REASONING TO REMEMBER: STRUCTURED KEYFRAME MEMORY FOR LONG-HORIZON ROBOTIC MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Long-horizon robotic manipulation requires historical information to guide decisions beyond what current observations reveal. Keyframe-based memory provides visual context for a high-level vision-language model (VLM), which predicts subgoals for a low-level vision-language-action (VLA) policy. However, whether selecting better keyframes or retaining more frames effectively improves planning remains unclear. To investigate this, we replace predicted keyframes with ground-truth keyframes and separately double the keyframe output frequency. Ground-truth keyframes do not improve average success, while increased frequency yields only modest gains. Further analysis identifies two limitations that visual selection alone does not directly address. First, task-relevant conclusions remain implicit in visual content, requiring subsequent planning to reconstruct them even when the original supporting context has changed or become unavailable. Second, keyframe selection can discard intervening changes and interactions, leaving distinct histories indistinguishable from the retained snapshots. To address these limitations, we propose R2Mem (*Reasoning to Remember*), built around Structured Keyframe Memory (SKM). SKM pairs keyframes with reasoning text organized around task-relevant key events. These events specify which object changes and completed operations to record beyond what individual snapshots convey. The records form observation memory and action memory, explicitly preserving object histories and execution progress, respectively. The high-level VLM jointly constructs SKM and predicts subgoals using these explicit records. Experiments on RoboMME, RoboMemArena, and real-world manipulation tasks show that R2Mem improves average task success over memory-augmented baselines, highlighting the value of explicitly recording task-relevant events for subsequent subgoal planning. The code will be released.

1 INTRODUCTION

Vision-language-action (VLA) models have advanced language-conditioned robotic control, enabling diverse manipulation skills across tasks and embodiments (Kim et al., 2024; Black et al., 2024; Qu et al., 2025; Kim et al., 2025). Long-horizon manipulation additionally requires interpreting task intent, organizing skills into subgoals, and revising these subgoals in response to execution feedback. Dual-system architectures address these demands by coupling a high-level vision-language model (VLM) that predicts language subgoals with a low-level VLA that executes them (Shi et al., 2025; Gemini Robotics Team et al., 2025). This separation allows the VLM to reason over task-level objectives and adapt the sequence of operations while reusing the VLA's learned motor skills. Effective subgoal planning, however, must account for partial observability: earlier object changes and completed operations can determine the next subgoal without being recoverable from the current observation. Current observations therefore need not provide a Markovian state for planning, making memory essential for preserving context for subsequent decisions.

Robotic policies incorporate history through multi-frame features, visual traces, and memories of executed actions (Li et al., 2026b; Zheng et al., 2025; Li et al., 2026d). For high-level VLM planning, providing the complete observation history is costly and introduces redundant views. Keyframe memory offers a compact alternative, and existing approaches seek to improve it by selecting his-

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

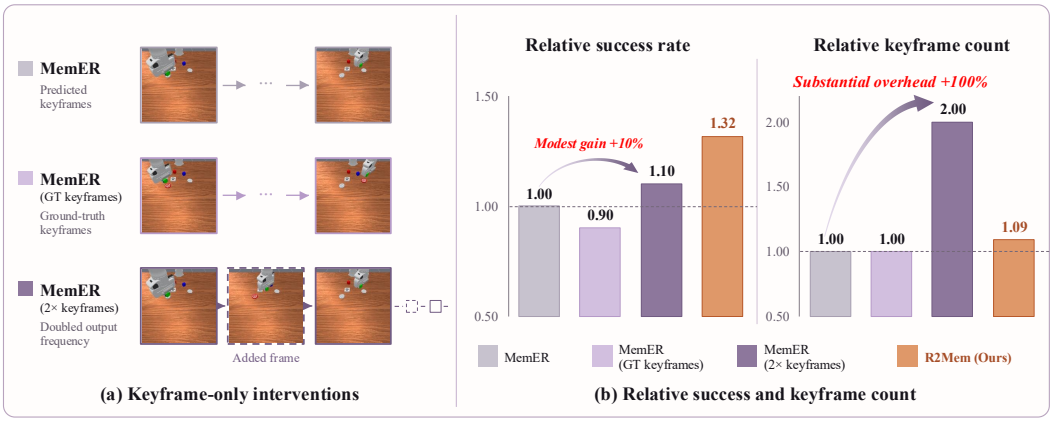


Figure 1: Keyframe-only controls on all RoboMME tasks with ground-truth keyframes available at evaluation, covering all task suites. Success rates and keyframe counts are averaged equally across tasks and normalized to MemER.

torical images that are more relevant to subsequent decisions (Sridhar et al., 2026; Lei et al., 2026). The extent to which better or denser visual selection improves planning, however, remains unclear.

In this paper, we investigate the effectiveness of visual-only keyframe memory for long-horizon planning. Our experiments reveal that selecting better keyframes or retaining more visual evidence offers limited benefits to task performance, as illustrated in Figure 1. These findings suggest that effective memory construction requires more than improving the visual evidence retained. To explain this discrepancy, we analyze the limitations of representing history solely through selected images. First, *task-relevant conclusions remain implicit*: retaining an image does not explicitly preserve conclusions drawn from its original observation context. Subsequent planning must infer these conclusions again, although changes in context and accumulated interactions can make the required reasoning increasingly complex. Information that was readily established earlier may therefore become difficult to recover later. Second, *temporal information can be lost through selection*: retained snapshots may omit task-relevant changes and interactions between them. Distinct histories can therefore produce similar retained images while requiring different next subgoals.

To address these limitations, we propose R2Mem (*Reasoning to Remember*), built around Structured Keyframe Memory (SKM). SKM pairs keyframes with reasoning text that explicitly preserves conclusions drawn from the observation history. Task-relevant key events organize this text, specifying which object changes and completed operations to record across the sequence. These records can preserve process information beyond what the associated images convey. They comprise *observation memory*, describing relevant objects and their evolution, and *action memory*, recording completed task steps. Shared object references link records across time and to visual evidence. The high-level VLM jointly constructs SKM and predicts subgoals using the accumulated records.

Our main contributions are:

- Controlled experiments reveal that improving visual keyframe selection alone yields limited gains. Our analysis identifies two representational limitations: task-relevant conclusions remain implicit, and frame selection can discard task-relevant temporal information.
- We propose R2Mem, built around Structured Keyframe Memory (SKM), which pairs keyframes with reasoning text to explicitly preserve conclusions drawn from observation history. Task-relevant key events organize these records to capture object histories and completed operations for subsequent subgoal planning.
- Experiments demonstrate improved performance on memory-dependent manipulation. Our findings motivate a broader objective for robotic memory construction: preserving both historical evidence and reusable reasoning outcomes to support future decisions.

