Memory in Motion: Learning Memory-Based Humanoid Loco-Manipulation from Visual Interaction and Motion

***Abstract*— Humanoid loco-manipulation requires retaining task cues that leave view and learning coordinated whole-body actions from limited visual-interaction demonstrations. We present MEMORY IN MOTION, a whole-body vision–language–action architecture centered on contextual memory writing. Its writer retrieves evidence from recent observations using current content, body motion, and existing memory. It fuses the retrieved evidence with current content and updates a persistent linear-attention state. The Action Expert combines this state with current visual information to generate actions. Visual-interaction and language–motion data jointly train the writer and Action Expert through one body–action interface, allowing trajectories without synchronized images to support whole-body learning. With task-specific fine-tuning across nine real-robot tasks, MEMORY IN MOTION achieves 75.6% mean success, with an improvement of 27.8% over $\pi_{0.5}$. On SIMPLE, a jointly fine-tuned 16-task policy achieves 62.4% mean success across the three evaluation levels. Controlled hidden-cue and delay experiments show higher first-choice accuracy than the evaluated memory variants. With approximately 400 hours of visual-interaction data held fixed, adding approximately 400 hours of language–motion data raises mean success on three whole-body tasks from 55% to 75%. Code will be released.**

I. INTRODUCTION

Humanoid loco-manipulation requires interpreting changing views while coordinating the legs, torso, and arms. As a humanoid walks, turns, and reaches, task-relevant cues can leave its field of view before they are needed. Consider a robot that observes a reference object, then turns to select a matching bottle. The current image shows the candidate bottles and their locations but no longer contains the reference needed to choose among them. The robot must use earlier observations to select the target and current vision to locate and manipulate it. Whole-body coordination also expands the range of movements that the policy must learn and demonstrations must cover.

Vision–language–action (VLA) models predict actions from images and language instructions [1]–[3]. Memory-aware methods extend these models by aggregating recent observations, retrieving stored records, or maintaining causal execution histories [4]–[7]. These approaches establish the importance of selecting and integrating task-relevant history. For a moving humanoid, similar views can occur at different movement phases or after different task histories. The usefulness of an observation therefore depends on the robot's motion and accumulated experience. We investigate how current content, body motion, and existing memory can jointly guide the selection of recent evidence and its incorporation into persistent memory for subsequent actions.

To address this question, we introduce MEMORY IN MOTION, a whole-body VLA architecture that combines contextual memory writing with joint learning from language–motion and visual-interaction data (Fig. 1). A recent-observation cache stores compact records that the writer can retrieve. The writer selects evidence using current content, body motion, and existing memory, then fuses it with current content. These contextual writes update a linear-attention state matrix that carries historical information across control steps (Fig. 2). The Action Expert reads this state alongside current visual information. Earlier cues thus inform target selection after leaving view, while current perception supplies the details needed for execution of the robot.

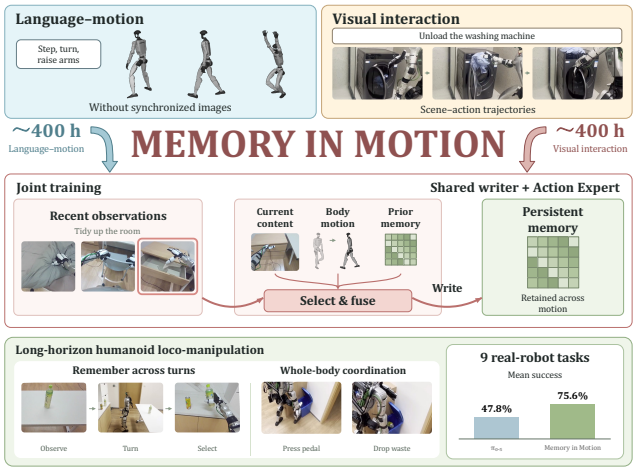


Fig. 1. **Memory in Motion.** Language–motion and visual-interaction data jointly train a contextual memory writer and Action Expert for long-horizon humanoid loco-manipulation. Illustrations highlight memory across whole-body coordination; bars report mean success on nine real-robot tasks.

Learning whole-body actions requires coverage of diverse postures and movement sequences. Expanding this coverage through visual-interaction demonstrations requires synchronized recordings of the scene and robot actions. Language–motion data pair instructions with body trajectories, supplying whole-body supervision without synchronized images. We train a *VL2Action* branch on *visual-interaction data* and a *Motion2Action* branch on language–motion data. VL2Action grounds robot actions in visual scenes and their associated language instructions. Motion2Action predicts subsequent whole-body actions from instructions, body states, and execution histories. Both branches share a body–action interface and jointly train the same memory writer and Action Expert. At inference, *motion context* conditions evidence retrieval and memory writing; while training, *Motion2Action* supplies supervision from language–motion trajectories. Section IV-G

