DIM-WAM: World-Action Modeling with Diverse Historical Event Memory

Kai Wang[1,2], Zhaopeng Gu[1,2], Yixiang Chen[1], Yuan Xu[1], Qisen Ma[1],
Peng Su[2], Zhaowen Li[2,*], Yan Huang[1,3,*], Liang Wang[1]

***Abstract*—World-action models have shown promising robot-manipulation performance by jointly predicting future visual states and actions. However, existing methods mainly rely on short-term history and short-horizon future prediction, which is insufficient for long-horizon tasks whose correct execution depends on earlier observations and task progress. Such temporally dependent tasks require effective use of complementary temporal information, including recent local context, cross-stage historical events, immediate future dynamics, and global task progress. To address long-term forgetting and poor awareness of the global task state, we introduce DIM-WAM, a memory-augmented world-action model that integrates multi-scale historical context, local future dynamics, and global task progress. The memory extracts compact visual event information from real observations, updates multiple memory banks through independent similarity-based merging, and then reads the bank-identity- and time-embedded long-term context to condition video and action denoising. A progress-supervision objective further encourages memory tokens to encode not only completed historical events but also the current task stage and its implications for the remaining task. On RMBench, DIM-WAM raises average success from 28.4% with LingBot-VA to 69.8%, exceeding the explicit-memory Mem-0 baseline at 42.0%. On four real-world Franka tasks, it improves average stage success from 70.7% to 91.5% and full-task success from 52.5% to 80.0%. Project page: `https://wangkai-casia.github.io/dim-wam/`.**

I. INTRODUCTION

Vision-language-action (VLA) models map visual observations, language instructions, and robot actions into a unified sequence-modeling framework. Early systems such as RT-2 [1] and OpenVLA [2] obtained strong semantic generalization from large-scale vision-language pretraining. Subsequent work has improved continuous control, bimanual coordination, and cross-embodiment transfer through flow matching in π_0 [3], diffusion action heads in RDT-1B [4], and dual-system architectures in GR00T N1 [5].

Despite this progress, the primary supervision signal for VLA models remains relatively sparse action labels. Compared with high-dimensional visual observations, action labels only indirectly describe fine-grained dynamics such as contact, occlusion changes, and stage transitions. As noted by GigaWorld-Policy [6], policies may therefore rely on static semantic correlations without sufficiently modeling how actions change the environment.

World-action models (WAMs), including GR-2 [7], LingBot-VA [8], and DreamZero [9], provide a complementary paradigm by jointly predicting actions and future visual states. By importing the spatiotemporal and physical priors learned by video-generation models into robot control, WAMs use visual evolution as dense supervision and explicitly model the relation between actions and environment changes.

Existing WAMs have demonstrated strong performance in short-horizon closed-loop manipulation, but extending them to long-horizon tasks remains difficult. Long-horizon manipulation requires more than local future prediction: the model must maintain, retrieve, and update task-relevant history over extended execution while also estimating long-range task state.

More concretely, long-horizon manipulation involves at least four complementary types of temporal information. Long-term history records completed subtasks and early states; short-term history confirms recent local changes; short-term future information describes immediate visual evolution and action consequences; and global task-progress information describes the goal-relative temporal position of the current state within the complete task.

In multi-step tidying, assembly, and tool-use tasks with strong non-Markovian structure, missing any of these information types can lead to repeated actions, skipped stages, or incorrect stage switching. Therefore, a long-horizon WAM should not merely extend its prediction horizon; it must organize context at different temporal scales and with different functional roles under a finite computational budget.

A direct solution is to keep increasing the historical context length. However, this increases both information volume and computation, while early but critical states may be weakened by attention competition. More importantly, WAM predictions recursively influence subsequent decisions. If intermediate prediction errors are written into long-term context, the model's internal state can gradually drift away from the real environment.

Recent memory-augmented VLA methods aggregate history through explicit memory banks in SAM2Act+ [10], perceptual-cognitive memory in MemoryVLA [11], multiscale embodied memory in MEM [12], or recurrent queries in ReMem-VLA [13]. These methods primarily target action prediction. In contrast, WAM memory also participates directly in future visual-state generation and update. Reusing a single memory mechanism may either discard fine-grained visual evidence or inject generated-state errors into subsequent history.

Motivated by this analysis, we propose DIM-WAM, a memory-augmented WAM for tasks with long-range tempo-

[1]NLPR, CASIA. [2]Yinwang Intelligent Technology. [3]FiveAges.
[*]Corresponding authors: Zhaowen Li `lizhaowen@yinwang.com`; Yan Huang `yhuang@nlpr.ia.ac.cn`.

ral dependencies. The key idea is to explicitly decompose the information needed for long-horizon decisions into local history, cross-stage history, immediate future dynamics, and global task-progress information, and to organize long-term history outside the local window through a multi-bank memory structure.

DIM-WAM introduces MULTI-TYPE HISTORICAL EVENT MEMORY, which first extracts compact visual event information from each real observation and then updates multiple memory banks through independent similarity-based merging. At read time, retained memory tokens receive bank-identity embeddings, are flattened in temporal order, and are encoded with RoPE time embeddings before conditioning video and action denoising. Compared with a single memory bank, this design reduces direct competition among heterogeneous information under the same capacity and allows the model to learn history subspaces specialized for different task functions. We further use task-progress prediction as an auxiliary objective, so that long-term memory encodes not only past events but also the current task stage and progress toward completion.

Experiments show that DIM-WAM improves the consistency of future-state prediction and the success rate of action decisions on RMBench [14] and real Franka Panda long-horizon tasks, with the largest gains on tasks with strong temporal dependence. On RMBench, using LingBot-VA [8] as the base WAM, fixed-capacity long-term memory increases average success over all long-horizon tasks from 28.4% to 69.8%, exceeding the explicit-memory Mem-0 baseline [14] at 42.0%. In real-robot experiments, DIM-WAM improves average stage success from 70.7% to 91.5% and full-task success from 52.5% to 80.0%. Ablations further show that both the multi-bank structure and progress supervision over long-term history are important, and cross-bank behavior analysis indicates that different banks learn complementary event selection patterns and representation subspaces rather than collapsing to identical memories.

II. RELATED WORK

A. Vision-Language-Action Models and Temporal Context

Representative VLA models have mainly focused on semantic transfer, action generation, and spatial representation, including RT-2 [1], OpenVLA [2], π_0 [3], RDT-1B [4], GR00T N1 [5], SpatialVLA [15], and BridgeVLA [16].

For partially observable and non-Markovian tasks, SAM2Act+ [10] adopts the memory structure of SAM2 and introduces memory banks, a memory encoder, and attention modules to enhance spatial memory; it also proposes MemoryBench for evaluating spatial memory and action recall. Its memory mainly stores multi-view visual features and focuses on target localization and spatial-state recovery after occlusion.

MemoryVLA [11] combines short-term working memory with a long-term Perceptual-Cognitive Memory Bank, preserving low-level visual details and high-level semantics separately while updating memory through relevance retrieval and redundancy merging. MEM [12] further organizes history along temporal-scale and modality dimensions, using visual representations for short-term image memory and language representations for long-term abstract events.

Different from explicit retrieval, ReMem-VLA [13] implicitly propagates short- and long-term history with frame-level and chunk-level queries, and strengthens visual memory through past observation prediction. EventVLA [17] treats long-term history as sparse visual evidence and writes key evidence before it becomes occluded or disappears. MemoryVLA++ [18] retrieves history and predicts future states simultaneously, fusing them into temporal-context tokens for action prediction.