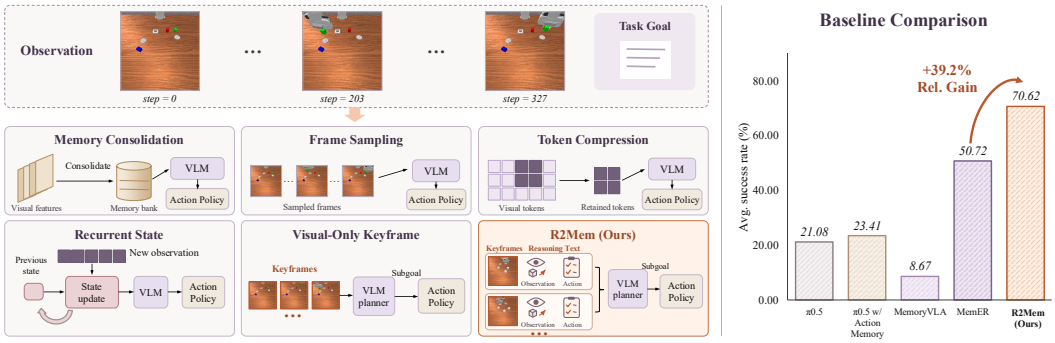


Figure 2: Memory designs (left) and comparison of average success rates across methods (right).

2 Related Work

2.1 Dual-System Vision-Language-Action Architectures

Dual-system architectures couple a high-level planner with a low-level executor. Here, a VLM interprets the task and visual context to predict a language subgoal, which a VLA policy executes through motor actions. This decomposition provides a semantic interface between task reasoning and physical control, allowing planners to revise task-level decisions while executors handle local interactions. Several recent systems adopt this architecture. RT-H establishes language as an intermediate representation for hierarchical action prediction, using language motions to connect task instructions with motor control (Belkhale et al., 2024). At the task level, Hi Robot uses a high-level VLM to interpret complex instructions and situated feedback, generating instructions for a low-level policy (Shi et al., 2025). Gemini Robotics 1.5 combines embodied reasoning and task orchestration with a VLA executing natural-language instructions (Gemini Robotics Team et al., 2025).

Recent research further examines how these components should interact. A systematic study of hierarchical VLA agents identifies planner and controller choices, subgoal-switching mechanisms, and context representations as important design factors (Hu et al., 2026). Cortex studies the planning–execution interface through executable subtask representations and alignment between high-level VLM planning and low-level VLA control (Peng et al., 2026). These architectures provide our setting: the high-level planner uses task context to select the next subgoal.

2.2 Memory for Robotic Decision-Making

Memory lets robotic policies use information no longer available in the current observation. Benchmarks such as RoboMME explicitly evaluate this dependence on historical observations and interactions (Dai et al., 2026). Existing methods incorporate history through several complementary representations. SAM2Act+ extends SAM2Act with a visual memory architecture (Fang et al., 2025), while HAMLET encodes observations into compact moment tokens and aggregates them into features for action prediction (Koo et al., 2026). NativeMEM compresses historical observations into memory tokens using the VLA's native visual encoder (Wang et al., 2026). Recurrent approaches propagate information across time: ReMem-VLA maintains frame-level and chunk-level memory queries (Li et al., 2026a), and Embodied-SlotSSM preserves object-centric histories through temporally consistent slots and state-space modeling (Chung et al., 2026).

Other approaches explicitly select and retrieve historical information. MemoryVLA maintains a perceptual-cognitive memory bank that retrieves and integrates historical visual details and semantic features for action generation (Shi et al., 2026). EventVLA retains sparse visual evidence through an event-driven keyframe memory (Yang et al., 2026). TRM-VLA combines keyframe-triggered chain-of-thought reasoning with a context memory of historical reasoning features (Li et al., 2026c). These methods preserve visual, semantic, and temporal context for action prediction.

In hierarchical systems, memory also supplies context for high-level subgoal planning. Drawing on the memory mechanisms examined in RoboMME (Dai et al., 2026), Figure 2 (left) summarizes representative designs based on memory consolidation, frame sampling, token compression, recurrent

states, and visual-only keyframes. MemER trains a high-level policy to select and track relevant historical keyframes, using them alongside recent observations to generate instructions for a low-level VLA (Sridhar et al., 2026). PrediMem, introduced with RoboMemArena, maintains recent-frame and keyframe buffers for a VLM planner and uses predictive coding during training to improve sensitivity to task dynamics (Lei et al., 2026). Semantic and structured representations provide additional ways to organize planning context. MEM combines short-term video memory for control with recursively updated textual memory for high-level subtask prediction (Torne et al., 2026). HiMe separates execution, monitoring, and planning into three layers, maintaining multimodal contextual memory and procedural memory that tracks subgoal status (Ji et al., 2026). BATON stores solutions acquired through subtask exploration and uses transition-aware memory to coordinate policy invocation and the handoff between successive subtasks (Xu et al., 2026). Our work focuses on memory construction for high-level subgoal prediction within this hierarchical planning-and-execution setting.

3 Background: Memory for Subgoal Planning

Following the dual-system architecture discussed in Section 2.1, we consider a high-level VLM planner coupled with a low-level VLA executor. The planner uses the task instruction, recent observations, and accumulated memory to predict the next subgoal; the executor translates that subgoal and current sensory inputs into robot actions. Memory provides historical evidence that is no longer available in recent observations but remains relevant to the planning decision.

3.1 Keyframes as Visual Memory

Keyframe memory retains selected historical observations as visual evidence for subsequent planning, reducing the redundancy of dense observation histories while preserving object appearance and spatial configurations. Existing approaches seek to improve this memory by selecting frames that are more relevant to subsequent decisions (Sridhar et al., 2026; Lei et al., 2026).

Given a task instruction g and visual history $\mathcal{H}_t$, a keyframe selector S constructs memory as

$$\mathcal{K}_t = S(\mathcal{H}_t, g), \qquad \mathcal{M}_t^{\text{vis}} = [I_\tau]_{\tau \in \mathcal{K}_t}, \tag{1}$$

where $\mathcal{K}_t$ contains the selected frame indices in temporal order. Under this formulation, improving memory through visual selection amounts to adjusting the history that retained keyframes cover.

