MOST: MEMORY OF OBJECT STATES AND TASK PROGRESS FOR LONG-HORIZON MOBILE MANIPULATION

Anonymous authors
Paper under double-blind review

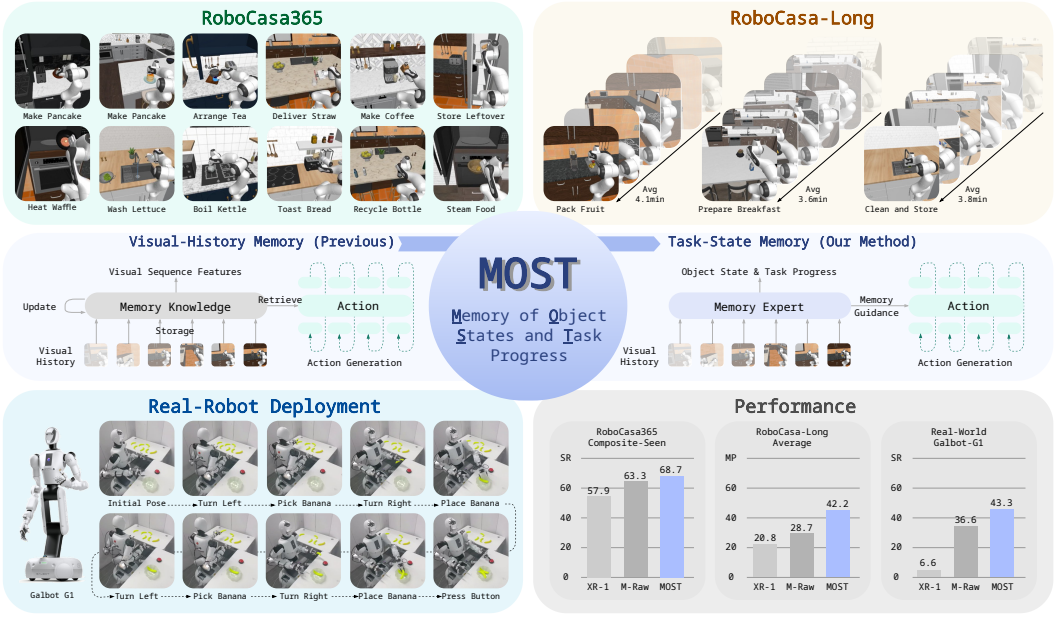

Figure 1: **MOST** structures policy-internal memory around object states and task progress for long-horizon mobile manipulation. We evaluate it on RoboCasa365, our newly introduced **RoboCasa-Long** benchmark of multi-minute, multi-stage tasks, and real-world deployment. M-Raw in the plots denotes MOST-Raw, a matched visual-history memory variant without task-state supervision or future prediction. Across these settings, MOST outperforms both the memoryless Xiaomi-Robotics-1 (XR-1) baseline and MOST-Raw.

ABSTRACT

Long-horizon embodied manipulation requires policies to act on information that is no longer visible. Modified objects may leave the field of view, while completed subtasks continue to constrain future actions. Existing approaches extend temporal context through recurrent state updates, test-time training, or storage and retrieval of past observations. As horizons grow, these mechanisms can still leave the policy to infer task state from long, redundant, and viewpoint-dependent visual experience. We argue that long-horizon agents benefit not merely from remembering more, but from remembering task-relevant state changes and progress. We propose **Memory of Object States and Task Progress (MOST)**, a policy-internal memory framework that compresses multi-view observations into temporal tokens. Historical and predictive future tokens are explicitly supervised with object states and subtask progress, structuring memory around what has changed in the world and what has already been accomplished. The resulting memory directly conditions action generation within a single end-to-end policy, without an external planner or retrieval stage. We further introduce **RoboCasa-Long**, a benchmark of multi-minute, multi-stage mobile manipulation tasks. Across Robo-

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.

Casa365, RoboCasa-Long, and real-world robot experiments, MOST consistently outperforms Xiaomi-Robotics-1 (XR-1), the previous state-of-the-art model. It raises average success from 65.9% to 72.1% on RoboCasa365, average maximum task progress from 20.83% to 42.22% on RoboCasa-Long, and real-world success from 6.66% to 43.33%. MOST also outperforms a matched visual-history memory variant, showing that effective long-horizon memory depends not only on how much experience is retained, but on whether it preserves decision-relevant progress and world-state changes.

1 INTRODUCTION

Learning-based robot policies are increasingly capable across tasks, objects, instructions, and embodiments (Open X-Embodiment Collaboration et al., 2024; Kim et al., 2024). Scaling robot data, multimodal representations, and action models has broadened task coverage and improved dexterous control (Physical Intelligence et al., 2025; Chi et al., 2023). Yet extending these policies to long-horizon embodied tasks introduces a qualitatively different challenge. As a robot moves across workspaces and executes multiple task stages, information that remains critical for future decisions can disappear from the current observation (Dai et al., 2026; Yang et al., 2026). An object may already have been moved, an appliance may have been turned on, or a subtask may have been completed several minutes earlier. The difficulty is therefore not only that the current observation is partial. The policy must preserve the task-relevant consequences of earlier interactions, particularly persistent object states and task progress. Losing this information can lead to repeated actions, skipped stages, incorrect object interactions, or premature termination (Bagaria et al., 2026). These dependencies are especially relevant in mobile, multi-workspace settings (Yenamandra et al., 2023; Fu et al., 2024).

Existing approaches extend temporal context through recurrent or compressed state updates, test-time training, and storage or retrieval of past observations (Cherepanov et al., 2026; Sun et al., 2025; Shi et al., 2026b; Shah et al., 2026b). Despite their different mechanisms, these approaches largely focus on how much history to retain and how it should be updated or accessed. Progress-critical events can nevertheless remain embedded in long visual streams, while visually similar stages may require different next actions. We instead focus on what information from interaction history should survive because it remains relevant for future decisions. This distinction defines our controlled comparison among a memoryless policy, a matched visual-history memory, and memory explicitly structured by task state.

To realize this structured alternative, we organize a single memory sequence around two complementary kinds of task-relevant information: task progress and object states. Task progress records the current phase and what has already been completed (Bagaria et al., 2026), while object states record action-induced changes to relevant objects that may no longer be visible (Chung et al., 2026). Their temporal evolution links earlier interactions to future decisions while suppressing visual detail that is no longer relevant. For example, packing three bananas requires tracking completed transfers and whether the completion button should be pressed after earlier bananas leave the current view.

Based on this representation, we propose **Memory of Object States and Task Progress (MOST)**, an instruction-conditioned, end-to-end policy with compact task-state memory. MOST compresses multi-view observations into temporal tokens and structures both historical and predictive tokens through subtask-progress and relevant object-state supervision. This memory directly conditions action generation, while the auxiliary supervision is removed at inference. MOST therefore operates as a single policy without an external planner or retrieval stage.

Evaluating this hypothesis requires tasks in which relevant state persists over substantially longer horizons than in standard manipulation benchmarks. RoboCasa365 offers broad task coverage, but its standard episodes place limited pressure on memory over multi-minute execution (Nasiriany et al., 2026). Existing memory-focused benchmarks introduce explicit long-term dependencies, often in constrained or purpose-built settings (Dai et al., 2026; Chen et al., 2026). This leaves room for natural household activities such as preparing food, cleaning, and tidying that span multiple work areas. We therefore introduce **RoboCasa-Long**, a benchmark of multi-minute, multi-stage mobile manipulation tasks that require policies to preserve task progress and world-state changes across extended navigation and interaction. As summarized in Fig. 1, we evaluate MOST on Robo-

Casa365, RoboCasa-Long, and real-world deployment. We compare against the previous state-of-the-art Xiaomi-Robotics-1 (XR-1) baseline (Fan et al., 2026) and a matched visual-history memory variant. Progress-aware metrics adapted from PRM-as-a-Judge 1.5 (Liu et al., 2026) and controlled representation studies separate access to history from how it is represented. MOST raises average success from 65.9% to 72.1% on RoboCasa365, average maximum task progress from 20.83% to 42.22% on RoboCasa-Long, and real-world success from 6.66% to 43.33%. These results support structuring temporal context around task progress and object states.

Our contributions are threefold:

- **Memory formulation.** We formulate long-horizon policy memory as a representation problem centered on what task-relevant information should survive, rather than only how much history is retained or how it is accessed.
- **MOST.** We introduce a policy-internal memory framework that learns compact temporal representations under subtask-progress and object-state supervision, directly conditioning action generation without external planning or retrieval.
- **Long-horizon evaluation.** We introduce **RoboCasa-Long**, a benchmark of multi-minute, multi-stage mobile manipulation, and demonstrate consistent gains across RoboCasa365, RoboCasa-Long, and real-world deployment.

