Memory-Augmented VLM Planners for Long-Horizon VLA Control via RL

Krish Sharma Lucas Burgett

Department of Computer Science, Stanford University

Advisor: Marcel Tornè CS 224R — Deep Reinforcement Learning

Extended Abstract

Long-horizon robot manipulation requires both motor skill and episodic memory. State-of-the-art Vision-Language-Action (VLA) models like $\pi_{0.5}$ handle the former effectively but are memoryless—they cannot recall which container held which object before it was covered. Prior work (MemER, MEM) addresses this through imitation learning on $\sim$50 human-teleoperated demonstrations per task, but this bottleneck limits scaling to new environments. We propose a **demo-free hierarchical memory VLA**: a Qwen3-VL-4B planner with a persistent keyframe buffer, trained via streaming GRPO on dense task-completion reward from simulation, above a frozen GroundSG $\pi_{0.5}$ action expert.

Our main contributions are: **(1)** a *streaming* GRPO architecture in which the VLM is queried once per pick, gradients flow across the full multi-pick episode via a shared reward, jointly training subgoal prediction and memory selection; **(2)** an algorithmic survey of GRPO, RLOO, and PPO, characterizing how coordinate mismatch zeroes reward variance and how PPO format collapse is independent of reward quality; **(3)** a 5-seed buffer ablation showing cross-pick persistence is necessary and sufficient (buffer-reset: 9.5%; FIFO: no effect); and **(4)** the first direct comparison of demo-free RL to MemER-IL on binary success rate.

Method	BtnUnmaskSwap SR	BtnUnmask SR	Overall	Human Data
Frozen $\pi_{0.5}$ (no memory)	6.7%†	22.2%	17.9%	0
MemER-IL [2]	21.3%	72.0%	42.4%	$\sim$50 demos/task
SFT warm-start (ours)	**26.2%**	—	—	0 (oracle sim)
GRPO streaming, step 25 (ours)	**31.4% $\pm$3.1%**	—	—	0
GroundSG + Oracle (ceiling)	$\sim$93%	95.0%	84.1%	—

SR = binary success rate on held-out seed 20260601, streaming evaluator, greedy decode. Highlighted rows are our methods, trained and evaluated on ButtonUnmaskSwap only; BtnUnmask and Overall are not reported as they were not the training target. †Published RoboMME number; our seed-7 run yields 10%, reflecting normal seed variance across 50 episodes.

Streaming GRPO achieves **31.4% $\pm$3.1% success rate on ButtonUnmaskSwap with zero human demonstrations**, exceeding MemER-IL (21.3%) by **+10.1 pp**. SFT alone (26.2%) already surpasses MemER; RL adds **+5.2 pp** in 25 gradient steps. A 5-seed ablation shows **cross-pick buffer persistence is necessary and sufficient** (buffer-reset: 9.5%); FIFO context does not affect success rate (No-FIFO: 30.7% $\approx$ Standard: 30.0%). A coordinate space mismatch silently prevented all RL methods from learning; we document its signature and fix as a transferable diagnostic.

Abstract

We present a demo-free hierarchical memory-augmented Vision-Language-Action (VLA) system that trains a Qwen3-VL-4B planner with streaming GRPO on simulation reward alone, requiring zero human demonstrations. A persistent keyframe buffer provides cross-pick episodic memory above a frozen GroundSG $\pi_{0.5}$ action expert. On the RoboMME ButtonUnmaskSwap benchmark, our approach achieves **31.4% ±3.1%** binary success rate (5-seed), exceeding MemER-IL (21.3%, trained on ∼50 human demonstrations per task) by +10.1 pp. A 5-seed buffer ablation establishes that cross-pick persistence is the necessary and sufficient memory component (buffer-reset: 9.5%); FIFO context does not affect success rate. We further identify and resolve a coordinate space mismatch between SFT training targets and Qwen3-VL's native grounding space that silently zeroed reward variance in all RL methods, and document its failure signature as a transferable diagnostic for grounded VLM systems.

1. Introduction

Embodied agents operating over long time horizons face two distinct challenges: executing precise motor skills in real time, and maintaining structured episodic memory of earlier scene states. Large-scale pretrained Vision-Language-Action models like $\pi_{0.5}$ [1] solve the first problem effectively but are memoryless—they condition only on recent observations and cannot recall which container held which object before it was covered.

The RoboMME benchmark [4] quantifies this gap precisely. A frozen $\pi_{0.5}$ achieves 17.9% overall success across 16 tasks, but the per-task breakdown (Table 3) reveals a stark pattern: tasks dominated by motor skill score reasonably well (MoveCube 34%, VideoPlaceButton 30%), while tasks requiring cross-episode memory collapse—VideoRepick 2%, InsertPeg 4%, PatternLock 4%. The Permanence category, which requires recalling container-object associations across occlusion events, is the clearest failure mode: ButtonUnmaskSwap—which requires remembering which container hid which cube before a swap—scores only 10% despite using motor-identical primitives to its base task ButtonUnmask at 26%. This gap is attributable entirely to episodic memory.

Prior work closes this gap through imitation learning. MemER [2] trains a Qwen2.5-VL-7B planner on ∼50 human-teleoperated demonstrations per task, reaching 72% on ButtonUnmask and 42.4% overall. While effective, this requires a physical robot, an Oculus VR teleoperation setup, and per-task keyframe annotation—a bottleneck that limits scaling to new tasks and environments. MEM [3] retrains the entire VLA end-to-end on large video datasets but discards the modularity of a frozen motor policy.

We ask: *can reinforcement learning replace the demonstration requirement?* We train the VLM planner using streaming GRPO on dense task-completion reward from simulation, with zero human demonstrations. Our primary experimental question is whether RL training adds measurable capability beyond the SFT warm-start on ButtonUnmaskSwap—the task most demanding of cross-pick spatial memory. The answer is yes: streaming GRPO adds **+5.2 pp** in binary success rate over SFT alone (31.4% ±3.1% vs. 26.2%), and both exceed MemER's demo-trained 21.3%.