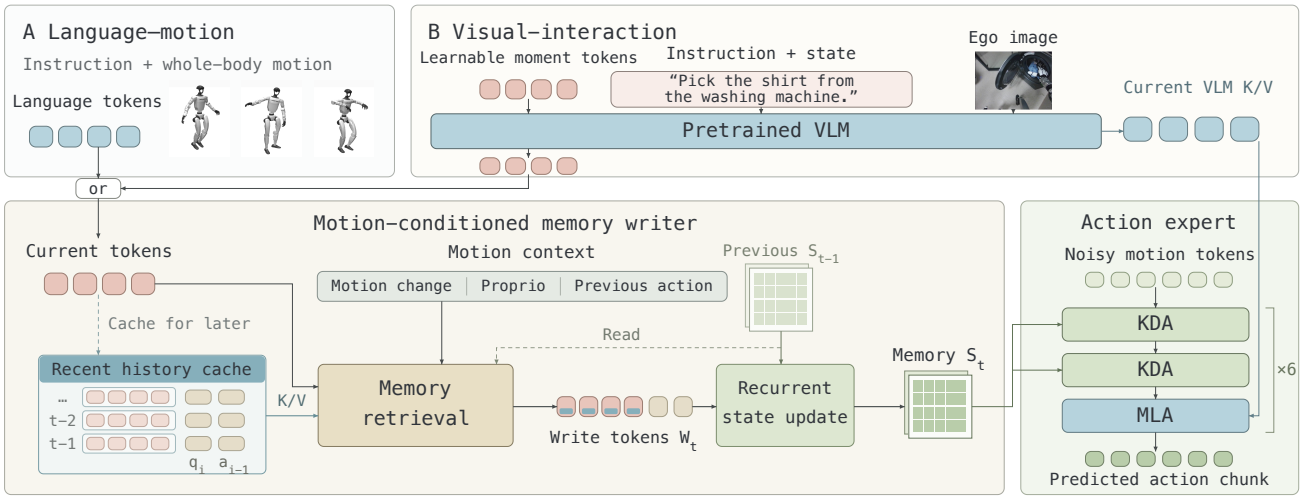


Fig. 2. **Contextual memory writing for humanoid loco-manipulation.** Current content, body motion, previous execution, and old memory condition recent-evidence retrieval. Retrieved evidence is fused with current content and combined with body and action tokens into a six-token write, which jointly updates persistent memory. KDA reads this state, while MLA provides current visual context. Language–motion and visual-interaction data train a shared writer and Action Expert through the same body–action interface and flow-matching objective.

evaluates the contribution of this additional training data.

We evaluate task-specific policies on nine real-robot tasks and a jointly fine-tuned policy on all 16 SIMPLE tasks. On the real robot, MEMORY IN MOTION improves mean success by 27.8 percentage points over $\pi_{0.5}$. On SIMPLE, it achieves 62.4% mean success across the three evaluation levels. Controlled hidden-cue experiments test whether earlier observations support later target selection. After an additional six-second cue-free delay, first-choice accuracy remains 75% among trials reaching the choice point, exceeding the other evaluated variants. With ∼400 hours of visual-interaction data held fixed, adding ∼400 hours of language–motion data raises mean success on three coordination-intensive tasks from 55% to 75%. These controlled comparisons assess the contributions of contextual memory writing and additional motion supervision to whole-body execution.

Our three main contributions are as follows:

- A contextual memory writer conditioned on current content, body motion, and existing memory, connecting recent-evidence retrieval with persistent state updates in linear attention layer of action expert.
- A joint *VL2Action* and *Motion2Action* training scheme with a shared memory writer and Action Expert, using language–motion trajectories without synchronized images to supplement whole-body action learning.
- An evaluation on 9 real-robot tasks and the SIMPLE benchmark, with experiments assessing memory writing, cue retention, and additional motion supervision.

II. RELATED WORK

A. Memory for Vision–Language–Action Policies

Memory-aware policies differ in how they represent, retrieve, and update historical information. HAMLET aggregates compact observation tokens across recent history [4]. MemoryVLA retrieves and fuses perceptual and cognitive records, using the fused representations for action prediction and memory consolidation [5]. AR-VLA maintains a causal state/action cache across control steps [6], while TemporalFlow-VLA uses robot-surface flow supervision to learn queries over execution history [7]. These methods establish the importance of selecting and integrating task-relevant history; RoboMME further examines how memory representations and integration sites interact with the specific demands that a task imposes [8].

Recurrent models retain history in a bounded state. Recurrent Memory Transformer passes learned memory tokens between segments [9], whereas delta-rule models revise key–value associations in a fixed-size matrix [10]. Gated Delta Networks add learned retention [11]; Kimi Linear combines KDA with MLA [12]. Within VLA control, MemBodied combines fixed-size associative interaction memory with an initial-scene anchor [13]. Such recurrent states provide a way to carry information beyond a finite observation window, making the construction and revision of their stored associations central design choices.

Our focus is the construction of persistent memory under changing body motion and execution history. Rather than treating the availability of historical context as sufficient, we examine how motion and accumulated state condition the evidence selected for a write. Writer ablations and cue-retention tests assess this distinction separately from aggregate measures of task performance.

B. Whole-Body Policies and Motion2Action

VLA models transfer pretrained vision–language representations to instruction-conditioned robot control [1]–[3]. Humanoid loco-manipulation additionally requires coordinated whole-body movement [14], [15], which visual-interaction data supply thinly. MotionWAM couples video prediction with a unified whole-body motion representation for humanoid control [16]. WB-WAM uses heterogeneous body

and hand data in a shared physical action space, followed by staged adaptation to real robots [17]. Learning Action Priors learns temporal structure from action trajectories before aligning the resulting action decoder with vision–language inputs [18]. These approaches show that motion structure can be learned from sources other than target-task visual demonstrations [19]. Our *Motion2Action* path carries that structure into the same writer and Action Expert that handle *VL2Action*, so language–motion data shape memory construction and action generation together rather than a separate motion module.

III. Method

At each observation step, the memory writer combines current content, body motion, and the previous memory state to retrieve evidence from a recent-observation cache (Sec. III-A). Retrieved evidence is fused with current content and combined with body state and the previous action into a write that jointly updates persistent memory (Sec. III-B). The Action Expert reads the updated state and, for visual-interaction samples, current VLM context to generate actions. Language–motion and visual-interaction data jointly train the writer and Action Expert through a shared body–action interface and action objective (Sec. III-C). Figure 2 summarizes the architecture and its information flow.

