiency rather than embodied episodic evidence and downstream control quality. AdaMem applies resource-aware maintenance directly to embodied memory. It evaluates feasible post-admission memory states with counterfactual future-action supervision and operates under explicit context-token and resident-memory budgets, while keeping the underlying VLA backbone frozen.

3 METHODS

3.1 Problem setup and resource model

At time t, a language-conditioned VLA policy receives the current observation o_t, proprioception s_t, instruction l, fixed visual anchors V_t, and a dynamic event bank M_t. It predicts an action chunk

$$A_t = \pi_\theta(o_t, s_t, l, V_t, M_t), \tag{1}$$

where the backbone parameters θ are frozen in the core experiments. An event proposal module supplies a candidate e_t only when the underlying event detector fires. Each event stores a single-frame feature, acquisition time, detector confidence, a compact proprioceptive summary, and either a raw image or a latent representation.

We impose two budgets rather than measuring only the number of frames:

$$C_{\mathrm{ctx}}(M_t) \le B_{\mathrm{ctx}}, \qquad C_{\mathrm{store}}(M_t) + C_{\mathrm{aux}} \le B_{\mathrm{store}}. \tag{2}$$

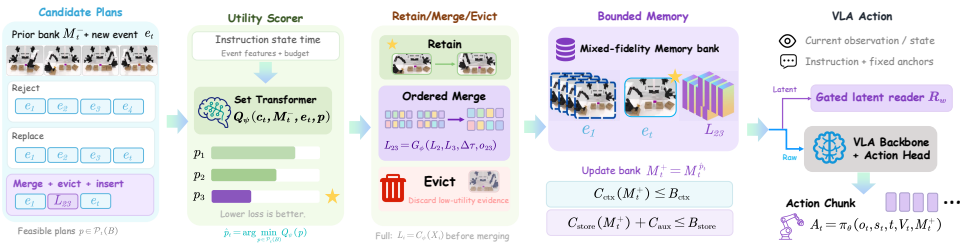


Figure 2: **Overview of the AdaMem method.** AdaMem scores candidate memory-update plans to adaptively retain, merge, or evict events under a bounded budget, maintaining a mixed-fidelity memory bank that is accessed through a gated latent reader to condition VLA action prediction.

C_{ctx} counts the visual and latent tokens consumed by the next policy call, while C_{store} counts resident CPU/GPU bytes. Auxiliary metadata and manager state are included in C_{aux}; deleted images and debugging logs cannot be used as hidden memory. The implementation always includes an empty-bank fallback, so every update is budget-feasible.

3.2 Online Memory Lifecycle

When an event arrives, AdaMem constructs a finite set $\mathcal{P}_t(B)$ of legal plans. A plan specifies which old events are retained, whether the new event is rejected or inserted, and (for AdaMem-Full) whether one ordered pair is merged into a fixed-size latent. The manager evaluates complete candidate banks rather than assigning independent scores to individual frames:

$$\widehat{p}_t = \arg\min_{p \in \mathcal{P}_t(B)} Q_\psi(c_t, M_t^-, e_t, p), \qquad M_t^+ = M_t^{\widehat{p}_t}, \tag{3}$$

where c_t contains only the instruction, current robot state, relative time, and detached context available at t. The manager runs once per event proposal and does not add a VLA forward pass at every low-level control tick.

Figure 3 gives an overview of the two instantiations of this framework. Both variants share the same event proposal mechanism and frozen VLA policy, but differ in the operation space available to the memory manager and in the fidelity at which past events are retained. The following sections describe these two designs in detail.

3.3 AdaMem-Select: Learned Raw-Frame Replacement

As illustrated in Figure 3, AdaMem-Select restricts memory management to full-fidelity raw observations while preserving the original EventVLA memory interface. Its operation space contains only raw-frame admission and replacement: when the bank is full, the incoming event can either be rejected or replace one resident frame. Consequently, every retained memory slot corresponds to an original observation. If the bank has fewer than K frames, the only candidate is acceptance. If it is full, the candidates are

$$\mathcal{P}_t = \{p_{\text{reject}}, p_{\text{replace}}(1), \ldots, p_{\text{replace}}(K)\}. \tag{4}$$

The replacement candidates remove one old frame and append the new frame while preserving acquisition-time order. This joint state view captures complementarity: two observations that are weak in isolation may jointly identify an object–goal correspondence.

Each event token contains a frozen single-frame visual feature, relative age, event confidence, a state summary, and representation cost. Task and budget tokens are appended to the event set. A two-layer, four-head set Transformer with width 256 and feed-forward width 1024 produces a scalar utility-loss estimate for every candidate bank. Bank indices are not semantic positions; time embeddings carry ordering information and padding is masked. The manager selects the lowest predicted loss.