These studies show that multiscale decomposition, selective writing, and long-term compression can improve history utilization in VLA models. However, such memory modules usually expose action prediction as the final interface. They rarely study how memory should participate in autoregressive visual-state updates or how closed-loop real observations can correct state bias introduced by generation. This distinction is central to memory-augmented long-horizon WAMs.

B. World Action Models

Text-guided video generation for policy learning [19] formulates sequential decision making as conditional future-video generation and recovers control actions from generated visual trajectories. RoboDreamer [20] improves planning under unseen object-goal compositions through compositional video generation, while Gen2Act [21] uses pretrained human-video generators to provide motion priors for robot policies.

The field has since moved from cascaded video-generation and control pipelines toward joint modeling of video and action. GR-2 [7] is pretrained on large-scale Internet video and jointly fine-tuned for video generation and action prediction on robot trajectories; Video Prediction Policy [22] and Seer [23] use future visual prediction to construct dynamic representations or auxiliary training objectives.

Recent WAMs directly use video-generation foundation models as control backbones. LingBot-VA [8] jointly models vision and action in a shared latent space and uses real observations for closed-loop correction; DreamZero [9] treats video as a dense representation of world evolution and achieves strong generalization across tasks, environments, and embodiments.

Cosmos Policy [24] simultaneously predicts actions, future visual/proprioceptive states, and values under a unified diffusion objective; DiT4DiT [25] cascades video diffusion and action diffusion modules so that action prediction can exploit dynamic features from video denoising. These results suggest that future modeling is valuable not only as visual planning but also as dense dynamic supervision for policy learning.

Several studies improve modeling efficiency. WoG [26] compresses future observations into condition space, and Fast-WAM [27] shows that the benefit of joint video training can be partially decoupled from explicit future generation at test time. GigaWorld-Policy [6] adopts action-centered joint training, using future visual dynamics as dense supervision

while allowing inference to decode only actions for lower control latency.

Nevertheless, these methods mainly address local future modeling, training efficiency, or short-horizon closed-loop performance. They leave cross-stage historical-state management relatively underexplored. Our work studies a complementary question: how to build a compressible, retrievable, and observation-corrected long-term memory for WAMs so that visual prediction and action decision making remain consistent during long-horizon execution.

C. Memory-Augmented Long Video Generation

Long-video generation typically extends fixed-length models autoregressively, either frame by frame or segment by segment. StreamingT2V [28] connects adjacent segments with short-term conditional attention and preserves long-term appearance consistency through the initial segment, but local windows remain insufficient for distant events.

For efficient long context, SSM-based video diffusion [29], long-context state-space video world models [30], and SANA-Video [31] replace global attention with state-space models, block scans, or linear attention, extending generation length with near-linear or constant memory cost. While unified state representations improve efficiency, they may mix information about different entities, events, and temporal scales.

For history compression, MALT [32], MAG [33], and VideoSSM [34] use recurrent memory vectors, independent KV-compression models, and hybrid local-lossless caches with global SSM memory, respectively. Context Forcing [35] addresses forgetting and drift in long contexts from the perspective of training consistency.

For selective and structured memory, Relax Forcing [36] partitions history by function, MemFlow [37] dynamically retrieves relevant clips, VideoMemory [38] maintains separate memories for characters, objects, and backgrounds, and Slot-Memory [39] reorganizes temporal caches into object-centric slots. These results indicate that memory performance depends on structured organization rather than capacity alone.

Long-video generation, however, mainly targets appearance, identity, and narrative consistency. Long-horizon robot control additionally requires memory to accurately represent task progress, object states, and action consequences, and to correct state through real interactions at every step. We therefore introduce structured memory into WAMs, using multiple memory banks to carry information with different temporal scales and functional roles while jointly supporting future prediction and action decisions.

III. Method

A. Overview

Long-horizon robot manipulation requires temporal information with different scales and functional roles. We therefore propose DiM-WAM, a world-action model that integrates multi-scale historical context, local future dynamics, and global task progress.

Given the i-th decision segment, where i indexes segments along a trajectory, we denote the temporal information relevant to the current decision by $\mathcal{I}_i$ and decompose it as

$$\mathcal{I}_i = \left(\mathcal{H}_i^{\text{long}}, \mathcal{H}_i^{\text{short}}, \mathcal{F}_i^{\text{short}}, \mathcal{G}_i^{\text{prog}}\right). \tag{1}$$

Here, $\mathcal{H}_i^{\text{short}}$ denotes recent observations and actions, $\mathcal{H}_i^{\text{long}}$ denotes cross-stage historical events, $\mathcal{F}_i^{\text{short}}$ denotes the immediate visual and action evolution predicted by the world-action model, and $\mathcal{G}_i^{\text{prog}}$ denotes the global task-progress state of the current segment.

These information types are complementary. Short-term history describes what has just happened, while long-term history records what has already been completed. Short-term future information captures what is likely to happen next, whereas global task-progress information indicates where the current state lies within the overall task. Unlike explicit long-horizon future prediction, $\mathcal{G}_i^{\text{prog}}$ does not specify future observations or actions. Instead, it provides a goal-relative temporal signal linking completed events to the remaining task.

The base WAM already models short-term history and short-term future information through local context and joint video-action generation. DiM-WAM augments this base model with Multi-Type Historical Event Memory to store long-term history, and uses task-progress supervision to shape memory representations toward the global task state. The resulting model integrates local history, cross-stage history, immediate future dynamics, and global task-progress information.

Fig. 1 illustrates the overall framework. At each decision segment, the model follows a read-predict-interact-update loop. It first reads long-term memory and combines it with local context to jointly predict future visual states and actions. It then executes the action, receives the real observation, and updates long-term memory using environmental feedback. Global task-progress information is not generated as an explicit future rollout; instead, it is learned during training through task-progress supervision.

This closed-loop design continuously aligns local prediction with global task state. It also avoids relying solely on autoregressive generated states to maintain long-term context, thereby reducing error accumulation.

B. Multi-Scale World-Action Modeling

Let c denote the language instruction, $\mathcal{C}_i$ the local context of the base model, and $\mathcal{M}_i$ the long-term memory state. The model predicts future visual latents and action chunks over horizon H, where $\hat{\mathbf{z}}_{i:i+H}$ denotes the predicted visual-latent sequence and $\hat{\mathbf{a}}_{i:i+H}$ denotes the corresponding action sequence:

$$(\hat{\mathbf{z}}_{i:i+H}, \hat{\mathbf{a}}_{i:i+H}) = f_\theta(\mathcal{C}_i, c; \mathcal{M}_i). \tag{2}$$

Here, f_θ is the WAM parameterized by θ, and the semicolon indicates that long-term memory $\mathcal{M}_i$ is used as an additional prediction condition.