A. Motion- and Memory-Conditioned Retrieval

a) Observation records: For visual-interaction samples, four learnable content tokens accompany the image, instruction, and state prefix through the VLM. Their outputs are projected and normalized to form the current content representation U_t. Proprioception $\boldsymbol{q}_t$ and the previous executed action $\boldsymbol{a}_{t-1}^{\text{exec}}$ are encoded as a body token p_t and a previous-action token e_{t-1}, respectively. These six tokens form the observation record we designed:

$$B_t = [U_t; p_t; e_{t-1}], \qquad C_t = P_C(B_t). \tag{1}$$

The MLA latent projection P_C reduces the feature dimension of each token without pooling the six tokens together. The resulting record still contains four content tokens, one body token, and one previous-action token, which remain individually accessible during retrieval. The cache stores a finite window of these records before fusion with retrieved evidence, while the persistent state S_{t-1} carries accumulated information from earlier steps. Retrieval accesses only previous records; the current record C_t is inserted afterward.

b) Motion and memory conditioning: Body state describes the current posture, while its change supplies motion context. We encode the change between consecutive proprioceptive observations and summarize the previous memory:

$$\begin{aligned} d_t &= \phi_m(\boldsymbol{q}_t - \boldsymbol{q}_{t-1}), \\ s_t &= P_S\big(\mathrm{Concat}_h(r^h S_{t-1}^h)\big)\,. \end{aligned} \tag{2}$$

Here ϕ_m is a learned motion encoder. In memory head h, a learned, normalized query r^h reads the previous state S_{t-1}^h. The head readouts are concatenated and mapped by a learned projection P_S into the state summary s_t, which conditions retrieval on information already held in memory.

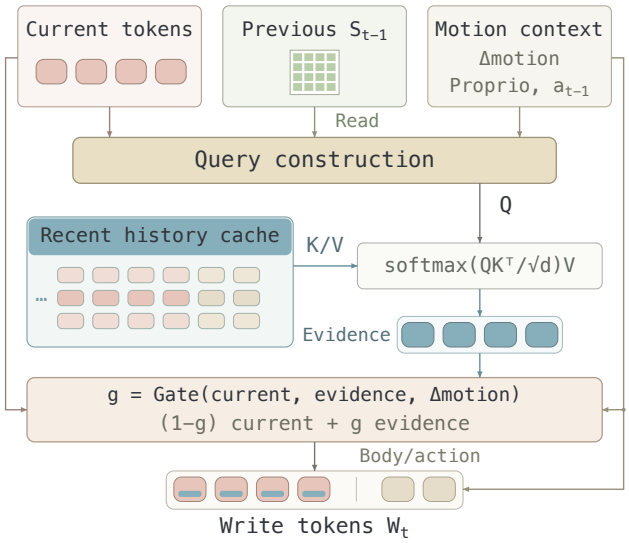


Fig. 3. **Constructing a write from recent experience.** Current content, motion context, and a readout of S_{t-1} jointly determine what to retrieve from the recent cache. A learned gate combines the retrieved evidence with current tokens; body/action tokens anchor the resulting W_t to current execution. The recurrent update then reconciles this write with the task associations already stored in persistent memory.

c) Conditional retrieval: Each content token forms a query with shared body, motion, and memory conditions:

$$Q_{t,i} = P_U(U_{t,i}) + P_E([p_t; e_{t-1}]) + d_t + s_t, \tag{3}$$

where P_U and P_E project the content token and body–action representation into the query space. The query thus combines current content, body state and previous action, recent body motion, and a readout of existing memory.

Valid historical records in the cache are normalized and projected into keys and values, with record-age encodings added to the keys. The four queries retrieve evidence using standard Softmax attention. The outputs are projected back to the content-token dimension to obtain four historical-evidence tokens R_t, which enter the memory write (Fig. 3).

B. From Retrieved Evidence to Persistent Memory

We fuse retrieved evidence with current content, then combine it with body state and the previous action so that the write carries the current execution context. A learned gate uses current content, retrieved evidence, and motion context to regulate the balance between current and retrieved content:

$$\begin{aligned} g_{t,i} &= \sigma(P_g[U_{t,i}; R_{t,i}; d_t]), \\ Z_{t,i} &= (1 - g_{t,i}) \odot U_{t,i} + g_{t,i} \odot R_{t,i}, \\ W_t &= [Z_t; p_t; e_{t-1}]. \end{aligned} \tag{4}$$

P_g is a learned projection, and $g_{t,i}$ controls fusion. Body and previous-action tokens accompany the fused content in the write. When no history is available, we set $Z_t = U_t$.

Retrieval and fusion decide what the write contains; the update decides how it revises the existing state. Following the key–value encoding in KDA [12], we map W_t to six normalized keys, corresponding values, and nonnegative write weights. The keys and values form the rows of K_t and V_t,

and the weight for pair i is $\lambda_{t,i}$. We describe one memory head below and omit its index.

The six key–value pairs describe the same observation step. Their keys need not be orthogonal, so changing the state to fit one pair can also change the readout for another. The six corrections therefore need to account for one another. We use a block delta update that jointly fits all six pairs while limiting changes to the previous state:

$$S_t = \arg\min_S \Big[\|S - S_{t-1}\|_F^2 + \sum_{i=1}^{6} \lambda_{t,i} \|(K_t S)_i - V_{t,i}\|_2^2 \Big]. \tag{5}$$

Here $K_t S$ reads the state at the write keys. The first term penalizes changes from S_{t-1}; the second fits the readouts to the new values [20]. A larger $\lambda_{t,i}$ gives greater weight to the fitting error of pair i. We obtain it from the learned gate $\beta_{t,i} = \sigma(P_\beta W_{t,i})$ as $\lambda_{t,i} = \beta_{t,i}/(1 - \beta_{t,i} + \eta)$, where P_β is a learned projection and η is a numerical stability constant.

The delta-update is driven by the residual between the new values and the previous state's readouts. If $K_t S_{t-1} = V_t$, the state remains unchanged. Otherwise, the joint correction accounts for interactions among the six keys. The objective has a closed-form solution, computed by solving a 6×6 linear system independently in each memory head. The system size is determined by the six write tokens, rather than by the dimensions of the memory state.

The Action Expert uses six KDA–KDA–MLA groups with a hidden width of 1024. Each KDA layer stores eight 128×128 state matrices, one per head. Action features generate queries to read S_t; gated and projected readouts are added residually to those features. Interleaved MLA layers provide current VLM context for visual-interaction samples [12], [21], combining historical information with current perception for action generation.

