REKI: REGISTERING KNOWLEDGE IN VLAS

Anonymous authors
Paper under double-blind review

ABSTRACT

Most vision-language-action models (VLAs) predict actions from the current observation alone, without retaining information from the past. Existing memory mechanisms either directly pass past frames into the vision–language backbone (VLM), unroll the model during training, or add dedicated memory modules, which increase training or inference costs for VLAs. To address this, we introduce REKI, a VLA with a lightweight memory mechanism that substantially improves performance with negligible overhead. A small set of *memory tokens* in the action head summarises outputs from the VLM at each step and is cached across time to inform next actions. We find that memory is particularly useful for reducing repetitive robot behaviours. Evaluations on LIBERO-Plus, RoboTwin, and real-world robot tasks demonstrate REKI's strong performance and generalisation. With only 2.2B parameters and trained only on public robot data, REKI achieves state-of-the-art results among models of comparable size and can even match or surpass larger models. We release our code, data curation pipeline, and models.

1 INTRODUCTION

Vision-language-action models (VLAs) have become an effective paradigm for language-conditioned robot control. A pretrained vision–language model (VLM) encodes camera views and instruction, based on which a smaller action expert predicts the next actions (Black et al., 2024; Physical Intelligence et al., 2025; Bjorck et al., 2025; Shukor et al., 2025; Yuan et al., 2026; Chi et al., 2025).

Most VLAs predict actions from the current observation alone (Kim et al., 2024; Black et al., 2024; Physical Intelligence et al., 2025; Bjorck et al., 2025; Zhao et al., 2023b; Lee et al., 2025; Qu et al., 2025; Liu et al., 2026), although incorporating past context can be helpful for control (Octo Model Team et al., 2024; Liu et al., 2025; Chi et al., 2025). Existing ways of incorporating such context, however, often introduce substantial overhead (Fig. 1): stacking past frames enlarges every VLM forward (Koo et al., 2026; Jang et al., 2025) and re-encodes the past at every step (Yuan et al., 2026), recurrent methods require sequential training (Cherepanov et al., 2026; Jiang et al., 2026), and others introduce additional, dedicated memory modules to the policy (Shi et al., 2026; Koo et al., 2026).

We propose REKI, which places the memory in the action expert: a small set of register tokens (Darcet et al., 2024) summarise each encoded frame inside the expert and are designed so that they can be efficiently cached across chunks (Fig. 2). Each frame is thus encoded only once and no separate module, sequential unroll during training or test-time update mechanisms are necessary. This yields clear gains with little overhead at training and inference. Our 2.2B model REKI, trained only on public data, achieves state-of-the-art results for its size (Fig. 4) across LIBERO-Plus and RoboTwin 2.0, and generalises to real-world experiments. It also outperforms or is on par with larger models.

Our ablations show that memory consistently improves performance, with the most pronounced gains for the first past frame (Fig. 5a). Beyond task success, we find that memory stabilises action generation: without memory, the policy often oscillates between poses across action chunks, whereas REKI substantially reduces such repetitive behaviour (Fig. 5b).

To summarise, our contributions are three-fold: (i) We introduce memory tokens in the action expert whose keys and values persist across frames, which gives the policy access to its recent past at marginal cost. (ii) We analyse how short-term memory reduces the failure modes observed with our memoryless policy. (iii) We release REKI, a 2.2B-parameter VLA trained on public data only, with state-of-the-art results for its size, together with the code, the data curation pipeline, and the pretrained and finetuned models.

