Extended Abstract

Motivation Vision-Language-Action (VLA) models such as $\pi_{0.5}$ Black et al. (2025) implicitly assume that the *current* RGB observation is a sufficient statistic for choosing the next action. This Markovian assumption breaks down in any task where the same scene demands different actions depending on what has already happened, a setting studied extensively in POMDP benchmarks and recurrent RL Morad et al. (2023); Ni et al. (2022). Our running example is *table wiping*: a Franka Panda arm must clean an entire table from randomized start poses, but a region that has already been wiped is visually identical to one that has not, so the correct next sweep cannot be inferred from a single frame. Under this partial observability, naively fine-tuning a VLA fails in a characteristic way: behavior cloning averages over conflicting sweep directions present in the demonstrations, collapsing lateral motion toward zero and causing the arm to stall. Equipping the policy with the right memory is therefore essential for the many real-world tasks—cleaning, sorting, inspection, assembly—that require acting on accumulated history rather than the latest frame.

Method We introduce a *pretrained, action-conditioned recurrent memory encoder* that augments a $\pi_{0.5}$ Black et al. (2025) policy. Given a short history of past observations and actions $(o_{t-K:t-1}, a_{t-K:t-1})$ and the previous memory state M_{t-1}, the encoder produces a fixed-size latent memory $M_t = E_{\text{mem}}(M_{t-1}, o_{t-K:t-1}, a_{t-K:t-1})$. Rather than training this memory only through downstream action supervision—a long, weak credit-assignment path—we pretrain it with a self-supervised *predictive* objective: from M_t a lightweight decoder predicts the fused future latent that a *frozen* $\pi_{0.5}$ teacher assigns to the next H steps, and we minimize the MSE to that target. This forces memory to store exactly the history that is predictive of future robot states and actions. After pretraining the decoder is discarded and the memory encoder is attached to $\pi_{0.5}$: its memory tokens are projected into the policy's prefix-token space and prepended as additional context, after which the policy is fine-tuned (with LoRA) on the target task.

Implementation The memory encoder factorizes attention spatio-temporally: a ViT-style image encoder over $K{=}8$ past 128×128 frames, an action encoder over the 8-dim joint-position actions, and a π_0/Perceiver-style cross-attention bottleneck in which $L{=}64$ memory tokens query the concatenated image, action, and previous-memory tokens. Pretraining targets come from a frozen $\pi_{0.5}$ (PaliGemma backbone) using the full RoboMME cross-robot dataset Dai et al. (2026). For the downstream task we built a RoboMME-format table-wipe environment in ManiSkill Gu et al. (2023) with a Franka Panda arm and collected 100 teleoperated demonstrations. We report table-area coverage averaged over 30 simulated rollouts with randomized initial poses.

Results We compare four conditions. (a) Zero-shot $\pi_{0.5}$ achieves **0.0%** coverage—severe out-of-distribution behavior without fine-tuning. (b) Fine-tuning without memory reaches only **5.0%**, exhibiting the predicted action-mode collapse. (c) Fine-tuning with a *randomly initialized* memory encoder gives **7.8%**: a memory architecture alone, with limited downstream data, learns little useful structure. (d) Fine-tuning with our *pretrained* memory encoder reaches **20.3%**, a roughly $4\times$ improvement over the memoryless policy and a $2.6\times$ improvement over the un-pretrained memory, while human teleoperation upper-bounds the task at **59.2%**.

Discussion The (c) vs. (d) gap is the central finding: the benefit comes not from *having* a memory module but from *pretraining* it. Cross-robot predictive pretraining gives the encoder a structured head start so that, with only 100 task demonstrations, it can compress interaction history into a representation that disambiguates which sweep direction is correct. The remaining gap to human performance reflects both the difficulty of the task and practical limits: rollout evaluation is time-consuming, and the limited number of demonstrations constrains the amount of downstream fine-tuning data.

Conclusion We show that pretraining an action-conditioned recurrent memory encoder—supervised to predict a frozen VLA's future latents—substantially improves a $\pi_{0.5}$ policy on a partially observable manipulation task, turning a stalled memoryless baseline into a policy that meaningfully covers the table. Future work includes comparing encoder-only and encoder–decoder memory designs, GRU- versus Transformer-based recurrent updates, tuning the history window K, horizon H, and latent dimension, and generalizing to other POMDP-style manipulation tasks.