We test whether improving keyframe selection or retaining more keyframes yields more effective memory. Specifically, we replace predicted keyframes with ground-truth keyframes and, in a separate control, double the keyframe output frequency. As shown in Figure 1, ground-truth keyframes do not improve average success, while doubling the keyframe output frequency produces only a modest gain. These limited gains motivate examining what task-relevant information remains implicit in, or absent from, the retained images—and why these issues are not directly addressed by the visual-only formulation in Equation 1.

3.2 Limitations of Visual-Only Memory

Implicit task semantics. Retaining an observation does not explicitly record what it establishes about the task. Object identities, changes in their states and relations, and evidence of task progress remain encoded in visual content and its temporal context. Subsequent planning must reconstruct these conclusions, although supporting context may no longer be available and intervening interactions can lengthen the reasoning chain. Conclusions established when an event was observed may therefore be difficult to recover later. More relevant keyframes can support this inference, but selecting them does not preserve the conclusions drawn from the original context.

Temporal information lost through selection. Selected keyframes preserve scene configurations but may omit intervening transitions and interactions. Distinct histories can therefore yield similar retained views while requiring different decisions. For example, two visually similar objects may exchange positions without an apparent change in the endpoint scenes, even though the location of the task-relevant object has changed. Additional or better-selected frames may provide the missing evidence, but improving visual coverage alone does not explicitly ensure that the relevant event is recorded. Absent evidence cannot be reliably recovered by further reasoning over retained images.

These limitations motivate complementing keyframe selection with explicit records of task-relevant conclusions and temporal information that retained snapshots may not convey.

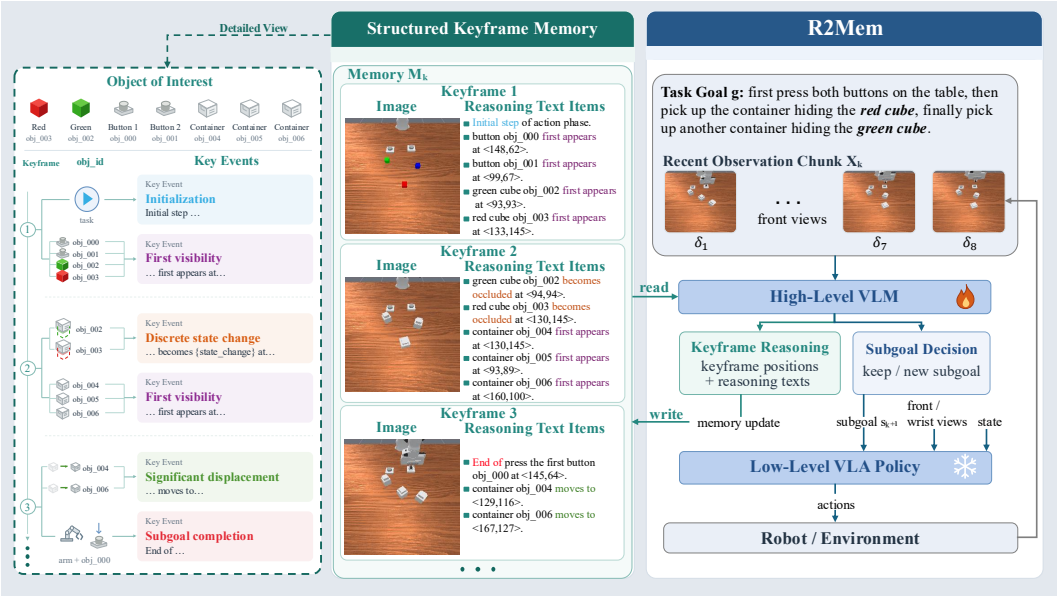


Figure 3: R2Mem overview: key events (left), SKM (center), and planning and execution (right).

4 R2MEM: REASONING TO REMEMBER

R2Mem addresses the two limitations of visual-only memory through explicit reasoning records and task-relevant key events. To address implicit task semantics, reasoning text explicitly records conclusions drawn from the observation history, making them available to subsequent planning without requiring their reconstruction from a potentially different context. To address temporal information lost through keyframe selection, we define key events that capture object changes and completed task steps across the observed sequence. These events specify what to preserve beyond individual snapshots. Pairing the resulting reasoning records with their associated images yields Structured Keyframe Memory (SKM), illustrated in Figure 3.

4.1 REASONING TEXT FOR TASK-RELEVANT EVENTS

Constructing reasoning records involves identifying task-relevant entities, associating their observations across time, and interpreting changes and execution progress. We organize the resulting information as observation memory and action memory: the former describes how task-relevant entities have evolved, while the latter records which task steps have been completed.

Observation memory. Observation memory explicitly records task-relevant entities and their evolution over time. We identify objects of interest from the object categories and constraints specified by the task instruction. Persistent object references associate observations of the same instance across time. The records describe changes in these objects' spatial configurations and task-relevant states, preserving their history for subsequent planning.

Action memory. Action memory records completed operations, their associated objects, and temporal order. An ongoing subgoal expresses the intended operation, whereas a completion record establishes progress through the task. This distinction matters because the same scene configuration can occur at different stages of execution. Completion records let the planner determine which operations remain, whether to repeat an operation, and when to advance to the next stage. They complement observation memory by linking scene evolution to execution history.

Recording key events. We define five key events to specify which information to extract from the observed sequence. Initialization and first visibility establish observation references; displacement and state changes update observation memory; subgoal completion supplies action memory. Let τ index observations sampled within a demonstration or execution phase and i an object of interest.

- *Stream initialization* occurs at $\tau = \tau_0$, where τ_0 is the first observation of the phase. The record identifies the phase beginning and retains its first image as an initial reference.
- *First visibility* establishes an object's identity and location when it first becomes observable in the phase. Let $v_i(\tau)$ indicate observability with a valid location. First visibility occurs at

$$\tau_i^{\text{first}} = \min\{\tau : v_i(\tau) = 1\}. \tag{2}$$

  The record links the object reference to its initial location, starting its history.
- *Significant displacement* records updated object locations when their accumulated displacement from recorded references reaches a threshold. Let $\mathbf{p}_i(\tau)$ denote an object's image-space location, a_i its most recent recorded reference frame, and $\mathcal{C}_\tau$ the set of eligible objects with valid current and reference locations. The displacement criterion is

$$D(\tau) = \sum_{i \in \mathcal{C}_\tau} \left\| \mathbf{p}_i(\tau) - \mathbf{p}_i(a_i) \right\|_2 \geq \delta_{\text{move}}. \tag{3}$$

  Here δ_{move} controls when accumulated spatial changes warrant a new record. The records explicitly associate the affected objects with their updated locations.
- *Discrete state change* records a transition in an object's task-relevant discrete state. Let $s_i(\tau)$ denote this state and τ^- the preceding sampled observation. An event satisfies

$$s_i(\tau) \neq s_i(\tau^-). \tag{4}$$

  Each record identifies the object and its resulting state for a task-relevant transition.
- *Subgoal completion* records completed task steps and their associated objects. Let u_k denote the kth task step and e_k its completion time. The action events at τ are

$$\mathcal{E}_\tau^{\text{act}} = \{\text{End}(u_k) : e_k = \tau\}. \tag{5}$$

  Each completion record includes the operation and its object references. Their temporal sequence preserves execution order, including separate occurrences of the same operation.