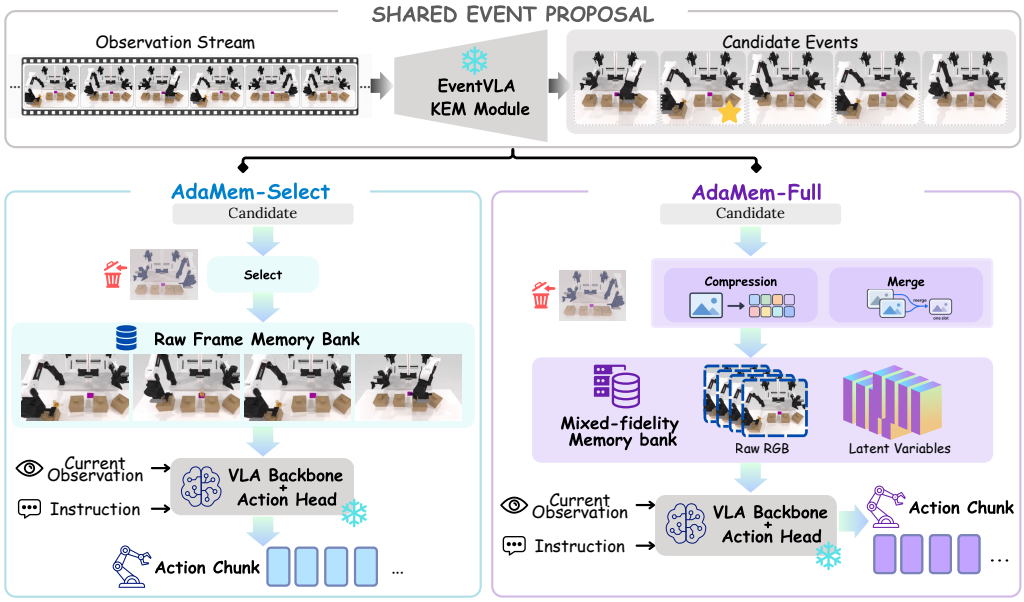


Figure 3: **Comparison of AdaMem-Select and AdaMem-Full under a shared frozen VLA policy.** AdaMem-Select operates on a full-fidelity raw-frame memory using admission and replacement decisions, whereas AdaMem-Full extends the decision space to mixed-fidelity memory with raw retention, single-event compression, and temporally ordered pairwise merging.

3.4 Counterfactual future-action supervision

The manager is trained from the action consequence of a memory choice. For a decision point t, let $\mathcal{Q}_t$ be the next valid policy queries (default offsets 1, 2, 4, and 8). For every candidate p, we keep the observed image, proprioception, instruction, and fixed anchors at each query unchanged, and replace only the dynamic bank by M_t^p. During this label window no new dynamic event is written and deleted frames cannot be restored. With expert action target a_q^*,

$$J_t(p) = \frac{\sum_{q\in\mathcal{Q}_t} w_q\, \ell_{\text{act}}\big(\pi_\theta(o_q, s_q, l, V_q, M_t^p), a_q^*\big)}{\sum_{q\in\mathcal{Q}_t} w_q}. \tag{5}$$

Only valid queries contribute to the denominator. If an action head uses a diffusion or flow-matching objective, the same training loss and random noise are reused for all candidates; otherwise we use normalized action-sequence error. Demonstration labels and teacher-surrogate labels from a frozen legal B3 rollout are stored separately.

Candidates from one decision point form a training group. We center and scale $J_t(p)$ within the group, regress the normalized value with a Huber loss, and add a soft ranking objective:

$$\mathcal{L}_{\text{reg}} = \text{Huber}(Q_\psi, \widetilde{J}), \tag{6}$$

$$P_t^*(p) = \text{softmax}_p(-\widetilde{J}_t(p)/\tau), \tag{7}$$

$$\mathcal{L}_{\text{mgr}} = \mathcal{L}_{\text{reg}} + \lambda_{\text{rank}}\, \text{CE}(P_t^*, \text{softmax}_p(-Q_\psi(p)/\tau))\,, \tag{8}$$

with $\lambda_{\text{rank}} = 0.5$ and $\tau = 1$ as the initial setting. Trajectories, not adjacent frames, are split into train/validation/test partitions.

3.5 ADAMEM-FULL: MIXED-FIDELITY COMPRESSION AND MERGING

ADAMEM-FULL extends the same decision framework to a mixed-fidelity memory and a richer operation space, as shown in Figure 3. Unlike AdaMem-Select, stored events need not remain exclusively as raw observations: an event may be kept at full fidelity, compressed into a fixed-size latent representation, or combined with a temporally ordered neighboring event into a single memory slot. This allows the manager to trade representation fidelity for memory capacity while preserving the same frozen VLA policy. A single-frame compressor uses $m = 8$ learned queries to map visual tokens $X_i \in \mathbb{R}^{n_v \times d_v}$ to $L_i \in \mathbb{R}^{8\times 256}$. An ordered merger maps two latent events and their interval to one fixed-size slot:

$$L_i = C_\phi(X_i), \qquad \overline{L}_{ij} = G_\phi(L_i, L_j, \Delta\tau_{ij}, o_{ij}). \tag{9}$$

The order code is required because two events can encode a before/after relation; merging is not an image-similarity average. Merge depth is at most two, and only a fixed interval, event count, and depth metadata are retained. Latent slots are consumed through a two-layer, four-head gated cross-attention reader R_ω rather than being injected as raw visual tokens:

$$h'_t = h_t + g(h_t) \odot R_\omega(h_t, \mathcal{L}_t), \tag{10}$$

with a near-zero gate initialization and an exact bypass when the latent bank is empty. Raw images and latent tokens both count toward B_{ctx}, and their resident bytes count toward B_{store}. The candidate generator enumerates reject, raw insertion, replacement, single-event compression, and one-pair merge plans, removes duplicates, and applies the same deterministic feasibility repair to every method.