Memory Pretraining for Vision-Language-Action Model

Pengyu Mo
Department of Electrical Engineering
Stanford University
pengyumo@stanford.edu

Haowen Wang
Department of Electrical Engineering
Stanford University
haowenw@stanford.edu

Zhen Jia
Department of Electrical Engineering
Stanford University
zhenjia1@stanford.edu

Abstract

Vision-Language-Action (VLA) models have become strong general-purpose manipulation policies, but they treat the current observation as a sufficient statistic for action selection and therefore struggle on partially observable tasks where visually identical states require different behaviors. We study this through table wiping, where an already-wiped region looks the same as an unwiped one and naive behavior cloning collapses by averaging over conflicting sweep directions. We propose to augment a $\pi_{0.5}$ Black et al. (2025) policy with a recurrent memory encoder that summarizes a short observation-action history into a fixed-size latent, and—crucially—to *pretrain* that encoder with a self-supervised predictive objective against a frozen $\pi_{0.5}$ teacher's future latents on the large cross-robot RoboMME dataset. The pretrained memory is then projected into the policy's prefix tokens and the model is fine-tuned with LoRA. On a RoboMME-format ManiSkill table-wipe task with a Franka Panda arm, the pretrained-memory policy reaches 20.3% table-area coverage, versus 5.0% for memoryless fine-tuning and 7.8% for a randomly initialized memory encoder, with human teleoperation at 59.2%. The ablation isolates pretraining—not merely the presence of a memory module—as the source of the gain.

1 Introduction

General-purpose Vision-Language-Action (VLA) models map language instructions and camera observations directly to robot actions and have shown strong cross-task and cross-embodiment generalization. Most of these policies, including $\pi_{0.5}$ Black et al. (2025) , are reactive: they condition on the current frame (and instruction) and implicitly assume it contains everything needed to choose the next action. This Markovian assumption is convenient and often adequate, but it fails on the large class of manipulation tasks that are *partially observable*, where the same visual observation maps to different correct actions depending on the agent's interaction history. The challenge of acting under partial observability is well documented in the POMDP and recurrent-RL literature Morad et al. (2023); Ni et al. (2022), yet it is comparatively underexplored for modern VLA policies on real manipulation skills.

We use *table wiping* as a concrete and demanding instance of this problem. A Franka Panda arm holding a sponge must wipe an entire table starting from randomized poses. Once a region has been cleaned it is visually indistinguishable from an uncleaned region, so the correct next sweep is not a

function of the current image alone—it depends on where the arm has already been. When a reactive VLA is fine-tuned on demonstrations of this task, behavior cloning averages over the conflicting sweep directions associated with near-identical observations. The averaged policy collapses its lateral action variance toward zero, and the arm effectively stalls. Solving the task therefore requires the policy to remember and reason about its own past interactions.

A natural fix is to give the policy memory, and recent work has begun to attach memory modules to VLA policies. However, these modules are typically trained from scratch per task through action supervision alone, which creates a long, weak credit-assignment path between what the memory stores and the downstream loss, and is data-hungry on the target task. Our key idea is to *pretrain* the memory encoder on large-scale cross-robot data with a direct, self-supervised objective before any task-specific fine-tuning. Concretely, we train an action-conditioned recurrent encoder so that its memory state predicts the future latent representation assigned by a frozen $\pi_{0.5}$ teacher. This aligns the memory with information that is provably useful for downstream prediction and decouples representation learning from the scarce, task-specific reward/imitation signal.

This report makes the following contributions:

- We formulate VLA fine-tuning under partial observability and characterize how reactive behavior cloning fails on table wiping via action-mode collapse.
- We propose a pretrained, action-conditioned recurrent memory encoder for VLA policies, trained with a predictive objective against a frozen $\pi_{0.5}$ teacher's fused future latents on the RoboMME cross-robot dataset Dai et al. (2026), and a mechanism to inject its memory tokens into the $\pi_{0.5}$ prefix for LoRA fine-tuning.
- We build a RoboMME-format table-wipe environment in ManiSkill Gu et al. (2023) with a Franka Panda arm and a SpaceMouse 3Dconnexion (2026) teleoperation pipeline, and use table-area coverage as the evaluation metric.
- Through a four-way comparison (zero-shot, memoryless fine-tuning, randomly initialized memory, and pretrained memory) we show that the pretrained memory encoder improves coverage to 20.3%—about $4\times$ over memoryless fine-tuning—and that an ablation against a randomly initialized memory isolates *pretraining* as the source of the gain.