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

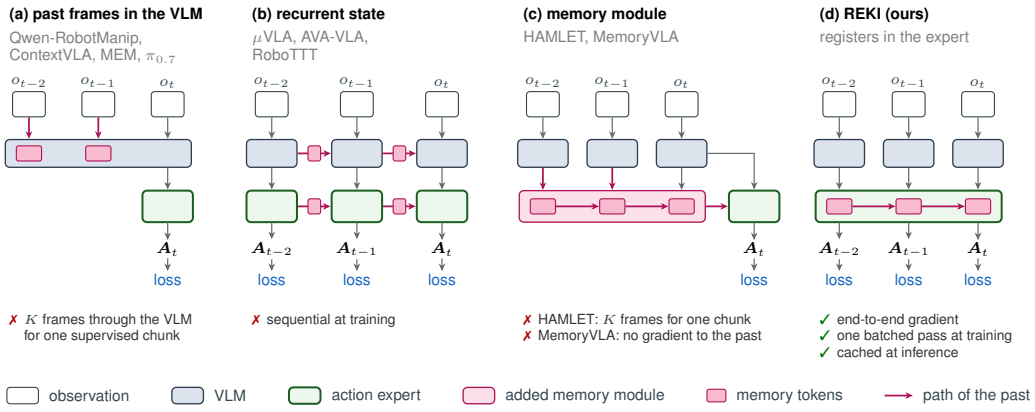


Figure 1: **How VLAs carry the past.** Each panel shows one training window of three frames, with a loss under each frame that receives one. **(a)** Past frames enter the VLM with the current one, in full (Yuan et al., 2026), compressed (Torne et al., 2026; Physical Intelligence et al., 2026) or from a cache (Jang et al., 2025). The VLM takes the whole window as input, and a loss is written for the last frame only. **(b)** A state is written at each step and read at the next, in the backbone (Cherepanov et al., 2026; Xiao et al., 2026) or as fast weights in the head (Jiang et al., 2026). Each step needs the output of the previous one, so the window is unrolled step by step. **(c)** Each frame is encoded on its own, and an added module mixes the memory tokens of past frames into the prediction for the current one. HAMLET (Koo et al., 2026) supervises the last frame only, and MemoryVLA (Shi et al., 2026) every frame, with the bank entries stored without gradient. **(d)** In REKI, the memory tokens are registers in the action expert. Their keys and values are kept across frames and read by the self-attention already present. The window is one batched pass, and every frame receives a loss.

2 Related Work

Most VLAs pair a VLM with an action expert that generates chunks of continuous actions (Black et al., 2024; Physical Intelligence et al., 2025; Bjorck et al., 2025; Shukor et al., 2025). REKI follows this design, with a flow-matching expert that cross-attends the VLM (§3.2). Many of these models release their weights and finetuning code, although the strongest ones today are pretrained partly on large-scale data, collected in-house (Wang et al., 2026; Physical Intelligence et al., 2025; 2026) or synthesised (Yuan et al., 2026), with optimised pretraining pipelines that are not released, and several have no public weights (Yuan et al., 2026; Wang et al., 2026; Physical Intelligence et al., 2026). Like the fully open SmolVLA and MolmoAct (Shukor et al., 2025; Lee et al., 2025; Fang et al., 2026), we train REKI on public data only and release the full pipeline, from data curation to pretraining.

Memory in VLAs. Most of these models act on the current observation alone, yet past context helps a policy identify the task state and recover from failed attempts (Shi et al., 2026; Torne et al., 2026; Gao et al., 2026), and methods differ in where they pay for it (Fig. 1). The most direct approach stacks or compresses past frames in the VLM (Yuan et al., 2026; Torne et al., 2026; Physical Intelligence et al., 2026; Jang et al., 2025). Its cost then grows with the window, at least in the vision encoder (Koo et al., 2026). A recurrent state (Cherepanov et al., 2026; Xiao et al., 2026) or fast weights (Jiang et al., 2026) keep the input short, but must be unrolled step by step in training. Others store past features or keyframes in an extra module (Shi et al., 2026; Koo et al., 2026; Mark et al., 2026; Sridhar et al., 2026). Like μVLA and HAMLET, REKI carries the past in a few learnable tokens, but as registers (Darcet et al., 2024) in the action expert. The VLM encodes each frame once, a training window is one batched pass, and at inference memory adds under 1% GPU time (Appendix C.1).

Consistency across action chunks. A memoryless policy can switch between modes at each new action chunk (Agarwal et al., 2026; Lian et al., 2026). Temporal ensembling (Zhao et al., 2023b) and bidirectional decoding (Liu et al., 2024b) smooth them with extra policy queries. In reinforcement learning, such cycling is attributed to perceptual aliasing, where different states look alike (Whitehead & Ballard, 1991; Singh et al., 1994; McCallum, 1996). We find that memory mitigates this switching in REKI, without any change to how the action chunks are sampled at inference (§6.2).