C. Joint VL2Action and Motion2Action Training

Both data sources train the same memory writer and Action Expert through a shared body–action interface. They differ in how current content is constructed and whether visual context is available, but share retrieval patterns, state updates, and the action objective.

VL2Action samples contain images, instructions, body states, and action sequences. Content tokens U_t are constructed as in Sec. III-A, with body state and the previous action encoded as p_t and e_{t-1}. Retrieval and memory updating follow Secs. III-A–III-B. Current VLM context also reaches the Action Expert through MLA, so action prediction uses both historical information and the current scene.

Motion2Action samples contain instructions, body states, and action sequences without synchronized images. We pool and project the instruction-token embeddings, then add four learned token embeddings to form U_t in the same format as the visual branch. Body state and the previous action use the shared encoders, followed by the same retrieval and state updates. These samples train the model to predict subsequent whole-body actions from language–motion data.

Visual-interaction trajectories provide action supervision paired with scenes, while language–motion trajectories expand supervision over postures and motion sequences. Because both sources pass through the shared writer and Action Expert, additional motion supervision trains the path from memory construction to action generation. The action-prediction loss supervises the writer through backpropagation, without separate labels specifying which historical observation to retrieve or retain.

Both sources use flow matching [22] to train action generation. Given a target action chunk A_t, Gaussian noise ε , and interpolation time $\tau \in [0, 1]$, we use

$$\begin{aligned} A_t^\tau &= (1-\tau)A_t + \tau\varepsilon, \\ \mathcal{L}_{\text{act}} &= \mathbb{E}\big[\|f_\theta(A_t^\tau, \tau; S_t, \mathcal{X}_t) - (\varepsilon - A_t)\|_{\text{valid}}^2\big]. \end{aligned} \tag{6}$$

Here S_t is the memory state after processing the current observation, and f_θ predicts velocity along the action interpolation path. The current VLM context $\mathcal{X}_t$ is available for visual-interaction samples, while language–motion samples omit this branch. The expectation covers both data sources, noise, and interpolation time. Source-specific mappings align body states and actions in a shared interface. The subscript valid restricts the action loss to annotated dimensions, excluding components unavailable in the source data.

During joint pretraining, we freeze the pretrained VLM weights and optimize the learned content tokens, input projections, shared memory writer, and Action Expert. Within each training window, action gradients guide earlier evidence selection, fusion, and state updates through this shared path. The memory state and recent cache carry across consecutive windows of the same episode, with gradients truncated at window boundaries. The state and cache are reset at the start of each episode. The task-specific fine-tuning settings are reported in the experimental training configuration.

IV. Experiments

Our experiments address four questions:

(1) Does MEMORY IN MOTION improve success on nine real-robot tasks and all three SIMPLE evaluation levels?
(2) How do contextual memory writing and its motion- and memory-conditioned retrieval contribute to performance?
(3) Can earlier evidence continue to support decisions as cue-free observations update the recent cache?
(4) Does additional language–motion supervision improve whole-body execution when the visual-interaction data are held fixed at approximately 400 hours?

A. Hardware, Data Collection, and Training

a) Hardware: Experiments use a Unitree G1 with 29 actuated body joints, two ROBOTERA XHAND 1 hands, and a head with actuated pitch and yaw. A head-mounted RGB camera supplies the egocentric view.

b) Data collection: We collect demonstrations with PHT [23]. A PICO 4 Ultra headset and body tracking drive its predictive retargeter to generate near-future robot reference motions; headset orientation sets the viewing direction. RGB images, body and hand feedback, measured head angles, and

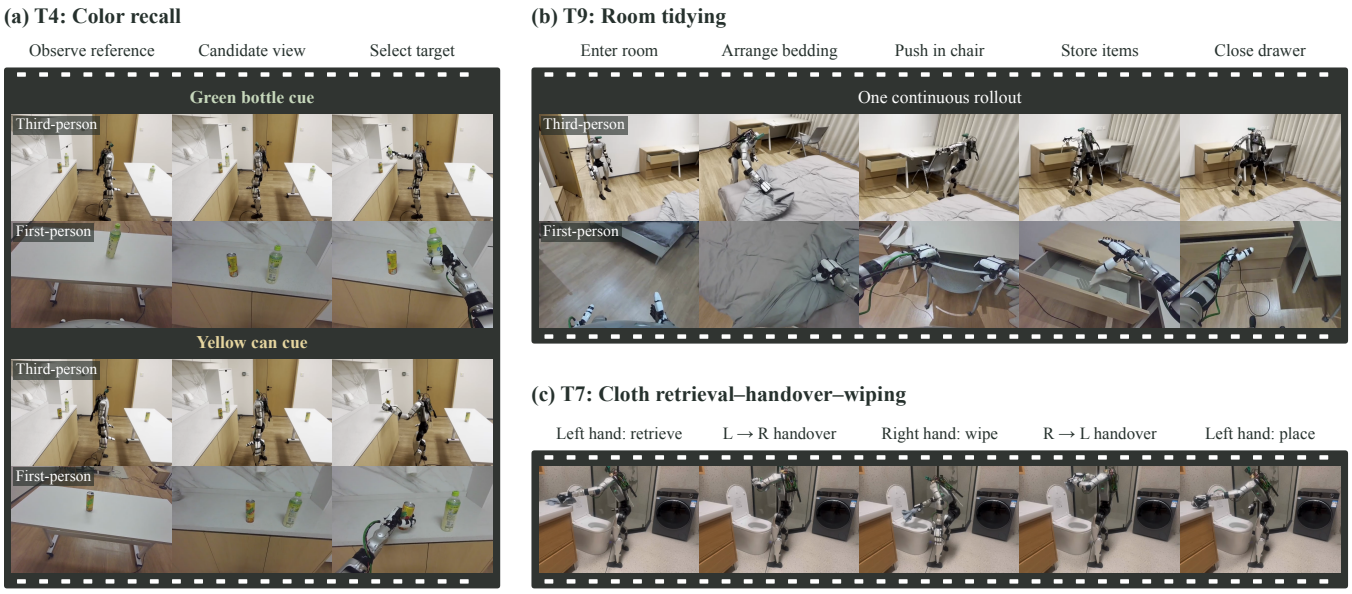


Fig. 4. **Key moments for memory-dependent and whole-body execution.** (a) Color recall (T4): different cues lead to different choices among the same candidates. (b) Room tidying (T9). Paired rows show third-person (top) and egocentric (bottom) views. (c) Cloth retrieval–handover–wiping (T7).