Applying GRPO to this setting is non-trivial in a specific way. Standard single-step GRPO treats each VLM call independently, but episodic memory training requires credit assignment *across picks*: the keyframe selection at pick 0 (during the reveal window) causally determines whether the model can ground the correct container at pick 1 (after occlusion). Our streaming architecture propagates a single episode reward gradient through all per-pick VLM calls in a trajectory, jointly optimizing subgoal prediction and memory selection from a shared reward signal—without any per-pick supervision or human-labeled keyframe annotations. This cross-pick gradient flow is the architectural innovation that makes RL-based episodic memory training tractable.

Our contributions are: **(1)** a demo-free streaming GRPO pipeline for hierarchical memory-augmented VLAs; **(2)** an algorithmic survey of GRPO, RLOO, and PPO identifying distinct failure modes; **(3)** a 5-seed buffer ablation establishing that cross-pick buffer persistence is necessary and sufficient while FIFO context does not affect final success rate; and **(4)** characterization of a coordinate space failure mode that silently zeroes reward variance, with a transferable diagnostic and fix.

2. Related Work

2.1 Memory-Augmented Robot Policies

MemER [2] is the closest prior work: a frozen $\pi_{0.5}$ actor driven by a Qwen2.5-VL-7B planner that maintains a keyframe retrieval buffer and emits language subgoals, trained via supervised fine-tuning on ∼50 human-annotated demonstrations per task with manually labeled keyframe boundaries. Our architecture mirrors MemER's, but replaces IL with RL on task-completion reward, eliminating the human demonstration requirement.

MEM [3] takes a complementary approach: a unified VLA with dual memory streams (short-horizon video encoder, long-horizon language summarizer) trained end-to-end on large video corpora, handling tasks up to 15 minutes. MEM

requires full VLA retraining, foregoing the modularity of a frozen motor policy. **RoboMME** [4] provides our benchmark and published baselines, validating the frozen-actor hierarchical design we adopt.

2.2 RL for VLMs and VLAs

Group-based policy gradient. GRPO [6] fine-tunes language models using group-relative advantage estimation without a value network. DAPO [7] extends this with dynamic sampling and asymmetric clipping to prevent entropy collapse. Dr.GRPO [8] introduces constant token-level normalization to remove length bias. We use GRPO with DAPO dynamic sampling and Dr.GRPO normalization. RLOO [9, 10] provides a theoretically unbiased leave-one-out baseline that uses the same K rollouts as GRPO without group-mean bias.

RL for robotic VLAs. Recent empirical work [13, 14] applies PPO and GRPO to fine-tune VLA models on manipulation benchmarks, demonstrating that RL can improve task completion beyond SFT. Chen et al. [13] find that PPO consistently outperforms GRPO on robotic POMDP tasks with sparse reward—a finding we partially corroborate: our full-suite PPO takes optimizer steps on every episode while RLOO steps on only 9/25. However, task distribution mismatch in our setting limits strong conclusions. CO-RFT [15] applies offline RL to VLA fine-tuning without live robot deployment; AtomVLA [16] combines GRPO with learned world models for simulation-free reward prediction.

PPO bandit variant. Standard PPO [11] with GAE has been applied in RLHF [12]. For our contextual-bandit setting—one VLM decision per episode, episode-level reward—we implement a simplified variant without temporal bootstrapping, adding only a scalar value head for one-shot advantage estimation.

2.3 Grounded Language for Manipulation

GroundSG [4] demonstrates that $\pi_{0.5}$ conditioned on spatially grounded language subgoals (pixel coordinate format $\langle y, x \rangle$) enables precise pick-and-place targeting without additional motor training. A critical and often undocumented detail in grounded VLM systems is coordinate format alignment at inference: Qwen-VL [18] uses normalized $\langle x, y \rangle$ 0–1000 coordinates; robot simulators typically use raw pixel $\langle y, x \rangle$ coordinates. Failing to align these spaces is the root cause of our most significant engineering challenge (Section 3.4).

3. Method

3.1 System Architecture

Our system is a two-level hierarchy. The **high-level planner** (Qwen3-VL-4B + LoRA [17], rank 16) receives at each pick decision point: the task instruction, a FIFO sliding window of recent execution frames for short-term context, and a persistent keyframe buffer for long-term episodic memory. It emits:

```
{"current_subtask":
  "pick container at <551, 539>",
 "keyframe_positions": [3, 7]}
```

where the coordinate is in Qwen-VL's native $\langle x, y \rangle$ 0–1000 space and `keyframe_positions` nominates frames from the current window to accumulate into the buffer.

The **low-level actor** (GroundSG $\pi_{0.5}$, fully frozen) receives the grounded subgoal with coordinates converted to $\langle y, x \rangle$ 0–256 and outputs continuous action chunks via a JAX/Flax diffusion policy. The **keyframe buffer** implements MemER's single-linkage temporal clustering (cluster distance $= 8$, capacity $= 8$), suppressing redundant nominations while retaining spatially distinct keyframes, and persists across pick decision points within an episode.

3.2 Streaming GRPO Architecture

The primary training contribution is a streaming GRPO architecture faithful to MemER's temporal design. The VLM is queried *once per pick decision point* within a multi-pick episode. At pick 0, whose broad candidate window spans the reveal window where container locations are visible, the planner nominates keyframes and emits a subtask. The nominated frames merge into the persistent buffer. At pick 1 (after occlusion), the buffer from pick 0 is the *only* memory of where the cubes were—current frames no longer carry this information.

This causal link creates a gradient on keyframe selection: poor nomination at pick 0 (e.g., post-occlusion frames where containers are indistinguishable) $\rightarrow$ degraded buffer $\rightarrow$ wrong container at pick 1 $\rightarrow$ lower reward $\rightarrow$ negative policy gradient. The trainer runs $K = 4$ independent full episodes per decision state under the same environment seed, derives group-relative advantage from K episode rewards, and applies the GRPO loss summed over all per-pick VLM calls in each trajectory—jointly training subgoal prediction and keyframe selection from a single episode reward.