3 BACKGROUND

Problem formulation. At each decision step t we predict the chunk $\boldsymbol{A}_t \in \mathbb{R}^{H \times d_a}$ of the next H actions of an embodied system, conditioned on the *current context*

$$c_t = (z_t,\, o_t,\, \boldsymbol{s}_t), \tag{1}$$

which gathers a text prompt z_t, camera observations o_t and the current robot state $\boldsymbol{s}_t \in \mathbb{R}^{d_s}$. More generally, a policy may condition on every context observed so far, $c_{1:t}$.

The camera observations $o_t = \{o_t^1, \ldots, o_t^V\}$ gather the V available views, static or embodied. The robot state $\boldsymbol{s}_t$ is expressed as *absolute joint angles* or an *absolute end-effector position*, and the actions as *absolute joint angles* or *delta end-effector positions* and yield the next H states at a given control frequency. Action and state dimensions (d_a, d_s) vary across embodiments and control modes.

3.1 DIFFUSION POLICY

Following prior work (Chi et al., 2025; Black et al., 2024), the action expert generates the chunk by flow matching (Lipman et al., 2023). Write $\boldsymbol{A} = \boldsymbol{A}_t$ for the chunk generated at step t and $c = c_{1:t}$ for the contexts observed up to it; the index t is fixed in this subsection and omitted. Given noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ and a flow time $\tau \sim p(\tau)$, the interpolant $\boldsymbol{A}^\tau = (1-\tau)\,\boldsymbol{A} + \tau\,\boldsymbol{\epsilon}$ moves from the clean chunk at $\tau = 0$ to noise at $\tau = 1$, and the expert v_θ is trained to regress its constant velocity $\boldsymbol{\epsilon} - \boldsymbol{A}$,

$$\ell_{\mathrm{fm}}(\theta;\, \boldsymbol{A}, c) = \mathbb{E}_{\tau, \boldsymbol{\epsilon}}\left[\left\| v_\theta(\boldsymbol{A}^\tau, \tau \mid c) - (\boldsymbol{\epsilon} - \boldsymbol{A}) \right\|_2^2\right]. \tag{2}$$

The action loss is the expectation of this term over the dataset $\mathcal{D}$ of (context, chunk) pairs,

$$\mathcal{L}_{\mathrm{act}}(\theta) = \mathbb{E}_{(c, \boldsymbol{A}) \sim \mathcal{D}}[\ell_{\mathrm{fm}}(\theta;\, \boldsymbol{A}, c)]\,. \tag{3}$$

At inference, $\boldsymbol{A}^1 \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$ is integrated from $\tau = 1$ to 0 in ten Euler steps.

3.2 CONDITIONING

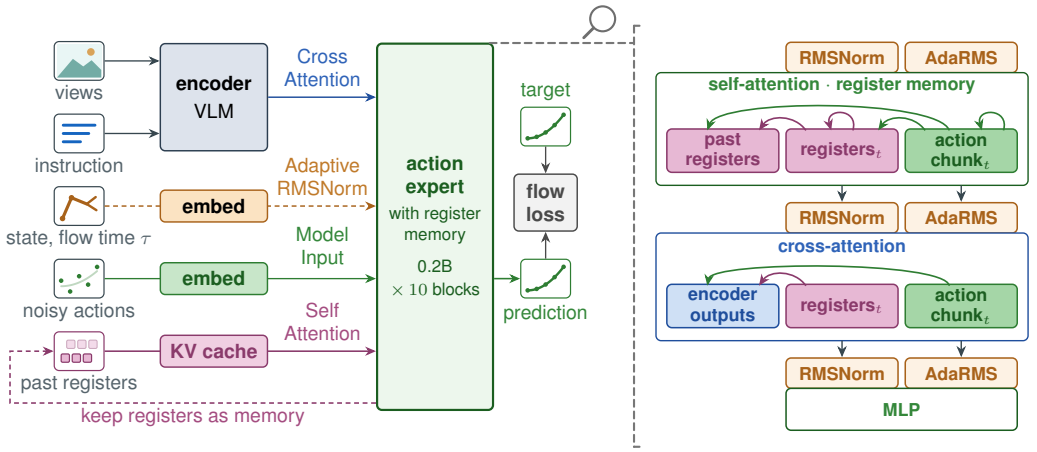
The action expert is conditioned on the camera observations, the text instruction and the robot state, through two separate paths. The observations and the instruction are variable-length token sequences. An encoder embeds them into an *encoded context*