The event records explicitly preserve object histories and execution progress inferred from the observation sequence, including information that the associated images alone may not convey.

4.2 Structured Keyframe Memory

SKM organizes the event records and their visual evidence into a sequence of image–text entries:

$$\mathcal{M}_t = \left[(I_{\tau_j}, \mathcal{R}_j) \right]_{j=1}^{N_t}. \tag{6}$$

Here t indexes high-level queries, N_t is the number of accumulated entries, I_{τ_j} is the selected observation at frame index τ_j, and $\mathcal{R}_j$ is a non-empty collection of reasoning records associated with that image. An entry can contain both observation and action records. The image preserves visual evidence, while the text explicitly states task-relevant information from the observation history.

Object references connect the records to their visual evidence. Each referenced object is assigned a persistent textual identifier of the form `obj_NNN`, and described by a semantic name. The identifier links observations of the same instance across entries; the semantic name specifies its category. An image-space point provides spatial grounding where applicable. Observation and action records share these identifiers to link object changes and completed operations to the same entities.

The event definitions specify the targets used to supervise memory construction. The conditions follow an OR rule: any one event can justify a record. When several events correspond to the same image, their records are aggregated into a single entry. A recent observation sequence can therefore contribute zero, one, or several entries. These targets are constructed from trajectory annotations; at deployment, the VLM predicts the entries from its available visual and textual inputs.

At each execution-phase query, the VLM reads the task instruction, recent observations, the ongoing subgoal, and the accumulated SKM. It jointly predicts new memory entries and a decision to retain or update the subgoal. Each new entry specifies a frame position within the recent observations and its reasoning records; the corresponding observed image is then paired with the text and appended to memory in temporal order. The resulting subgoal is passed to the low-level VLA for execution. Memory writes and subgoal transitions are separate decisions, allowing relevant changes to be recorded while the current operation continues.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Benchmarks and tasks. We evaluate R2Mem on RoboMME (Dai et al., 2026) and RoboMemArena (Lei et al., 2026), two manipulation benchmarks requiring information from observation history to guide subsequent decisions. RoboMME comprises 16 tasks organized into Counting, Permanence, Reference, and Imitation, assessing temporal, spatial, object, and procedural memory through prior interactions or demonstrations. We exclude three tasks that are particularly sensitive to low-level VLA execution noise to focus the evaluation on the high-level VLM's ability to construct memory and predict subsequent subgoals from memory and current observations. RoboMemArena complements this evaluation with 26 tasks covering transferring, occlusion, counting, and sequential execution. Its trajectories average over 1,000 control steps, requiring agents to retain object information and execution progress across extended sequences of manipulation.

Baselines. We compare R2Mem with dual-system methods that predict language subgoals for low-level VLA execution, alongside VLA policies and memory-augmented approaches. GroundSG (Gemini) and GroundSG (QwenVL) predict grounded subgoals from current observations and prior subgoals, using prompted Gemini and task-fine-tuned QwenVL, respectively. MemER (Sridhar et al., 2026) uses selected historical keyframes for subgoal prediction. On RoboMemArena, we include the pretrained Qwen3-VL-8B (Bai et al., 2025) and closed-source GPT-5.4 planners, and the official PrediMem planner, which maintains recent-frame and keyframe buffers and uses predictive training supervision (Lei et al., 2026). GroundSG (GT) and RoboMemArena's Ground Truth configuration use ground-truth subgoals as oracle references. Other comparisons include $\pi_{0.5}$ (Physical Intelligence et al., 2025) and its past-action variant, uniform frame sampling (FrameSamp+Modul), token compression (TokenDrop+Modul), recurrent memory (RMT+Modul and TTT+Expert), MemoryVLA (Shi et al., 2026), SAM2Act+ (Fang et al., 2025), and HiF-VLA (Lin et al., 2026). Baseline results follow official reports and repositories.

Implementation. For R2Mem, we keep the low-level VLA frozen in all experiments, using the pretrained policy provided by each benchmark to evaluate memory improvements under fixed low-level execution. To supervise reasoning-text generation, we extract key events using the construction code and raw data of both benchmarks, and convert them into textual records with event-specific templates detailed in the supplementary material. We match each benchmark's VLM initialization and training hyperparameters: Qwen3-VL-4B on RoboMME and Qwen3-VL-8B on RoboMemArena, trained for 2 epochs.

5.2 MAIN RESULTS AND ANALYSIS

Performance on RoboMME. Table 1 shows that R2Mem achieves **70.62%** average success, improving on MemER by **+19.9** percentage points. The gains are particularly pronounced on VideoUnmaskSwap and ButtonUnmaskSwap, where visually similar containers exchange positions after the target becomes hidden. These tasks require maintaining object correspondence through intervening changes, since the final scene alone does not identify the target. The results are consistent with the role of explicit event records in preserving object histories for subsequent subgoal prediction. R2Mem does not outperform MemER on PickHighlight and MoveCube. A possible explanation is that, when retained images already make the relevant visual cue or demonstrated motion clear, textual records provide less additional information and can introduce errors in interpretation.

Table 1: Success rates (%) on RoboMME tasks. R2Mem results are averaged over runs with seeds aligned with the official evaluation settings. Bold indicates the best non-oracle result in each column.