The local context $\mathcal{C}_i$ is provided by a sliding window and a key-value (KV) cache, which preserve recent action

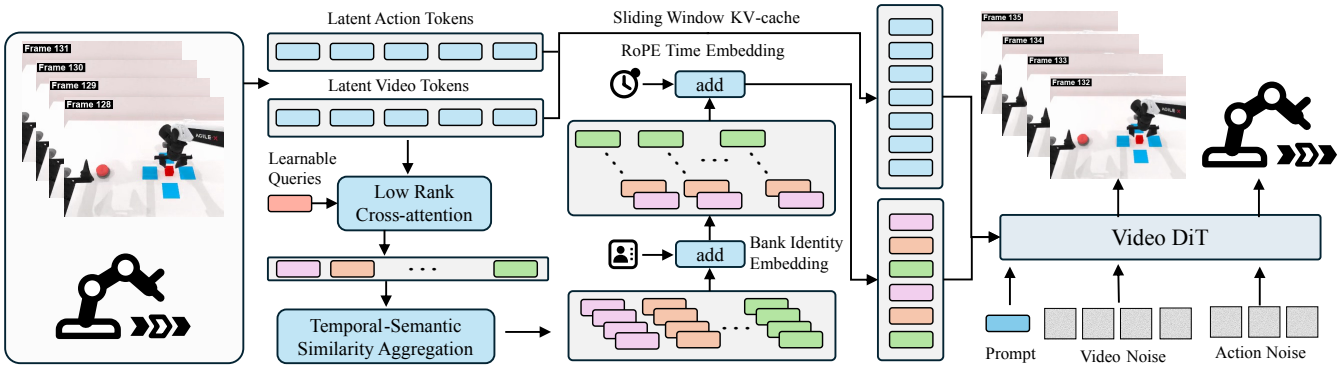


Fig. 1. Overall framework of DIM-WAM. The model reads long-term memory, predicts short-term future video and action, executes the action, and updates memory using real observations.

continuity. The visual branch predicts short-term environment evolution, and the action branch generates continuous controls conditioned on the corresponding dynamic representation.

With only $\mathcal{C}_i$, the model can handle local dynamics but cannot disambiguate visually similar states with different task meanings. For example, an object at the center may indicate either that a button has not yet been pressed or that the object should now be returned after the button press.

Long-term memory $\mathcal{M}_i$ supplements the current state with cross-stage evidence, and progress supervision further constrains the memory to encode the goal-relative temporal position of the current state within the task. Together, they resolve temporal ambiguity in local observations and keep video prediction and action decisions consistent with the global task state.

C. Multi-Bank Temporal Memory

a) Why multiple memory banks?: Long-term history is not a homogeneous sequence. Initial object locations, completed subtasks, key interaction outcomes, and recent state changes have different semantic functions, retention horizons, and update requirements.

A single memory bank compresses all of this information into the same representation space and applies one capacity and update rule. As the trajectory grows, different event types compete directly for limited slots. Frequent local changes may overwrite sparse but critical early evidence, while excessive retention of early information may reduce sensitivity to new states.

Instead of compressing long-term history into a single state, we construct MULTI-TYPE HISTORICAL EVENT MEMORY from K parallel banks:

$$\mathcal{M}_i = \{\mathcal{M}_i^1, \ldots, \mathcal{M}_i^K\}, \qquad \mathcal{M}_i^k = \{\mathbf{m}_{i,k,1}, \ldots, \mathbf{m}_{i,k,N}\}. \tag{3}$$

Here, $\mathcal{M}_i^k$ is the memory of bank k at segment i, and $\mathbf{m}_{i,k,n}$ is the memory token in its n-th event slot. Each bank contains N event slots, yielding a fixed total capacity of KN tokens. The banks are not simple copies of the same memory. Through independent similarity-based compression and bank identities at read time, they form complementary historical views of the same trajectory.

We do not prescribe a semantic role for each bank. Functional specialization is learned jointly from video prediction, action generation, task progress, and cross-bank diversity objectives, allowing the model to discover memory subspaces suited to different event types.

b) Compressed visual event extraction.: After the i-th interaction segment, the model encodes the real observation into visual features $\mathbf{X}_i$ and compresses them into a compact visual event token:

$$\tilde{\mathbf{u}}_i = g_\phi(\mathbf{X}_i). \tag{4}$$

Here, g_ϕ denotes a visual compression module that summarizes the observation into the information written to long-term memory. The same compressed visual information is provided to every memory bank; bank identity is not injected during this extraction step.

Each bank then decides independently whether and how to retain this candidate under its current memory state. The specialization of banks is therefore produced by their independent merge histories, downstream denoising losses, progress supervision, and the bank-identity embeddings applied during reading, rather than by using bank identity to extract different visual candidates from the same observation.

This ordering matches the framework in Fig. 1: compact visual information is first extracted from real observations, then written into each bank through similarity-based merging, and only after compression are bank identities and temporal encodings attached for cooperative reading.

c) Independent bounded compression.: Each bank independently maintains a fixed-length event sequence. A sequence contains an initial-state anchor, compressed historical slots, and recent event slots, respectively preserving the task starting point, cross-stage evidence, and local state continuity.

Under fixed capacity, when a bank reaches its slot limit, the model compares semantic similarity and temporal distance inside that bank and preferentially merges the most redundant

adjacent historical events:

$$\rho_k(u,v) = \frac{1+\cos(\mathbf{m}_{k,u}, \mathbf{m}_{k,v})}{2} \exp\left(-\frac{|t_u - t_v|}{\tau}\right). \quad (5)$$

Here, $\rho_k(u,v)$ is the merge priority of event slots u and v in bank k, $\cos(\cdot,\cdot)$ denotes cosine similarity, t_u and t_v are their temporal positions, and τ is a temperature controlling temporal decay.

If the new event is highly redundant with the recent state, memory is left unchanged. Otherwise, the most redundant historical pair is merged to release a slot for the new event. The merged event is updated by accumulated-mass weighting:

$$\mathbf{m}_{uv} = \frac{\alpha_u \mathbf{m}_u + \alpha_v \mathbf{m}_v}{\alpha_u + \alpha_v}, \qquad t_{uv} = \frac{\alpha_u t_u + \alpha_v t_v}{\alpha_u + \alpha_v}. \quad (6)$$

The merged mass is

$$\alpha_{uv} = \alpha_u + \alpha_v. \quad (7)$$

Here, $\mathbf{m}_u$ and $\mathbf{m}_v$ are the two event representations to be merged, α_u and α_v are their accumulated masses, and $\mathbf{m}_{uv}$, t_{uv}, and α_{uv} are the representation, temporal position, and mass after merging. Accumulated mass records how much historical evidence a slot compresses, so higher-mass events contribute more to the merged result.

d) Concrete memory-update rule.: We use one-indexed slot notation below. In every bank, slot 1 is the fixed initial-state anchor, slot N is the latest retained event, and slots $2,\ldots,N-1$ store compressed middle history. The latest slot is protected from merging: it is either preserved by moving the old latest event into the middle history before accepting a new event, or left unchanged when the new event is judged redundant.

For bank k, the candidate merge-pair set contains only valid adjacent middle-history pairs,

$$\mathcal{A}_{i,k} = \{(s, s+1) \mid 2 \le s \le N-2,\ v_{k,s} = v_{k,s+1} = 1\}, \quad (8)$$

where $v_{k,s}$ is the validity flag of slot s. The effective write threshold is adaptive rather than a fixed cosine cutoff. Let

$$\rho_{i,k}^{\text{new}} = \rho_k(N, \tilde{\mathbf{u}}_i), \qquad \rho_{i,k}^{\text{hist}} = \max_{(u,v)\in\mathcal{A}_{i,k}} \rho_k(u,v). \quad (9)$$

When the bank is full, the new candidate is skipped if

$$\rho_{i,k}^{\text{new}} \ge \rho_{i,k}^{\text{hist}}. \quad (10)$$

Thus the current latest event acts as the novelty threshold: a candidate is written only when inserting it is more informative than keeping the latest event and compressing the most redundant adjacent historical pair. In our implementation, the temporal-decay temperature is set to $\tau = N-1$, matching the per-bank slot horizon. When memory is initialized by filling all slots with the first-frame anchor, duplicated anchor copies in middle slots are evicted before applying Eq. (10); this removes initialization artifacts without changing the normal update rule.

Independent compression allows each bank to assess information value relative to its own retained history. An event that is redundant for one bank may still be preserved by another as a key state change, reducing direct competition among heterogeneous historical evidence under a single capacity.