$$\boldsymbol{E}_t = \mathrm{Enc}_\phi(z_t,\, o_t) \in \mathbb{R}^{L_t \times d}, \tag{4}$$

whose tokens the expert reads as keys and values in its cross-attention blocks. The state is a single low-dimensional vector. It is pooled into one *state embedding* $\boldsymbol{g}_t = \mathrm{Pool}_\psi(\boldsymbol{s}_t) \in \mathbb{R}^{d_e}$, which modulates the expert through its adaptive normalisation (Peebles & Xie, 2023), alongside the flow-time embedding. Since $\boldsymbol{g}_t$ does not depend on τ, it is computed once per frame and reused across the Euler steps.

Encoded context. Following recent practice (Yuan et al., 2026; Yang et al., 2026), $\boldsymbol{E}_t$ is the final-layer hidden states of a VLM, read by cross-attention. This keeps the encoder and the action expert independent, and is reported to match or outperform alternatives such as processing the encoded context jointly with the noisy chunk by self-attention (Wang et al., 2026), or attending the backbone at single or all layers (Black et al., 2024; Physical Intelligence et al., 2025; Shukor et al., 2025; Bjorck et al., 2025). We use Qwen3.5-2B (Qwen Team, 2026) and pack the prompt and the camera views into a single stream (Appendix Fig. 8). The prompt z_t carries the language instruction, the embodiment, the control mode, the control frequency, a binned episode length, and, when available, a success flag for mistake conditioning (Appendix C). Fields are dropped independently with probability 0.2 during training. Each view o_t^v is embedded by the VLM's visual encoder at native resolution, up to 448^2 pixels, and is preceded by a camera-role token (`side`, `overhead`, `wrist left`, `wrist right`). Views are shuffled and randomly dropped during training.

State embedding. Pool_ψ is one linear layer with a GELU per state layout, followed by a shared zero-initialised projection at the expert's width. Its output modulates the expert through the adaptive normalisation. The state is dropped with probability 0.2 during training.



(a) REKI architecture. (b) One expert block.

Figure 2: **REKI. (a)** Camera views and instruction are encoded as context $\boldsymbol{E}_t$, which every expert block reads by cross-attention. Robot state and flow time are embedded into $\boldsymbol{g}_t$ and modulate the actions through adaptive normalisation. The noisy chunk enters the expert's stream, and the flow loss compares the predicted velocity with its target. The register keys and values of past frames are kept in a cache and read by self-attention, which is the memory of §4.1. **(b)** One of the ten blocks, holding $n_r = 32$ registers and the action tokens of the current frame. Arrows run from a query to what it reads. Action tokens read the current and past registers and themselves, and registers never read them.

4 REKI MODEL

Building on the policy of §3, REKI gives the action expert a memory of prior frames through registers (§4.1), which are trained end to end with the flow-matching objective and cached at inference (§4.2).

4.1 PRESERVING PAST CONTEXT

Standard VLAs condition on the current context c_t alone. We give the expert access to the preceding frames through n_r memory tokens per frame, implemented as registers (Darcet et al., 2024). The registers of frame t cross-attend the encoded prompt and views of that frame, self-attend the registers of the earlier frames, and persist as keys and values in the expert's cache, implemented as a rolling window over the $K-1$ preceding frames with $K=3$ (§4.2). Since they do not attend the action tokens, they depend only on the observations and can be shared across frames and denoising steps.

Cost and consistency. The memory is added without changing the encoder, which still processes one frame at a time, so the encoder cost is that of a memoryless policy. During training, the frames are encoded in a single batched pass with no recurrence to unroll, and each frame is supervised on its own chunk (Cherepanov et al., 2026; Jiang et al., 2026), so a window of K frames yields K training signals rather than one (Fig. 1).