2 Related Work

Memory for long-horizon embodied tasks. Torne et al. Torne et al. (2026) introduce MEM, a multi-scale memory module that combines a short-term perceptual buffer with a long-term semantic store of compressed language summaries, enabling agents to retain information over extended interactions. Their short-term buffer motivates our windowed history encoder, but MEM trains its memory only indirectly through action supervision, yielding a long and weak credit-assignment path. We instead supply a direct, self-supervised pretraining signal so the memory learns predictive structure before task-specific fine-tuning.

Cognitively-inspired memory for VLA models. Shi et al. Shi et al. (2025) propose MemoryVLA, which pairs a short-term working memory with a long-term bank of perceptual-cognitive representations that are retrieved and fused with the current observation to guide action generation. While effective, the memory is trained from scratch for each task and its design is largely fixed; the work does not study how *pretraining* the memory on large-scale data affects downstream performance. Our contribution is orthogonal and complementary: a pretraining recipe for the memory encoder itself.

Benchmarking memory under partial observability. Morad et al. Morad et al. (2023) compare many memory architectures across partially observable environments and find that recurrent models such as GRUs are consistently strong, while Ni et al. Ni et al. (2022) show that carefully tuned recurrent model-free RL is a strong POMDP baseline. These studies establish that memory matters under partial observability, but they evaluate on synthetic, game-like, or locomotion tasks rather than visuomotor manipulation, and they do not address how to pretrain a memory module for a downstream VLA.

Object-centric memory for non-Markovian manipulation. Chung et al. Chung et al. (2026) propose a slot-centric VLA that tracks per-object interaction histories, with an accompanying LIBERO-Mem benchmark. Their focus is on *which* object to act on and relies on structured, object-centric

scene representations. Our task instead hinges on *whether a region was already acted upon* under ambiguous visual feedback, and our memory operates on a compact latent summary rather than explicit per-object slots.

Our gap. Across this prior work, no method applies a principled, large-scale pretraining objective on diverse cross-robot data to initialize a memory encoder *before* VLA fine-tuning on a partially observable manipulation task. RoboMME Dai et al. (2026) supplies the cross-robot data and a memory benchmark but does not pretrain such an encoder. We fill this gap with an action-conditioned recurrent memory encoder pretrained to predict a frozen $\pi_{0.5}$ teacher's future latents, and we quantify its benefit with a pretrained-vs-random ablation.

3 Method

3.1 Problem Formulation

We study long-horizon table wiping as a partially observable manipulation problem. At each control step t, the robot observes RGB images from front and wrist cameras together with proprioceptive state $s_t \in \mathbb{R}^8$, consisting of seven Franka joint angles and gripper width. The policy predicts an action chunk $a_{t:t+H_a-1}$ in joint-position space, where each action contains seven joint targets and one gripper command.

This task requires memory because the relevant task state, namely which regions of the table have already been wiped, is not directly observable from a single camera frame. Episodes last 600–1200 control steps at 20 Hz, and a reactive policy must repeatedly infer the latent coverage state from a narrow visual field. We therefore augment a pretrained VLA policy with an explicit recurrent memory module.

3.2 Recurrent Memory Backbone

Our memory backbone summarizes a sliding window of past observations and actions into a fixed-size recurrent memory state. Let K denote the history length and let $M_t \in \mathbb{R}^{L\times d}$ be the memory state at time t, where L is the number of memory tokens and d is the token width. The memory update is

$$M_t = f_\theta \left(I_{t-K:t-1}, a_{t-K:t-1}, M_{t-1} \right). \tag{1}$$

The backbone contains three components. First, a spatio-temporal image encoder patchifies each historical frame and applies factorized Transformer blocks: spatial attention within each frame followed by temporal attention across frames. Second, an action encoder embeds each scalar action dimension and applies factorized attention over action dimensions and time. Third, a Perceiver-style memory encoder uses the previous memory tokens as queries and cross-attends to the concatenated image tokens, action tokens, and previous memory state:

$$M_t = \mathrm{CrossAttn}\left(M_{t-1}, [Z_t^{\mathrm{img}} \| Z_t^{\mathrm{act}} \| M_{t-1}] \right). \tag{2}$$

Because the output memory has a fixed size, the model can propagate information through arbitrarily long episodes with constant per-step memory cost. This design allows the policy to retain task-relevant history, such as previously wiped table regions, beyond the local observation window.