Algorithm 1 Per-bank memory update at segment i

Require: Bank memory $\mathcal{M}^k = \{(\mathbf{m}_s, t_s, \alpha_s, v_s)\}_{s=1}^{N}$, compressed visual event $\tilde{\mathbf{u}}_i$
Ensure: Updated bank memory $\mathcal{M}^k$

$t_{\text{new}} \leftarrow i,\ \alpha_{\text{new}} \leftarrow 1$
if $v_1 = 0$ **then**
  Initialize slot 1 as the initial anchor and slot N as the latest event
  Optionally fill all slots with the initial anchor during first-frame initialization
  return $\mathcal{M}^k$
end if
Save old latest $(\mathbf{m}_N, t_N, \alpha_N)$
if slot N is an initialization duplicate of slot 1 **then**
  Replace slot N with $(\tilde{\mathbf{u}}_i, t_{\text{new}}, \alpha_{\text{new}}, 1)$
  return $\mathcal{M}^k$
end if
if there exists a middle slot $r \in \{2,\ldots,N-1\}$ that duplicates slot 1 **then**
  Shift middle slots $r+1,\ldots,N-1$ one position left
  Move the old latest event to slot $N-1$
  Write $(\tilde{\mathbf{u}}_i, t_{\text{new}}, \alpha_{\text{new}}, 1)$ to slot N
  return $\mathcal{M}^k$
end if
if there exists an invalid middle slot $r \in \{2,\ldots,N-1\}$ **then**
  Move the old latest event to slot r
  Write $(\tilde{\mathbf{u}}_i, t_{\text{new}}, \alpha_{\text{new}}, 1)$ to slot N
  return $\mathcal{M}^k$
end if
Build $\mathcal{A}_{i,k} = \{(s, s+1) \mid 2 \le s \le N-2,\ v_s = v_{s+1} = 1\}$
if $\mathcal{A}_{i,k} = \emptyset$ **then**
  return $\mathcal{M}^k$
end if
$(u^\star, v^\star) \leftarrow \arg\max_{(u,v)\in\mathcal{A}_{i,k}} \rho_k(u,v)$
if $\rho_k(N, \tilde{\mathbf{u}}_i) \ge \rho_k(u^\star, v^\star)$ **then**
  return $\mathcal{M}^k$ ▷ candidate is redundant with the latest event
end if
Merge slots $u^\star$ and $v^\star$ by mass-weighted averaging
Shift middle slots $v^\star+1,\ldots,N-1$ one position left
Move the old latest event to slot $N-1$
Write $(\tilde{\mathbf{u}}_i, t_{\text{new}}, \alpha_{\text{new}}, 1)$ to slot N
return $\mathcal{M}^k$

Unlike extending the context window, this mechanism bounds memory cost to KN tokens. As the trajectory grows, only the retained events in each bank change; the long-term context length does not grow.

D. Cooperative Memory Reading

Before predicting the current segment, the model first attaches a bank-identity embedding $\mathbf{b}_k$ to every valid token in

Algorithm 2 Memory readout for video-action denoising

Require: Updated banks $\{\mathcal{M}_i^k\}_{k=1}^K$, bank embeddings $\{\mathbf{b}_k\}_{k=1}^K$

Ensure: Memory condition $\mathbf{R}_i$ for video/action denoising

$\mathcal{S} \leftarrow \emptyset$
for $k = 1, \ldots, K$ **do**
  for each valid slot n in $\mathcal{M}_i^k$ **do**
    Append $(\mathbf{m}_{i,k,n} + \mathbf{b}_k,\ t_{i,k,n})$ to $\mathcal{S}$
  end for
end for
Sort $\mathcal{S}$ by time index t
$\mathbf{R}_i \leftarrow \mathrm{RoPE}_t(\mathcal{S})$
Use $\mathbf{R}_i$ as the shared memory condition for video and action denoising
return $\mathbf{R}_i$

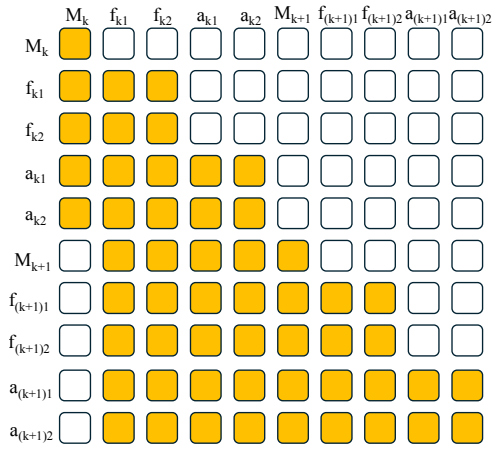


Fig. 2. Visibility mask among memory, video latent, and action tokens. Memory tokens provide long-term context while preserving the causal structure of video-action prediction.

bank k. Tokens are then concatenated in chronological order across banks and equipped with RoPE time embeddings:

$$\mathbf{R}_i = \mathrm{RoPE}_t\left(\mathrm{ChronoFlatten}\left(\{\mathbf{m}_{i,k,n} + \mathbf{b}_k\}_{k,n}\right)\right). \quad (11)$$

Here, $\mathrm{ChronoFlatten}(\cdot)$ sorts valid memory tokens by their retained time indices and keeps the bank identity as an additive type embedding, while $\mathrm{RoPE}_t(\cdot)$ injects continuous temporal order into the read sequence. This ensures that bank identity is applied after memory compression, and temporal information is applied when memory is read by the denoising model.

The banks remain independent during writing and similarity-based merging, but they cooperate through Transformer attention during reading. The resulting memory sequence controls both video denoising and action denoising: video and action tokens can dynamically select relevant evidence from different banks according to the current local state, without predefining which bank must be used for a particular decision.

As shown in Fig. 2, memory conditions both the video denoising branch and the action denoising branch. The full

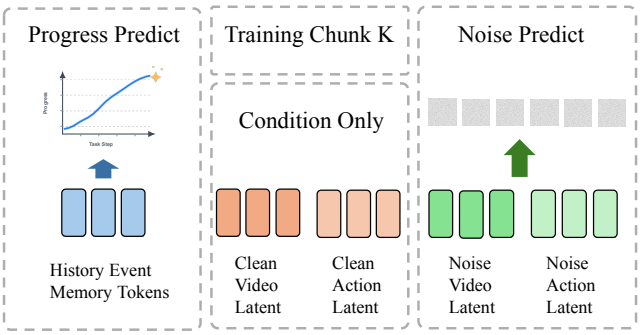


Fig. 3. Task-progress supervision for long-term memory. The auxiliary objective encourages memory tokens to encode completed events, the current task stage, and the amount of task progress remaining.

model uses joint conditioning so that future visual states and actions are grounded in the same long-term task state and local dynamics, preventing the two branches from forming inconsistent estimates of the current stage.

At inference time, memory tokens serve only as temporary attention conditions for the current segment. They do not permanently extend the base model's KV cache. After prediction, the model clears temporary memory K/V entries and updates persistent memory only after receiving real environmental feedback.

Thus, the generated short-term future guides the current control action, while real observations correct long-term history. This information flow separates predictive hypotheses from factual memory, allowing the model to exploit the foresight of a world model while suppressing propagation of generated errors into later segments.

E. Task-Progress-Aware Memory Learning

Long-horizon manipulation requires not only remembering the past but also estimating how far the current state is from task completion. We therefore predict a discrete task-progress distribution $\mathbf{p}_i$ from the aggregated long-term memory:

$$\mathbf{p}_i = \mathrm{Softmax}\left(h_{\mathrm{prog}}(\mathrm{Pool}(\mathbf{R}_i))\right), \qquad \mathbf{p}_i \in \mathbb{R}^B. \quad (12)$$

Here, $\mathrm{Pool}(\cdot)$ aggregates the long-term read sequence, h_{prog} is the progress-prediction head, $\mathrm{Softmax}$ normalizes logits into class probabilities, and B is the number of discrete progress bins.