	Counting			Permanence				Reference				Imitation		
Method	Bin Fill	Pick Xtimes	Swing Xtimes	Video Unmask	Button Unmask	Video Unmask Swap	Button Unmask Swap	Pick Highlight	Video Repick	Video Place Button	Video Place Order	Move Cube	Pattern Lock	AVG
Dual System: Oracle Subgoals														
GroundSG (GT)	85.78	100.00	100.00	98.78	95.00	99.22	80.22	83.33	97.33	100.00	100.00	87.78	97.00	94.19
Dual System: Subgoal History														
GroundSG (Gemini)	26.00	18.00	4.00	36.00	14.00	13.00	0.00	9.00	17.00	12.00	7.00	17.00	7.00	13.85
GroundSG (QwenVL)	52.00	92.67	7.33	88.67	24.00	30.67	14.00	15.11	25.33	54.00	31.78	71.56	6.67	39.52
Uniform Frame Sampling														
FrameSamp+Modul	39.56	87.33	**92.00**	32.67	25.11	24.44	18.22	22.89	30.44	60.00	32.00	77.78	53.56	45.85
Token Dropping														
TokenDrop+Modul	34.44	83.56	86.00	28.22	29.33	28.44	21.33	21.33	22.00	59.56	36.00	62.00	32.44	41.90
Recurrent State Update														
RMT+Modul	33.33	60.78	37.78	31.11	11.78	17.78	2.44	17.11	4.22	32.00	31.11	24.67	3.78	23.68
TTT+Expert	34.89	63.78	41.33	31.78	22.44	19.56	18.00	12.22	9.56	34.00	22.89	33.56	3.11	26.70
Similarity-Based Memory Consolidation														
MemoryVLA (ICLR 2026)	10.00	17.33	1.33	16.00	8.00	4.67	4.67	7.33	0.00	13.33	4.00	14.00	12.00	8.67
Other Baselines														
$\pi_{0.5}$	30.00	42.89	35.56	20.44	22.22	18.67	6.67	11.33	0.44	31.11	25.78	26.00	2.89	21.08
$\pi_{0.5}$ w/ Past Actions	26.67	58.33	26.67	30.67	23.67	20.67	16.00	12.33	8.67	24.00	18.67	34.00	4.00	23.41
SAM2Act+ (ICML 2025)	40.00	76.00	25.33	27.33	32.00	18.00	26.67	17.33	5.33	24.67	20.00	29.33	0.00	26.31
Dual System: Visual Keyframe Memory														
MemER (ICLR 2026)	56.67	79.33	59.33	81.33	72.00	38.00	21.33	**70.67**	25.33	30.00	26.00	**82.67**	16.67	50.72
Dual System: Structured Keyframe Memory														
R2Mem (Ours)	**66.67**	**94.00**	72.67	**94.00**	**92.67**	**85.33**	**74.67**	62.00	**50.67**	**62.00**	**40.00**	63.33	**60.00**	**70.62**

Table 2: Task success rate on RoboMemArena. †: frozen.

Method	**TSR (%)**
Non-Memory VLA	
$\pi_{0.5}$	21.5
End-to-End Memory VLAs	
HiF-VLA (CVPR 2026)	16.9
MemoryVLA (ICLR 2026)	15.0
Dual-System Memory VLAs	
Ground Truth	46.1
Qwen3-VL-8B†	6.0
GPT-5.4†	8.7
MemER (ICLR 2026)	27.3
PrediMem	38.5
R2Mem (Ours)	**43.8**

Table 3: Ablations on RoboMME (success rate, %). Columns report suite means; AVG averages all tasks. Full-model results are from Table 1.

Variant	Counting	Permanence	Reference	Imitation	**AVG**
R2Mem (full)	**77.78**	**86.67**	**53.67**	**61.67**	**70.62**
Train-and-test ablation (retrained)					
w/o reasoning text	66.00	58.50	41.00	50.00	53.54
Test-time ablations (full-model checkpoint)					
w/o stream initialization	74.00	85.00	47.00	45.00	64.62
w/o first visibility	73.33	55.50	43.00	50.00	54.92
w/o significant displacement	52.67	64.50	42.50	60.00	54.31
w/o discrete state change	74.00	85.50	46.00	59.00	66.62
w/o subgoal completion	20.67	40.00	14.00	19.00	24.31
w/o object identifiers	73.33	55.00	47.50	53.00	56.62

Performance on RoboMemArena. R2Mem retains **95.0%** of the performance achieved with ground-truth subgoals and delivers a **+13.8%** relative improvement over PrediMem (Table 2). R2Mem's explicit records of object histories and completed operations yield stronger performance than PrediMem's predictive supervision for visual keyframe selection. This favors explicit task records over improved visual selection. These records tell the planner how objects have changed and which steps are complete, supporting subgoal prediction without reconstructing it visually.

Qualitative analysis. Figure 4 illustrates the complementary roles of observation and action memory on RoboMME. Observation records link the red cube's disappearance to the target container and track it as `obj_003` through subsequent swaps. Action records establish that both button presses have been completed. Together, these records provide the target's updated location and task progress for the final subgoal. R2Mem selects the correct container, while MemER selects an incorrect one in this rollout. SKM thus makes key events and task progress explicit for later planning.

Ablation study. Table 3 evaluates reasoning text and the planner's reliance on its memory components. Retraining without reasoning text reduces average success by 17.08 percentage points, supporting the benefit of explicitly preserving task-relevant information alongside visual keyframes. We further remove individual event types or object identifiers at test time using the full-model checkpoint. Removing subgoal-completion records causes the largest decline, indicating that the planner

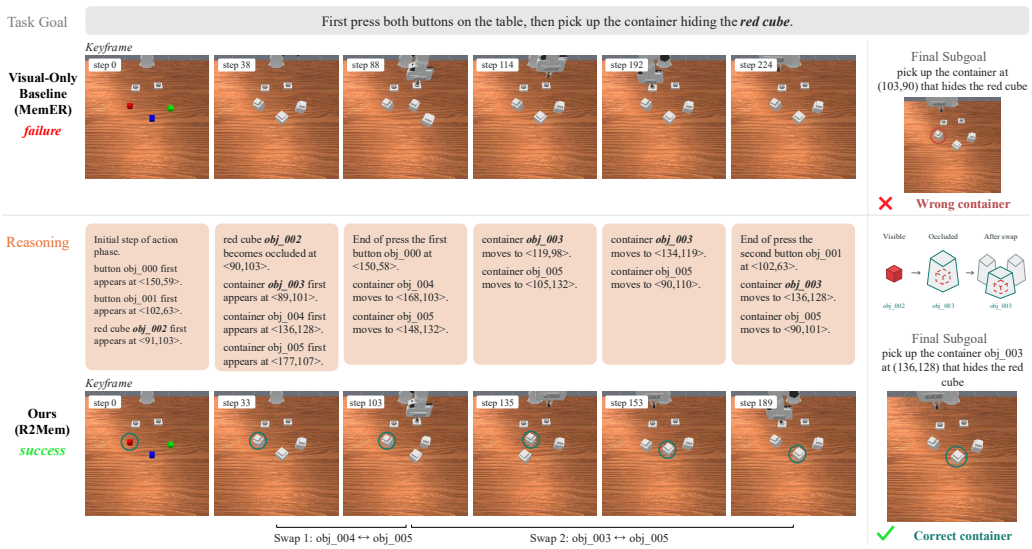


Figure 4: Qualitative comparison on ButtonUnmaskSwap: MemER (top) and R2Mem (bottom).

relies heavily on explicit records of completed operations to determine subsequent subgoals. Removing first-visibility or displacement records also substantially reduces performance, consistent with their roles in establishing object references and preserving spatial changes across observations. The decline without object identifiers supports linking records to the same entities over time. These ablations indicate that the planner uses both object histories and execution progress.