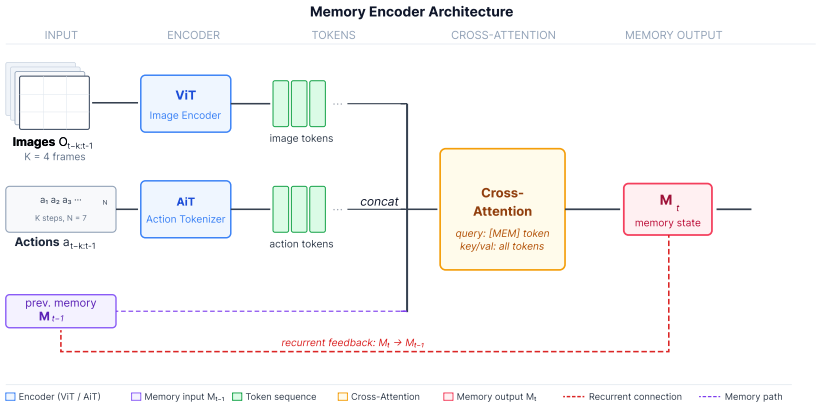

Figure 1: Recurrent Memory Backbone.

3.3 Memory Pretraining

The goal of memory pretraining is to teach the encoder to retain past information that is predictive of future behavior. Thus, we pretrain the memory backbone using a frozen $\pi_{0.5}$ teacher on RoboMME Dai et al. (2026) dataset. For each training window, the memory encoder compresses recent observations and actions into memory tokens, while the teacher processes a short future horizon and produces a fused latent target z_t from its image and action representations. A lightweight decoder reads the memory state and predicts this teacher latent:

$$\mathcal{L}_{\text{mem}} = \|\text{Dec}(M_t) - z_t\|_2^2 . \tag{3}$$

This distillation objective encourages the memory state to capture temporal context that is useful for predicting future visual and action representations, rather than only reconstructing the recent observation window. Pretraining is performed sequentially within each episode. The memory state is rolled forward across time, and gradients are propagated through the recurrence using truncated backpropagation through time. During downstream finetuning, this pretrained memory can help the policy infer hidden task state, such as which table regions have already been wiped.

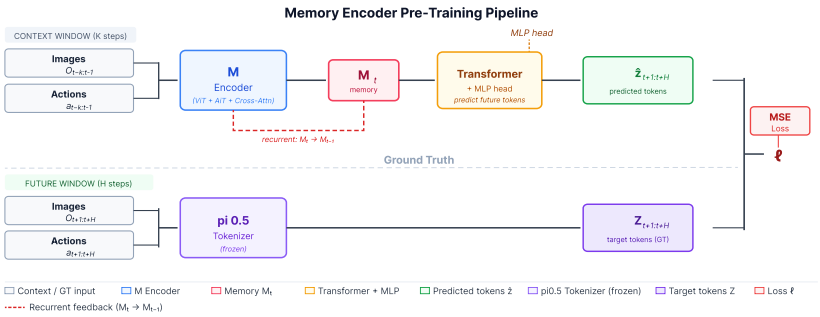

Figure 2: Memory Pretraining Pipeline.