Given normalized progress $r_i \in [0, 1]$, we define the discrete label as $y_i = \mathrm{bin}(r_i)$. Progress supervision uses cross entropy:

$$\mathcal{L}_{\mathrm{prog}} = \mathrm{CE}(\mathbf{p}_i, y_i). \quad (13)$$

Here, r_i is the completion ratio of segment i within the full task, $\mathrm{bin}(\cdot)$ maps continuous progress to a discrete bin, y_i is the corresponding label, and $\mathrm{CE}(\cdot, \cdot)$ denotes cross-entropy loss.

Fig. 3 illustrates task-progress supervision. The progress head is not an independent planner and does not explicitly predict future observations, actions, or task events. Instead, it provides a global, goal-relative temporal signal that requires the memory to infer the current task stage from completed

historical events. It therefore connects the historical events retained in memory with their implications for the current task stage and the amount of work remaining.

Consequently, MULTI-TYPE HISTORICAL EVENT MEMORY is not a passive storage mechanism. It forms a history representation oriented toward goal completion. Although task progress does not explicitly predict future observations or actions, it provides a goal-relative temporal signal that reflects the implications of completed events for the remaining task. In this sense, progress supervision connects retained historical events with their consequences for future task completion, without requiring explicit long-horizon rollout. The world model explicitly predicts short-term visual and action evolution, while the progress objective provides a global temporal constraint on the current task state. Together, they encourage the current action to remain consistent with both immediate dynamics and overall task completion.

F. Training Objective

To prevent different banks from converging to identical representations, we impose a diversity constraint on aggregated bank features. Let $\bar{\mathbf{m}}_{i,k}$ be the normalized mean representation of bank k, and let $\mathcal{P}$ be the set of all index pairs from different banks. The diversity loss is

$$\mathcal{L}_{\text{div}} = \frac{1}{|\mathcal{P}|} \sum_{(k,l)\in\mathcal{P}} \left(\bar{\mathbf{m}}_{i,k}^{\top} \bar{\mathbf{m}}_{i,l}\right)^2. \tag{14}$$

Here, $|\mathcal{P}|$ is the number of bank pairs, (k, l) denotes a pair of distinct banks, and $(\cdot)^{\top}$ is the vector inner product. This objective encourages banks to retain complementary information and works together with bank-specific queries and independent compression to induce multi-bank specialization.

The final training objective is

$$\mathcal{L} = \mathcal{L}_{\text{video}} + \lambda_a \mathcal{L}_{\text{action}} + \lambda_{\text{div}} \mathcal{L}_{\text{div}} + \lambda_{\text{prog}} \mathcal{L}_{\text{prog}}. \tag{15}$$

Here, $\mathcal{L}_{\text{video}}$ and $\mathcal{L}_{\text{action}}$ are video-prediction and action-generation losses, and λ_a, λ_{div}, and λ_{prog} balance the loss terms. The video and action objectives learn local future prediction and control, the progress objective provides global task-progress supervision, and the diversity objective promotes cross-bank decomposition of long-term history. These four information sources close the loop within a unified model: local history provides the current state, long-term history resolves cross-stage ambiguity, short-term future prediction guides immediate control, and global task-progress supervision constrains decisions to remain consistent with the overall task state.

IV. EXPERIMENTS

A. Experimental Questions

The experiments are designed to answer four questions: whether long-term memory improves success on long-horizon temporally dependent manipulation tasks; whether the gains transfer to real robots; whether multiple banks learn complementary behavior; and how memory-bank structure and progress supervision contribute to performance.

B. Baseline: LingBot-VA

We use LingBot-VA [8] as the base WAM and main baseline. LingBot-VA models future visual states and actions in latent space and uses real observations to correct local state during closed-loop execution, making it a representative recent WAM.

LingBot-VA obtains short-term context from a sliding-window KV cache. Given a language instruction, recent visual observations, and action history, it generates short-term future video latents and corresponding action sequences. After executing the action, the next prediction is conditioned again on the new observation.

This mechanism is suitable for short-horizon closed-loop control but has no explicit long-term memory. Once a key early observation leaves the window, the original LingBot-VA must decide only from the current local state. It therefore provides a clear control: if a task requires out-of-window history, performance degradation should mainly reflect missing long-term information rather than poor local control.

DIM-WAM preserves the video-action joint-modeling interface of LingBot-VA and adds an external fixed-capacity long-term memory. All subsequent protocol audits and major ablations use the same window, chunk, and camera inputs as this base model, so that the benefit of long-term memory can be separated from the local modeling capability of the underlying WAM.

C. RMBench Benchmark

RMBench [14] is a bimanual robot-manipulation benchmark built on RoboTwin 2.0. It contains nine non-Markovian tasks that require historical observations, enabling systematic evaluation of how policies retain, retrieve, and use task-relevant information.

The tasks are divided into five $M(1)$ tasks and four $M(n)$ tasks. The former require retaining one or a fixed small number of key observations, whereas the latter require accumulating multiple historical outcomes through repeated exploration, trial, or interaction.

In `put_back_block`, for example, the robot first moves a block from one of four candidate regions to the center, then presses a button, and finally returns the block to its initial location. At the final stage, the initial location is no longer uniquely determined by the current observation, so the task requires cross-stage preservation of early visual evidence.

As shown in Fig. 4, trajectories from different initial states become highly similar in the middle stages. If the initial observation has entirely left the local context and the four initial positions are balanced in the training data, the chance level for the final target-location choice is 25%. This 25% value refers only to the key position decision, not to the theoretical upper bound of full-task success. Because the remaining operations are relatively stable, the full-task success rate of a history-free policy is expected to approach this level.

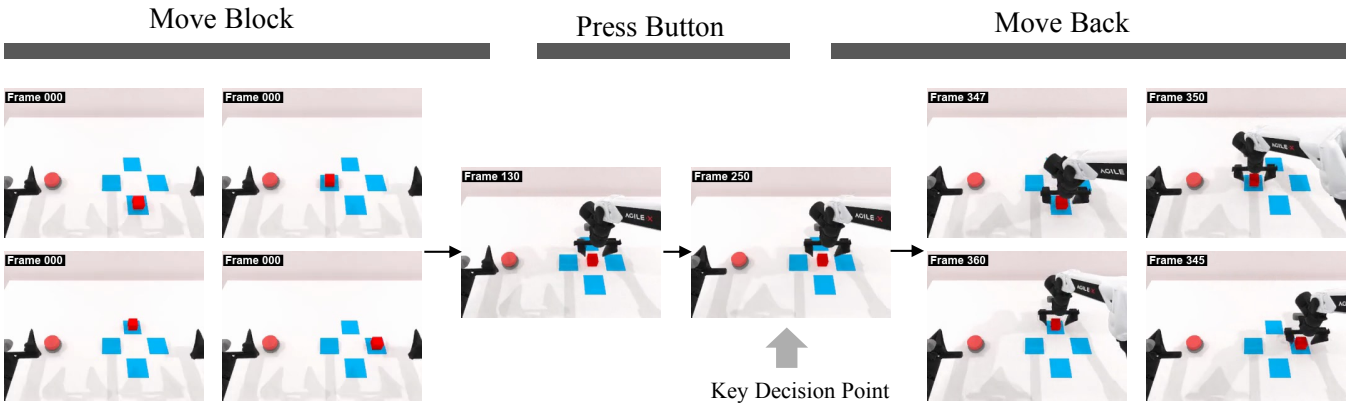


Fig. 4. Key frames of the RMBench `put_back_block` task. Different initial block locations lead to similar intermediate states, while the final move-back target still depends on the initial location that may fall outside the local observation window.