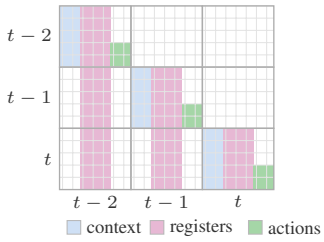
Registers. As in Qwen-RobotManip (Yuan et al., 2026), n_r learnable register tokens $\boldsymbol{M} \in \mathbb{R}^{n_r \times d_e}$ are prepended to the noisy actions in the expert's stream, so that frame t enters it as $\boldsymbol{X}_t = [\,\boldsymbol{M}\,;\,\boldsymbol{A}_t^\tau\,]$, with the actions linearly projected to the expert's width d_e; we write $\boldsymbol{M}_t$ for the register tokens of frame t. While Qwen-RobotManip discards the registers after each forward pass, we keep their keys and values across frames, allowing the action expert access to past observations.

Attention. Every token of $\boldsymbol{X}_t$ cross-attends $\boldsymbol{E}_t$. In self-attention, writing $\mathrm{Attn}(\boldsymbol{Q};\,\boldsymbol{V})$ for attention with queries $\boldsymbol{Q}$ over the tokens of $\boldsymbol{V}$ as keys and values,

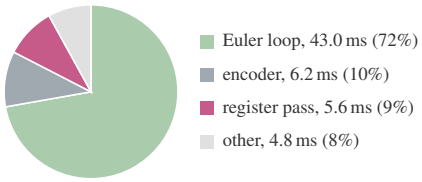
$$\boldsymbol{M}_t \leftarrow \mathrm{Attn}\big(\boldsymbol{M}_t;\,[\boldsymbol{M}_1;\ldots;\boldsymbol{M}_t]\big), \qquad \boldsymbol{A}_t^\tau \leftarrow \mathrm{Attn}\big(\boldsymbol{A}_t^\tau;\,[\boldsymbol{M}_1;\ldots;\boldsymbol{M}_t;\,\boldsymbol{A}_t^\tau]\big). \tag{5}$$

The history thus enters the velocity field through the attention support alone (Fig. 2b).

Positions. Self-attention positions are 2D, the frame index and the slot within the frame, encoded by the temporal and spatial axes of the VLM's multimodal rotary embedding (Appendix C).



(a) Attention pattern, $K=3$



(b) GPU time of one chunk, $K=1$

K	GPU time (ms)	extra over $K=1$
1	59.5	–
2	59.8	0.6 %
4	60.0	0.9 %
8	60.6	1.8 %
16	61.6	3.4 %
32	63.6	6.9 %

(c) GPU time against K

Figure 3: **Attention pattern and inference cost.** The registers and noisy action tokens of a frame cross-attend the context of that frame and self-attend the registers of the K frames up to it, so the past reaches the current chunk only through the registers, and only these are kept for later frames (a). For a memoryless policy ($K=1$), most of the GPU time of a chunk goes to the ten Euler steps (b). Reading the registers of past frames adds a negligible overhead: under 1% at $K=3$, the window used for training and evaluation (c). GPU kernel time was measured on a single H100.

4.2 Training and inference

Objective. A training sample is a window of K frames from one episode, taken at a 1 s stride and indexed from 1, the oldest, to K, the most recent. Each frame t has its own chunk $\boldsymbol{A}_t$, and its contexts $c_{1:t}$ cover the frames up to it, without past states. The loss averages the flow-matching loss of Eq. (2) over the chunks of all frames, each conditioned on the contexts up to its frame,

$$\mathcal{L}_{\text{act}}(\theta) = \mathbb{E}_{(c_{1:K},\, \boldsymbol{A}_{1:K}) \sim \mathcal{D}} \left[\frac{1}{K} \sum_{t=1}^{K} \ell_{\text{fm}}(\theta;\, \boldsymbol{A}_t,\, c_{1:t}) \right]. \tag{6}$$

Since every later frame of the window reads the registers of a frame, these registers receive gradient from all of their losses. At pretraining, the LM head of the VLM is also trained to predict the subtask when one is annotated and the answers of the vision–language data, with a cross-entropy loss of weight 0.1 (Appendix Fig. 8).