3.4 Memory-Conditioned $\pi_{0.5}$ Policy

We integrate the pretrained memory backbone into $\pi_{0.5}$, a flow-matching vision-language-action policy. Given an action chunk a, Gaussian noise ϵ, and flow time τ, $\pi_{0.5}$ constructs the interpolated action $x_\tau = \tau\epsilon + (1-\tau)a$ and learns the velocity field toward the clean action:

$$\mathcal{L}_{\pi_0} = \|v_\phi(x_\tau, \tau, \text{context}) - (\epsilon - a)\|_2^2 . \tag{4}$$

We inject memory as additional prefix tokens. The frozen memory backbone produces M_t, which is mapped into the $\pi_{0.5}$ prefix embedding space by a trainable projection, layer normalization, and

learned scale parameter:

$$\widetilde{M_t} = \gamma \cdot \mathrm{LN}(W M_t). \tag{5}$$

The projected memory tokens are prepended to the original image-language prefix tokens and are visible to all downstream action tokens through full attention. During task finetuning, we freeze both the memory backbone and the base $\pi_{0.5}$ weights, and train only the memory projection and LoRA adapters on $\pi_{0.5}$.

3.5 Ablation Baseline

To isolate the effect of recurrent memory, we compare against an otherwise identical $\pi_{0.5}$ LoRA finetuning baseline with the memory pathway disabled. This baseline uses the same task data, optimization schedule, and adapter configuration, but receives no recurrent memory tokens. Any performance difference therefore reflects the contribution of the pretrained memory module rather than additional model capacity or training data.

4 Experimental Setup

We evaluate whether memory pretraining improves the performance of Vision-Language-Action (VLA) models on partially observable manipulation tasks. Our experiments are conducted in a custom ManiSkill/SAPIEN table wiping environment using a Franka Panda robot.

4.1 Task Description

The robot is required to wipe the entire tabletop from randomized initial poses. This task is inherently partially observable because previously cleaned regions become visually indistinguishable from untouched regions. Consequently, the correct action cannot always be inferred from the current RGB observation alone, making memory essential for successful task completion.

4.2 Dataset

For downstream fine-tuning, we collect 100 teleoperated demonstrations using a SpaceMouse controller. Each demonstration contains RGB observations and corresponding robot actions. For memory pretraining, we use the full RoboMME Dai et al. (2026) dataset, which contains large-scale cross-robot manipulation trajectories spanning diverse embodiments and tasks.

4.3 Training Conditions

To isolate the effects of memory and memory pretraining, we evaluate four conditions:

1. Zero-shot $\pi_{0.5}$ without task-specific training.
2. Fine-tuned $\pi_{0.5}$ without a memory encoder.
3. Fine-tuned $\pi_{0.5}$ with a randomly initialized memory encoder.
4. Fine-tuned $\pi_{0.5}$ with a pretrained memory encoder.

The pretrained memory encoder is first trained on RoboMME Dai et al. (2026) using a future prediction objective. During downstream training, the memory encoder remains frozen and provides memory representations to the policy, while only the memory projection and $\pi_{0.5}$ LoRA adapters are fine-tuned on the table wiping demonstrations.

4.4 Evaluation Protocol

Each model is evaluated over 30 simulated rollouts with randomized initial conditions. Performance is measured using table coverage, defined as the percentage of the tabletop successfully wiped by the robot during an episode.

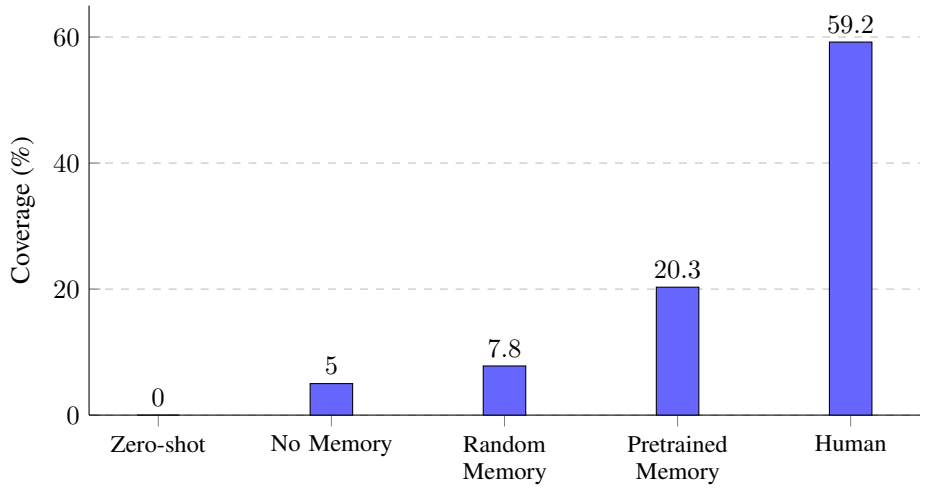

Figure 3: Table wiping coverage under different training conditions. Pretraining the memory encoder improves coverage by approximately $4\times$ compared with the memoryless baseline.