5.3 Real-World Experiments

We construct three real-world tasks to evaluate the high-level VLM's ability to predict subgoals from memory and current observations. *TrackFruit* adapts VideoUnmaskSwap, requiring the model to track a hidden fruit through container swaps. *PickFruit* follows BinFill, requiring it to keep count while collecting a specified number of fruits. *RepickBlock* follows VideoRepick, requiring it to identify the block manipulated in a demonstration and repeat the operation a specified number of times. Figure 5 shows higher R2Mem success across all three tasks. These results suggest that SKM generalizes to real-world subgoal prediction using explicit object histories and task progress.

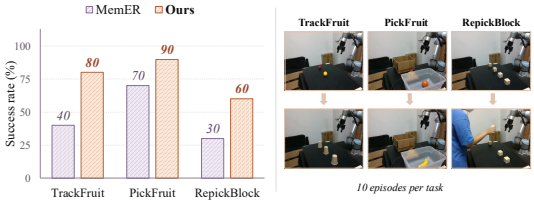


Figure 5: Real-world success rates (left) and task examples (right).

6 Conclusion and Limitations

We examined visual-only keyframe memory and found limited gains from improving keyframe selection or increasing retention frequency in our controlled experiments. To make task-relevant conclusions explicit and preserve information between retained observations, we introduced R2Mem and its Structured Keyframe Memory (SKM). By organizing reasoning text around task-relevant key events, SKM records object histories and completed operations alongside their visual evidence. Evaluations on two memory benchmarks show improved task success, with real-world tests further supporting its applicability to subgoal prediction. Our findings highlight the importance of explicitly preserving conclusions about objects and task progress in memory for long-horizon manipulation.

However, SKM offers limited additional benefit on tasks that can be solved readily through visual imitation, where retained images already provide sufficient guidance for subsequent decisions. Future work could adapt the use of reasoning records to the memory demands of each task.

AI USE STATEMENT

We used generative AI tools to assist with discussing the method formulation and experimental methodology, interpreting results, and translating manuscript text. We also used these tools for data visualization and figure creation, literature research, and manuscript editing. All AI-assisted content was reviewed and revised by the authors. We take responsibility for the final content of this work, including text, claims, and artifacts produced with the assistance of generative AI.

ETHICS STATEMENT

This work studies memory for robotic manipulation in simulated and real-world settings. We do not identify specific ethical risks associated with the research described in this paper.

REPRODUCIBILITY STATEMENT

Appendix A provides the complete prompt templates and key-event text templates. Appendix B describes the reasoning data construction pipeline, including how each event type is extracted from simulator records. The code will be released.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Suneel Belkhale, Tianli Ding, Ted Xiao, Pierre Sermanet, Quon Vuong, Jonathan Tompson, Yevgen Chebotar, Debidatta Dwibedi, and Dorsa Sadigh. Rt-h: Action hierarchies using language. *arXiv preprint arXiv:2403.01823*, 2024.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Nhat Chung, Taisei Hanyu, Toan Nguyen, Huy Le, Frederick Bumgarner, Duy Minh Ho Nguyen, Khoa Vo, Kashu Yamazaki, Chase Rainwater, Tung Kieu, et al. Rethinking progression of memory state in robotic manipulation: An object-centric perspective. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 3407–3415, 2026.

Yinpei Dai, Hongze Fu, Jayjun Lee, Yuejiang Liu, Haoran Zhang, Jianing Yang, Chelsea Finn, Nima Fazeli, and Joyce Chai. Robomme: Benchmarking and understanding memory for robotic generalist policies. *arXiv preprint arXiv:2603.04639*, 2026.

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. *arXiv preprint arXiv:2501.18564*, 2025.

Gemini Robotics Team, Abbas Abdolmaleki, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Ashwin Balakrishna, Nathan Batchelor, Alex Bewley, Jeff Bingham, et al. Gemini robotics 1.5: Pushing the frontier of generalist robots with advanced embodied reasoning, thinking, and motion transfer. *arXiv preprint arXiv:2510.03342*, 2025.

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. *arXiv preprint arXiv:2606.10267*, 2026.

Li Ji, Siyin Wang, Pengfang Qian, Xiaopeng Yu, Yihai Tian, Zhaoye Fei, Jingjing Gong, and Xipeng Qiu. Hime: Hierarchical embodied memory for long-horizon vision-language-action control. *arXiv preprint arXiv:2607.03449*, 2026.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. OpenVLA: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Moo Jin Kim, Chelsea Finn, and Percy Liang. Fine-tuning vision-language-action models: Optimizing speed and success. *arXiv preprint arXiv:2502.19645*, 2025.

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*, volume 2026, pp. 101537–101558, 2026.

Huashuo Lei, Wenxuan Song, Huarui Zhang, Jieyuan Pei, Jiayi Chen, Haodong Yan, Han Zhao, Pengxiang Ding, Zhipeng Zhang, Lida Huang, et al. Robomemarena: A comprehensive and challenging robotic memory benchmark. *arXiv preprint arXiv:2605.10921*, 2026.

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. *arXiv preprint arXiv:2603.12942*, 2026a.

Hao Li, Shuai Yang, Yilun Chen, Xinyi Chen, Xiaoda Yang, Yang Tian, Hanqing Wang, Tai Wang, Dahua Lin, Feng Zhao, et al. Towards efficient and robust manipulation via multi-frame vision-language-action modeling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18388–18396, 2026b.

Xiang Li, Ya-Li Li, Yuan Wang, and Shengjin Wang. Trm-vla: Temporal-aware chain-of-thought reasoning and memorization for vision-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10943–10953, 2026c.

Zaijing Li, Bing Hu, Rui Shao, Gongwei Chen, Dongmei Jiang, Pengwei Xie, Jianye Hao, and Liqiang Nie. Global prior meets local consistency: Dual-memory augmented vision-language-action model for efficient robotic manipulation. *arXiv preprint arXiv:2602.20200*, 2026d.