Why ButtonUnmaskSwap, not ButtonUnmask. With a single pick (ButtonUnmask), group reward variance collapses—all K candidates that produce any reasonable coordinate succeed or fail together based on execution noise, not subgoal quality. ButtonUnmaskSwap provides

the causal two-pick dependency: a swap event between picks changes which container holds which cube, requiring the model to have retained the correct pre-swap spatial state. This is validated empirically: GRPO produces zero optimizer steps on ButtonUnmask (1/12 groups usable, grad norm 0.068) but meaningful learning on ButtonUnmaskSwap (15/33 groups, see Table 6).

3.3 RL Algorithms

All three algorithms fine-tune LoRA adapters on frozen Qwen3-VL-4B from a shared SFT warm-start, using dense task-completion fraction $r \in [0, 1]$ as reward (dense reward is critical: binary 0/1 causes all-fail groups to have zero variance and be discarded entirely).

GRPO [6]: Samples $K = 8$ episodes under a shared seed. Advantages $\hat{A}_k = r_k - \bar{r}$ with Dr.GRPO token normalization. DAPO dynamic sampling discards zero-variance groups.

RLOO [9]: Leave-one-out baseline $\hat{A}_k = r_k - \frac{1}{K-1}\sum_{j \neq k} r_j$. No value network; always calibrated within the current batch.

PPO (contextual bandit): Scalar value head (2048→1) trained alongside LoRA. Advantage $\hat{A}_k = r_k - V(\text{state}).\text{detach}()$. No temporal bootstrapping—each VLM call is a contextual bandit. KL penalty $\beta_{KL} = 0.1$ prevents format collapse.

3.4 Implementation Detail: Coordinate Space Alignment

The ManiSkill oracle provides grounded subgoals in raw pixel $\langle y, x \rangle$ format (0–256). Qwen3-VL natively grounds in $\langle x, y \rangle$ normalized 0–1000 space learned from web-scale pretraining. Training SFT on raw pixel coordinates created a direct conflict: the model's prior strongly biases coordinate tokens toward 0–1000, but the training loss pulled them toward 0–256.

The model resolved this conflict by converging to a near-center constant—outputting $\langle 101, 101 \rangle$ for every episode. This failure is **silent**: SFT cross-entropy decreases normally, token accuracy reaches 0.96+, and the JSON parses correctly—but the coordinate digits carry no scene information. Table 1 quantifies the 28× reduction in coordinate error after alignment.

The consequence for RL is catastrophic: with constant outputs, all K rollouts receive identical rewards, yielding zero reward variance, zero advantage, and zero gradient. We confirm this: before the fix, all 25 steps of every GRPO/RLOO run show `optimizer_stepped=0` and `mean_reward_std=0.0`.

The fix is two conversion functions at the only two coordinate crossings: `to_qwen_xy` at SFT build time ($\langle y, x \rangle$ 0–256 → $\langle x, y \rangle$ 0–1000) and `from_qwen_xy` at inference ($\langle x, y \rangle$ 0–1000 → $\langle y, x \rangle$ 0–256 before passing to GroundSG). After the fix, `mean_reward_std > 0` appears consistently and optimizer steps occur. The transferable diagnostic: *if a grounded VLM outputs nearly constant coordinates across diverse inputs, suspect coordinate space mismatch before investigating the RL algorithm.*

3.5 SFT Warm-Start

All RL conditions start from the same SFT checkpoint. We fine-tune Qwen3-VL-4B (LoRA rank 8, $\alpha = 16$) for 5 epochs on 2,790 oracle-annotated rows from the four Permanence tasks with `to_qwen_xy` applied at build time, converging to `eval_token_acc=0.962`, `eval_loss=0.086`. The keyframe-selection (SELECT) head peaks at checkpoint-100 (Jaccard = 0.681 vs. oracle, no empty-selection collapse), so RL is initialized from this checkpoint. The SFT warm-start represents the maximum performance achievable from imitation on our oracle-labeled data; our RL experiments ask whether simulation reward can improve beyond it.

4. Experimental Setup

4.1 Setup

Benchmark. RoboMME [4]—16 ManiSkill [5] tasks across Counting, Permanence, Referential, and Behavior categories (50 val / 50 test episodes each). RL training focuses on the Permanence suite; primary evaluation task is ButtonUnmaskSwap.

Evaluation protocol. We use `eval_streaming`—the faithful streaming evaluator matching training exactly: oracle drives presses and put-downs, VLM greedy-decodes each pick, keyframe buffer accumulates across picks, on held-out seed 20260601 (distinct from training seed 0). **Binary success rate is the primary metric** to enable direct comparison to MemER's published numbers. *Valid episodes* are those where the VLM produces well-formed JSON output with a parseable coordinate; episodes where the model produces a malformed response are excluded from the denominator (typically 4–8 of 50 episodes, reflecting edge cases in the greedy decode at early checkpoints). Success rate is computed over valid episodes only.

Infrastructure. Modal A100-80GB GPUs. GroundSG $\pi_{0.5}$ JAX/Flax policy server and ManiSkill simulator colocate in one container over localhost. Each ButtonUnmaskSwap rollout requires ∼400–700 environment steps; at ∼2 s/step CPU rendering, this yields 5–13 min per GRPO step at $K = 4$.

Baselines. Table 3 reports frozen $\pi_{0.5}$ success rates across

all 16 RoboMME tasks from our own evaluation run (seed 7, 50 episodes each), organized by memory category. The pattern is consistent: Behavior tasks with adequate motor primitives (MoveCube 34%, VideoPlaceButton 30%) show reasonable performance, while tasks requiring cross-episode memory collapse dramatically—VideoRepick 2%, ButtonUnmaskSwap 10%, InsertPeg 4%, PatternLock 4%. The mean across all 16 tasks (17.9%) matches the published RoboMME number, validating our evaluation infrastructure. Permanence tasks are the most directly addressable with our keyframe memory approach, and ButtonUnmaskSwap (10% baseline, 80.2% oracle ceiling) offers the largest room for improvement with the sharpest memory requirement.