2 Related Work

2.1 Memory for Long-Horizon Robot Control

Long-horizon robot control requires policies to preserve information after relevant evidence leaves the current view. Existing methods extend temporal context through recurrence and attention (Parisotto et al., 2020; Cherepanov et al., 2026; Bagaria et al., 2026), test-time training (Sun et al., 2025; Jiang et al., 2026), compressed multi-frame memory (Jang et al., 2025; Torne et al., 2026; Shah et al., 2026a; Wang et al., 2026), or event selection and retrieval (Yang et al., 2026; Shah et al., 2026b). Structured maps provide another route by organizing past observations spatially (Parisotto et al., 2018; Chaplot et al., 2020). Hierarchical systems instead externalize long-horizon reasoning through language-model plans or executable programs (Ahn et al., 2022; Huang et al., 2022; Liang et al., 2023). These approaches establish several effective ways to preserve and access longer context. MOST complements them by shifting the focus from how history is retained or accessed to what information should persist in policy memory.

2.2 Task-State Representations for Long-Horizon Manipulation

For extended manipulation, useful memory should retain the consequences of earlier interactions even when their visual evidence is no longer available. Recurrent models summarize latent execution context (Cherepanov et al., 2026; Bagaria et al., 2026), while perceptual-cognitive and imagination-based memories integrate prior observations with predicted future context (Shi et al., 2026b;a). Object-centric memory further organizes temporal state around persistent entities (Chung et al., 2026). High-level planners can also express task state through intermediate plans, feedback, or executable steps (Ahn et al., 2022; Huang et al., 2022; Singh et al., 2023), although this state is maintained outside the action policy. Building on these related directions, MOST jointly structures policy-internal memory around subtask progress and the states of relevant objects. The resulting representation preserves both what has been accomplished and how earlier actions changed the world.

2.3 Long-Horizon Manipulation Benchmarks

Evaluating long-horizon memory requires tasks in which earlier events remain relevant throughout extended execution. CALVIN and HULC evaluate language-conditioned skill sequences (Mees et al., 2022a;b), while LIBERO studies transfer across manipulation task suites (Liu et al., 2023). ALFRED, RoboCasa, and RoboCasa365 broaden evaluation to household activities with multiple objects, locations, and state changes (Shridhar et al., 2020; Nasiriany et al., 2024; 2026). Mobile manipulation systems additionally demonstrate operation across work areas (Yenamandra et al., 2023; Fu et al., 2024; Wu et al., 2023; Liu et al., 2024). These settings contain important sequential

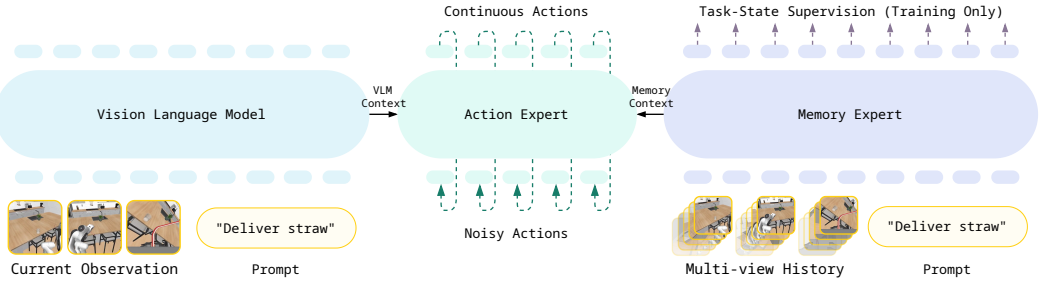

Figure 2: **MOST overview.** MOST integrates a Vision Language Model, a Memory Expert, and an Action Expert within one end-to-end policy. The Vision Language Model encodes the current observation and instruction. The Memory Expert represents sampled history as object states and task progress, and the Action Expert uses both contexts to generate continuous actions. Task-state supervision is training-only.

structure, but are not designed specifically to isolate persistent task state throughout multi-minute interaction across several workspaces. RoboMME and RMBench directly probe memory-dependent manipulation through controlled task designs (Dai et al., 2026; Chen et al., 2026). Complementing both lines, RoboCasa-Long evaluates natural multi-stage household activities whose progress and world-state changes must remain available as the robot moves among work areas. It therefore provides a focused testbed for the memory representation studied by MOST.

3 Method

3.1 MOST Overview

MOST is an instruction-conditioned, end-to-end policy with three components (Fig. 2). The **Vision Language Model** encodes the current multi-view observation and instruction. The **Memory Expert** converts sampled history into an instruction-conditioned representation of object states and task progress. The **Action Expert** uses both contexts to predict continuous actions. The two experts are depth aligned, letting task-state information influence action generation without an external planner or retrieval module.

3.2 Compact Task-State Memory

Fig. 3 details the Memory Expert. At sampled timestep i, the multi-view observation is $O_i = \{I_i^{(v)}\}_{v=1}^{V}$, where $I_i^{(v)}$ is the image from view v among V views. A DINOv2 image encoder (Oquab et al., 2024) extracts each view's global [CLS] feature $c_i^{(v)}$. A learned view query q_{view} cross-attends to these features to produce one history token h_i. One token per sampled timestep makes memory growth independent of views and image patches, keeping long histories computationally manageable.

To make the compressed memory decision-relevant, we structure it to preserve observed task progress while also encoding anticipated future task states. We append K copies of a learned future query q_{fut} to the N history tokens. Let ℓ denote the encoded instruction and E_{mem} the Memory Expert. The history tokens and complete memory sequence are

$$h_i = \text{ViewFuse}\Big(q_{\text{view}}, [c_i^{(1)}, \ldots, c_i^{(V)}]\Big),$$
$$M = [m_1, \ldots, m_{N+K}] = E_{\text{mem}}\left([h_1, \ldots, h_N, \underbrace{q_{\text{fut}}, \ldots, q_{\text{fut}}}_{K \text{ future queries}}], \ell\right). \tag{1}$$

Here, ViewFuse is learned-query cross-attention over the views. The output sequence M contains a memory token m_i at position i.

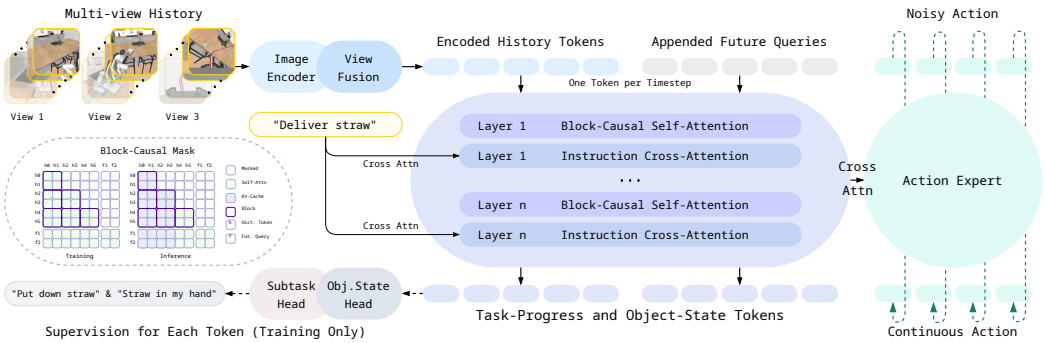

Figure 3: **Memory Expert architecture.** Historical multi-view observations are encoded into history tokens, and future queries are appended to form the memory sequence. The sequence is processed with block-causal self-attention and conditioned on the instruction through cross-attention. During training, each token predicts its subtask and the state of every relevant object. The output of each Memory Expert layer conditions the corresponding Action Expert layer.

The Memory Expert groups the history into consecutive blocks of B tokens. Block-causal self-attention is bidirectional within a block and causal across blocks, so history tokens cannot access later blocks or future queries. Future queries attend to the full history and one another. Each layer then cross-attends to the encoded instruction ℓ, conditioning memory on the commanded task. During inference, completed blocks are cached while only the current block and future queries are updated. This maintains ordered context without reprocessing the full history.

3.3 Memory-Conditioned Action Generation

Each Action Expert layer attends to its paired Memory Expert output, making task-state memory directly available during control (Fig. 3). At layer r, let $M^{(r)}$ and $X^{(r)}$ denote the memory and action features. The action features query the memory through gated cross-attention:

$$Q^{(r)} = \mathrm{Query}\left(X^{(r)}\right), \qquad K^{(r)} = \mathrm{Key}\left(M^{(r)}\right), \qquad V^{(r)} = \mathrm{Value}\left(M^{(r)}\right),$$
$$\widetilde{X}^{(r)} = X^{(r)} + g^{(r)} \cdot \mathrm{Attention}\left(Q^{(r)}, K^{(r)}, V^{(r)}\right). \tag{2}$$

Here, $\widetilde{X}^{(r)}$ is the memory-conditioned action representation and $g^{(r)}$ is a learned scalar gate initialized to zero. Applying this interaction at every paired layer exposes action generation to observed progress and anticipated task states throughout the network.

3.4 Task-Progress and Object-State Supervision

We supervise memory with subtask and object-state targets. At position i, label s_i identifies the execution stage, such as picking up or placing an object. For each relevant object o, label $z_{i,o}$ describes its state, such as being on a counter, in hand, or at the destination. We combine its category and episode-local instance embeddings into e_o, allowing the state head to distinguish instances.