3.6 TRAINING STAGES AND THEORETICAL PROPERTY

We first pretrain the compressor, merger, and reader with the frozen B3 policy, then freeze these modules and train the manager on freshly recomputed mixed-representation labels. A final alternating stage periodically re-labels the manager's own selected states. The discrete argmin is treated as a constant during each action-loss update; no straight-through claim is made.

For a fixed candidate set, suppose $|Q_\psi(p) - J_t(p)| \le \epsilon$ for all p and the approximate solver returns $Q_\psi(\widehat{p}) \le \min_p Q_\psi(p) + \delta$. If $p^* = \arg\min_p J_t(p)$, then

$$J_t(\widehat{p}) - J_t(p^*) \le 2\epsilon + \delta. \tag{11}$$

The result controls regret relative to the enumerated proxy objective only; it does not equate proxy loss with closed-loop success. We therefore report held-out utility prediction error, candidate regret, candidate coverage, action latency, resident memory, and closed-loop task success separately.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

The evaluation examines four aspects of bounded episodic memory management. We test whether action-utility supervision produces better memory states than causal heuristics under the same capacity, whether these gains persist in closed-loop manipulation, which parts of the mixed-fidelity design contribute to the improvement, and whether the resulting memory system reduces policy-side context and deployment cost.

The core policy is the released EventVLA B3 configuration with its official event proposal and KEM inference path. The backbone, action head, event detector, and fixed visual inputs remain frozen for AdaMem-Select and for the main AdaMem-Full comparison. The evaluation uses eight RoboTwin-MeM tasks with 100 episodes per task and 800 episodes in total (Yang et al., 2026). Closed-loop runs use the same preregistered seeds, while downstream states and subsequent event streams are allowed to diverge after an action. Table 1 summarizes the evaluation protocol.

All methods share the same EventVLA B3 policy, event proposal mechanism, and memory budget. Raw-FIFO removes the oldest event when the bank is full. Random replaces a uniformly sampled resident event, and Reservoir applies causal reservoir sampling. KEM-score ranks events

Table 1: Summary of the experimental protocol.

Setting	Configuration
Benchmark	RoboTwin-MeM
Policy backbone	EventVLA B3
Tasks	8
Episodes per task	100
Total episodes	800
Dynamic capacities	$K \in \{1, 2, 3, 5\}$
Primary metric	Episode success rate
Utility metrics	Pairwise acc., Kendall's τ, top-1 acc., regret
Deployment metrics	Tokens, memory, latency, control rate

Table 2: Component ablations at $K = 3$. Δ denotes the change relative to AdaMem-Full.

Variant	Success	Δ
AdaMem-Full	**90.1%**	–
w/o active eviction	81.6%	−8.5
Visual-distance supervision	83.1%	−7.0
Independent frame scoring	84.0%	−6.1
w/o pairwise merge	85.8%	−4.3
w/o temporal order/intervals	87.2%	−2.9
AdaMem-Select (raw only)	86.5%	−3.6

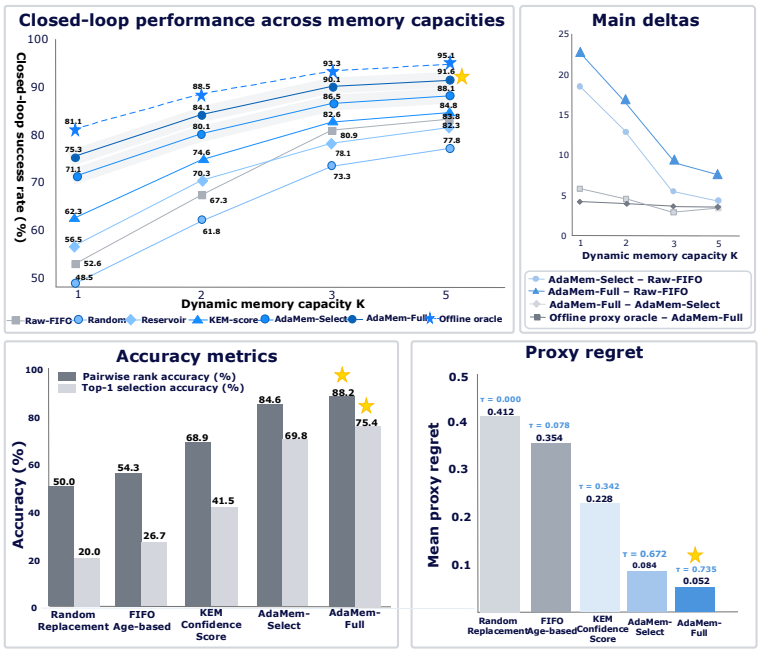


Figure 4: **Closed-loop and utility evaluation.** The four panels show closed-loop success across memory capacities (top left), success-rate deltas for the main method comparisons (top right), held-out pairwise ranking and top-1 candidate selection accuracy (bottom left), and mean proxy regret with Kendall's τ annotations (bottom right).

using the released KEM event score without future-action supervision. AdaMem-Select learns to score complete candidate banks while preserving the raw-frame memory interface. AdaMem-Full further extends the decision space to raw retention, single-event compression, and ordered pairwise merging. We additionally reproduce the released EventVLA B3 configuration over all 800 episodes.