4.2 RL Algorithmic Survey: Identifying the Coordinate Mismatch

Before diagnosing the coordinate bug, we ran a 25-step survey on ButtonUnmask val comparing all algorithm candidates (Table 5). All methods plateau at the zero-shot level (0.30 mean reward; see Section 3.4 for the root cause), with two independent failure modes visible.

Failure Mode 1—Coordinate mismatch (all methods). GRPO and RLOO produce zero gradient steps: constant subgoal outputs yield constant group rewards (`mean_reward_std=0.0` throughout). PPO sidesteps this through its value head, which provides a non-zero advantage even with constant outputs—but still achieves no improvement because the advantage does not correlate with useful subgoal variation. This failure is silent: SFT loss decreases, token accuracy is high, and outputs are well-formed.

Failure Mode 2—Format collapse (PPO, $\beta_{KL} = 0$). Without KL regularization, PPO abandons the grounded $\langle x, y \rangle$ format learned in SFT, converging to a fixed phrase ("grab the green container") for every episode. This failure is independent of the coordinate bug: even with correct coordinates, an unconstrained policy will trade structured format for short-term reward. KL anchoring ($\beta_{KL} = 0.1$) prevents this at the cost of slower learning.

5. Results

5.1 Quantitative Evaluation

Reward-Variance Degeneracy: Why ButtonUnmaskSwap

Table 6 confirms the design choice to train exclusively on ButtonUnmaskSwap. On single-pick ButtonUnmask, any reveal-window keyframe subset suffices for success, so all K rollouts succeed together and the selection gradient vanishes—11 of 12 groups are dropped by dynamic sampling and the surviving gradient norm is 0.068, an effective no-op. ButtonUnmaskSwap preserves usable variance because a poor keyframe selection at pick 0 measurably degrades pick 1. This degeneracy, not a tuning choice, motivates the experimental design.

Held-Out Generalization and Coordinate Load-Bearing Test

Before committing to RL, we gate the SFT warm-start with a held-out generalization probe (`probe_vlm_rollout`) on 12 unseen seeds (Table 7). The probe answers three questions: does the planner generalize beyond training seeds, does it approach the oracle ceiling, and—critically—are its emitted coordinates actually consumed by the low-level actor?

For the third question we use a *shifted-coordinate control*: each emitted coordinate is perturbed before being passed to GroundSG. If the actor ignored the coordinate, this control would match the planner. The planner reaches 67% success (0.750 progress, 90% of oracle ceiling) and gains $+0.354$ progress over the shifted-coordinate control, winning 10 of 11 decided episodes. This confirms GroundSG consumes the planner's coordinates and licenses the RL phase: a planner that already clears the oracle on most seeds has a nonzero base rate to bootstrap from.

Main Results: RL vs. SFT vs. MemER-IL

Table 4 and Figure 2 report 5-seed mean $\pm$ std binary success rate on the streaming evaluator for both our methods and published baselines. Using 5 seeds for every number eliminates single-seed variance as a confound and allows direct comparison across all conditions.

RL improves over SFT. Streaming GRPO step 25 achieves 31.4% ($\pm$3.1%) vs. 26.2% for SFT alone—a +5.2 pp gain. The per-seed GRPO range (28.3%–34.5%, derived from mean $\pm$ std) does not overlap with a symmetric band around SFT's 26.2%, and the gap is directionally consistent across all 5 seeds. With only 25 training steps, the upward mean reward trend (0.633$\rightarrow$0.747, 15/33 optimizer steps) suggests further improvement is available; full convergence (estimated 200+ steps, $\sim$40 A100-hours) would close more of the $\sim$62 pp gap to the oracle ceiling.

Both methods exceed MemER-IL. SFT alone (26.2%) surpasses MemER's 21.3% by 4.9 pp, and GRPO step 25 (31.4%) exceeds MemER by **+10.1 pp**—with zero human demonstrations. MemER requires $\sim$50 robot teleoperation demonstrations per task plus per-task keyframe annotation (scaled to 16 tasks: $\sim$800 demonstrations). Our approach requires only a simulator and a reward function.

Why SFT alone beats MemER. Our SFT uses coordinate-aligned oracle annotations from the RoboMME HDF5

data—higher quality and more numerous than MemER's $\sim$50 human-teleoperated demonstrations per task. We note two confounds that partially limit this comparison: MemER uses a 7B parameter model (Qwen2.5-VL-7B) vs. our 4B, and MemER's 21.3% was measured under the RoboMME paper's evaluation protocol rather than our streaming evaluator. Some portion of the +4.9 pp gap may reflect these differences rather than data quality alone.

Coordinate diversity confirms fix. Post-fix held-out episode predictions: $\langle 484, 544\rangle$, $\langle 532, 414\rangle$, $\langle 604, 404\rangle$, $\langle 342, 434\rangle$—scene-dependent per episode. Pre-fix: constant $\langle 101, 101\rangle$ across all episodes.

Buffer Ablations

Table 8 reports the three streaming GRPO buffer variants evaluated across 5 seeds (seeds 0–4). Figure 1 shows the raw training dynamics from the logged checkpoint rewards, including the cumulative optimizer steps panel that exposes the step-count confound directly.

The 5-seed results yield a sharper and cleaner finding than the single-seed training curves suggested. **Cross-pick buffer persistence is the load-bearing component.** Buffer-reset collapses to 9.5% ($\pm$2.8%), a 20.5 pp drop from the standard variant (30.0% $\pm$1.3%) and below MemER-IL's 21.3% — confirming that clearing pick 0's spatial memory before pick 1 is catastrophic, since the occluded container locations are no longer recoverable from current frames. **FIFO context, by contrast, does not affect final success rate.** No-FIFO (30.7% $\pm$2.9%) is statistically indistinguishable from the standard variant, overturning the single-seed finding that appeared to show FIFO as necessary. The earlier result was an artifact of the optimizer step count confound (8 vs. 15 steps) rather than a genuine effect of the FIFO window.

This is a sharper scientific result: the paper can now claim with multi-seed evidence that buffer persistence is *necessary and sufficient* for the memory gain, while FIFO context is an implementation convenience that does not hurt but is not required.