Minghui Lin, Pengxiang Ding, Shu Wang, Zifeng Zhuang, Yang Liu, Xinyang Tong, Wenxuan Song, Shangke Lyu, Siteng Huang, and Donglin Wang. Hif-vla: Hindsight, insight and foresight through motion representation for vision-language-action models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20732–20742, 2026.

Jiaqi Peng, Xiqian Yu, Delin Feng, Yuqiang Yang, Wenzhe Cai, Jing Xiong, Ganlin Yang, Jinliang Zheng, Jiafei Cao, Xueyuan Wei, et al. Cortex: A bidirectionally aligned embodied agent framework for long-horizon manipulation. *arXiv preprint arXiv:2607.05377*, 2026.

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

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, et al. Spatialvla: Exploring spatial representations for visual-language-action model. *arXiv preprint arXiv:2501.15830*, 2025.

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*, volume 2026, pp. 18567–18602, 2026.

Lucy Xiaoyang Shi, Brian Ichter, Michael Equi, Liyiming Ke, Karl Pertsch, Quan Vuong, James Tanner, Anna Walling, Haohuan Wang, Niccolo Fusai, et al. Hi robot: Open-ended instruction following with hierarchical vision-language-action models. *arXiv preprint arXiv:2502.19417*, 2025.

Ajay Sridhar, Jennifer Pan, Satvik Sharma, and Chelsea Finn. Scaling up memory for robotic control via experience retrieval. In *International Conference on Learning Representations*, volume 2026, pp. 97142–97166, 2026.

Marcel Torne, Karl Pertsch, Homer Walke, Kyle Vedder, Suraj Nair, Brian Ichter, Allen Z Ren, Haohuan Wang, Jiaming Tang, Kyle Stachowicz, et al. Mem: Multi-scale embodied memory for vision language action models. *arXiv preprint arXiv:2603.03596*, 2026.

Ziye Wang, Modi Shi, Chaojun Ni, Jiazhi Yang, Mengdi Li, Zhizhong Su, Tianwei Lin, and Hongyang Li. Nativemem: Native memory compression for long-horizon robotic manipulation. *arXiv preprint arXiv:2607.06678*, 2026.

Bingxin Xu, Yuzhang Shang, and Emilio Ferrara. Don't drop the baton: Long-horizon robot manipulation via agentic subtask exploration and transition-aware memory. *arXiv preprint arXiv:2608.16889*, 2026.

Ganlin Yang, Zhangzheng Tu, Yuqiang Yang, Sitong Mao, Junyi Dong, Tianxing Chen, Jiaqi Peng, Jing Xiong, Jiafei Cao, Jifeng Dai, et al. Eventvla: Event-driven visual evidence memory for long-horizon vision-language-action policies. *arXiv preprint arXiv:2606.20092*, 2026.

Ruijie Zheng, Yongyuan Liang, Shuaiyi Huang, Jianfeng Gao, Hal Daumé III, Andrey Kolobov, Furong Huang, and Jianwei Yang. Tracevla: Visual trace prompting enhances spatial-temporal awareness for generalist robotic policies. In *International Conference on Learning Representations*, volume 2025, pp. 54277–54296, 2025.

SUPPLEMENTARY MATERIAL

A PROMPT TEMPLATE

The VLM receives a shared system prompt and a phase-specific user prompt. Braced fields are filled at runtime; <image> denotes an image input. The field action_memory stores execution-phase SKM, including both observation and action records.

A.1 SYSTEM PROMPT

```
You are a robot keyframe-reasoning subgoal predictor.

You are given a task goal, past structured keyframe memory, and current
    front-view observations.

You may operate in two phases:
Demo video observation phase: Some tasks have a demo video for the initial
    setup, while others do not. If a demo video is provided, identify
    keyframes that record important visual evidence from the current
    observations. The selected keyframes will be added to
    "demo_video_memory" in the structured keyframe memory.
Action phase: Identify keyframes that record important visual evidence
    from the current observations, and decide whether the ongoing subgoal
    should be kept or changed. The selected keyframes will be added to
    "action_memory" in the structured keyframe memory.

Each keyframe memory entry contains a keyframe image and a list of
    keyframe reasoning texts.
A keyframe image records a task-relevant visual state, including cases
    where:
- a phase begins;
- an object of interest becomes visible for the first time in the current
    phase;
- an object of interest has a clear position change compared with its
    previously recorded keyframe state;
- an object of interest has an important discrete state change, such as
    becoming highlighted, becoming occluded, or becoming unoccluded;
- the current subgoal has visually ended.

Each attached reasoning text should briefly describe one task-relevant
    event. When mentioning a specific object, always include its
    identifier, such as obj_000. In the demo phase, assign object
    identifiers by first-mention order, starting from obj_000. In the
    action phase, reuse the same identifier for an object that corresponds
    to an object already recorded in demo_video_memory; assign a new
    identifier only to an action-phase object that was not recorded in
    demo_video_memory, continuing after the largest existing obj_xxx. If
    no demo video memory is available, assign action-phase identifiers by
    first-mention order starting from obj_000.
```

A.2 PHASE-SPECIFIC USER PROMPTS

Demonstration observation. This prompt constructs memory from a demonstration without predicting an execution subgoal.

```
The task goal is: {task_goal}

Phase: demo video observation

Past Structured Keyframe Memory:
{
  "demo_video_memory": {video_memory_json}
}

Current Observations:
Frame 1: <image>
Frame 2: <image>
...
Frame N: <image>

Predict all keyframes in the current observations and provide reasoning
    texts for each selected keyframe.

If "demo_video_memory" is empty, this is the initial observation chunk;
    you must select at least Frame 1 as a keyframe.

Return only valid JSON matching the following schema template. The
    angle-bracket text describes what each value should contain; do not
    copy the descriptions literally:
{
  "keyframes": [
    {
      "position": <the 1-indexed integer index of the selected keyframe in
    the Current Observations list>,
      "reasoning_texts": [
        "<at least one reasoning string for why this frame should be
    written into memory>",
        "<another reasoning string for the same keyframe if needed>"
      ]
    }
  ]
}

If there is no keyframe, return {"keyframes": []}.
Order the reasoning_texts list according to the object identifier
    referenced by each text, so texts about obj_000 come before texts
    about obj_001, etc.
```