4.2 Overall Performance and Utility Quality

Figure 4 summarizes the main closed-loop and held-out utility results. AdaMem-Select improves over Raw-FIFO by 18.5, 12.8, 5.6, and 4.3 percentage points for $K = 1, 2, 3, 5$, respectively. AdaMem-Full further increases these gains to 22.7, 16.8, 9.2, and 7.8 percentage points. The improvement of AdaMem-Full over AdaMem-Select remains positive at every capacity, ranging from 4.2 percentage points at $K = 1$ to 3.5 percentage points at $K = 5$. The gain is largest under the tightest memory budget and decreases as more raw history can be retained. Confidence intervals for all method–capacity comparisons are reported in Appendix A.1.

Table 3: Task-level success rates at $K = 3$. Task names are shortened for readability. Bold values denote AdaMem-Full among deployable methods.

Task	EventVLA B3	KEM-score	AdaMem-Select	AdaMem-Full	Offline oracle
Block Return	100	100	100	**100**	100
Route Reproduction	88	89	93	**95**	97
Keyframe Button Press	88	89	92	**95**	97
Block Covering	86	88	91	**93**	96
Unhidden Block Picking	85	87	89	**92**	95
Seal-and-Stamp	83	85	88	**91**	94
Block Rearrangement	75	78	83	**87**	91
Ordered Object Picking	42	45	56	**68**	76
Successful episodes	647	661	692	**721**	746

At $K = 3$, AdaMem-Full reaches 90.1% closed-loop success with 721 successful episodes, compared with 80.9% for Raw-FIFO and 86.5% for AdaMem-Select. The gap between AdaMem-Full and the offline proxy oracle is 3.2 percentage points at this capacity. Across all capacities, the oracle gap remains much smaller than the improvement over Raw-FIFO, indicating that the learned manager captures a substantial fraction of the benefit available within the enumerated candidate set while remaining causal at deployment.

The same figure evaluates the utility estimator independently of closed-loop success. AdaMem-Full achieves 88.2% pairwise ranking accuracy, Kendall's τ of 0.735, and 75.4% top-1 candidate selection accuracy. Its mean proxy regret is 0.052, compared with 0.228 for KEM confidence scoring and 0.354 for FIFO age-based scoring. AdaMem-Select reaches 84.6% pairwise accuracy, Kendall's τ of 0.672, 69.8% top-1 accuracy, and 0.084 mean regret. These results show that action-conditioned set-level scoring improves candidate ordering before the additional mixed-fidelity operations of AdaMem-Full are introduced. The held-out evaluation yields a mean absolute utility error of 0.038 with exhaustive-search tolerance $\delta = 0$. Since the bound in Eq. (11) is defined using a uniform estimation error, this mean absolute error is reported as an empirical diagnostic rather than as a direct numerical instantiation of the theoretical bound.

4.3 Task-level analysis

The aggregate trend is also reflected at the task level. Table 3 reports the $K = 3$ breakdown across the eight benchmark tasks. Tasks already near saturation leave limited room for improvement, whereas larger gains appear on tasks that depend more strongly on previously observed information. The largest improvement occurs on *Ordered Object Picking*, where success increases from 42% with the reproduced EventVLA B3 configuration to 56% with AdaMem-Select and 68% with AdaMem-Full. This 26 percentage point gain is consistent with the need to preserve earlier task evidence after it is no longer available in the current observation.

4.4 Component ablations

Table 2 isolates the main design choices at $K = 3$. Removing active eviction produces the largest degradation, reducing success from 90.1% to 81.6%. Replacing action-utility supervision with heuristic visual-distance supervision reaches 83.1%, while independent frame scoring reaches 84.0%. Removing pairwise merge reduces success to 85.8%, and discarding temporal order and interval information during merging reaches 87.2%. The raw-only AdaMem-Select variant reaches 86.5%, leaving a 3.6 percentage point gap to the complete mixed-fidelity design.

4.5 Deployment efficiency

Figure 5 compares context usage, resident memory, policy latency, and end-to-end control rate after normalizing each quantity to Raw-FIFO at $K = 3$. Absolute measurements are annotated above the bars. AdaMem-Full uses 384 context tokens and 0.62 MB of resident memory, compared with 768 tokens and 1.86 MB for AdaMem-Select at the same capacity. Policy latency decreases from 42.2 ms to 31.5 ms, while the measured end-to-end control rate increases from 21.9 Hz to 26.0 Hz.