5 Results

Table 1 summarizes the performance of all evaluated methods on the table wiping task.

5.1 Quantitative Evaluation

Table 1: Table wiping performance measured by coverage (%).

Method	Coverage (%)
Zero-shot $\pi_{0.5}$	0.0 ± 0.0
Fine-tuned $\pi_{0.5}$ (No Memory)	5.0 ± 3.75
Fine-tuned $\pi_{0.5}$ + Random Memory	7.8 ± 4.12
Fine-tuned $\pi_{0.5}$ + Pretrained Memory	20.3 ± 3.71
Human Teleoperation	59.2 ± 1.32

The zero-shot policy completely fails to solve the task, achieving 0% coverage. This result highlights the significant distribution shift between the pretraining data and the table wiping environment.

Fine-tuning without memory improves performance only marginally, reaching 5.0% coverage. Because visually identical observations can correspond to different desired actions, behavior cloning struggles to learn a consistent policy and often averages over conflicting action modes.

Adding a randomly initialized memory encoder yields only a modest improvement to 7.8%, suggesting that learning useful memory representations from the limited downstream dataset is difficult.

In contrast, the pretrained memory encoder achieves 20.3% coverage, representing approximately a fourfold improvement over the memoryless baseline. This result demonstrates that large-scale memory pretraining provides a strong initialization that transfers effectively to downstream partially observable tasks.

5.2 Qualitative Analysis

The experiment rollouts show clear behavioral differences between the memoryless and memory-augmented policies. The memoryless baseline often stalls after wiping a small region or repeatedly sweeps over the same area. Since the wiped surface leaves only a weak visual trace, a reactive policy cannot infer from the current frame alone which regions have already been covered, resulting in redundant or oscillatory motions. Contact diagnostics also show that many baseline episodes end with the sponge hovering above the surface rather than maintaining sustained wiping pressure.

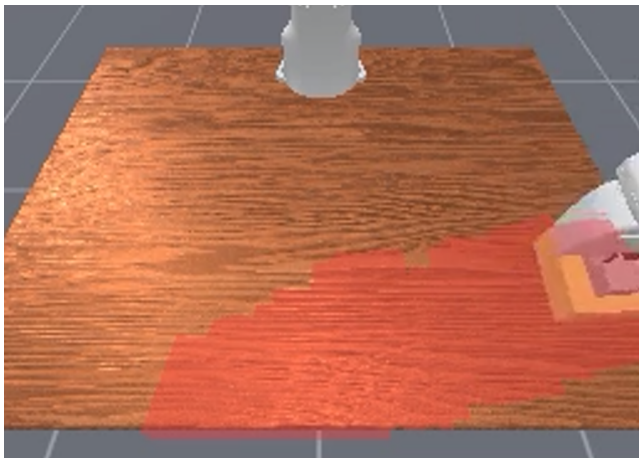

Figure 4: Rollout visualization of the memory-pretrained policy on the table-wiping task, with an achieved table coverage of 19.2%.

In contrast, the pretrained memory policy produces more coherent wiping trajectories. Its strokes are longer, more continuous, and more likely to move from previously wiped regions toward adjacent uncovered areas. It also maintains table contact more consistently, suggesting that memory helps both with selecting the next region to wipe and with sustaining useful surface interaction.

Overall, these results support our hypothesis that table wiping is partially observable and benefits from recurrent memory. A pretrained memory encoder helps disambiguate visually similar states by encoding recent observation-action history, reducing repetitive behavior and improving coverage. Remaining failures mostly occur when the policy fails to establish stable contact early, suggesting that memory improves but does not fully solve the contact-stability challenge.

6 Discussion

Our results demonstrate that memory plays a critical role in partially observable robotic manipulation tasks. In the table wiping environment, the current observation alone is often insufficient to determine the appropriate action because visually similar states may correspond to different cleaning histories. As a result, policies trained purely through behavior cloning struggle to disambiguate these situations and frequently exhibit repetitive or inconsistent behaviors.