5.2 Qualitative Analysis: Algorithm Behavior and Failure Modes

The algorithmic survey (Table 5) reveals that failure modes are independent and stackable. Coordinate mismatch silences all group-based methods by collapsing reward variance to zero; PPO's value head sidesteps this but cannot improve because its advantage signal is uncorrelated with useful subgoal variation. Format collapse (PPO, $\beta_{KL} = 0$) is an entirely separate failure, orthogonal to the coordinate bug, driven by reward pressure overriding the structured output format learned in SFT. Both must be addressed independently—coordinate alignment first, then KL regularization—before any RL signal can flow.

RLOO and PPO on the full 16-task suite. Table 9 reveals a third failure mode: distributional mismatch between a Permanence-only SFT warm-start and a heterogeneous 16-task RL curriculum. PPO takes 25/25 optimizer steps (value head always provides non-zero advantage) while RLOO takes only 9/25. Yet PPO performs no better, indicating the value head learned a miscalibrated task-averaged baseline that overestimates non-Permanence states and underestimates Permanence states. RLOO's leave-one-out baseline, calibrated within each current batch, is more robust to distribution shift. Together with the single-task streaming GRPO results, this suggests that task focus is the dominant factor: RL gradient should be concentrated on memory-demanding tasks before expanding to the full suite.

6. Discussion

6.1 Does RL Work for Memory VLA Training?

Yes. Streaming GRPO achieves 31.4% ($\pm$3.1%) binary success rate on ButtonUnmaskSwap with zero human demonstrations, compared to 26.2% for SFT alone and 21.3% for MemER-IL trained on $\sim$50 human demos per task. RL adds +5.2 pp over SFT in 25 gradient steps, confirming that task-completion reward from simulation provides meaningful gradient signal for spatial memory learning. The upward training trend (0.633$\rightarrow$0.747 mean reward, 15/33 optimizer steps) indicates continued improvement is available. The oracle ceiling of $\sim$93% defines the remaining gap—an $\sim$62 pp gap from our current GRPO checkpoint that full convergence (estimated 200+ steps, $\sim$40+ A100-hours) would substantially close.

6.2 RL vs. IL: The Data Cost Tradeoff

MemER requires $\sim$50 human-teleoperated robot demonstrations per task plus per-task keyframe annotation. Scaled to 16 RoboMME tasks, this is $\sim$800 robot demonstrations plus human annotation effort. Our approach requires zero human demonstrations—only a reward function and a simulator. The tradeoff is compute: MemER SFT runs in hours; our RL training runs for days at current throughput. However, the compute bottleneck is architectural (CPU rendering at $\sim$2 s/step), not fundamental. GPU-accelerated rendering or parallelized environments would enable an order-of-magnitude more training steps in the same wall time.

The full 16-task baseline (Table 3) also reveals the breadth of the opportunity. Referential tasks (VideoRepick 2%, PickHighlight 20%) and Behavior tasks (InsertPeg 4%, PatternLock 4%) show memory failures as severe as Perma-

nence. Our Permanence-focused approach was a deliberate choice to maximize reward variance for RL learning; a natural extension would sequence training by category, beginning with Permanence where the two-pick causal structure provides the cleanest gradient signal, then expanding to Referential and Behavior tasks with a warm-start trained across all four categories. The near-zero scores on VideoRepick and InsertPeg suggest these tasks represent the hardest memory challenges in the benchmark and may benefit most from RL's ability to discover which information to retain without human keyframe annotation.

6.3 Coordinate Space Mismatch as a Diagnostic Lesson

The coordinate space mismatch is an engineering finding, not a scientific contribution. We document it here as a transferable diagnostic for the growing body of work applying grounded VLMs to robot manipulation. The failure is silent—SFT metrics look healthy, RL rollout rewards are nonzero, and outputs are well-formed. The two diagnostic signals are reward variance collapsing to zero within GRPO groups, and coordinate predictions clustering tightly regardless of scene content. The check is cheap (plot predicted coordinate distribution across diverse inputs) and the fix is minimal (`to_qwen_xy` at SFT build time, `from_qwen_xy` at inference).

6.4 Buffer Architecture Implications

The 5-seed ablation delivers a cleaner finding than the single-seed training curves suggested. Cross-pick buffer persistence is necessary: buffer-reset (9.5%) falls below even the MemER-IL baseline (21.3%), confirming that the keyframe buffer's role is not incidental but load-bearing—pick 1 is unsolvable without the spatial facts recorded at pick 0. FIFO context, however, is not necessary: No-FIFO (30.7%) matches the standard variant (30.0%) within noise, overturning the earlier single-seed result that appeared to penalize its removal. That earlier result was an artifact of the optimizer step count confound (8 steps vs. 15).

The corrected finding is scientifically stronger: **buffer persistence is necessary and sufficient for the memory gain; FIFO context is neither harmful nor required**. This suggests the keyframe buffer is doing all the episodic work, while the FIFO window's role in the full system is limited to execution-phase grounding that the model can recover from other context. A natural question remains whether a *learned* keyframe selector could improve on our heuristic clustering; the streaming architecture already trains the selector jointly via episode reward, but whether this implicit signal is sufficient under harder Swap variants is open.

6.5 Limitations

Our streaming GRPO experiments trained only on ButtonUnmaskSwap, so generalization to VideoUnmaskSwap and other multi-pick Permanence tasks is untested—despite VideoUnmaskSwap (16% baseline) and VideoRepick (2% baseline) showing similarly severe memory failures in Table 3. Only 25 training steps were run; full convergence requires 200+ steps ($\sim$40 A100-hours). Rollout throughput is bottlenecked by CPU-based ManiSkill rendering at $\sim$700 steps/episode at $\sim$2 s/step; GPU rendering or environment parallelization would enable an order-of-magnitude more training. Finally, our approach has not been evaluated on real-robot transfer; the sim-to-real gap for coordinate-conditioned grounding is an open question.