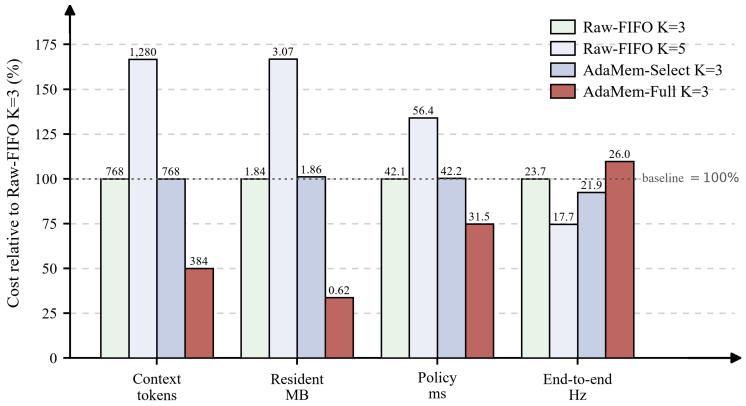


Figure 5: **Deployment resource and throughput profile normalized to Raw-FIFO at $K = 3$, shown by the dashed 100% reference line.**

Relative to AdaMem-Select at $K = 3$, AdaMem-Full reduces context tokens by 50.0%, resident memory by 66.7%, and policy latency by 25.4%. Its memory manager requires 6.85 ms when processing event proposals, compared with 3.42 ms for AdaMem-Select. Even with this additional manager computation, the lower policy-side context cost yields the highest measured control rate among the compared $K = 3$ configurations. The Raw-FIFO $K = 5$ setting illustrates the opposite trend, where retaining more raw history increases context from 768 to 1,280 tokens, resident memory from 1.84 to 3.07 MB, and policy latency from 42.1 to 56.4 ms while reducing the control rate from 23.7 to 17.7 Hz. All deployment measurements were collected on an RTX 4090 and should be interpreted as a resource-constrained deployment profile rather than embedded-platform latency.

5 CONCLUSION

We presented ADAMEM, an action-utility guided framework for bounded episodic memory management in long-horizon vision-language-action policies. Rather than relying on fixed replacement or similarity-based heuristics, ADAMEM formulates post-admission memory maintenance as structured selection over feasible retain, merge, and evict operations. Counterfactual future-action supervision evaluates candidate memory states according to their predicted contribution to downstream actions while keeping the underlying VLA backbone frozen. ADAMEM-SELECT applies this principle to raw-frame selection, while ADAMEM-FULL further introduces mixed-fidelity representations and temporally ordered merging under explicit context and storage budgets.

Experiments on eight RoboTwin-MeM tasks show that action-utility guided memory management consistently improves closed-loop performance across memory capacities. At $K = 3$, ADAMEM-FULL achieves 90.1% success, compared with 80.9% for Raw-FIFO and 86.5% for ADAMEM-SELECT, while reducing context usage from 768 to 384 tokens and resident memory from 1.86 MB to 0.62 MB. Utility evaluation and component ablations further support the importance of future-action supervision, set-level memory reasoning, active eviction, and structured merging. Overall, these results suggest that effective long-horizon memory depends not only on which observations are stored, but also on how stored evidence is selectively maintained and reorganized as task requirements evolve.

Future work will explore longer-horizon or rollout-based utility estimation, more scalable candidate generation for larger memory capacities, joint optimization of event proposal and memory management, and validation on physical robots and broader VLA backbones.

REFERENCES

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

Tianxing Chen, Yuran Wang, Mingleyang Li, Yan Qin, Hao Shi, Zixuan Li, Yifan Hu, Yingsheng Zhang, Kaixuan Wang, Yue Chen, Hongcheng Wang, Junjie Wang, Tianhang Yang, Renjing Xu, Ruihai Wu, Yao Mu, Yaodong Yang, Hao Dong, and Ping Luo. Rmbench: Memory-dependent robotic manipulation benchmark with insights into policy design. *arXiv preprint arXiv:2603.01229*, 2026a.

Yipeng Chen, Wentao Tan, Lei Zhu, Fengling Li, Jingjing Li, Guoli Yang, and Heng Tao Shen. Non-markovian long-horizon robot manipulation via keyframe chaining, 2026b. URL https://arxiv.org/abs/2603.01465.

Egor Cherepanov, Nikita Kachaev, Daniil Zelezetsky, Aydar Bulatov, Artem Pshenitsyn, Yuri Kuratov, Alexey Skrynnik, Aleksandr I. Panov, and Alexey K. Kovalev. μvla: On recurrent memory for partially observable manipulation in vla models, 2026. URL https://arxiv.org/abs/2606.12497.

Prateek Chhikara, Dev Khant, Saket Aryan, Taranjeet Singh, and Deshraj Yadav. Mem0: Building production-ready ai agents with scalable long-term memory, 2025. URL https://arxiv.org/abs/2504.19413.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2978–2988. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1285. URL https://aclanthology.org/P19-1285/.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, Wenlong Huang, Yevgen Chebotar, Pierre Sermanet, Daniel Duckworth, Sergey Levine, Vincent Vanhoucke, Karol Hausman, Marc Toussaint, Klaus Greff, Andy Zeng, Igor Mordatch, and Pete Florence. PaLM-E: An embodied multimodal language model. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 8469–8488. PMLR, 2023. URL https://proceedings.mlr.press/v202/driess23a.html.