TABLE I
EVALUATION-PROTOCOL AUDIT ON `PUT_BACK_BLOCK`. "CHANCE" DENOTES WHETHER THE INITIAL-POSITION CHOICE CAN BE INFERRED FROM THE LOCAL CONTEXT, NOT THE FULL-TASK THEORETICAL SUCCESS RATE.

Camera views	Window	Stride	Initial visible	Chance	Success
front + wrist + head	30	1	False	25	86
head + front	30	1	False	25	24
head + front	60	1	False	25	28
head + front	120	1	True	100	88
front + wrist + head	30	4	True	100	**100**
head + front	30	4	True	100	98

D. Evaluation-Protocol Audit

Long-horizon memory evaluation can be affected by camera viewpoint, temporal downsampling, and context-window length. Before the main experiments, we therefore audit the evaluation protocol by varying camera combinations, history windows, and video strides.

We analyze the duration of the consistent stage before the key decision point in `put_back_block` training data. Based on the relation between this duration and the LingBot-VA window length, we estimate the ideal success probability under perfect action denoising. This ideal probability refers to the chance of selecting the correct one among four candidate return locations after the button press, assuming no low-level action errors. Table I reports the theoretical and empirical success rates under different settings.

The comparison shows that when the history window or downsampling stride brings the initial observation back into context, success increases sharply, confirming that the task depends on long-term history. In contrast, the short-window head+front setting cannot access the initial location and performs near the 25% chance level.

However, the short-window front+wrist+head setting still reaches high success, indicating that the wrist view may retain shortcut features correlated with the initial position. To reduce potential information leakage, the main RMBench experiments use no wrist view and no temporal downsampling. Unless otherwise noted, our RMBench setting therefore uses head+front views, a 30-frame window, and stride 1; the † marker in Table II denotes results obtained under this audited setting.

E. Implementation Details

In RMBench simulation, LingBot-VA and DIM-WAM are trained and evaluated with head+front views, a 30-frame window, stride 1, and 128×128 image resolution. Except for the camera combinations, window lengths, and strides explicitly modified in the protocol audit, all simulation experiments use this unified protocol.

Both WAM methods use the same optimization configuration: 20 video-denoising steps, 50 action-denoising steps, chunk size 4, AdamW optimizer, learning rate 10^{-5}, batch size 1, gradient accumulation over 10 steps, and 5,000 optimization steps on eight H200 GPUs.

DIM-WAM uses eight banks, each containing 12 memory slots. Memory tokens condition both the video branch and the action branch. During training, we remove historical clean KV entries with probability 0.3 to encourage utilization of memory tokens and reduce the distribution gap between memory tokens and the clean KV cache. The truncation length for backpropagation through time is set to 4 to avoid excessive graph length and memory consumption in long episodes.

Real-robot experiments use a single third-person camera with 224×224 input resolution. LingBot-VA and DIM-WAM follow the WAM training configuration from RMBench except for the input resolution.

The real-robot baselines use their recommended training settings. Fast-WAM is trained for 150 epochs with a cosine learning-rate schedule, base learning rate 10^{-4}, weight decay 10^{-2}, and effective batch size 16. $\pi_{0.5}$ is fine-tuned for 20K steps using bfloat16 precision, batch size 24, a cosine decay schedule, peak learning rate 2.5×10^{-5}, 1K warmup steps, 30K decay steps, and final learning rate 2.5×10^{-6}. Real-robot

experiments use absolute joint-position control with an eight-dimensional action space consisting of seven joint positions and a discretized binary gripper state.

F. Simulation Experiments

We compare against Diffusion Policy [40], ACT [41], $\pi_{0.5}$ [3], X-VLA [42], and Mem-0 [14], the official explicit-memory policy from RMBench. The reported results for these methods are taken from the official RMBench report. For LingBot-VA and DIM-WAM, we use the same RMBench training data as the official protocol, with 50 demonstrations per task. Evaluation is also conducted in the same official environments with the default official random seeds. Each task is evaluated for 100 rollouts, and the reported success rate is the average over these trials. Because training recipes, implementation details, and data usage may still differ across methods, our comparison is intended to test whether the proposed memory design mitigates long-term historical forgetting relative to the base WAM and representative baselines, rather than to claim a fully optimized state-of-the-art result obtained by exhaustively tuning configurations or task-specific tricks.

Table II reports success rates on RMBench long-horizon tasks. In addition to per-task results, we report average success over $M(1)$, $M(n)$, and all tasks.

The results show that DIM-WAM obtains the highest success rate on all RMBench tasks except `Cover Blocks`, and improves the overall average success of LingBot-VA from 28.4% to 69.8%. Compared with the official explicit-memory Mem-0 baseline, DIM-WAM also improves the total average from 42.0% to 69.8%. The gain is especially pronounced on $M(1)$ tasks, where average success increases from 22.8% for LingBot-VA and 52.8% for Mem-0 to 80.6%. On $M(n)$ tasks, which require accumulating multiple historical outcomes, DIM-WAM reaches 56.3%, outperforming LingBot-VA at 35.5% and Mem-0 at 28.5%. These results indicate that the multi-bank structure and progress supervision organize long-term history more effectively.

G. Real-World Experiments

We design four long-horizon manipulation tasks on a Franka Panda platform to evaluate extraction and retrieval of spatial state, event order, target identity, and other historical information. Each task contains multiple consecutive substages, and a single current frame is insufficient to determine the correct action. Fig. 5 shows the task designs and stage partitions.

We compare $\pi_{0.5}$ [3], Fast-WAM [27], LingBot-VA [8], and DIM-WAM. Each task uses 15–25 demonstrations for training, and each method is evaluated in 10 independent trials per task.

Table III reports stage success rate (SSR) and full-task success rate (SR). Full success requires all substages to be completed in order. SSR is the ratio of completed stages to all stages, which distinguishes early control failures from stage-switching errors caused by long-term memory failure.

The real-robot results show that DIM-WAM is never worse than LingBot-VA across the four tasks. It improves average

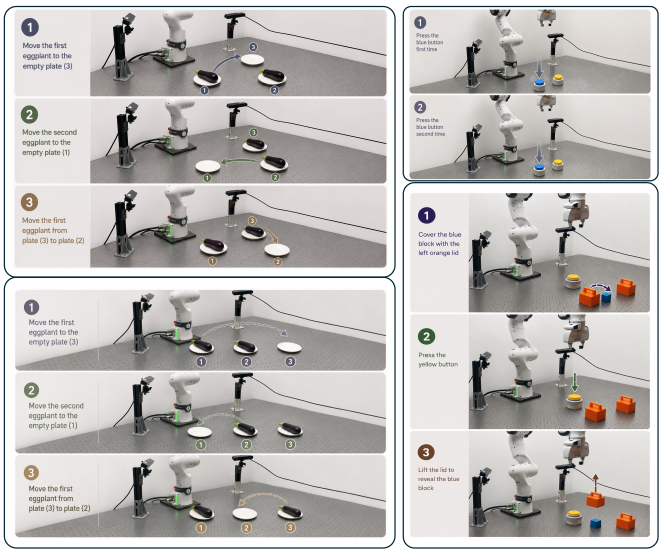


Fig. 5. Real-world Franka Panda tasks. The four panels show Triangle Swap (top-left), Line Swap (bottom-left), Press Twice (top-right), and Find Blue Block (bottom-right).