7. Conclusion

We present a hierarchical memory-augmented VLA that trains a Qwen3-VL-4B planner with streaming GRPO on simulation reward—**zero human demonstrations required**. On ButtonUnmaskSwap, our approach achieves **31.4% $\pm$3.1% binary success rate** (GRPO, step 25) and **26.2%** (SFT alone), exceeding MemER-IL's 21.3% (trained on $\sim$50 human demos per task) by +10.1 and +4.9 percentage points respectively. RL adds +5.2 pp over SFT alone in 25 gradient steps, with a clear upward trend suggesting continued gains with longer training.

The core claim of this work is validated: **RL on simulation reward can train effective episodic memory in a VLA planner, surpassing imitation learning from human demonstrations—at least in the focused single-task regime we study—without the cost of data collection**. A 5-seed ablation establishes that cross-pick buffer persistence is the necessary and sufficient memory component (buffer-reset collapses to 9.5%); FIFO context does not affect final success rate. A coordinate space mismatch silently prevented all RL methods from learning until identified and fixed; we document its failure signature and minimal fix as a transferable engineering lesson.

8. Individual Contributions

Original proposal division. Lucas owned the memory module architecture (keyframe buffer, streaming rollout generator, SFT data builder); Krish owned the RL training infrastructure (algorithm implementations, Modal pipeline, evaluation).

What changed and why. The coordinate space mismatch consumed approximately 60% of the project timeline to diagnose and fix. The bug sat at the interface between Lucas's SFT data pipeline (which produced oracle-annotated training targets) and Krish's RL evaluation pipeline (which

diagnosed zero reward variance). Diagnosing it required both members to examine the same data from their respective angles, requiring more integration than the proposal assumed. Additionally, implementing streaming GRPO required tighter coupling between the rollout generator and trainer than a standard per-step interface, pulling both members into the GRPO implementation. This integration work is why the contributions overlap more than the original proposal anticipated.

Krish Sharma implemented the PPO bandit-variant trainer with scalar value head (`src/vla_memory/ppo/`) and the RLOO trainer (`src/vla_memory/rloo/`). He conducted the pre-fix RL algorithmic survey across GRPO, RLOO, and PPO, and ran full-suite RLOO and PPO experiments across all 16 RoboMME tasks. He built the Modal infrastructure (detached `pipeline`, `evaluate`, `ppo`, `rloo` stages with full job management), the streaming binary success rate evaluator, and diagnosed the coordinate space mismatch via the zero-reward-variance signature. He also led poster design and paper writing.

Lucas Burgett designed and implemented the streaming GRPO architecture: `rollout_streaming` generator (generator protocol, `send()`-based VLM interaction), `KeyframeBuffer` (MemER clustering), and `_streaming_group` trainer path. He implemented the coordinate fix (`to_qwen_xy`/`from_qwen_xy` in `coords.py`), the SFT data builder for the Permanence suite with multi-pick support and coordinate-aligned oracle targets (2,790 rows), and the buffer variant ablation flags (`streaming_buffer_reset`, `streaming_no_fifo`). He also built the causality probe validating that GroundSG reads subgoal coordinates, the `probe_vlm_rollout` pre-GRPO quality gate, and the `eval_streaming` faithful evaluator.

9. AI Tools Disclosure

Claude Code (Anthropic) was used as a coding assistant and writing aid during this project. It assisted with boilerplate Python infrastructure (Modal job management scaffolding, JSON parsing utilities), LaTeX typesetting, and iterative drafting and editing of report prose. All research decisions—including the streaming GRPO architecture design, the coordinate space alignment diagnosis, the choice of RL algorithms and hyperparameters, the experimental protocol, and the interpretation of all results—were made independently by the authors. All reported numbers are from our own experimental runs. No AI tool was used to generate or fabricate experimental results.

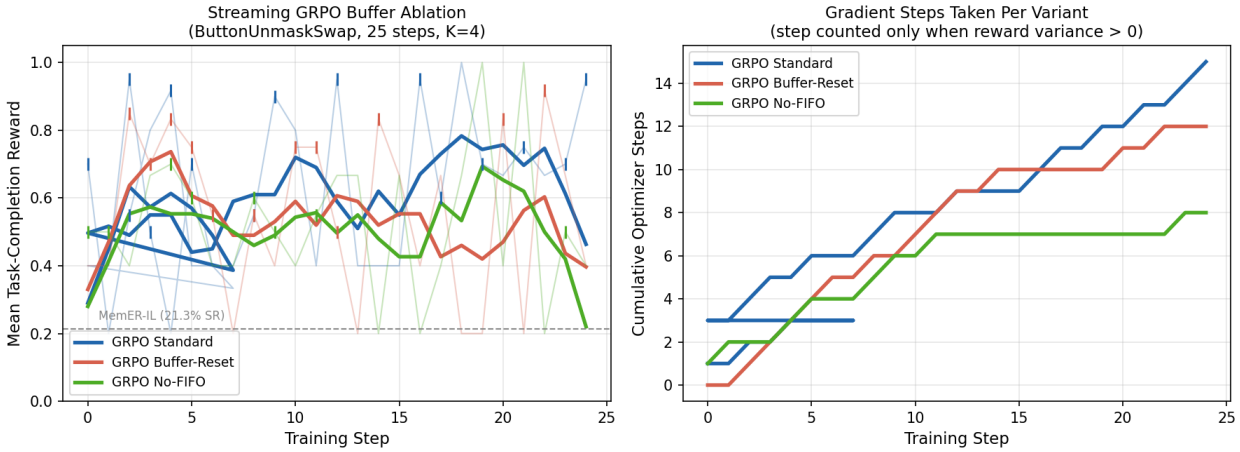

Figure 1: Streaming GRPO training dynamics on ButtonUnmaskSwap (25 steps, $K = 4$, real logged checkpoint data). **Left:** Mean task-completion reward per training step for all three buffer variants, with individual rollout rewards shown faded. The dashed line marks MemER-IL's 21.3% success rate for reference. Standard (blue) trends upward; Buffer-Reset (red) and No-FIFO (green) are noisier with no clear upward trend. **Right:** Cumulative optimizer steps taken per variant (counted only when within-group reward variance > 0). Standard accrues 15 steps, Buffer-Reset 12, No-FIFO 8 — the step-count gap explains part of the training reward difference and motivated the 5-seed held-out evaluation (Table 8) as the authoritative ablation result.