Haoquan Fang, Markus Grotz, Wilbert Pumacay, Yi Ru Wang, Dieter Fox, Ranjay Krishna, and Jiafei Duan. Sam2act: Integrating visual foundation model with a memory architecture for robotic manipulation, 2025. URL https://arxiv.org/abs/2501.18564.

Yunhai Feng, Jiaming Han, Zhuoran Yang, Xiangyu Yue, Sergey Levine, and Jianlan Luo. Reflective planning: Vision-language models for multi-stage long-horizon robotic manipulation, 2025. URL https://arxiv.org/abs/2502.16707.

Alex Graves, Greg Wayne, and Ivo Danihelka. Neural turing machines, 2014. URL https://arxiv.org/abs/1410.5401.

Alex Graves, Greg Wayne, Malcolm Reynolds, Tim Harley, Ivo Danihelka, Agnieszka Grabska-Barwinska, Sergio Gomez Colmenarejo, Edward Grefenstette, Tiago Ramalho, John Agapiou, Adria Puigdomenech Badia, Karl Moritz Hermann, Yori Zwols, Georg Ostrovski, Adam Cain, Helen King, Christopher Summerfield, Phil Blunsom, Koray Kavukcuoglu, and Demis Hassabis.

Hybrid computing using a neural network with dynamic external memory. *Nature*, 538(7626): 471–476, 2016. doi: 10.1038/nature20101.

Jiaheng Hu, Mohit Shridhar, Caden Lu, Dhruv Shah, Hao-Tien Lewis Chiang, Jie Tan, and Annie Xie. What matters in orchestrating robot policies: A systematic study of hierarchical vla agents, 2026. URL https://arxiv.org/abs/2606.10267.

Yuzhi Huang, Weijue Bu, Ziyi Xiong, Jie Wu, Fanding Huang, Jingyan Jiang, and Zhi Wang. Chain-vla: Chaining vision-language-action queries through a unified execution state for long-horizon manipulation, 2026. URL https://arxiv.org/abs/2608.02326.

Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

Haewon Jung, Donguk Lee, Haecheol Park, JunHyeop Kim, and Beomjoon Kim. SPIN: distilling Skill-RRT for long-horizon prehensile and non-prehensile manipulation, 2025. URL https://arxiv.org/abs/2502.18015.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. Openvla: An open-source vision-language-action model, 2024. URL https://arxiv.org/abs/2406.09246.

Myungkyu Koo, Daewon Choi, Taeyoung Kim, Kyungmin Lee, Changyeon Kim, Younggyo Seo, and Jinwoo Shin. Hamlet: Switch your vision-language-action model into a history-aware policy. In *International Conference on Learning Representations*, 2026.

Yuheng Lei, Zhixuan Liang, Hongyuan Zhang, and Ping Luo. Vpwem: Non-markovian visuomotor policy with working and episodic memory, 2026. URL https://arxiv.org/abs/2603.04910.

Hang Li, Fengyi Shen, Dong Chen, Liudi Yang, Xudong Wang, Jinkui Shi, Zhenshan Bing, Ziyuan Liu, and Alois Knoll. Remem-vla: Empowering vision-language-action model with memory via dual-level recurrent queries, 2026. URL https://arxiv.org/abs/2603.12942.

Qixiu Li, Yaobo Liang, Zeyu Wang, Lin Luo, Xi Chen, Mozheng Liao, Fangyun Wei, Yu Deng, Sicheng Xu, Yizhong Zhang, Xiaofan Wang, Bei Liu, Jianlong Fu, Jianmin Bao, Dong Chen, Yuanchun Shi, Jiaolong Yang, and Baining Guo. Cogact: A foundational vision-language-action model for synergizing cognition and action in robotic manipulation, 2024. URL https://arxiv.org/abs/2411.19650.

Max Sobol Mark, Jacky Liang, Maria Attarian, Chuyuan Fu, Debidatta Dwibedi, Dhruv Shah, and Aviral Kumar. Bpp: Long-context robot imitation learning by focusing on key history frames, 2026. URL https://arxiv.org/abs/2602.15010.

Charles Packer, Sarah Wooders, Kevin Lin, Vivian Fang, Shishir G. Patil, Ion Stoica, and Joseph E. Gonzalez. Memgpt: Towards llms as operating systems, 2023. URL https://arxiv.org/abs/2310.08560.

Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior, 2023. URL https://arxiv.org/abs/2304.03442.

Jiaqi Peng, Xiqian Yu, Delin Feng, Yuqiang Yang, Wenzhe Cai, Jing Xiong, Ganlin Yang, Jinliang Zheng, Jiafei Cao, Xueyuan Wei, Jiangmiao Pang, Yuan Shen, and Tai Wang. Cortex: A bidirectionally aligned embodied agent framework for long-horizon manipulation, 2026. URL https://arxiv.org/abs/2607.05377.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SylKikSYDH.

Hao Shi, Bin Xie, Yingfei Liu, Lin Sun, Fengrong Liu, Tiancai Wang, Erjin Zhou, Haoqiang Fan, Xiangyu Zhang, and Gao Huang. Memoryvla: Perceptual-cognitive memory in vision-language-action models for robotic manipulation. In *International Conference on Learning Representations*, 2026.