SSR from 70.7% to 91.5% and average SR from 52.5% to 80.0%. The gains mainly come from tasks requiring cross-stage preservation of target identity or spatial state. For example, full-task success on Find Blue Block increases from 10.0% to 70.0%, and Line Swap improves from 60.0% to 90.0%.

$\pi_{0.5}$ and Fast-WAM almost never complete these few-shot long-horizon real tasks, suggesting that general action priors or efficient short-term WAMs are insufficient for strong history dependence. The Press Twice demonstrations average 70 frames, so the local sliding window already covers about 85% of the maximum length. Both LingBot-VA and DIM-WAM reach 100.0%, indicating that long-term memory does not harm the base control ability when local observations are already sufficient for stage switching.

H. Ablation Studies

We ablate memory-bank structure and progress supervision. Table IV reports success rates on `Swap T` and `Swap Blocks`, two representative tasks used to examine the effects of multi-bank organization, retained event slots, and the progress head on long-term state retention.

Increasing the number of banks and retaining richer event slots consistently improves performance: 1×32 memory reaches 75.5% average success, 4×8 improves to 90.0%, and the default 8×12 reaches 96.5%. This indicates that multi-bank decomposition improves not only capacity but also the organization of different historical events.

Removing the progress head reduces average success from 96.5% to 90.5%, with a larger drop on `Swap Blocks`. This supports the design motivation: long-term memory must preserve past events while also learning a representation of the global task state and progress toward completion.

TABLE II
MAIN SIMULATION RESULTS ON RMBENCH LONG-HORIZON TASKS. VALUES ARE SUCCESS RATES IN PERCENT. TMC DENOTES THE RMBENCH $M(1)/M(n)$ TASK-MEMORY CATEGORY.

Task	TMC	DP	ACT	$\pi_{0.5}$	X-VLA	Mem-0	LingBot-VA†	DIM-WAM†
Observe and Pick Up	$M(1)$	1.0	1.0	9.0	9.0	4.0	4.0	**13.0**
Rearrange Blocks	$M(1)$	0.0	29.0	13.0	13.0	89.0	27.0	**99.0**
Put Back Block	$M(1)$	0.0	0.0	11.0	18.0	90.0	24.0	**98.0**
Swap Blocks	$M(1)$	11.0	2.0	24.0	16.0	67.0	34.0	**96.0**
Swap T	$M(1)$	20.0	2.0	15.0	3.0	14.0	25.0	**97.0**
Average	$M(1)$	6.4	6.8	14.4	11.8	52.8	22.8	**80.6**
Battery Try	$M(n)$	10.0	19.0	16.0	26.0	28.0	33.0	**48.0**
Blocks Ranking Try	$M(n)$	10.0	0.0	6.0	1.0	18.0	48.0	**87.0**
Cover Blocks	$M(n)$	0.0	0.0	0.0	2.0	**68.0**	42.0	56.0
Press Button	$M(n)$	0.0	0.0	0.0	0.0	0.0	19.0	**34.0**
Average	$M(n)$	5.0	4.8	5.5	7.3	28.5	35.5	**56.3**
Total Average	–	5.8	5.9	10.4	9.8	42.0	28.4	**69.8**

TABLE III
REAL-WORLD FRANKA PANDA RESULTS. VALUES ARE PERCENTAGES. SSR DENOTES STAGE SUCCESS RATE AND SR DENOTES FULL-TASK SUCCESS RATE.

Method	Find Blue Block		Line Swap		Triangle Swap		Press Twice		Avg.	
	SSR	SR	SSR	SR	SSR	SR	SSR	SR	SSR	SR
$\pi_{0.5}$	0.0	0.0	5.0	0.0	0.0	0.0	0.0	0.0	1.3	0.0
Fast-WAM	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
LingBot-VA	52.5	10.0	78.4	60.0	51.8	40.0	**100.0**	**100.0**	70.7	52.5
DIM-WAM(ours)	**92.5**	**70.0**	**95.0**	**90.0**	**78.4**	**60.0**	**100.0**	**100.0**	**91.5**	**80.0**

TABLE IV
ABLATION STUDY ON MEMORY STRUCTURE AND PROGRESS SUPERVISION. VALUES ARE SUCCESS RATES IN PERCENT.

Variant	Swap T	Swap Blocks	Avg.
1×32 memory	71.0	80.0	75.5
4×8 memory	88.0	92.0	90.0
8×12 memory	**97.0**	**96.0**	**96.5**
8×12 w/o prog head	92.0	89.0	90.5

I. Cross-Bank Behavior Analysis

After structural ablations show that multi-bank memory outperforms fewer-bank variants, we further visualize the memory behavior of the default model in one successful `put_back_block` episode. The goal is to determine whether the performance gain corresponds to observable bank specialization.

Fig. 6 shows the final retained events in each bank and the representation-space structure of memory tokens from the same episode.

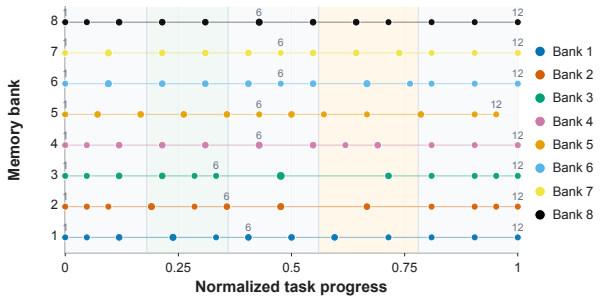
The timeline shows that different banks do not retain identical event positions in the same trajectory. Some banks tend to preserve early events or events near stage boundaries, while others retain middle- or late-stage state changes. Larger retained mass is often concentrated near stage transitions, suggesting that compression increases the weight of key events in long-term memory.

The PCA visualization further shows that memory tokens from different banks form distinguishable distributions while retaining some overlap. This observation is consistent with Table IV: multi-bank memory does not merely duplicate the same history. Instead, it learns complementary event-selection behavior and representation subspaces within a shared episode.

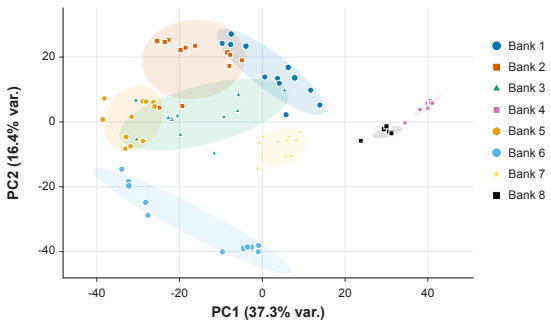
V. CONCLUSION

We presented DIM-WAM, a memory-augmented world-action model for long-horizon robot manipulation. The central claim is that long-horizon decision making should not rely solely on a longer local window; it must jointly organize local history, cross-stage history, immediate future dynamics, and global task-progress information.

DIM-WAM realizes this view through MULTI-TYPE HISTORICAL EVENT MEMORY, which extracts compact visual event information from real observations, updates multiple banks through independent similarity-based merging, and reads the retained tokens with bank-identity and RoPE time embeddings. With independent compression, cooperative reading, and updates from real observations, the memory supports future visual prediction and action generation under a fixed capacity. This design alleviates competition among heterogeneous information in a single memory bank and reduces the risk that generated prediction errors contaminate long-term state.



(a) Retained memory events over task progress.



(b) PCA of memory tokens.

Fig. 6. Memory behavior during one complete episode. (a) Bank-wise timeline of retained memory events over normalized task progress. Each row corresponds to one memory bank, each dot denotes a retained event token, and annotated numbers indicate retained mass when shown. The shaded vertical bands separate successive task phases. (b) Principal component analysis (PCA) of memory tokens from the same episode. Colors and marker shapes indicate different memory banks, while translucent ellipses summarize the bank-wise token distributions in the PCA space.