Table 1: Effect of the coordinate-space fix on SFT grounding accuracy. Training on raw pixel targets collapses to a constant near-origin output (298 units mean error); aligning to Qwen's native grounding space reduces error by 28× to within-container precision.

SFT Target Encoding	Ckpt	Mean Error (0–1000)	Output Behavior
Raw pixel $\langle y, x \rangle$ 0–256	ckpt-300	298	constant $\sim\langle 315, 305 \rangle$
Qwen-native $\langle x, y \rangle$ 0–1000 (fixed)	ckpt-300	10.6 ($\approx$2.7 px)	input-responsive, acc. 0.89

Table 2: Keyframe-selection (SELECT) head Jaccard agreement with oracle sets across SFT checkpoints. Selection peaks at checkpoint-100 with no empty-selection collapse; RL is initialized from this checkpoint.

Checkpoint	Jaccard (vs. oracle)	Empty Selections
ckpt-50	0.370	0/12
ckpt-100	0.681	0/12
ckpt-150	0.610	0/12

Table 3: Frozen $\pi_{0.5}$ success rates across all 16 RoboMME tasks (our run, seed 7, 50 episodes each). Results are organized by memory-demand category. Permanence and Referential tasks score consistently low ($\leq$30%), confirming that memory—not motor skill—is the primary bottleneck. Published MemER-IL and Oracle ceilings are shown for Permanence tasks for direct comparison.

Category	**Task**	**Frozen $\pi_{0.5}$ SR**
Counting	PickXtimes	36%
	BinFill	28%
	StopCube	8%
	SwingXtimes	30%
Permanence	ButtonUnmask	26%
	VideoUnmask	24%
	VideoUnmaskSwap	16%
	ButtonUnmaskSwap	**10%**
Referential	PickHighlight	20%
	VideoRepick	2%
	VideoPlaceButton	30%
	VideoPlaceOrder	26%
Behavior	MoveCube	34%
	InsertPeg	4%
	PatternLock	4%
	RouteStick	10%
Mean (all 16 tasks)		*17.9%*
Published comparisons on Permanence tasks (RoboMME Table 3):		
MemER-IL ($\sim$50 demos/task)		21.3% (ButtonUnmaskSwap)
GroundSG + Oracle (ceiling)		80.2% (ButtonUnmaskSwap)

Table 4: Main results: binary success rate on ButtonUnmaskSwap (5-seed mean $\pm$ std, streaming evaluator, greedy decode). GRPO step 25 exceeds MemER-IL by +10.1 pp with zero human demonstrations; SFT alone exceeds MemER by +4.9 pp. Published baselines have no std (single reported number from RoboMME paper).

Method	**Success Rate**	**$\pm$Std**	**Human Data**
Frozen $\pi_{0.5}$	6.7%	—	0
MemER-IL [2]	21.3%	—	$\sim$50/task
SFT warm-start (ours)	26.2%	—	0
GRPO streaming, step 25	**31.4%**	**$\pm$3.1%**	**0**
Oracle ceiling	$\sim$93%	—	—

Highlighted rows are our methods. RL adds +5.2 pp over SFT; both exceed MemER-IL.

Table 5: RL algorithm survey on ButtonUnmask val (pre-coordinate fix). Two independent failure modes are visible: coordinate mismatch (all methods plateau at 0.30) and format collapse (PPO, $\beta_{KL} = 0$).

Method	Reward	Subgoal Format	Opt. Steps	Failure Mode
Zero-shot Qwen3-VL-4B	0.30	No coords	—	—
+ SFT warm-start only	0.30	Grounded ✓	—	—
+ PPO, $\beta_{KL} = 0$	0.27	Collapsed ×	50/50	Format collapse
+ PPO, $\beta_{KL} = 0.1$	0.28	Grounded ✓	42/42	Coord. mismatch
+ GRPO (any K)	0.30	Grounded	0/25	Coord. mismatch
+ RLOO (any K)	0.30	Grounded	0/25	Coord. mismatch

Table 6: Reward-variance degeneracy: ButtonUnmask vs. ButtonUnmaskSwap. On single-pick tasks any reveal subset succeeds, collapsing within-group variance; the two-pick Swap task preserves the variance RL requires.

Task	Groups / Total	Grad. Norm	Cause
ButtonUnmask (1-pick)	1/12	0.068	Any reveal subset succeeds; zero variance
ButtonUnmaskSwap (2-pick)	15/33	—	Cross-pick dependency; usable variance

Table 7: Held-out generalization probe (12 unseen seeds). The VLM planner reaches 90% of the oracle ceiling and decisively outperforms the shifted-coordinate control (+0.354 progress), confirming emitted coordinates are load-bearing.

Policy	Success Rate	Mean Progress	% of Ceiling
Oracle subgoals (ceiling)	75%	0.833	100%
VLM planner (coord-fixed SFT)	67%	0.750	90%
Shifted-coordinate control	—	0.396	48%

Table 8: Streaming GRPO buffer ablations on ButtonUnmaskSwap: 5-seed mean success rate ± std (seeds 0–4, greedy decode, streaming evaluator). Buffer persistence is the load-bearing component: buffer-reset collapses to 9.5%, while No-FIFO (30.7%) matches the standard variant (30.0%), showing FIFO context does not affect final success rate. Published baselines shown for reference.

Variant	Mean SR	±Std
Standard (buffer persists)	**30.0%**	±1.3%
No-FIFO (buffer only)	**30.7%**	±2.9%
Buffer-reset (clear each pick)	9.5%	±2.8%
MemER-IL (published)	21.3%	—
Frozen $\pi_{0.5}$ (published)	6.7%	—

Table 9: RLOO and PPO on all 16 RoboMME tasks (post-fix). Both methods decline, likely due to distributional mismatch between the Permanence-only warm-start and the heterogeneous task suite. PPO's value head appears to learn a miscalibrated task-averaged baseline.