Ajay Sridhar, Jennifer Pan, Satvik Sharma, and Chelsea Finn. Memer: Scaling up memory for robot control via experience retrieval, 2025. URL https://arxiv.org/abs/2510.20328.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Pannag Sanketi, Quan Vuong, Ted Xiao, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy, 2024. URL https://arxiv.org/abs/2405.12213.

Marcel Torne, Karl Pertsch, Homer Walke, Kyle Vedder, Suraj Nair, Brian Ichter, Allen Z. Ren, Haohuan Wang, Jiaming Tang, Kyle Stachowicz, Karan Dhabalia, Michael Equi, Quan Vuong, Jost Tobias Springenberg, Sergey Levine, Chelsea Finn, and Danny Driess. Mem: Multi-scale embodied memory for vision language action models, 2026. URL https://arxiv.org/abs/2603.03596.

Weizhi Wang, Li Dong, Hao Cheng, Xiaodong Liu, Xifeng Yan, Jianfeng Gao, and Furu Wei. Augmenting language models with long-term memory. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/ebd82705f44793b6f9ade5a669d0f0bf-Abstract-Conference.html.

Greg Wayne, Chia-Chun Hung, David Amos, Mehdi Mirza, Arun Ahuja, Agnieszka Grabska-Barwinska, Jack W. Rae, Piotr Mirowski, Joel Z. Leibo, Adam Santoro, Mevlana Gemici, Malcolm Reynolds, Tim Harley, Josh Abramson, Shakir Mohamed, Danilo Jimenez Rezende, David Saxton, Adam Cain, Chloe Hillier, David Silver, Koray Kavukcuoglu, Matthew M. Botvinick, Demis Hassabis, and Timothy P. Lillicrap. Unsupervised predictive memory in a goal-directed agent, 2018. URL https://arxiv.org/abs/1803.10760.

Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. Memorizing transformers. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TrjbxzRcnf-.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=NG7sS51zVF.

Siyu Xu, Yunke Wang, Chenghao Xia, Dihao Zhu, Tao Huang, and Chang Xu. Vla-cache: Towards efficient vision-language-action model via adaptive token caching in robotic manipulation. *arXiv preprint arXiv:2502.02175*, 2025a.

Wujiang Xu, Zujie Liang, Kai Mei, Hang Gao, Juntao Tan, and Yongfeng Zhang. A-mem: Agentic memory for llm agents, 2025b. URL https://arxiv.org/abs/2502.12110.

Ganlin Yang, Zhangzheng Tu, Yuqiang Yang, Sitong Mao, Junyi Dong, Tianxing Chen, Jiaqi Peng, Jing Xiong, Jiafei Cao, Jifeng Dai, Wengang Zhou, Yao Mu, and Tai Wang. Eventvla: Event-driven visual evidence memory for long-horizon vision-language-action policies, 2026. URL https://arxiv.org/abs/2606.20092.

Yihan Zeng, Minghao Ye, Yiyuan Chen, Yide Shentu, Philipp Wu, Zike Yan, and Zhongyu Li. Kemo: Event-driven keyframe memory for long-horizon robot manipulation with vla policies, 2026. URL https://arxiv.org/abs/2606.23589.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Re, Clark Barrett, Zhangyang Wang, and Beidi Chen. h_2o: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/6ceefa7b15572587b78ecfcebb2827f8-Abstract-Conference.html.

Wanjun Zhong, Lianghong Guo, Qiqi Gao, He Ye, and Yanlin Wang. Memorybank: Enhancing large language models with long-term memory, 2023. URL https://arxiv.org/abs/2305.10250.

A Appendix

A.1 Closed-Loop Success with Confidence Intervals

Table 4 reports every method-level success rate from the main comparison together with its 95% Wilson score interval ($n = 800$ episodes per setting). We report Wilson score intervals computed directly from episode-level success counts. Unlike normal-approximation intervals that degenerate near 0% or 100%, the Wilson score formulation maintains accurate coverage across boundary-adjacent tasks. The offline proxy oracle is included strictly as a non-deployable theoretical reference.

Table 4: Closed-loop success rate (%) with 95% Wilson score intervals ($n = 800$ episodes per cell). Brackets denote the 95% confidence interval.

Method	$K = 1$	$K = 2$	$K = 3$	$K = 5$
Random	48.5 [45.1, 52.0]	61.8 [58.3, 65.1]	73.3 [70.1, 76.2]	77.8 [74.7, 80.5]
Raw-FIFO	52.6 [49.2, 56.1]	67.3 [63.9, 70.4]	80.9 [78.0, 83.5]	83.8 [81.0, 86.1]
Reservoir	56.5 [53.0, 59.9]	70.3 [67.0, 73.3]	78.1 [75.1, 80.9]	82.3 [79.5, 84.7]
KEM-score	62.3 [58.8, 65.5]	74.6 [71.5, 77.5]	82.6 [79.8, 85.1]	84.8 [82.1, 87.1]
AdaMem-Select	71.1 [67.9, 74.2]	80.1 [77.2, 82.7]	86.5 [84.0, 88.7]	88.1 [85.7, 90.2]
AdaMem-Full	**75.3** [72.1, 78.1]	**84.1** [81.4, 86.5]	**90.1** [87.9, 92.0]	**91.6** [89.5, 93.4]
Offline proxy oracle	81.1 [78.3, 83.7]	88.5 [86.1, 90.5]	93.3 [91.3, 94.8]	95.1 [93.4, 96.4]