Execution with demonstration memory. The VLM reads the completed demonstration memory together with the memory accumulated during execution.

```
The task goal is: {task_goal}

The original demo video is not provided directly in this phase. Use the
    "demo_video_memory" field in Past Structured Keyframe Memory as the
    summarized representation of the demo video.

Phase: action

Past Structured Keyframe Memory:
{
  "demo_video_memory": {video_memory_json},
  "action_memory": {action_memory_json}
}

Ongoing Subgoal: {ongoing_subgoal}

Current Observations:
Frame 1: <image>
Frame 2: <image>
...
Frame N: <image>

Predict all keyframes in the current observations, provide reasoning texts
    for each selected keyframe, and decide whether the ongoing subgoal
    should be kept or changed.

If "action_memory" is empty, this is the initial action chunk; you must
    select the Frame 1 as a keyframe, and "current_subgoal" must be a new
    grounded subgoal, not "keep".

Return only valid JSON matching the following schema template. The
    angle-bracket text describes what each value should contain; do not
    copy the descriptions literally:
{
  "current_subgoal": "<keep, or one new grounded subgoal if the ongoing
    subgoal should change>",
  "keyframes": [
    {
      "position": <the 1-indexed integer index of the selected keyframe in
    the Current Observations list>,
      "reasoning_texts": [
        "<at least one reasoning string for why this frame should be
    written into memory>",
        "<another reasoning string for the same keyframe if needed>"
      ]
    }
  ]
}
If there is no keyframe, set "keyframes" to [].
Order the reasoning_texts list according to the object identifier
    referenced by each text, so texts about obj_000 come before texts
    about obj_001, etc.
```

Execution without a demonstration. When no demonstration is available, the input contains only execution-phase memory.

```
The task goal is: {task_goal}

Phase: action

Past Structured Keyframe Memory:
{
  "action_memory": {action_memory_json}
}

Ongoing Subgoal: {ongoing_subgoal}

Current Observations:
Frame 1: <image>
Frame 2: <image>
...
Frame N: <image>

Predict all keyframes in the current observations, provide reasoning texts
    for each selected keyframe, and decide whether the ongoing subgoal
    should be kept or changed.

If "action_memory" is empty, this is the initial action chunk; you must
    select the Frame 1 as a keyframe, and "current_subgoal" must be a new
    grounded subgoal, not "keep".

Return only valid JSON matching the following schema template. The
    angle-bracket text describes what each value should contain; do not
    copy the descriptions literally:
{
  "current_subgoal": "<keep, or one new grounded subgoal if the ongoing
    subgoal should change>",
  "keyframes": [
    {
      "position": <the 1-indexed integer index of the selected keyframe in
    the Current Observations list>,
      "reasoning_texts": [
        "<at least one reasoning string for why this frame should be
    written into memory>",
        "<another reasoning string for the same keyframe if needed>"
      ]
    }
  ]
}
If there is no keyframe, set "keyframes" to [].
Order the reasoning_texts list according to the object identifier
    referenced by each text, so texts about obj_000 come before texts
    about obj_001, etc.
```

A.3 KEY-EVENT TEXT TEMPLATES

Table S1 lists the event-text templates. The fields {semantic_name} and {object_id} denote an object's descriptive name and persistent identifier of the form obj_NNN. Coordinates use image-space <{x},{y}> order. For subgoal completion, {subgoal_text} describes the completed operation, including object references and coordinates where available. Multiple event texts may accompany the same image.

Table S1: Examples of text templates for the five key-event types.

Key event	Reasoning text
Stream initialization	Initial step of demo video. Initial step of action phase.
First visibility	{semantic_name} {object_id} first appears at <{x},{y}>.
Significant displacement	{semantic_name} {object_id} moves to <{x},{y}>.
Discrete state change	{semantic_name} {object_id} becomes occluded at <{x},{y}>. {semantic_name} {object_id} becomes unoccluded at <{x},{y}>. {semantic_name} {object_id} becomes highlighted at <{x},{y}>.
Subgoal completion	End of {subgoal_text}.

Each historical memory entry contains keyframe_index, keyframe_image, and a list of keyframe_reasoning_text strings. Entries are ordered by time within each memory stream. In the training representation, image-space coordinates are serialized as <bbox> placeholders with the corresponding points stored in objects.bbox.

B REASONING DATA CONSTRUCTION PIPELINE

We extract key events from simulation trajectories, then assemble instruction-tuning examples with the prompts in Appendix A.

Trajectory records and object correspondence. Front-view images are aligned with simulator object identities, visibility, image-space and 3D positions, discrete states, and subgoal annotations. The task instruction determines the object categories and constraints of interest. Actor identities and subgoal coordinate references link records of the same object. Public identifiers are assigned in first-mention order from obj_000, reused across demonstration and execution for corresponding objects, and extended for newly encountered objects.

Temporal alignment. The initial execution frame forms a one-frame chunk; the remaining execution frames and demonstration frames are divided into chunks of 16 simulator steps. We sample every second frame within each chunk and retain the final partial chunk for training. Events are generated or aligned to these sampled observations, ensuring that target keyframes occur in the VLM input. Visibility history and spatial references are reset between streams, while object identifiers persist.

Event extraction. The five event types use the following simulator signals:

- *Stream initialization.* The first observation of each stream is retained and paired with its initialization text.
- *First visibility.* For each object of interest, we find the first sampled observation with a positive visibility flag and a valid tracked image-space location, then record its identity and location.
- *Significant displacement.* Each object's spatial reference is the last keyframe that explicitly recorded it. Candidates require valid current and reference locations and sufficient 3D

displacement to exclude projection and tracking jitter. Their image-space displacements are summed; exceeding the configured threshold triggers records of their updated locations.

- *Discrete state change.* We detect transitions in visibility and task-state annotations, recording the resulting state without repeating unchanged states. Occlusion records can use the last reliable visible location. Restored visibility is recorded after a short delay, provided a valid visible location is available.
- *Subgoal completion.* Recorded subgoal-boundary flags mark the end of the preceding valid operation, even for consecutive operations with identical descriptions. For button presses, an earlier switch in the simulator's online subgoal signal can establish completion; duplicate records are suppressed. The completed operation's text supplies the record, with coordinate references mapped to the shared object identifiers.

Any event can trigger a keyframe; events on the same frame share one entry. A displacement-only event may be merged into a nearby subsequent completion keyframe if its displacement criterion still holds there. Annotations retain frame indices, event types, object references, and template-generated text.

Instruction-tuning examples. Each example combines the task instruction, prior memory, and sampled images. Targets specify one-indexed keyframe positions and reasoning texts. Execution examples also supervise `current_subgoal`: the first grounded subgoal at initialization, a new subgoal at a transition, or `keep` otherwise. Chunks with multiple transitions are omitted from supervision. Ground-truth entries supply subsequent training contexts. Simulator metadata constructs the labels; at deployment, the VLM predicts memory and subgoals from visual and textual inputs.