Experiments validate the effectiveness of this design on temporally dependent long-horizon tasks. On RMBench, DIM-WAM improves the average success rate of LingBot-VA from 28.4% to 69.8%, surpassing the explicit-memory Mem-0 baseline at 42.0%. On real Franka Panda tasks, it improves average stage success from 70.7% to 91.5% and full-task success from 52.5% to 80.0%, with particularly large gains on tasks such as Find Blue Block and Line Swap that require cross-stage preservation of target identity or spatial state. The RMBench protocol audit further shows that when the initial observation cannot be recovered from the local window, short-window policies approach the chance level of the key decision; when the initial state is reintroduced through a longer window or temporal downsampling, success rises substantially. Ablations show that the default 8×12 multi-bank memory reaches 96.5% average success on representative tasks, compared with 75.5% for a 1×32 memory, and removing the progress head reduces performance to 90.5%. These results suggest that fixed-capacity long-term memory benefits from both cross-bank event decomposition and task-progress-aware supervision when supporting future prediction and action generation in long-horizon tasks.

REFERENCES

[1] "Rt-2: Vision-language-action models transfer web knowledge to robotic control," 2023. [Online]. Available: https://arxiv.org/abs/2307.15818

[2] "Openvla: An open-source vision-language-action model," 2024. [Online]. Available: https://arxiv.org/abs/2406.09246

[3] "pi0: A vision-language-action flow model for general robot control," 2024. [Online]. Available: https://arxiv.org/abs/2410.24164

[4] "Rdt-1b: A diffusion foundation model for bimanual manipulation," 2024. [Online]. Available: https://arxiv.org/abs/2410.07864

[5] "Gr00t n1: An open foundation model for generalist humanoid robots," 2025. [Online]. Available: https://arxiv.org/abs/2503.14734

[6] "Gigaworld-policy: An efficient action-centered world-action model," 2026. [Online]. Available: https://arxiv.org/abs/2603.17240

[7] "Gr-2: A generative video-language-action model with web-scale knowledge for robot manipulation," 2024. [Online]. Available: https://arxiv.org/abs/2410.06158

[8] "Causal world modeling for robot control (lingbot-va)," 2026. [Online]. Available: https://arxiv.org/abs/2601.21998

[9] "World action models are zero-shot policies (dreamzero)," 2026. [Online]. Available: https://arxiv.org/abs/2602.15922

[10] "Sam2act: Integrating visual foundation model with a memory architecture for robotic manipulation," 2025. [Online]. Available: https://arxiv.org/abs/2501.18564

[11] "Memoryvla: Perceptual-cognitive memory in vision-language-action models for robotic manipulation," 2025. [Online]. Available: https://arxiv.org/abs/2508.19236

[12] "Mem: Multi-scale embodied memory for vision language action models," 2026. [Online]. Available: https://arxiv.org/abs/2603.03596

[13] "Remem-vla: Empowering vision-language-action model with memory via dual-level recurrent queries," 2026. [Online]. Available: https://arxiv.org/abs/2603.12942

[14] T. Chen, Y. Wang, M. Li, Y. Qin, H. Shi, Z. Li, Y. Hu, Y. Zhang, K. Wang, Y. Chen *et al.*, "RMBench: Memory-dependent robotic manipulation benchmark with insights into policy design," *arXiv preprint arXiv:2603.01229*, 2026. [Online]. Available: https://arxiv.org/abs/2603.01229

[15] "Spatialvla: Exploring spatial representations for visual-language-action model," 2025. [Online]. Available: https://arxiv.org/abs/2501.15830

[16] "Bridgevla: Input-output alignment for efficient 3d manipulation learning with vision-language models," 2025. [Online]. Available: https://arxiv.org/abs/2506.07961

[17] "Eventvla: Event-driven visual evidence memory for long-horizon vision-language-action policies," 2026. [Online]. Available: https://arxiv.org/abs/2606.20092

[18] "Memoryvla++: Temporal modeling via memory and imagination in vision-language-action models," 2026. [Online]. Available: https://arxiv.org/abs/2606.09827

[19] "Learning universal policies via text-guided video generation," 2023. [Online]. Available: https://arxiv.org/abs/2302.00111

[20] "Robodreamer: Learning compositional world models for robot imagination," 2024. [Online]. Available: https://arxiv.org/abs/2404.12377

[21] "Gen2act: Human video generation in novel scenarios enables generalizable robot manipulation," 2024. [Online]. Available: https://arxiv.org/abs/2409.16283

[22] "Video prediction policy: A generalist robot policy with predictive visual representations," 2024. [Online]. Available: https://arxiv.org/abs/2412.14803

[23] "Predictive inverse dynamics models are scalable learners for robotic manipulation," 2024. [Online]. Available: https://arxiv.org/abs/2412.15109

[24] "Cosmos policy: Fine-tuning video models for visuomotor control and planning," 2026. [Online]. Available: https://arxiv.org/abs/2601.16163

[25] "Dit4dit: Jointly modeling video dynamics and actions for generalizable robot control," 2026. [Online]. Available: https://arxiv.org/abs/2603.10448

[26] "World guidance: World modeling in condition space for action generation," 2026. [Online]. Available: https://arxiv.org/abs/2602.22010

[27] "Fast-wam: Do world action models need test-time future imagination?" 2026. [Online]. Available: https://arxiv.org/abs/2603.16666

[28] “Streamingt2v: Consistent, dynamic, and extendable long video generation from text,” 2024. [Online]. Available: https://arxiv.org/abs/2403.14773
[29] “Ssm meets video diffusion models: Efficient long-term video generation with selective state spaces,” 2024. [Online]. Available: https://arxiv.org/abs/2403.07711
[30] “Long context state space video world models,” 2025. [Online]. Available: https://arxiv.org/abs/2505.20171
[31] “Sana-video: Efficient video generation with block linear diffusion transformer,” 2025. [Online]. Available: https://arxiv.org/abs/2509.24695
[32] “Malt diffusion: Memory-augmented latent transformers for any-length video generation,” 2025. [Online]. Available: https://arxiv.org/abs/2502.12632
[33] “Memorize-and-generate: Towards long-term consistency in real-time video generation,” 2025. [Online]. Available: https://arxiv.org/abs/2512.18741
[34] “Videossm: Autoregressive long video generation with hybrid state-space memory,” 2025. [Online]. Available: https://arxiv.org/abs/2512.04519
[35] “Context forcing: Consistent autoregressive video generation with long context,” 2026. [Online]. Available: https://arxiv.org/abs/2602.06028
[36] “Relax forcing: Relaxed kv-memory for consistent long video generation,” 2026. [Online]. Available: https://arxiv.org/abs/2603.21366
[37] “Memflow: Flowing adaptive memory for consistent and efficient long video narratives,” 2025. [Online]. Available: https://arxiv.org/abs/2512.14699
[38] “Videomemory: Toward consistent video generation via memory integration,” 2026. [Online]. Available: https://arxiv.org/abs/2601.03655
[39] “Slotmemory: Object-centric kv memory for streaming long-video generation,” 2026. [Online]. Available: https://arxiv.org/abs/2605.31033
[40] “Diffusion policy: Visuomotor policy learning via action diffusion,” 2023. [Online]. Available: https://arxiv.org/abs/2303.04137
[41] “Learning fine-grained bimanual manipulation with low-cost hardware,” 2023. [Online]. Available: https://arxiv.org/abs/2304.13705
[42] “X-vla: Soft-prompted transformer as scalable cross-embodiment vision-language-action model,” 2025. [Online]. Available: https://arxiv.org/abs/2510.10274