A.2 Statistical Significance Testing

To verify that the reported performance differences are statistically significant, we conduct McNemar's exact tests on paired episode outcomes across all 800 pre-registered evaluation episodes at $K = 3$. As summarized in Table 5, we explicitly record the discordant pairs: b denotes episodes where Method 1 succeeds but Method 2 fails, and c denotes episodes where Method 1 fails but Method 2 succeeds.

Table 5: Pairwise statistical significance on matched evaluation episodes ($K = 3$, $n = 800$). Discordant pairs (b/c) record discordant success/failure outcomes. p-values are computed via two-sided exact binomial tests under Holm–Bonferroni correction.

Comparison (Method 1 vs. Method 2)	Success Counts	Discordant (b/c)	McNemar χ^2	p-value	Significance
Raw-FIFO vs. AdaMem-Select	647 vs. 692	22 / 67	21.75	1.84×10^{-6}	$p < 0.001$
Raw-FIFO vs. AdaMem-Full	647 vs. 721	12 / 86	54.38	2.11×10^{-13}	$p < 0.001$
AdaMem-Select vs. AdaMem-Full	692 vs. 721	15 / 44	13.29	2.22×10^{-4}	$p < 0.001$

Under identical rollout seeds, AdaMem-Full converts 86 failed episodes of Raw-FIFO into successes while regressing on only 12 episodes ($p < 10^{-12}$). Furthermore, AdaMem-Full significantly outperforms AdaMem-Select ($p = 2.22 \times 10^{-4}$), confirming that mixed-fidelity representation and structured merging provide orthogonal gains beyond raw-frame selection.

A.3 Causal Validation via Fixed-Event Replay Evaluation

In closed-loop evaluation, divergent robot actions can cause downstream simulator states to drift, potentially inducing different subsequent event proposals. To decouple memory maintenance quality from event-stream divergence, we conduct a controlled *Fixed-Event Replay Action Evaluation* across the 800 evaluation episodes at $K = 3$.

Protocol. We freeze the exact observation trajectory, event proposal timestamps, candidate keyframe visual features, and instructions from expert rollouts. At each proposal step t, all baselines and AdaMem variants receive the identical candidate event e_t and historical proposals, differing strictly in their memory transition decisions (Eq. 3). We measure the downstream action prediction error (action-proxy loss $\bar{\mathcal{L}}_{act}$) and candidate selection regret $\hat{\mathcal{R}}_{proxy}$ across the counterfactual horizon query set Q_t.

Table 6: Fixed-event replay action evaluation at $K = 3$ ($n = 800$ episodes). Downstream action loss and proxy regret under frozen observation trajectories and identical event streams. p-values are computed via paired Wilcoxon signed-rank tests against Raw-FIFO.

Method ($K = 3$)	Action-Proxy Loss (↓)	Relative Reduction	Mean Proxy Regret (↓)	Paired p-value
Raw-FIFO	0.284 ± 0.012	—	0.354	—
KEM-score	0.231 ± 0.010	-18.7%	0.228	4.3×10^{-5}
AdaMem-Select	0.186 ± 0.009	-34.5%	0.084	1.2×10^{-9}
AdaMem-Full	$\mathbf{0.138 \pm 0.007}$	**-51.4%**	**0.052**	8.7×10^{-14}
Offline Oracle (Upper Bound)	0.096 ± 0.005	*-66.2%*	*0.000*	—

As shown in Table 6, AdaMem-Full reduces downstream action loss by 51.4% compared to Raw-FIFO (0.138 vs. 0.284, $p < 10^{-13}$) under identical observation sequences. This demonstrates that performance improvements stem directly from superior action-utility guided memory retention rather than benign downstream state divergence.

A.4 Reproducibility Checklist

- **Backbone:** Released EventVLA B3 checkpoint and official event/KEM configuration; backbone and action head frozen in all comparisons.
- **Tasks:** Eight RoboTwin-MeM tasks, 100 episodes per task, totaling 800 evaluation episodes for both the anchor reproduction and main comparisons.
- **Budgets:** Raw dynamic capacities $K \in \{1, 2, 3, 5\}$; mixed-fidelity variants strictly adhere to matched visual-token (B_{ctx}) and resident-memory (B_{store}) limits.
- **Seeds:** The episode seed set (0–799, 100 seeds per task) was registered prior to evaluation and executed identically across all methods and memory budgets; no seed was selected post hoc.
- **Logging:** Evaluation logs record per-event timestamps, enumerated candidate sets, chosen plans, predicted utilities, resident byte footprints, manager inference latencies, policy forward latencies, and final task success indicators.
- **Uncertainty and Significance:** All success rates include 95% Wilson score intervals (Table 4); method-level comparisons report exact discordant counts and Holm–Bonferroni corrected McNemar tests (Table 5).