TABLE I

REAL-ROBOT TASK SUCCESS. SUCCESSFUL/ATTEMPTED TRIALS.

Method	T1 Chair	T2 Open bin	T3 Low-table	T4 Color recall	T5 Pedal bin	T6 Pouring	T7 Cloth	T8 Laundry	T9 Tidying	Mean (%)
$\pi_{0.5}$	17/20	8/20	9/20	6/20	10/20	13/20	9/20	12/20	2/20	47.8
Memory in Motion	20/20	14/20	15/20	19/20	13/20	15/20	17/20	17/20	6/20	**75.6**

commanded targets are aligned into approximately 30-Hz trajectories. Commanded references serve as action labels, and measured feedback supplies proprioceptive inputs.

c) Whole-body execution: PHT's RL-trained generalist tracker follows these references at 50 Hz in closed loop. At deployment, the Action Expert supplies body-reference sequences through the data-collection interface, preserving the correspondence between demonstration labels and policy outputs. PHT handles balance and reference tracking, while the VLA generates task motions.

d) Policy inference: Policy inference runs on one NVIDIA RTX 5880 Ada GPU with approximately 3.39B model parameters, including the VLM. Ten flow-integration steps generate each action chunk. Executing six frames at 30 Hz gives a nominal 5-Hz request cadence; memory updates once per observation and stays fixed during denoising.

e) Training configuration: Pretraining uses the full 808.10 h corpus (Fig. 5). Each step draws 64 language–motion and 64 visual-interaction sequences, with windows of 32 and 16 observation steps, respectively. Training runs for 40,000 steps on 128 H20-3e GPUs at a peak learning rate of 2.5×10^{-5}, with the pretrained PaliGemma-VLM [24] frozen. Task policies are fine-tuned on 50–90 demonstrations per real-robot task with batch size 32, learning rate of 10^{-5}, and LoRA [25] on the language model while the remaining VLM weights stay frozen. Both stages use a recent-cache

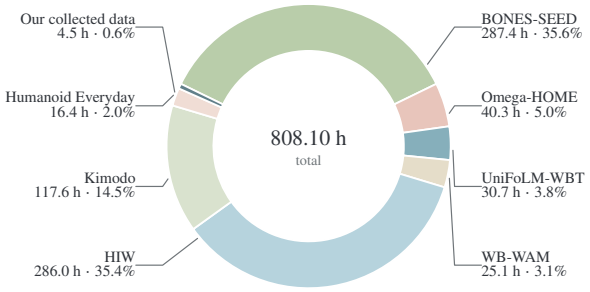


Fig. 5. **Training corpus.** Share of accepted trajectory hours per source, totaling 808.10 h. Labels give hours and percent of total.

capacity of 8 observation records and action chunk of 50.

B. Tasks and Evaluation Protocol

a) Tasks: The tasks comprise chair placement beneath a table (T1), open-bin disposal (T2), low-table retrieval (T3), color recall (T4), pedal-bin disposal (T5), pouring into a receiving bottle (T6), cloth retrieval–handover–wiping (T7), laundry (T8), and room tidying (T9). They span changing views, whole-body contacts, and extended execution (Fig. 4). We fine-tune a separate policy for each real-robot task and jointly fine-tune one policy across all 16 SIMPLE tasks.

b) Evaluation: The main real-robot comparison uses 20 trials per method and task, with shared target-task demon-

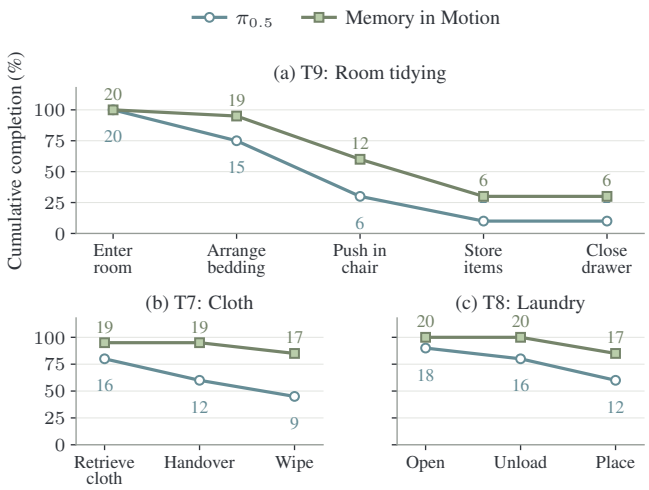


Fig. 6. **Progress through staged long-horizon routines.** Cumulative completion uses the same 20 trials per method and task as Table I. Each label counts trials completing that stage and all preceding stages; a failure at any stage ends the trial without attempting subsequent stages.

strations, robot action interface, and low-level controller. Complete-task success requires finishing without operator assistance. We report successful trials and the mean task success rate. For extended routines, cumulative stage completion measures how far each rollout proceeds. During the process, memory is reset between episodes.

c) Comparison methods: We use $\pi_{0.5}$ [26] as the external baseline, adapted to the same robot interface and fine-tuned on the target tasks. To examine memory design within our architecture, we compare MEMORY IN MOTION with three internal variants that share its backbone, Action Expert, training data, and training recipe. *Current-only* uses a fresh state at every observation and clears temporal buffers, retaining current content, proprioception, and the previous executed action. *Window fusion* aggregates current and recent records with a two-layer temporal Transformer before conditioning a fresh action state, following the temporal-aggregation approach of HAMLET [4]. *Direct write* writes B_t into persistent memory without retrieval or fusion. Each internal configuration is trained with its corresponding memory interface. Current-only and Window fusion are tested on T4; the general execution tasks T2 and T7 provide additional settings for the writer-query ablations in Table III.