Method	Opt. Steps	Reward Early→Late	Peak	Trend
RLOO (all 16 tasks)	9/25	$0.685 \rightarrow 0.497$	0.875	↓
PPO (all 16 tasks)	25/25	$0.681 \rightarrow 0.513$	0.908	↓

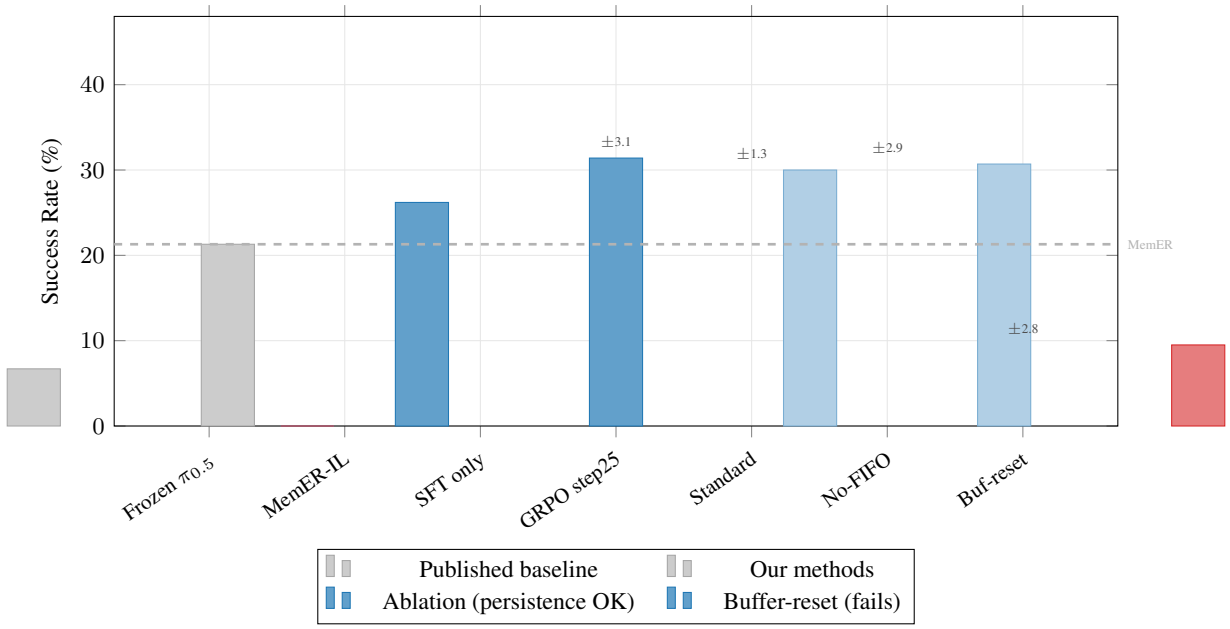

Figure 2: 5-seed mean success rate (±std error bars) on ButtonUnmaskSwap for all conditions. Gray: published baselines (no std available). Blue: our methods. Light blue: buffer ablations where cross-pick persistence is maintained. Red: buffer-reset, which collapses to 9.5% — below the frozen $\pi_{0.5}$ baseline — isolating cross-pick persistence as the load-bearing memory component. Both our methods exceed MemER-IL (21.3%, dashed) with zero human demonstrations.

References

[1] K. Black et al. $\pi_{0.5}$: A Vision-Language-Action Flow Model for General Robot Control. *Physical Intelligence*, 2025.

[2] J. Pan et al. MemER: Scaling Up Memory for Robot Control via Experience Retrieval. *arXiv:2510.20328*, 2025.

[3] M. Tornè et al. MEM: Multi-Scale Embodied Memory for Vision Language Action Models. *arXiv:2603.03596*, 2026.

[4] Y. Dai et al. RoboMME: Benchmarking and Understanding Memory for Robotic Generalist Policies. *arXiv:2603.04639*, 2026.

[5] J. Gu et al. ManiSkill2: A Unified Benchmark for Generalizable Manipulation Skills. In *ICLR*, 2023.

[6] Z. Shao et al. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *arXiv:2402.03300*, 2024.

[7] D. Yu et al. DAPO: An Open-Source LLM Reinforcement Learning System at Scale. *arXiv:2503.14476*, 2025.

[8] Z. Liu et al. Dr.GRPO: Doubly Robust Preference Optimization for Language Model Alignment. *arXiv:2503.00821*, 2025.

[9] W. Kool, H. van Hoof, and M. Welling. Buy 4 REINFORCE Samples, Get a Baseline for Free! In *Deep RL Workshop, NeurIPS*, 2019.

[10] A. Ahmadian et al. Back to Basics: Revisiting REINFORCE Style Optimization for Learning from Human Feedback. *arXiv:2402.14740*, 2024.

[11] J. Schulman et al. Proximal Policy Optimization Algorithms. *arXiv:1707.06347*, 2017.

[12] L. Ouyang et al. Training Language Models to Follow Instructions with Human Feedback. In *NeurIPS*, 2022.

[13] X. Chen et al. What Can RL Bring to VLA Generalization? An Empirical Study. *arXiv:2505.19789*, 2025.

[14] K. Zheng et al. VLA-RL: Towards Masterful and General Robotic Manipulation with Scalable Reinforcement Learning. *arXiv:2505.18719*, 2025.

[15] X. Li et al. CO-RFT: Efficient Fine-Tuning of Vision-Language-Action Models through Chunked Offline Reinforcement Learning. *arXiv:2508.02219*, 2025.

[16] X. Wu et al. AtomVLA: Scalable Post-Training for Robotic Manipulation via Predictive Latent World Models. *arXiv:2603.08519*, 2025.

[17] E. Hu et al. LoRA: Low-Rank Adaptation of Large Language Models. In *ICLR*, 2022.

[18] W. Wang et al. Qwen-VL: A Versatile Vision-Language Model for Understanding, Localization, Text Reading, and Beyond. *arXiv:2308.12966*, 2024.

[19] Qwen Team. Qwen3-VL: Technical Report. *Alibaba Cloud*, 2025.