Multi-draw variance reduction. The loss of Eq. (2) is an expectation over the flow time and the noise, and a single $(\tau, \boldsymbol{\epsilon})$ draw per frame gives a noisy estimate of it. An extra draw costs one pass of the expert, whereas the context and the registers it reads cost one pass of the encoder, which dominates the cost of a training step. As in Yuan et al. (2026) and Allshire et al. (2026), we therefore estimate the expectation from eight independent draws per frame. The eight noisy chunks share the encoded context and the registers of their frame, so the encoder runs once for all of them. The registers of frame s thus receive gradient from the eight draws of every frame from s to K.

Implementation. The frames of a window are packed into a single batch and kept independent in the backbone by `cu_seqlens` boundaries. In the expert, the keys and values of Eq. (5) split into two sets: the register tokens $\boldsymbol{M}_{1:t}$, which every query reads, and the action tokens of the frame, which only its action queries read (Fig. 3a). Each set is attended by a separate variable-length FlashAttention call, and the two results are merged exactly through their log-sum-exps in `fp32`.

Inference. At inference, each frame needs one encoder pass over the current observation, one pass over the register tokens and ten Euler steps on the action tokens, and the Euler steps take most of the GPU time of a chunk (Fig. 3b). The registers read no action token and skip adaptive normalisation (§4.1), so neither they nor the context depend on the flow time τ or on the state. Their keys and values are thus computed once per frame and reused at every Euler step. The register keys and values are also kept for the frames that follow, $(K-1)\, n_r$ pairs per expert block, and reading them adds little to the GPU time of a chunk (Fig. 3c). The cache is a rolling window, so the registers of the oldest frame are dropped when those of a new frame enter. One difference remains between training and inference. At training, the first frame of a window has no past, whereas at inference the oldest cached registers were themselves computed with memory of earlier frames. Appendix C.1 details the caching.

5 EXPERIMENTAL SETUP

5.1 PRETRAINING DATA

Robot manipulation. We train on a broad mixture of public teleoperation datasets: RoboSet (Bharadhwaj et al., 2024), Galaxea (Jiang et al., 2025), AgiBot World (Bu et al., 2025), RoboCOIN (Wu et al., 2025), RoboMIND V2 (Hou et al., 2025), RDT-1B (Liu et al., 2025), DROID (Khazatsky et al., 2024), BridgeData V2 (Walke et al., 2023), RT-1 (Brohan et al., 2023), and BC-Z (Jang et al., 2022), complemented by motion-planned simulation from InternData-A1 (Tian et al., 2026) and GR00T-X-Embodiment-Sim (Bjorck et al., 2025) and bimanual YAM demonstrations from ABC (Allshire et al., 2026). These span single-arm, bimanual, mobile, and humanoid platforms across tabletop, in-the-wild, and loco-manipulation settings (Appendix Table 3). Action and state dimensions, control conventions and units vary across sources, and a robot may appear under several conventions. Following Qwen-VLA (Wang et al., 2026), we keep each dataset's native control convention rather than remapping to a shared action space, normalise each action dimension to $[-1, 1]$ by per-dataset quantiles, and zero-pad actions to a common width under a loss mask. States are absolute, actions are absolute joint positions or delta end-effector poses, and rotations are harmonised to one Euler parameterisation. The prompt and the per-layout state projections of §3.2 tell the policy which convention a sample follows. In the spirit of Qwen-RobotManip's multi-stage filtering (Yuan et al., 2026), we curate every source before training: we keep only the degrees of freedom the policy commands, discard static and near-zero-variance episodes, and correct per-dataset sign conventions. Appendix D.2 gives the curation stages and the action representation.

Egocentric human manipulation. We add three egocentric human manipulation datasets, EgoDex (Hoque et al., 2026), EgoVerse (Punamiya et al., 2026) and Xperience (Ropedia, 2026), recorded from head-mounted or hand-held cameras. They contribute $\sim$350k episodes and 170M frames, as much as our largest robot source. The action target follows Qwen-VLA's 32D bimanual layout (Wang et al., 2026), wrist motion and hand articulation per hand, here with the wrist expressed in the camera frame; no proprioceptive state is used, and each is resampled to the pace of robot teleoperation (Yuan et al., 2026). Appendix D.4 gives the sources, target definition and resampling.