C. Real-World Humanoid Performance

MEMORY IN MOTION achieved 75.6% mean success across the nine tasks, compared with 47.8% for $\pi_{0.5}$, an improvement of 27.8 percentage points (Table I). Success was higher on every task. The largest gain, 65 points, occurred on color recall, where target selection depends on a reference observed before the robot turns away.

The advantage extended through the final stages of the longer routines (Fig. 6). MEMORY IN MOTION completed the cloth and laundry routines in 17 trials each, versus 9 and 12 for $\pi_{0.5}$. Room tidying further separates reaching the task area from completing the sequence: both methods completed entering the room in all trials, but MEMORY IN MOTION completed the whole routine in 6 trials versus 2. These results show improved completion of routines requiring successive locomotion and manipulation actions.

TABLE II

SIMPLE BENCHMARK. MEAN SUCCESS (%) ON ALL 16 TASKS.

Method	L0	L1	L2	Mean
$\pi_{0.5}$ (2025.9)†	40.6	28.8	30.6	33.3
GR00T N1.6 (2025.9)†	60.0	55.6	42.2	52.6
ψ_0 (2026.3)†	64.4	**59.7**	48.8	57.6
Memory in Motion	**76.0**	58.3	**53.0**	**62.4**

† As reported in SIMPLE, Table 8 [27].

TABLE III

MEMORY INTERFACES AND QUERY ABLATIONS. PERCENTAGES.

	T4: color recall		T2	T7
Method	First choice	Success	Open bin	Cloth
$\pi_{0.5}$	45.0	30.0	40.0	45.0
Current-only	55.0	55.0	—	—
Window fusion	70.0	55.0	—	—
Direct write	65.0	55.0	45.0	40.0
No motion query	75.0	50.0	55.0	35.0
No state query	85.0	75.0	30.0	45.0
Memory in Motion	**100.0**	**95.0**	**70.0**	**85.0**

First choice over 20 choice-point arrivals; task success over 20 attempted episodes.

D. Public SIMPLE Benchmark

We evaluate a jointly fine-tuned policy on all 16 tasks in SIMPLE [27]. L0 varies distractors and backgrounds; L1 adds material and lighting changes; L2 further varies initial object and robot poses. Table II compares average task success with three entries reported on the same benchmark.

MEMORY IN MOTION achieved the highest three-level mean among the listed methods (62.4%). It led at L0 and L2 and exceeded the reported $\pi_{0.5}$ scores at all three levels, while ψ_0 was 1.4 points higher at L1. The comparison extends the evaluation to a single multitask policy under varied appearances and initial configurations.

E. Contextual Memory Writing

a) Hidden-cue task and metrics: Color recall (T4) requires the robot to observe a rear reference bottle, turn toward two front-table candidates, select the matching color, and return the chosen bottle. The instruction stays fixed while the reference color and candidate arrangement vary, so the current image alone does not identify the correct target. Task success uses 20 attempted episodes per method. For first-choice accuracy, we collect additional episodes as needed to obtain 20 choice-point arrivals per method. Failures before the choice point count against task success but do not yield a choice to evaluate. The additional episodes equalize the number of evaluated decisions across methods; this decision set is distinct from the fixed task-success cohort.

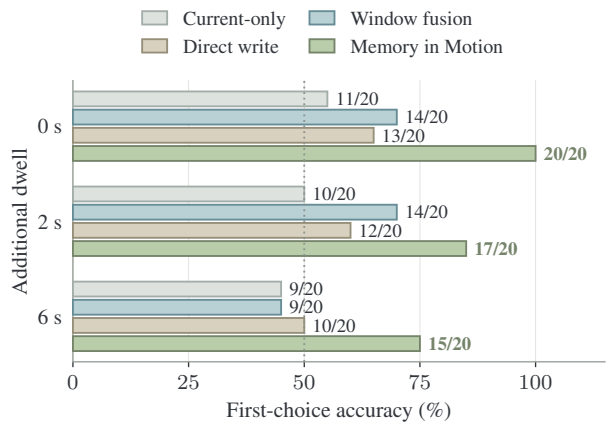


Fig. 7. **Memory retention after the cue leaves view.** Bars show first-choice accuracy over 20 choice-point arrivals per condition; labels give correct/20. Zero-delay results reuse Table III. The dotted line marks the 50% chance accuracy for selecting one of the two candidate objects.

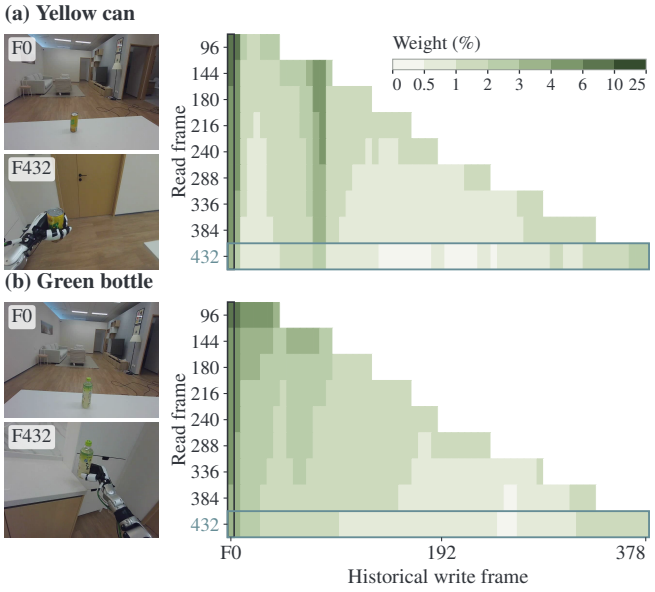


Fig. 8. **Early writes remain associated with later queries.** Two bottle-task trajectory replays from a checkpoint after 20,000 training steps. Rows index read frames and columns write frames. Read coefficients are normalized over the full history for each query and averaged across layers, heads, flow evaluations, and action queries. Only writes more than 48 frames old are shown, producing the staircase boundary. Left images show the initial target view (F0) and the object already held at F432.