Each memory output m_i predicts one subtask and one state for every relevant object. The subtask head H_{sub} receives m_i, while the shared object-state head H_{obj} also receives e_o. Thus, a task with three relevant objects gives each token one subtask target and three object-state targets. Let $\mathcal{O}$ be the set of relevant objects. The cross-entropy objectives are

$$\mathcal{L}_{\mathrm{sub}} = \operatorname*{mean}_i \mathrm{CE}(H_{\mathrm{sub}}(m_i), s_i), \qquad \mathcal{L}_{\mathrm{obj}} = \operatorname*{mean}_{i,\, o\in\mathcal{O}} \mathrm{CE}(H_{\mathrm{obj}}(m_i, e_o), z_{i,o}),$$
$$\mathcal{L} = \mathcal{L}_{\mathrm{policy}} + \lambda_{\mathrm{sub}}\mathcal{L}_{\mathrm{sub}} + \lambda_{\mathrm{obj}}\mathcal{L}_{\mathrm{obj}}. \tag{3}$$

The means cover the available annotations in each training sample. Here, $\mathcal{L}_{\mathrm{policy}}$ is the action-prediction objective, and $\lambda_{\mathrm{sub}}/\lambda_{\mathrm{obj}}$ denotes the auxiliary loss weight. The subtask and object-state vocabularies are closed-set and constructed from each training split. The auxiliary heads and object embeddings are removed at inference, while the Memory Expert, future queries, and memory-to-action pathway remain active. Further details appear in Appendix A.

Table 1: Success rate (%) on the 50-task RoboCasa365 target suite under Target-10% and Target-100%. GR00T N1.5 split rates are published results, while XR-1, MOST-Raw, and MOST use our matched protocol. Averages are recomputed from the displayed split rates with 18/16/16 task weighting and truncated to one decimal place. The best result in each setting is shown in **bold**.

Data	Method	Atomic↑	Composite-S↑	Composite-U↑	Average↑
Target 10%	GR00T N1.5	57.0	28.7	32.3	40.0
	XR-1	60.1	44.3	43.1	49.6
	MOST-Raw	61.1	48.3	45.2	51.9
	MOST (Ours)	**63.3**	**51.0**	**48.7**	**54.6**
Target 100%	GR00T N1.5	70.0	41.2	44.0	52.4
	XR-1	78.5	57.9	60.0	65.9
	MOST-Raw	77.5	63.3	62.2	68.0
	MOST (Ours)	**81.8**	**68.7**	**64.7**	**72.1**

Table 2: Progress-aware evaluation (%) on RoboCasa365 composite tasks. XR-1, MOST-Raw, and MOST use matched data, training budgets, and evaluation protocols. Best results within each setting are highlighted in blue and **bold**. Second-best results use a lighter blue.

Data	Method	SR↑	MC@25↑	MC@50↑	MC@75↑	MP↑	PPL↑	OPS↑	UPS↑
Com-S-10%	XR-1	44.37	88.12	77.29	61.45	72.34	57.64	66.42	79.22
	MOST-Raw	48.33	86.45	77.70	64.16	73.78	58.54	67.12	79.46
	MOST (Ours)	**51.04**	**89.16**	**82.50**	**68.75**	**77.29**	**62.00**	**71.86**	**81.76**
Com-U-10%	XR-1	43.12	94.37	80.83	58.75	73.30	56.02	66.92	81.08
	MOST-Raw	45.20	94.16	81.66	**61.04**	74.41	56.99	68.69	81.05
	MOST (Ours)	**48.75**	**95.41**	**83.33**	60.83	**75.79**	**57.55**	**69.11**	**83.49**
Com-S-100%	XR-1	57.91	90.83	84.37	70.41	79.48	63.55	74.83	84.56
	MOST-Raw	63.33	90.62	85.00	73.95	81.68	65.06	76.67	86.82
	MOST (Ours)	**68.75**	**94.79**	**89.16**	**80.20**	**85.88**	**68.50**	**82.09**	**89.37**
Com-U-100%	XR-1	60.00	95.41	87.50	73.75	82.03	64.54	76.70	88.12
	MOST-Raw	62.29	**97.50**	88.95	74.79	83.39	67.16	78.10	**89.23**
	MOST (Ours)	**64.79**	96.45	**90.20**	**76.25**	**84.66**	**68.38**	**79.71**	89.13

4 EXPERIMENTS

We test whether representing policy memory through object states and task progress improves control beyond unstructured visual history. We report success and execution progress on RoboCasa365, multi-minute mobile manipulation in simulation, and real-world count-conditioned manipulation.

4.1 EXPERIMENTAL SETUP

Benchmarks and protocol. The RoboCasa365 target suite (Nasiriany et al., 2026) contains 18 Atomic, 16 Composite-Seen, and 16 Composite-Unseen tasks. We first pretrain on Human300 and then finetune on the target suite. Target-10% and Target-100% use 50 and 500 target demonstrations per task, respectively, with 60k finetuning steps. Our **RoboCasa-Long** benchmark contains three multi-minute, multi-stage household tasks across work areas. Each provides 200 demonstrations averaging 3.8 min, and joint training runs for 20k steps. We evaluate 30 rollouts per task with matched data and schedules. Further details are provided in Appendix B.

Comparisons. We report published GR00T N1.5 results (Nasiriany et al., 2026; Bjorck et al., 2025). XR-1 (Fan et al., 2026), MOST-Raw, and MOST use matched data, training budgets, and evaluation. XR-1 is memoryless. **MOST-Raw** retains visual-history tokens and the memory-to-action interface but removes task-state supervision and future queries, providing an unstructured visual-history baseline.

Metrics. Alongside success rate (SR), we adapt Milestone Coverage (MC@25/50/75), Max Progress (MP), and Path-weighted Progress Length (PPL) from PRM-as-a-Judge 1.5 (Liu et al., 2026). MC measures threshold attainment, MP furthest progress, and PPL path efficiency. Ordered Progress Score (OPS) and Unordered Progress Score (UPS) measure ordered and unordered sub-

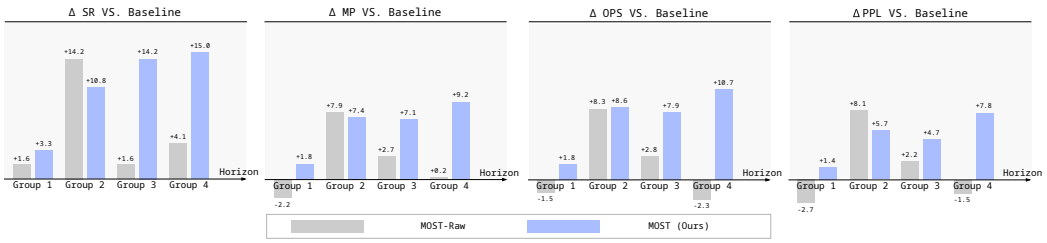

Figure 4: Horizon analysis on RoboCasa365 Composite-Seen under Target-100%. We sort 16 tasks by demonstration length into four groups from shortest (G1) to longest (G4) and report mean changes from XR-1. MOST separates most clearly from unstructured visual history on G3 and G4.

Table 3: Evaluation on our RoboCasa-Long benchmark. We compare XR-1, MOST-Raw, and MOST on three multi-minute, multi-stage manipulation tasks. Best results, including ties, are highlighted in blue and **bold**; a distinct second-best result uses lighter blue.

Task	Method	SR↑	MC@25↑	MC@50↑	MC@75↑	MP↑	PPL↑	OPS↑	UPS↑
PackFruit	XR-1	0.00	43.33	6.66	0.00	18.05	10.93	14.16	27.50
	MOST-Raw	0.00	76.66	40.00	0.00	38.61	23.72	34.16	57.50
	MOST (Ours)	**10.00**	**93.33**	**76.66**	**30.00**	**61.84**	**43.57**	**58.84**	**82.99**
Prepare Breakfast	XR-1	0.00	66.66	0.00	0.00	26.66	17.93	20.33	35.00
	MOST-Raw	0.00	76.66	10.00	0.00	27.66	15.76	19.33	42.33
	MOST (Ours)	**6.66**	**86.66**	**16.66**	**6.66**	**37.14**	**36.19**	**35.95**	**71.66**
CleanAndStore AfterMeal	XR-1	0.00	**33.33**	3.33	0.00	17.77	6.87	10.27	29.16
	MOST-Raw	0.00	**33.33**	**10.00**	0.00	20.00	14.00	14.66	38.66
	MOST (Ours)	**3.33**	20.00	**10.00**	**6.66**	**27.69**	**19.32**	**21.53**	**73.84**
Average	XR-1	0.00	47.77	3.33	0.00	20.83	11.91	14.92	30.55
	MOST-Raw	0.00	62.22	20.00	0.00	28.75	17.82	22.71	46.16
	MOST (Ours)	**6.66**	**66.66**	**34.44**	**14.44**	**42.22**	**33.03**	**38.77**	**76.17**

task completion. Simulation uses simulator state, while real trials use the same stages scored from recordings. Appendix B.5 gives full definitions.