Vision–language data. We train on the FineVision (Wiedmann et al., 2025) and LLaVA-OneVision-1.5 (An et al., 2025) datasets most relevant for spatial reasoning (Appendix D.5), as well as targeted spatial-grounding and pointing corpora: RoboPoint (Yuan et al., 2024), RefSpatial (Zhou et al., 2025), and PixMo (Deitke et al., 2025). This auxiliary supervision preserves the backbone's perception and spatial reasoning, and is mixed in at a small ratio with the robot and egocentric data.

5.2 MODELS

REKI pairs the Qwen3.5-2B vision–language backbone with an action expert of 10 blocks of width 768, 32 registers per frame and a memory window of $K=3$ frames. It is pretrained on the mixture above for 100k steps on 32 H100 GPUs with a global batch of 2,880 samples, then finetuned on each benchmark separately. The backbone is trained at $0.1\times$ the learning rate of the expert, and 10% of the sampled frames are vision–language data. Finetuning steps, node counts and the number of evaluation rollouts per task are in Appendix Table 5, the remaining hyperparameters in Appendix Table 4. Rows marked *scratch* skip the pretraining and finetune the initialised model directly on the benchmark. All reported numbers are success rates averaged over tasks.[1]

6 EXPERIMENTS

6.1 RESULTS

Benchmarks. LIBERO (Liu et al., 2023a) has four suites of ten single-arm tabletop tasks, and LIBERO-Plus (Fei et al., 2025) perturbs its 40 tasks along seven axes, from the camera viewpoint and the initial robot state to lighting and object layout. The models are finetuned on the four LIBERO suites and evaluated on both. RoboTwin 2.0 (Chen et al., 2025) has 50 tasks on the dual-arm AgileX

[1] Baseline numbers are taken from published tables (Appendix B). Qwen-RobotManip, in grey, is not released.

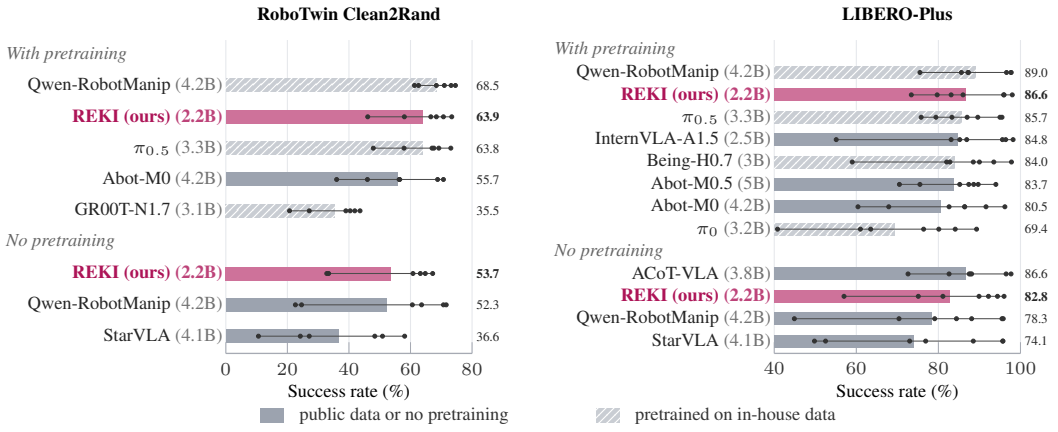


Figure 4: **Out-of-distribution success rate** on RoboTwin Clean2Rand (left) and LIBERO-Plus (right), with and without pretraining, with the total parameters (VLM and expert) of each model. The bars give the mean over the six Clean2Rand conditions and the overall LIBERO-Plus score, and the dots the individual conditions. Full results, including in distribution, are in Appendix Tables 1 and 2.