b) Memory interfaces: MEMORY IN MOTION selected the correct target in all 20 eligible decisions and completed 19 of 20 task attempts (Table III). Current-only and Window fusion reached 55% and 70% first-choice accuracy. Direct write reached 65% first-choice accuracy and 55% task success, compared with 100% and 95% for MEMORY IN MOTION. Because Direct write and MEMORY IN MOTION both maintain persistent memory, this comparison supports constructing writes from retrieved evidence and current execution context within the shared recurrent architecture.

c) Query conditioning: We ablate the retrieval-query inputs while keeping persistent memory and ordinary policy

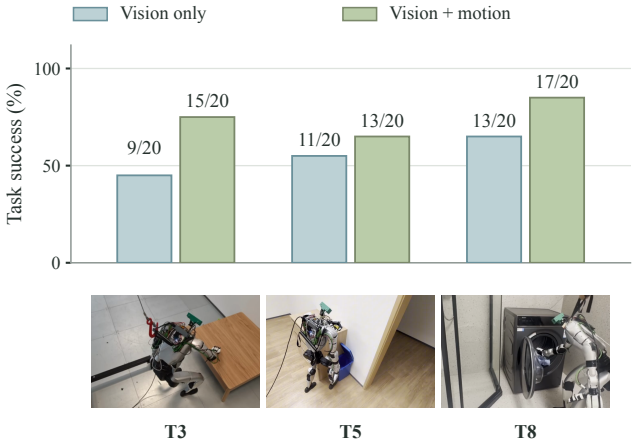


Fig. 9. **Additional language–motion supervision.** Success on three tasks requiring lower-body coordination with manipulation. Bars report success over 20 trials per task; labels give successful/attempted trials. The joint-training results reuse the corresponding task outcomes in Table I.

inputs intact. In *No motion query*, we remove $P_E([p_t; e_{t-1}])$ and d_t from Eq. (3); body/action write tokens remain. *No state query* removes only s_t, preserving recurrent updates and state reads by the Action Expert.

Removing the body–action and motion terms reduced first-choice accuracy from 100% to 75% and task success from 95% to 50%. Removing the state term reduced them to 85% and 75%, respectively. The full writer also achieved higher task success on T2 and T7 than either query ablation (Table III). These comparisons support using both embodied context and accumulated memory to condition evidence retrieval for subsequent writes.

F. Retention beyond the Recent Cache

a) Cue-free observations: We insert an additional cue-free dwell of 0, 2, or 6 seconds before the front-table choice. Each condition uses 20 choice-point arrivals, collecting additional episodes as needed under the protocol above. The configured six-frame execution stride at 30 Hz gives a nominal memory-update rate of 5 Hz. With observation processing active, a 6-second dwell corresponds to approximately 30 writes, exceeding the eight subsequent writes needed to evict a cue record from the recent cache.

MEMORY IN MOTION achieved 100%, 85%, and 75% first-choice accuracy at 0, 2, and 6 seconds, respectively (Fig. 7). The other three variants reached 50–70% at 2 seconds and 45–50% at 6 seconds. The higher recall therefore persisted during additional cue-free updates, including a dwell longer than the nominal cache window.

b) Historical associations: We expand the Action Expert's recurrent reads into implicit coefficients on earlier writes (Fig. 8). At F432, the initial write F0 ranked first among the 64 eligible older writes in both replays, with weights of 3.56% and 3.36%. These replays illustrate how early observations remain associated with later action queries after their records have left the recent cache.

G. Language–Motion Data for Whole-Body Execution

We compare VL2Action training with joint VL2Action and Motion2Action training while keeping the full writer and visual-interaction data fixed. Both variants use approximately 400 hours of visual-interaction data; the joint variant adds approximately 400 hours of language–motion data.

Joint training raised mean success from 55% to 75% across low-table retrieval, pedal-bin disposal, and laundry (Fig. 9). All three tasks improved, with the largest gain in low-table retrieval, from 45% to 75%. These results show that additional language–motion trajectories improve whole-body execution without increasing the visual-interaction data.

V. Conclusion and Limitation

MEMORY IN MOTION trains a shared humanoid Action Expert on language–motion and visual-interaction data, and preserves task evidence through contextual memory writes. Motion and accumulated execution state guide recent-evidence retrieval; a block delta update integrates the resulting writes into persistent memory. The real-robot and SIMPLE results show improved execution across changing views and whole-body interactions, while the comparison at matched visual-interaction data shows what language–motion data add to whole-body execution. Hidden-cue and delay tests show improved target selection through contextual writing, with recall declining as cue-free intervals lengthen.

Limitations. Task fine-tuning relies on real-robot demonstrations, which constrains how far the system scales to new tasks and embodiments. The evaluation also covers loco-manipulation routines with comparatively coarse contact; finer manipulation tasks would test both the memory interface and control precision more sharply.