4.2 RoboCasa365

RoboCasa365 spans diverse household tasks with multiple objects, state changes, and stages, providing a controlled test of what policy memory should represent. We compare three matched policies. XR-1 is memoryless. MOST-Raw encodes visual history without task-state supervision or future queries, whereas MOST supervises historical and future tokens with subtasks and relevant object states. Tab. 1 reports SR across all splits and both data regimes. MOST-Raw improves average SR over XR-1 in both regimes. MOST reaches 54.6% and 72.1% and leads every split. MOST-Raw's Target-100% Atomic regression shows that extra context is not uniformly beneficial, supporting memory structured around task state.

To evaluate behavior before terminal success, Tab. 2 measures milestone attainment, progress efficiency, and ordered and unordered completion. MOST surpasses MOST-Raw in SR, MP, PPL, and OPS across all four composite settings. Higher PPL and OPS indicate that task-state supervision supports efficient, correctly ordered execution, not only greater unordered completion.

We next test horizon effects by sorting the 16 Composite-Seen Target-100% tasks by demonstration length into four groups. Fig. 4 reports each method's change from XR-1. MOST improves all four metrics in every group. MOST-Raw remains competitive in G2, but the SR, MP, and OPS gaps are largest in G3 and G4. This growing separation suggests that structured task-state memory becomes more useful as the horizon grows.

4.3 Multi-Minute Evaluation on RoboCasa-Long

To evaluate policy memory beyond the shorter RoboCasa365 episodes, we construct RoboCasa-Long, a benchmark of three multi-minute, multi-stage mobile manipulation tasks. Each requires the robot to navigate across work areas and execute a sequence of object interactions while retaining

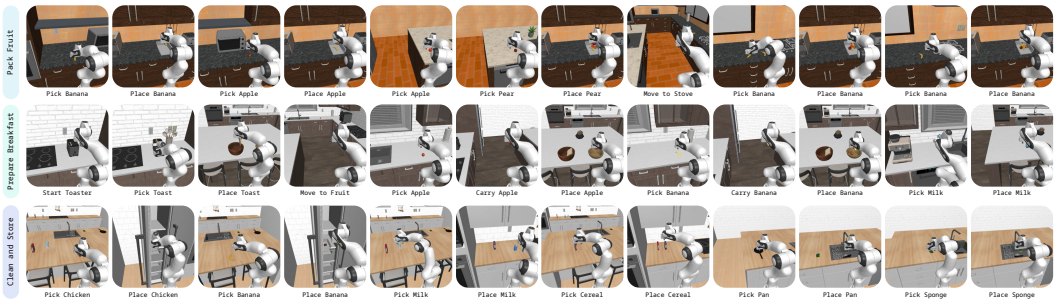

Figure 5: **RoboCasa-Long task demonstrations.** From top to bottom, each row follows one *PackFruit*, *PrepareBreakfast*, or *CleanAndStoreAfterMeal* demonstration from left to right.

Table 4: Controlled memory-representation study on RoboCasa365 Target-100% Composite-Seen. Darker blue denotes higher performance; the lowest result is unshaded and the best is **bold**.

Memory Representation	SR↑	MC@25↑	MC@50↑	MC@75↑	MP↑	PPL↑	OPS↑	UPS↑
No Memory (XR-1)	57.91	90.83	84.37	70.41	79.48	63.55	74.83	84.56
Visual History (MOST-Raw)	63.33	90.62	85.00	73.95	81.68	65.06	76.67	86.82
Current Task State	56.45	89.37	81.66	68.54	77.50	61.35	72.24	83.00
Task-State History	65.20	91.04	84.79	75.83	82.18	66.23	77.88	86.67
Subtask Trajectory	67.08	92.91	86.45	76.66	83.67	67.88	79.25	88.02
Object-State Trajectory	66.04	92.70	87.08	76.66	83.67	68.09	79.31	86.87
Task-State Trajectory (Ours)	**68.75**	**94.79**	**89.16**	**80.20**	**85.88**	**68.50**	**82.09**	**89.37**

earlier progress and persistent changes. We collect 200 demonstrations per task, averaging 3.8 minutes, and evaluate each method over 30 rollouts per task. Fig. 5 visualizes one demonstration from each task and shows the complete navigation and manipulation sequence.

Using matched data and training schedules, we compare memoryless XR-1, visual-history MOST-Raw, and MOST. Tab. 3 reports terminal success and progress-aware performance. MOST-Raw improves average progress over XR-1, while MOST is the only method with nonzero SR on all three tasks and leads every average metric. It leads every metric on *PackFruit* and *PrepareBreakfast*. *CleanAndStoreAfterMeal* remains challenging. MOST-Raw reaches the first milestone more often, but MOST matches MC@50 and leads all remaining completion and progress metrics. These results suggest that task-state memory better sustains progress across long action sequences, although the low absolute SR shows that full-task execution remains challenging.

4.4 What Should Task Memory Represent?

To isolate how memory content and temporal extent affect control, Tab. 4 evaluates seven variants on Target-100% Composite-Seen. *No Memory* removes the temporal pathway. *Visual History* retains past observations without task-state supervision or future queries. *Current Task State* predicts the current subtask and relevant object states from only the current observation. *Task-State History* applies the same supervision across observed history without future prediction. The three trajectory variants append future queries and supervise the sequence using subtasks, object states, or both.

Visual History improves SR and most progress measures over No Memory, showing the value of temporal context. Current Task State performs worst across all metrics, indicating that an instantaneous state estimate cannot replace history. Supervising the observed history recovers performance, while future prediction yields further gains. The subtask-only and object-state-only variants show complementary strengths, and their joint Task-State Trajectory is best on every metric. This supports representing memory through both execution progress and persistent changes in the world.

4.5 Real-World Long-Horizon Manipulation

To test task-state memory under real perception and control, we use a wheeled manipulator whose embodiment closely matches our simulated mobile platform. A single policy is jointly trained on *Pack 1 Banana*, *Pack 2 Bananas*, and *Pack 3 Bananas*. During evaluation, its base remains fixed

Table 5: Real-world long-horizon evaluation on three count-conditioned task variants. For each instruction, the robot must transfer the requested number of bananas one by one and then press the completion button.

Task	Method	SR↑	MC@25↑	MC@50↑	MC@75↑	MP↑	PPL↑
Pack 1 Banana	XR-1	0.00	**80.00**	50.00	0.00	43.33	38.33
	MOST-Raw	**40.00**	70.00	50.00	**40.00**	53.33	50.00
	MOST (Ours)	**40.00**	**80.00**	**70.00**	**40.00**	**63.33**	**61.67**
Pack 2 Bananas	XR-1	10.00	60.00	50.00	50.00	50.00	48.00
	MOST-Raw	40.00	**90.00**	70.00	50.00	68.00	65.00
	MOST (Ours)	**50.00**	**90.00**	**90.00**	**80.00**	**82.00**	**79.50**
Pack 3 Bananas	XR-1	10.00	80.00	60.00	40.00	57.14	54.17
	MOST-Raw	30.00	70.00	50.00	50.00	55.71	53.93
	MOST (Ours)	**40.00**	**100.00**	**80.00**	**70.00**	**81.43**	**78.10**
Average	XR-1	6.66	73.33	53.33	30.00	50.15	46.83
	MOST-Raw	36.66	76.66	56.66	46.66	59.01	56.31
	MOST (Ours)	**43.33**	**90.00**	**80.00**	**63.33**	**75.58**	**73.09**

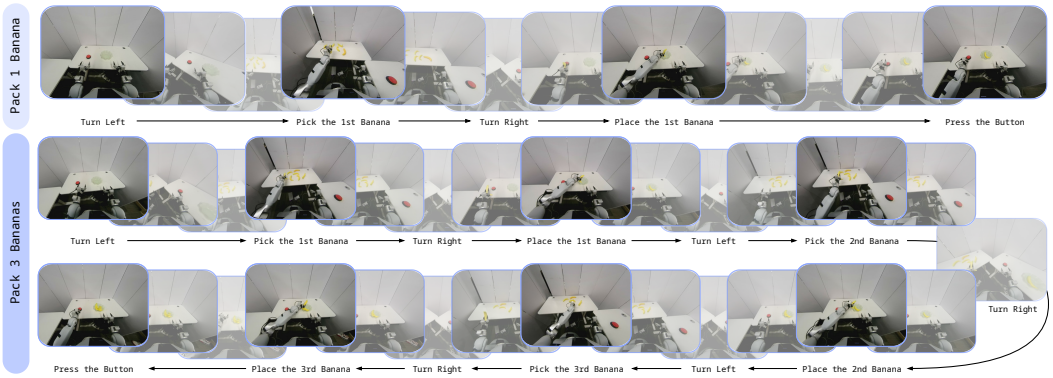

Figure 6: **Representative successful real-world rollouts on a wheeled manipulator.** MOST transfers one banana and stops in the top row. For *Pack 3 Bananas*, the trajectory continues from the middle row into the bottom row, which is read from right to left, before the robot presses the completion button.

while the robot rotates its waist between pickup and placement tables, transfers bananas one at a time, and presses the completion button. This setting isolates memory-dependent counting and stopping from navigation failures. We run 10 trials per method and task.