The comparison between randomly initialized memory and pretrained memory highlights the importance of large-scale memory pretraining. Although both approaches provide the policy with an additional memory pathway, only the pretrained memory encoder leads to substantial performance gains. This suggests that the downstream dataset is too small to learn effective memory representations from scratch and that pretraining provides a useful inductive bias that transfers across tasks.

Despite the improvement achieved by pretrained memory, the performance gap between the learned policy and human teleoperation remains significant. Human operators achieve nearly three times higher coverage, indicating that the current memory representation captures only part of the information required for long-horizon task execution. In particular, failures often occur when the policy revisits already cleaned regions or loses track of unexplored areas. These observations suggest that more structured forms of memory, such as explicit spatial representations or task-progress modeling, may further improve performance.

Several limitations should also be acknowledged. First, the evaluation is conducted on a single table wiping task with a relatively limited amount of downstream data. Additional experiments on more diverse manipulation tasks would be necessary to assess the generality of the findings. Second, coverage alone does not fully characterize task performance, and future evaluations could incorporate metrics related to efficiency, trajectory quality, and completion time. Finally, the current work focuses on frozen memory pretraining and does not investigate alternative pretraining objectives or memory architectures.

Future work may explore larger-scale memory pretraining, integration with spatial world models, and application to more challenging long-horizon robotic tasks. We believe that effective memory representations will become increasingly important as robotic systems are deployed in complex real-world environments where partial observability is unavoidable.

7 Conclusion

In this work, we investigated the impact of memory pretraining on robotic table wiping under partial observability. We compared zero-shot policies, fine-tuned policies without memory, policies equipped with randomly initialized memory, and policies using a pretrained memory encoder. Experimental results show that memory pretraining substantially improves task performance, increasing coverage from 5.0% to 20.3% and outperforming both the memoryless and randomly initialized baselines.

Qualitative observations further reveal that pretrained memory enables more coherent and exploratory behaviors, allowing the policy to better utilize information from previous interactions. These findings suggest that memory pretraining provides a transferable representation that helps resolve ambiguities arising from partial observability.

Although there remains a considerable gap between the learned policy and human performance, our results provide evidence that large-scale memory pretraining is a promising direction for improving long-horizon robotic manipulation. We hope this study motivates future research on memory-augmented robotic policies and more effective methods for learning and utilizing long-term experience.

8 Team Contributions

- **Pengyu Mo:** Proposed the memory framework, built the simulation environment, collected fine-tune data, and ran fine-tuning experiments.
- **Haowen Wang:** Built the memory pretraining pipeline, pretrained the memory module, and created the poster.
- **Zhen Jia:** Built the memory pretraining pipeline, fine-tuned $\pi_{0.5}$ with the memory module, and ran ablation study experiments.

Changes from Proposal Mostly similar, and each of us had a chance to run part of the training and experiments.

References

3Dconnexion. 2026. SpaceMouse: 3D Navigation Devices. `https://3dconnexion.com/us/spacemouse/`. Accessed: 2026-04-30.

Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. 2025. $\pi_{0.5}$: a Vision-Language-Action Model with Open-World Generalization. *arXiv preprint arXiv:2504.16054* (2025).

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

Yinpei Dai, Hongze Fu, Jayjun Lee, Yuejiang Liu, Haoran Zhang, Jianing Yang, Chelsea Finn, Nima Fazeli, and Joyce Chai. 2026. RoboMME: Benchmarking and Understanding Memory for Robotic Generalist Policies. *arXiv preprint arXiv:2603.04639* (2026).

Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, et al. 2023. ManiSkill2: A Unified Benchmark for Generalizable Manipulation Skills. *arXiv preprint arXiv:2302.04659* (2023).

Steven Morad, Ryan Kortvelesy, Matteo Bettini, Stephan Liwicki, and Amanda Prorok. 2023. PopGym: Benchmarking Partially Observable Reinforcement Learning. In *International Conference on Learning Representations*.