References

[1] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski *et al.*, "RT-2: Vision-language-action models transfer web knowledge to robotic control," 2023. [Online]. Available: https://arxiv.org/abs/2307.15818

[2] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair *et al.*, "OpenVLA: An open-source vision-language-action model," 2024. [Online]. Available: https://arxiv.org/abs/2406.09246

[3] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn *et al.*, "π_0: A vision-language-action flow model for general robot control," 2024. [Online]. Available: https://arxiv.org/abs/2410.24164

[4] M. Koo, D. Choi, T. Kim, K. Lee, C. Kim, Y. Seo *et al.*, "HAMLET: Switch your vision-language-action model into a history-aware policy," 2026. [Online]. Available: https://arxiv.org/abs/2510.00695

[5] H. Shi, B. Xie, Y. Liu, L. Sun, F. Liu, T. Wang *et al.*, "MemoryVLA: Perceptual-cognitive memory in vision-language-action models for robotic manipulation," 2025. [Online]. Available: https://arxiv.org/abs/2508.19236

[6] Y. Hu, J.-N. Zaech, N. Nikolov, Y. Yao, S. Dey, G. Albanese *et al.*, "AR-VLA: True autoregressive action expert for vision-language-action models," 2026. [Online]. Available: https://arxiv.org/abs/2603.10126

[7] J. Yang, Y. Lu, Y. Su, Y. Zhong, Y. Xie, Y. Zhang *et al.*, "TemporalFlow-VLA: Learning physically grounded execution history for long-horizon robot manipulation," 2026. [Online]. Available: https://arxiv.org/abs/2608.26821

[8] Y. Dai, H. Fu, J. Lee, Y. Liu, H. Zhang, J. Yang *et al.*, "RoboMME: Benchmarking and understanding memory for robotic generalist policies," 2026. [Online]. Available: https://arxiv.org/abs/2603.04639

[9] A. Bulatov, Y. Kuratov, and M. S. Burtsev, "Recurrent memory transformer," 2022. [Online]. Available: https://arxiv.org/abs/2207.06881

[10] I. Schlag, K. Irie, and J. Schmidhuber, "Linear transformers are secretly fast weight programmers," in *Proceedings of the 38th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 139, 2021, pp. 9355–9366. [Online]. Available: https://proceedings.mlr.press/v139/schlag21a.html

[11] S. Yang, J. Kautz, and A. Hatamizadeh, "Gated delta networks: Improving Mamba2 with delta rule," 2025. [Online]. Available: https://arxiv.org/abs/2412.06464

[12] Kimi Team, Y. Zhang, Z. Lin, X. Yao, J. Hu, F. Meng *et al.*, "Kimi Linear: An expressive, efficient attention architecture," 2025. [Online]. Available: https://arxiv.org/abs/2510.26692

[13] T. D. Pala, N. Majumder, B. Goh, R. Yee, J. Yang, L. Chen *et al.*, "MemBodied: Recurrent associative memory for vision-language-action models," 2026. [Online]. Available: https://arxiv.org/abs/2609.28256

[14] Z. Fu, Q. Zhao, Q. Wu, G. Wetzstein, and C. Finn, "HumanPlus: Humanoid shadowing and imitation from humans," in *Proceedings of the 8th Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, vol. 270, 2025, pp. 2828–2844. [Online]. Available: https://proceedings.mlr.press/v270/fu25a.html

[15] Q. Ben, F. Jia, J. Zeng, J. Dong, D. Lin, and J. Pang, "HOMIE: Humanoid loco-manipulation with isomorphic exoskeleton cockpit," 2025. [Online]. Available: https://arxiv.org/abs/2502.13013

[16] J. Zheng, T. Ma, Y. Fan, Z. Wang, S. Yang, and J. Liang, "MotionWAM: Towards foundation world action models for real-time humanoid loco-manipulation," 2026. [Online]. Available: https://arxiv.org/abs/2606.09215

[17] C. Qin, S. Zhu, S. Luo, S. Huang, H. Zhao, S. Baduge *et al.*, "WB-WAM: Heterogeneous body-hand pre-training for humanoid loco-manipulation," 2026. [Online]. Available: https://arxiv.org/abs/2609.34199

[18] D. Jing, T. Zhang, J. Liu, J. Zhao, Z. Sun, L. E. Li *et al.*, "Learning action priors for cross-embodiment robot manipulation," 2026. [Online]. Available: https://arxiv.org/abs/2606.26095

[19] M. Shi, S. Peng, J. Chen, H. Jiang, T. Li, D. Huang *et al.*, "EgoHumanoid: Unlocking in-the-wild loco-manipulation with robot-free egocentric demonstration," 2026. [Online]. Available: https://arxiv.org/abs/2602.10106

[20] B. Liu, R. Wang, L. Wu, Y. Feng, P. Stone, and Q. Liu, "Longhorn: State space models are amortized online learners," in *International Conference on Learning Representations*, 2025. [Online]. Available: https://openreview.net/forum?id=8jOqCcLzeO

[21] DeepSeek-AI, "DeepSeek-V2: A strong, economical, and efficient mixture-of-experts language model," 2024. [Online]. Available: https://arxiv.org/abs/2405.04434

[22] Y. Lipman, R. T. Q. Chen, H. Ben-Hamu, M. Nickel, and M. Le, "Flow matching for generative modeling," in *International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=PqvMRDCJT9t

[23] C. Zhu, L. Xing, L. Li, Y. Liu, and Y. Xu, "PHT: Predictive whole-body humanoid teleoperation," 2026, manuscript.

[24] L. Beyer, A. Steiner, A. S. Pinto, A. Kolesnikov, X. Wang, D. Salz *et al.*, "PaliGemma: A versatile 3B VLM for transfer," 2024. [Online]. Available: https://arxiv.org/abs/2407.07726

[25] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang *et al.*, "LoRA: Low-rank adaptation of large language models," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=nZeVKeeFYf9

[26] Physical Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess *et al.*, "$\pi_{0.5}$: A vision-language-action model with open-world generalization," 2025. [Online]. Available: https://arxiv.org/abs/2504.16054

[27] S. Wei, Z. Ni, J. Liu, Z. Zhao *et al.*, "SIMPLE: Simulation-based policy learning and evaluation for humanoid loco-manipulation," *arXiv preprint arXiv:2606.08278*, 2026, version 2. [Online]. Available: https://arxiv.org/abs/2606.08278v2