In Tab. 5, visual history raises average SR from 6.66% for XR-1 to 36.66% for MOST-Raw, while MOST reaches 43.33%. MOST ties MOST-Raw on *Pack 1 Banana*, improves SR by 10 points on both longer variants, and raises MP and PPL on every task. These results show that task-state memory helps track repetitions and stop correctly. Given only 10 trials per task, we interpret SR with the progress metrics. Fig. 6 illustrates this behavior. MOST stops after one placement for the one-banana instruction, but maintains progress through repeated waist rotations and transfers before stopping after the third placement for *Pack 3 Bananas*.

5 Conclusion

We introduced MOST, an end-to-end policy that represents internal memory as the evolution of object states and task progress rather than as unstructured visual history. Historical tokens summarize observed task state, while future queries encourage the memory to anticipate its subsequent evolution and provide context for action generation. Results on RoboCasa365, our newly introduced RoboCasa-Long benchmark, and count-conditioned manipulation with a Galbot G1 support this representation for long and repetitive tasks. The current study relies on closed-set state supervision, and full-task success remains limited on the longest tasks. The real-world evaluation also isolates memory under a fixed-base setting. Future work should learn more open-ended task-state abstractions and evaluate MOST on tasks that require navigation, manipulation, and longer-term adaptation.

AI USE STATEMENT

Generative AI tools were used to assist with drafting and polishing the writing of this manuscript. The authors directed the content, reviewed and revised all AI-assisted text, and take full responsibility for the final paper.

REPRODUCIBILITY STATEMENT

Implementation and optimization details, benchmark construction, task-state annotations, evaluation protocols, and complete metric definitions are provided in the appendix.

REFERENCES

Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Chuyuan Fu, Keerthana Gopalakrishnan, Karol Hausman, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022.

Vaidehi Bagaria, Bijo Sebastian, and Nirav Patel. Recursive belief vision language model. *arXiv preprint arXiv:2602.20659*, 2026.

Johan Bjorck, Valts Blukis, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, et al. GR00T N1.5: An improved open foundation model for generalist humanoid robots. NVIDIA Research, 2025. URL `https://research.nvidia.com/labs/gear/gr00t-n1_5/`.

Devendra Singh Chaplot, Dhiraj Gandhi, Abhinav Gupta, and Ruslan Salakhutdinov. Learning to explore using active neural SLAM. In *International Conference on Learning Representations*, 2020.

Tianxing Chen, Yuran Wang, Mingleyang Li, Yan Qin, Hao Shi, Zixuan Li, Yifan Hu, Yingsheng Zhang, Kaixuan Wang, Yue Chen, et al. RMBench: Memory-dependent robotic manipulation benchmark with insights into policy design. *arXiv preprint arXiv:2603.01229*, 2026.

Egor Cherepanov, Nikita Kachaev, Daniil Zelezetsky, Aydar Bulatov, Artem Pshenitsyn, Yuri Kuratov, Alexey Skrynnik, Aleksandr I. Panov, and Alexey K. Kovalev. μVLA: On recurrent memory for partially observable manipulation in VLA models. *arXiv preprint arXiv:2606.12497*, 2026.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems*, 2023.

Nhat Chung, Taisei Hanyu, Toan Nguyen, Huy Le, Frederick Bumgarner, Duy Minh Ho Nguyen, Khoa Vo, Kashu Yamazaki, Chase Rainwater, Tung Kieu, Anh Nguyen, and Ngan Le. Rethinking progression of memory state in robotic manipulation: An object-centric perspective. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 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. In *International Conference on Machine Learning*, 2026.

Shichao Fan, Kun Wu, Zhengping Che, Xinhua Wang, Di Wu, Fei Liao, Ning Liu, Yixue Zhang, Zhen Zhao, Zhiyuan Xu, et al. XR-1: Towards versatile vision-language-action models via learning unified vision-motion representations. In *International Conference on Machine Learning*, 2026. Oral; arXiv:2511.02776.

Zipeng Fu, Tony Z. Zhao, and Chelsea Finn. Mobile ALOHA: Learning bimanual mobile manipulation using low-cost whole-body teleoperation. In *Conference on Robot Learning*, 2024.

Wenlong Huang, Pieter Abbeel, Deepak Pathak, and Igor Mordatch. Inner monologue: Embodied reasoning through planning with language models. In *Conference on Robot Learning*, 2022.

Huiwon Jang, Sihyun Yu, Heeseung Kwon, Hojin Jeon, Younggyo Seo, and Jinwoo Shin. ContextVLA: Vision-language-action model with amortized multi-frame context. *arXiv preprint arXiv:2510.04246*, 2025.

Yunfan Jiang, Yevgen Chebotar, Ruijie Zheng, Fengyuan Hu, Yunhao Ge, Jimmy Wu, Tianyuan Dai, Scott Reed, Li Fei-Fei, Yuke Zhu, and Linxi Jim Fan. RoboTTT: Context scaling for robot policies. *arXiv preprint arXiv:2607.15275*, 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. In *Conference on Robot Learning*, 2024.

Jacky Liang, Wenlong Huang, Fei Xia, Peng Xu, Karol Hausman, Brian Ichter, Pete Florence, and Andy Zeng. Code as policies: Language model programs for embodied control. In *IEEE International Conference on Robotics and Automation*, 2023.

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

Peiqi Liu, Yaswanth Orru, Pavel Paetzel, Elliot Chane-Sane, Arjun Majumdar, Peter Stone, and Lerrel Pinto. OK-Robot: What really matters in integrating open-knowledge models for robotics. In *Robotics: Science and Systems*, 2024.

Yuyang Liu, Yanqing Shen, Ruike Chen, Jifan Zhao, Yuxuan Tian, Yichi Zhang, Tianfeng Long, Zixuan Yin, Yipu Wang, Ziheng Qin, et al. PRM-as-a-Judge 1.5: A toolkit for robot process assessment. *arXiv preprint arXiv:2608.14284*, 2026.

Oier Mees, Lukas Hermann, and Wolfram Burgard. CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation. *IEEE Robotics and Automation Letters*, 2022a.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. What matters in language conditioned robotic imitation learning over unstructured data. *IEEE Robotics and Automation Letters*, 2022b.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. RoboCasa: Large-scale simulation of everyday tasks for generalist robots. In *Robotics: Science and Systems*, 2024.

Soroush Nasiriany, Sepehr Nasiriany, Abhiram Maddukuri, and Yuke Zhu. RoboCasa365: A large-scale simulation framework for training and benchmarking generalist robots. In *International Conference on Learning Representations*, 2026.

Open X-Embodiment Collaboration, Abby O'Neill, Abdul Rehman, Abhinav Gupta, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, et al. Open X-embodiment: Robotic learning datasets and RT-X models. In *IEEE International Conference on Robotics and Automation*, 2024. arXiv:2310.08864.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024.

Emilio Parisotto, Jimmy Lei Ba, and Ruslan Salakhutdinov. Neural map: Structured memory for deep reinforcement learning. In *International Conference on Learning Representations*, 2018.

Emilio Parisotto, H. Francis Song, Jack W. Rae, Razvan Pascanu, Caglar Gulcehre, Siddhant M. Jayakumar, Max Jaderberg, Raphael Lopez Kaufman, Aidan Clark, Seb Noury, et al. Stabilizing transformers for reinforcement learning. In *International Conference on Machine Learning*, 2020.

Physical Intelligence, Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, et al. $\pi_{0.5}$: A vision-language-action model with open-world generalization. In *Conference on Robot Learning*, 2025.

Rutav Shah, Rajat Kumar Jenamani, Xiaohan Zhang, Lingfeng Sun, Roberto Martín-Martín, Yuke Zhu, Deva Ramanan, and Karl Schmeckpeper. Scaling short-term memory of visuomotor policies for long-horizon tasks. *arXiv preprint arXiv:2606.16178*, 2026a.

Rutav Shah, Yisu Li, Femi Bello, Yuke Zhu, and Roberto Martín-Martín. Memory retrieval in visuomotor policies for long-horizon robot control. *arXiv preprint arXiv:2606.25136*, 2026b.

Hao Shi, Weiye Li, Bin Xie, Yulin Wang, Renping Zhou, Tiancai Wang, Xiangyu Zhang, Ping Luo, and Gao Huang. MemoryVLA++: Temporal modeling via memory and imagination in vision-language-action models. *arXiv preprint arXiv:2606.09827*, 2026a.

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*, 2026b. arXiv:2508.19236.

Mohit Shridhar, Jesse Thomason, Daniel Gordon, Yonatan Bisk, Winson Han, Roozbeh Mottaghi, Luke Zettlemoyer, and Dieter Fox. ALFRED: A benchmark for interpreting grounded instructions for everyday tasks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. ProgPrompt: Generating situated robot task plans using large language models. In *IEEE International Conference on Robotics and Automation*, 2023.

Yu Sun, Xinhao Li, Karan Dalal, Jiarui Xu, Arjun Vikram, Genghan Zhang, Yann Dubois, Xinlei Chen, Xiaolong Wang, Sanmi Koyejo, Tatsunori Hashimoto, and Carlos Guestrin. Learning to (Learn at test time): RNNs with expressive hidden states. In *International Conference on Machine Learning*, pp. 57503–57522, 2025.

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.