Tianwei Ni, Benjamin Eysenbach, and Ruslan Salakhutdinov. 2022. Recurrent Model-Free RL Can Be a Strong Baseline for Many POMDPs. In *Proceedings of the 39th International Conference on Machine Learning*. PMLR. `https://proceedings.mlr.press/v162/ni22a.html`

Hao Shi, Bin Xie, Yingfei Liu, Lin Sun, Fengrong Liu, Tiancai Wang, Erjin Zhou, Haoqiang Fan, Xiangyu Zhang, and Gao Huang. 2025. MemoryVLA: Perceptual-Cognitive Memory in Vision-Language-Action Models for Robotic Manipulation. *arXiv preprint arXiv:2508.19236* (2025).

Marcel Torne, Karl Pertsch, Homer Walke, Kyle Vedder, Suraj Nair, Brian Ichter, Allen Z Ren, Haohuan Wang, Jiaming Tang, Kyle Stachowicz, et al. 2026. MEM: Multi-Scale Embodied Memory for Vision Language Action Models. *arXiv preprint arXiv:2603.03596* (2026).

A Implementation Details

A.1 Architecture Configuration

The memory backbone uses an embedding width of $d = 384$ with 8 attention heads, corresponding to 48 dimensions per head, and an MLP ratio of 4.0. Input frames are resized to 128×128 and tokenized using 8×8 non-overlapping patches, resulting in $P = 256$ visual tokens per frame. The image encoder contains $N_s = 2$ spatial attention layers and $N_t = 2$ temporal attention layers. The action encoder contains $N_w = 1$ within-step attention layer and $N_{at} = 1$ temporal attention layer. Dropout is set to 0.1 throughout the memory backbone. The history window length is $K = 8$, corresponding to 0.4 seconds at 20 Hz, and the recurrent memory state contains $L = 64$ tokens. The distillation decoder uses a hidden width of 1024 and predicts a 3072-dimensional fused teacher latent. Past actions provided to the memory backbone are 8-dimensional joint actions and are z-score normalized using statistics computed over the full RoboMME training pool.

The base policy is initialized from the $\pi_{0.5}$ `pi05_droid_jointpos` checkpoint. We adapt it using LoRA with rank 8, $\alpha = 16$, scaling $\alpha/r = 2$, and dropout 0. LoRA adapters are applied to both attention and MLP projection layers. $\pi_{0.5}$ states and action chunks use quantile normalization, mapping the q_{01} and q_{99} percentiles to $[-1, 1]$.

A.2 Datasets and Task

Memory pretraining corpus. We pretrain the memory backbone on RoboMME, which contains 16 Franka manipulation tasks with 100 episodes per task, for a total of 1600 episodes. Each episode contains front-view RGB images, wrist-view RGB images, robot joint states, and joint/end-effector actions. We use all 1600 episodes for memory pretraining.

Target task. The downstream task is a custom WIPETABLE environment built in ManiSkill/SAPIEN. A 7-DoF Franka Panda holds a compound foam sponge and must wipe a table of approximately $0.97\,\mathrm{m} \times 0.81\,\mathrm{m}$. We collect 100 demonstrations in the RoboMME format. The robot is controlled in absolute joint-position mode, with 8-dimensional actions consisting of seven joint targets and one gripper command. We verified that this control mode preserves demonstration quality by replaying the demonstrations in simulation.

A.3 Training Procedure

We use AdamW with learning rate 1.0×10^{-4}, weight decay 1.0×10^{-4}, a linear warmup of 500 steps followed by cosine decay, gradient clipping at norm 1.0, and batch size 16.

Memory backbone pretraining. The memory backbone is trained for 15 epochs on RoboMME in sequential mode. During pretraining, episodes are rolled forward recurrently, $M_{t-1} \to M_t$, using truncated backpropagation through time with a window length of 6. We set `detach_memory=False`

so gradients can flow through the recurrent memory update window. Each optimization step processes 4 parallel episode streams.

Policy finetuning. The memory-conditioned $\pi_{0.5}$ policy is finetuned for 5000 steps with batch size 16, and checkpoints are saved every 1000 steps. The memory backbone is loaded from the pretrained checkpoint and kept frozen during policy finetuning. The checkpoint also stores the action mean and standard deviation, ensuring that the memory backbone receives actions normalized consistently with pretraining. During finetuning, we update only the memory projection module and the LoRA adapters on $\pi_{0.5}$.