Jimmy Wu, Rika Antonova, Adam Kan, Marion Lepert, Andy Zeng, Shuran Song, Jeannette Bohg, Szymon Rusinkiewicz, and Thomas Funkhouser. TidyBot: Personalized robot assistance with large language models. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2023.

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.

Sriram Yenamandra, Arun Ramachandran, Karmesh Yadav, Austin Wang, Mukul Khanna, Theophile Gervet, Tsung-Yen Yang, Vidhi Jain, Alexander Clegg, John Turner, et al. HomeRobot: Open-vocabulary mobile manipulation. In *Conference on Robot Learning*, 2023.

A Implementation Details

A.1 Model Configuration

The Memory Expert uses 384-dimensional tokens and 36 layers aligned with the action network. It has six self-attention heads, eight view-fusion heads, and eight heads in each memory-to-action attention layer. We use a block size of $B = 8$, $K = 16$ future queries, and at most 128 instruction tokens. The memory image encoder is initialized from DINOv2 ViT-S/14 (Oquab et al., 2024). Three 224×224 views are encoded independently, and a learned query fuses their global features into one token at each sampled timestep. During training, gradients update the image encoder through the newest memory observation, while earlier history frames are encoded without image-encoder gradients. At inference, completed eight-token blocks are cached. The current block and future queries are updated at each replanning step, and past RGB frames are discarded after encoding. MOST-Raw retains the historical visual tokens and memory-to-action pathway, but removes the future queries and task-state supervision.

A.2 Optimization

All compared variants use the same benchmark-specific initialization, data, and optimization schedule. We train with BF16 mixed precision and DeepSpeed using FusedAdam with $\beta_1 = 0.9$, $\beta_2 = 0.95$, weight decay 0.1, and gradient-norm clipping at 1.0. The learning rate increases linearly from 5×10^{-7} to 2×10^{-5} over 500 warmup steps and then follows a cosine schedule to 5×10^{-6}. We use an effective global batch size of 128 and random seed 42. The subtask and object-state loss weights are both set to 0.25 for RoboCasa365, 1.0 for RoboCasa-Long, and 0.75 for the real-world experiments. RoboCasa365 finetuning runs for 60k steps. RoboCasa-Long uses only the three Long tasks and is trained for 20k steps without RoboCasa365 replay. Real-world training also runs for 20k steps. Training is distributed over eight NVIDIA A800 80 GB GPUs.

A.3 Visual and Temporal Inputs

RoboCasa365 and RoboCasa-Long use the left agent view, right agent view, and wrist view at a control frequency of 20 Hz. The current-observation pathway receives four frames separated by two source frames and predicts 16-step action chunks. The memory pathway samples one multi-view observation every 16 frames, which gives one memory token every 0.8 s. The real-world policy uses the left and right head cameras and the left-arm camera at 30 Hz. Its current images are resized to 448×320, while memory images are resized to 224×224. It predicts 30-step action chunks and samples memory every 30 frames, which gives one memory token per second.

B Benchmarks and Datasets

B.1 RoboCasa365

We use the 50-task RoboCasa365 target suite (Nasiriany et al., 2026), which contains 18 Atomic, 16 Composite-Seen, and 16 Composite-Unseen tasks. All policies start from the same checkpoint pretrained on Human300, and each target split is finetuned independently. Target-10% and Target-100% use fixed episode subsets containing 50 and 500 demonstrations per task, respectively. We train each target setting for 60k steps and evaluate 30 rollouts per task. Evaluation uses the target kitchens and the task-specific horizons registered by RoboCasa365. A rollout terminates when the simulator reports success, termination, truncation, or the maximum horizon. The overall results average the three splits according to their 18, 16, and 16 tasks.

B.2 RoboCasa-Long

We introduce RoboCasa-Long to evaluate memory on multi-minute mobile manipulation beyond the shorter RoboCasa365 episodes. It contains *PackFruit*, *PrepareBreakfast*, and *CleanAndStoreAfterMeal*. Their language instructions are:

- **PackFruit:** "Collect and pack all the fruit from the counter near the microwave, the dining counter, and the counter near the stove into the tupperware."
- **PrepareBreakfast:** "Toast the bread and put it in the bowl on the left, put the milk beside that bowl, and put the apple and banana in the bowl on the right."
- **CleanAndStoreAfterMeal:** "Put the chicken drumstick and banana in the tupperware on the second shelf of the freezer, put the cereal box and milk carton beside their matching items on the counter by the stove, and put the pan and sponge in the island sink."

Together, these tasks require repeated object transfers, persistent tracking of changes that leave the current view, and navigation among several kitchen work areas. Representative demonstrations are shown in Fig. 5.

We collect 200 demonstrations for each task. Their average durations are approximately 4.1, 3.6, and 3.8 minutes for *PackFruit*, *PrepareBreakfast*, and *CleanAndStoreAfterMeal*, respectively. The three tasks are trained jointly for 20k steps using only RoboCasa-Long data, without replay from RoboCasa365. We evaluate 30 rollouts per task under the same task environments used for data collection. Success is determined by each environment's complete task success condition rather than by reaching an intermediate stage.

B.3 Real-World Setup

Our real-world evaluation uses a Galbot G1 in a fixed-base setup with a pickup table and a placement table. The robot rotates its waist to switch between the two work areas and uses its left arm and gripper to transfer bananas into a bowl before pressing a completion button. We jointly train one instruction-conditioned policy on *Pack 1 Banana*, *Pack 2 Bananas*, and *Pack 3 Bananas*. Their instructions request the corresponding number of transfers followed by a button press.

The training set contains 180 demonstrations in LeRobot format, with 60 episodes for each task. The dataset records four camera streams, while training uses the left and right head cameras and the left-arm camera. We evaluate each method in 10 trials per task. A rollout is successful only when the robot transfers exactly the requested number of bananas into the bowl and then successfully presses the button. Incorrect counts, incomplete transfers, premature termination, and a missing or failed button press are counted as failures. Progress is scored from the recorded trials using the same predefined task stages for all methods.

B.4 Task-State Annotation Format

Each episode is paired with a JSON file containing `action_config` and `state_config`. For RoboCasa365 and RoboCasa-Long, these annotations are generated automatically from privileged simulator information. The real-world data are annotated manually using the same schema. `action_config` divides the episode into inclusive frame intervals. Each entry contains `start_frame`, `end_frame`, an `action_text` describing the active subtask, and a coarser `skill` type such as `Pick`, `Move`, or `Place`. We use `action_text`, rather than `skill`, as the subtask prediction target.

`state_config` lists only the task-relevant objects rather than every object in the scene. Each object entry contains `object_id`, `category`, `instance_id`, and a list of inclusive intervals paired with `state_text`. Distinct instance indices allow multiple objects from the same category to receive separate state targets. Object-state boundaries follow physical state changes and are not forced to coincide with subtask boundaries. For example, a pick subtask may continue while the arm retracts after a successful grasp, even though the object's state has already changed to `in hand`.

A complete annotation example is shown below.

```
{
  "episode_id": "000000",
  "label_info": {
    "action_config": [
      {
        "start_frame": 0,
        "end_frame": 78,
        "action_text":
```

```
          "Pick up the kettle from the counter",
        "skill": "Pick"
      },
      {
        "start_frame": 79,
        "end_frame": 200,
        "action_text":
          "Place the kettle on the tray",
        "skill": "Place"
      },
      {
        "start_frame": 201,
        "end_frame": 375,
        "action_text":
          "Pick up the mug from the cabinet",
        "skill": "Pick"
      },
      {
        "start_frame": 376,
        "end_frame": 503,
        "action_text":
          "Place the mug on the tray",
        "skill": "Place"
      },
      {
        "start_frame": 504,
        "end_frame": 970,
        "action_text":
          "Close the cabinet doors",
        "skill": "Close"
      }
    ],
    "state_config": [
      {
        "object_id": "mug_0",
        "category": "mug",
        "instance_id": 0,
        "state": [
          {
            "start_frame": 0,
            "end_frame": 374,
            "state_text": "in the cabinet"
          },
          {
            "start_frame": 375,
            "end_frame": 503,
            "state_text": "in hand"
          },
          {
            "start_frame": 504,
            "end_frame": 970,
            "state_text": "on the tray"
          }
        ]
      },
      {
        "object_id": "kettle_0",
        "category": "kettle",
        "instance_id": 0,
        "state": [
          {
            "start_frame": 0,
            "end_frame": 77,
            "state_text": "on the counter"
          },
          {
            "start_frame": 78,
            "end_frame": 200,
            "state_text": "in hand"
          },
          {
            "start_frame": 201,
            "end_frame": 970,
            "state_text": "on the tray"
          }
        ]
      },
      {
        "object_id": "cabinet doors_0",
        "category": "cabinet doors",
        "instance_id": 0,
```

```
        "state": [
          {
            "start_frame": 0,
            "end_frame": 954,
            "state_text": "open"
          },
          {
            "start_frame": 955,
            "end_frame": 970,
            "state_text": "closed"
          }
        ]
      }
    ]
  }
}
```


For a sampled frame, the active `action_text` supplies its subtask target, and the active `state_text` of every listed object supplies the corresponding object-state targets. Historical observations are sampled from the episode prefix at interval Δ, where $\Delta = 16$ in simulation and $\Delta = 30$ for the real-world data. For a current frame t, future query k is supervised using the labels at $t + k\Delta$. Positions outside the episode or without a valid interval are ignored by the auxiliary losses. The subtask, object-state, and object-category vocabularies are constructed from the unique labels in the selected training data. None of these labels or object identifiers is required at inference.

B.5 METRICS

A progress curve records how much of a task has been achieved throughout one rollout. For a task with N predefined milestones, let $m_j(t) \in \{0, 1\}$ indicate whether milestone j is satisfied at control step t. We define the normalized progress as

$$p_t = \frac{1}{N} \sum_{j=1}^{N} m_j(t), \qquad p_t \in [0, 1]. \tag{4}$$

Completing one new milestone therefore increases progress by $1/N$ when the remaining milestone states do not change. If a reversible milestone becomes unsatisfied, progress decreases by the same amount. In simulation, the milestone indicators are read directly from task-specific predicates using privileged simulator state at every control step. The curve remains constant between milestone transitions and is therefore staircase-shaped. When the simulator reports full task success, its final value is set to 1. For real-world trials, we obtain the same discrete representation by manually scoring the predefined milestones from the recorded rollout.

This construction differs from PRM-as-a-Judge 1.5 (Liu et al., 2026), which uses a learned progress reward model to estimate a dense curve from rollout video before post-processing the prediction. We do not use a learned judge. The input to our metric computation is the discrete curve defined above, obtained from simulator predicates or manual annotation, and the reported curve remains staircase-shaped. The resulting metric values are therefore not directly comparable with those computed from PRM-predicted curves.

Across R rollouts, let $p^{(r)}_{0:T_r}$ be the progress curve of rollout r, where T_r is its final curve index. Let $y_r \in \{0, 1\}$ be its task-success indicator, L_r its executed control steps, and H_r its maximum rollout horizon. For subtask-based metrics, let M_r be the number of required subtasks, c_r^{ord} the number completed in the required order, and c_r^{any} the number completed regardless of order. The notation $\mathbb{I}[\cdot]$ denotes an indicator function. We compute the metrics as follows.

- **Success Rate (SR)** is the fraction of rollouts that satisfy the complete task success condition:

$$\mathrm{SR} = \frac{1}{R} \sum_{r=1}^{R} y_r. \tag{5}$$

- **Ordered Progress Score (OPS)** is the fraction of required subtasks completed in the prescribed order, $\mathrm{OPS}^{(r)} = c_r^{\mathrm{ord}} / M_r$.

- **Unordered Progress Score (UPS)** is the fraction completed without considering order, $\mathrm{UPS}^{(r)} = c_r^{\mathrm{any}}/M_r$.
- **Normalized Completion Time (NCT)** measures the fraction of the rollout budget used. Successful rollouts use their actual completion time, while failures are assigned the full budget:

$$\mathrm{NCT}^{(r)} = \begin{cases} \min(1, L_r/H_r), & y_r = 1, \\ 1, & y_r = 0. \end{cases} \tag{6}$$

  Lower NCT indicates faster completion.
- **Max Progress (MP)** is the furthest progress reached during a rollout, $\mathrm{MP}^{(r)} = \max_{0 \le t \le T_r} p_t^{(r)}$.
- **Milestone Coverage (MC@q)** measures the fraction of rollouts that reach progress threshold q. For $q \in \{25, 50, 75\}$,

$$\mathrm{MC@}q = \frac{1}{R} \sum_{r=1}^{R} \mathbb{I}\left[\mathrm{MP}^{(r)} \ge q/100\right]. \tag{7}$$

- **Path-weighted Progress Length (PPL)** rewards reaching high progress through a direct path:

$$\mathrm{PPL}^{(r)} = \frac{\left(\mathrm{MP}^{(r)}\right)^2}{\sum_{t=1}^{T_r} \left| p_t^{(r)} - p_{t-1}^{(r)} \right|}. \tag{8}$$

  PPL is set to zero when the denominator is zero. A monotonic curve has PPL equal to MP, while regression and repeated recovery reduce PPL.
- **Cumulative Regret Area (CRA)** measures the severity and duration of regression below the best progress achieved so far. Let $b_t^{(r)} = \max_{0 \le i \le t} p_i^{(r)}$. Then

$$\mathrm{CRA}^{(r)} = \frac{1}{(T_r + 1)\mathrm{MP}^{(r)}} \sum_{t=0}^{T_r} \left(b_t^{(r)} - p_t^{(r)} \right). \tag{9}$$

  We define CRA $= 0$ for a zero-progress rollout, for which both the numerator and MP are zero. Lower CRA indicates less severe or persistent regression.
- **Drawdown Recovery Ratio (DRR)** measures recovery after the largest loss of progress. Define the drawdown $d_t^{(r)} = b_t^{(r)} - p_t^{(r)}$ and let $t_r^\star \in \arg\max_t d_t^{(r)}$ be the step of maximum drawdown. For rollouts with $d_{t_r^\star}^{(r)} > 0$,

$$\mathrm{DRR}^{(r)} = \min\left(1, \frac{\max_{t_r^\star \le t \le T_r} p_t^{(r)} - p_{t_r^\star}^{(r)}}{d_{t_r^\star}^{(r)}}\right). \tag{10}$$

  DRR is averaged only over rollouts that experience a drawdown. A value of 1 indicates full recovery. If no rollout for a task experiences a drawdown, DRR is undefined and reported as "–". This means that no progress regression was observed and should not be interpreted as poor recovery.
- **Failure Near-Success (FNS)** measures how close a failed rollout came to completion:

$$\mathrm{FNS}^{(r)} = 0.5\,\mathrm{MP}^{(r)} + 0.3\,\mathbb{I}\left[\mathrm{MP}^{(r)} \ge 0.75\right] + 0.2\,\mathbb{I}\left[\mathrm{MP}^{(r)} \ge 0.50\right]. \tag{11}$$

  FNS is averaged only over failed rollouts. Higher FNS indicates failure at a later task stage.

The main paper reports SR, OPS, UPS, MP, MC@25/50/75, and PPL. The appendix additionally reports NCT, CRA, DRR, and FNS. Unless a table explicitly averages task-level rows, we use the arithmetic mean over all eligible rollouts in the reported set, rather than the median aggregation used by PRM-as-a-Judge 1.5 for continuous metrics. DRR includes only rollouts with a drawdown, and FNS includes only failed rollouts. Higher defined values are better for all reported metrics except NCT and CRA. Scores are expressed as percentages in the tables and truncated to the displayed precision rather than rounded.

Table 6: Per-task performance (%) of MOST on RoboCasa365 Target-10% Composite-Seen.

Task	SR↑	OPS↑	UPS↑	NCT↓	MP↑	MC@25↑	MC@50↑	MC@75↑	PPL↑	CRA↓	DRR↑	FNS↑
DeliverStraw	33.33	50.00	75.55	86.80	68.88	83.33	73.33	50.00	50.81	11.05	25.00	46.16
GetToastedBread	83.33	90.83	95.83	68.03	90.83	93.33	93.33	90.00	89.72	0.00	100.00	46.50
KettleBoiling	70.00	81.11	86.66	67.96	83.33	90.00	90.00	70.00	83.33	0.00	–	35.55
LoadDishwasher	66.66	78.66	90.00	73.75	84.66	93.33	90.00	76.66	62.40	6.94	77.27	50.00
PackIdenticalLunches	20.00	47.50	69.16	92.56	67.50	100.00	76.66	43.33	41.96	18.87	62.20	52.60
PreSoakPan	53.33	86.66	89.33	76.38	87.33	96.66	93.33	93.33	81.50	1.12	100.00	79.28
PrepareCoffee	13.33	33.33	60.00	93.92	50.00	96.66	40.00	13.33	40.55	34.21	6.25	27.30
RinseSinkBasin	56.66	70.83	70.83	74.22	70.83	76.66	76.66	73.33	70.83	0.00	–	37.11
ScrubCuttingBoard	40.00	78.88	78.88	88.43	78.88	100.00	96.66	40.00	77.55	0.21	100.00	51.29
SearingMeat	20.00	40.66	73.33	87.01	52.00	50.00	50.00	40.00	39.06	42.77	30.43	35.00
SetUpCuttingStation	63.33	82.50	88.33	68.17	85.83	96.66	96.66	86.66	63.40	3.24	90.00	67.95
StackBowlsCabinet	86.66	93.33	96.66	40.83	96.66	100.00	100.00	100.00	67.05	5.37	94.73	87.50
SteamInMicrowave	53.33	80.55	88.33	85.21	81.66	86.66	86.66	80.00	62.87	5.84	91.66	61.78
StirVegetables	16.66	81.66	83.33	92.71	82.22	96.66	93.33	93.33	48.09	3.12	96.66	85.33
StoreLeftoversInBowl	90.00	93.33	96.11	55.53	96.11	96.66	96.66	96.66	52.96	3.23	96.66	63.88
WashLettuce	50.00	60.00	65.83	72.87	60.00	70.00	66.66	53.33	60.00	0.00	–	18.66
Average	51.04	71.86	81.76	76.52	77.29	89.16	82.50	68.75	62.00	8.50	74.68	52.87

Table 7: Per-task performance (%) of MOST on RoboCasa365 Target-10% Composite-Unseen.

Task	SR↑	OPS↑	UPS↑	NCT↓	MP↑	MC@25↑	MC@50↑	MC@75↑	PPL↑	CRA↓	DRR↑	FNS↑
ArrangeBreadBasket	86.66	88.00	94.83	40.12	89.50	93.33	86.66	86.66	51.16	2.57	96.42	10.62
ArrangeTea	60.00	80.66	89.33	85.53	85.33	100.00	93.33	73.33	72.97	3.45	57.14	58.33
BreadSelection	50.00	72.50	79.16	76.79	75.83	100.00	93.33	60.00	47.06	5.90	87.50	49.16
CategorizeCondiments	13.33	40.83	73.33	96.22	55.83	100.00	76.66	33.33	37.69	21.05	69.56	46.05
CuttingToolSelection	63.33	68.33	80.00	64.37	73.33	83.33	83.33	63.33	42.50	3.91	85.00	24.54
GarnishPancake	80.00	87.77	93.33	64.05	92.22	100.00	96.66	80.00	82.59	3.27	55.55	47.22
GatherTableware	40.00	60.83	60.83	86.53	64.16	76.66	76.66	53.33	64.16	0.00	–	39.02
HeatKebabSandwich	26.66	54.44	77.22	93.83	63.88	90.00	73.33	43.33	39.78	13.06	80.95	44.92
MakeIceLemonade	56.66	80.00	84.66	84.28	82.00	96.66	86.66	66.66	78.72	2.36	40.00	50.00
PanTransfer	30.00	61.11	83.33	83.50	65.55	100.00	63.33	33.33	54.48	6.33	86.36	36.34
PortionHotDogs	16.66	52.08	79.58	95.13	65.41	100.00	73.33	43.33	27.13	17.72	91.55	52.45
RecycleBottlesByType	60.00	82.77	93.88	69.11	87.77	100.00	100.00	70.00	68.61	4.25	84.37	62.22
SeparateFreezerRack	56.66	70.00	82.50	79.95	75.83	100.00	76.66	66.66	68.26	7.79	72.72	38.26
WaffleReheat	66.66	89.33	93.33	61.59	90.66	100.00	100.00	86.66	90.66	0.00	–	74.00
WashFruitColander	43.33	80.47	86.19	81.44	84.28	93.33	93.33	83.33	47.22	5.94	92.85	74.95
WeighIngredients	30.00	36.66	84.44	81.36	61.11	93.33	60.00	30.00	47.77	41.46	12.50	30.79
Average	48.75	69.11	83.49	77.74	75.79	95.41	83.33	60.83	57.55	8.69	72.32	46.18

C MORE RESULTS

Tab. 6 and Tab. 7 report per-task results for the Target-10% Composite-Seen and Composite-Unseen settings. Tab. 8 and Tab. 9 report the corresponding Target-100% results.

Table 8: Per-task performance (%) of MOST on RoboCasa365 Target-100% Composite-Seen.

Task	SR↑	OPS↑	UPS↑	NCT↓	MP↑	MC@25↑	MC@50↑	MC@75↑	PPL↑	CRA↓	DRR↑	FNS↑
DeliverStraw	73.33	81.11	94.44	67.17	92.22	96.66	96.66	83.33	66.78	7.26	65.00	64.16
GetToastedBread	86.66	88.33	94.16	67.99	89.16	90.00	90.00	90.00	83.20	0.72	80.00	21.87
KettleBoiling	90.00	94.44	96.66	57.19	96.66	100.00	100.00	90.00	89.11	0.08	100.00	53.33
LoadDishwasher	96.66	99.33	99.33	58.80	99.33	100.00	100.00	100.00	73.32	0.23	100.00	90.00
PackIdenticalLunches	40.00	66.25	78.75	82.80	78.33	100.00	93.33	60.00	62.69	13.69	66.66	59.72
PreSoakPan	73.33	90.66	93.33	75.65	92.66	96.66	96.66	96.66	86.54	1.62	66.66	80.00
PrepareCoffee	20.00	51.11	68.88	92.27	63.33	100.00	66.66	23.33	54.77	16.23	16.66	40.00
RinseSinkBasin	63.33	83.33	83.33	68.97	83.33	93.33	93.33	83.33	83.33	0.00	–	60.00
ScrubCuttingBoard	60.00	64.44	87.77	84.59	72.22	93.33	63.33	60.00	65.66	13.66	22.22	16.94
SearingMeat	33.33	52.00	78.00	81.78	60.00	60.00	60.00	53.33	41.59	29.07	53.84	37.00
SetUpCuttingStation	90.00	95.00	97.50	55.88	95.00	100.00	100.00	90.00	61.22	1.77	100.00	45.00
StackBowlsCabinet	93.33	96.66	98.33	31.28	98.33	100.00	100.00	100.00	68.89	2.57	100.00	87.50
SteamInMicrowave	70.00	91.11	93.88	86.52	91.66	96.66	96.66	93.33	75.38	2.99	95.00	77.22
StirVegetables	46.66	85.55	88.33	82.55	85.55	96.66	93.33	90.00	50.27	1.09	100.00	78.33
StoreLeftoversInBowl	93.33	96.66	98.88	49.77	98.88	100.00	100.00	100.00	55.83	2.39	96.66	91.66
WashLettuce	70.00	77.50	78.33	62.91	77.50	93.33	76.66	70.00	77.50	0.00	–	16.94
Average	68.75	82.09	89.37	69.13	85.88	94.79	89.16	80.20	68.50	5.83	75.90	57.48

Table 9: Per-task performance (%) of MOST on RoboCasa365 Target-100% Composite-Unseen.

Task	SR↑	OPS↑	UPS↑	NCT↓	MP↑	MC@25↑	MC@50↑	MC@75↑	PPL↑	CRA↓	DRR↑	FNS↑
ArrangeBreadBasket	86.66	89.33	91.33	40.80	90.00	90.00	86.66	86.66	53.16	1.38	96.29	12.50
ArrangeTea	76.66	90.66	93.33	79.05	92.00	100.00	96.66	86.66	83.86	1.32	77.77	62.85
BreadSelection	83.33	91.66	95.00	66.05	92.50	100.00	100.00	86.66	64.37	0.68	95.65	53.50
CategorizeCondiments	46.66	70.00	85.83	88.36	76.66	100.00	90.00	66.66	67.36	6.93	54.16	55.62
CuttingToolSelection	46.66	50.00	71.66	74.73	65.00	83.33	83.33	46.66	45.83	8.01	43.75	30.93
GarnishPancake	70.00	81.11	91.11	72.22	90.00	100.00	100.00	70.00	78.74	6.92	33.33	53.33
GatherTableware	56.66	73.33	73.33	79.00	73.33	80.00	80.00	66.66	73.33	0.00	–	36.92
HeatKebabSandwich	23.33	67.77	82.77	96.66	74.44	93.33	86.66	63.33	48.44	9.18	86.00	65.50
MakeIceLemonade	56.66	84.66	91.33	89.93	86.66	96.66	90.00	80.00	82.62	2.20	60.00	66.15
PanTransfer	63.33	80.00	93.33	58.66	83.33	100.00	86.66	63.33	77.96	6.95	75.00	40.00
PortionHotDogs	63.33	76.66	93.33	75.17	85.83	100.00	86.66	80.00	43.08	8.86	90.22	57.04
RecycleBottlesByType	93.33	96.66	98.88	50.69	96.66	100.00	96.66	93.33	84.92	0.14	100.00	35.00
SeparateFreezerRack	56.66	73.33	86.66	81.40	81.66	100.00	90.00	80.00	71.80	10.38	50.00	60.38
WaffleReheat	96.66	99.33	99.33	38.83	99.33	100.00	100.00	100.00	99.33	0.00	–	90.00
WashFruitColander	50.00	84.28	86.66	84.38	87.14	100.00	96.66	83.33	51.93	0.44	100.00	75.80
WeighIngredients	66.66	66.66	92.22	54.59	80.00	100.00	73.33	66.66	67.42	28.11	28.57	24.00
Average	64.79	79.71	89.13	70.66	84.66	96.45	90.20	76.25	68.38	5.72	70.76	51.22

D VISUALIZATIONS

Fig. 7 and Fig. 8 visualize evaluation rollouts produced by MOST for all 16 Composite-Seen and 16 Composite-Unseen tasks under Target-100%. Each row follows one rollout across ten sampled timesteps together with its exact simulator-derived progress trajectory.



Figure 7: **RoboCasa365 Composite-Seen visualizations under Target-100%.** Each row shows ten sampled frames from one rollout of a different task. The staircase beneath each frame traces exact simulator-derived task progress over the rollout, and the red marker indicates the displayed timestep.



Figure 8: **RoboCasa365 Composite-Unseen visualizations under Target-100%.** Each row shows ten sampled frames from one rollout of a different task. The staircase beneath each frame traces exact simulator-derived task progress over the rollout, and the red marker indicates the displayed timestep.