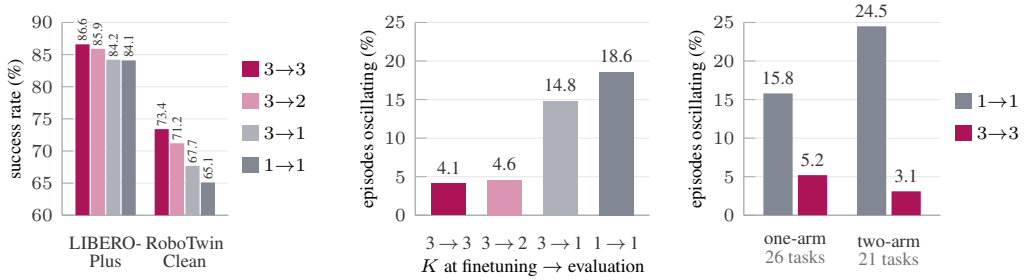


Figure 5: **Memory window and oscillation.** (a) Success rate on LIBERO-Plus and RoboTwin Clean, for REKI trained with $K=3$ and evaluated with $K=3, 2, 1$ ($3 \to 3$, $3 \to 2$, $3 \to 1$), and for a memoryless model ($1 \to 1$). (b) RoboTwin Clean episodes in which an arm reverses its motion at three or more consecutive chunks, over 50 tasks with 20 matched-seed episodes each. (c) The same share by task class, for $1 \to 1$ and $3 \to 3$. Two-arm tasks move both arms in at least 80% of the successful episodes with memory, one-arm tasks a single arm. The three other tasks are left out.

ALOHA, in clean scenes and in scenes with a randomised background, lighting, clutter and table height. For the out-of-distribution evaluation, the models are finetuned on clean scenes only and evaluated on clean scenes (Easy) and under the Clean2Rand conditions (Yuan et al., 2026), in which the randomisations are applied one at a time or all together (Hard). The out-of-distribution results are illustrated in Fig. 4, and all results are detailed in Appendix Tables 1 and 2.

Pretrained models. Among the models pretrained on public data, REKI has the highest out-of-distribution success on both benchmarks, although it is the smallest (Fig. 4). It reaches 63.9 on Clean2Rand against 55.7 for Abot-M0, and 86.6 on LIBERO-Plus against 84.8 for InternVLA-A1.5. It performs on par with $\pi_{0.5}$ (63.8 and 85.7), which is larger and pretrained on in-house data. Only Qwen-RobotManip, with 4.2B parameters and in-house pretraining data, scores higher, with 68.5 and 89.0. REKI also holds up in its hardest conditions. Its lowest score, 46.1 on Clean2Rand Hard and 73.4 under perturbed initial robot states on LIBERO-Plus, is the highest among the models pretrained on public data, whose lowest scores reach at most 36.0 and 70.5.

Without pretraining. Finetuned directly on each benchmark, REKI leads on Clean2Rand with 53.7, ahead of Qwen-RobotManip (52.3) and StarVLA (36.6), which both have 4B VLMs. On LIBERO-Plus it reaches 82.8, behind ACoT-VLA (86.6, 3.8B) and ahead of the other two.

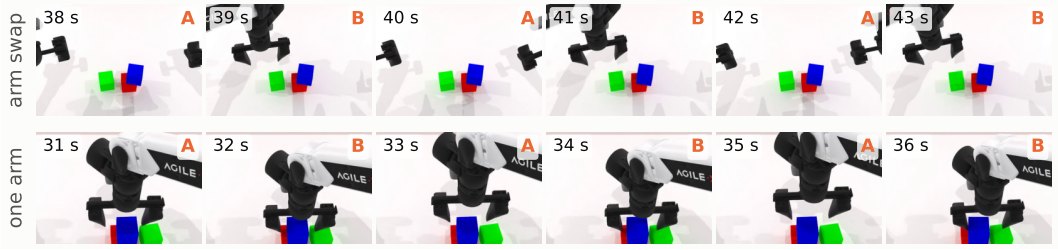


Figure 6: **Oscillation without memory.** Two RoboTwin rollouts of REKI trained without memory, stacking three blocks. Frames are one chunk apart (1 s), and A and B mark the two poses the arm alternates between. In the top row, the two arms swap at every chunk, one reaching while the other retracts. In the bottom row, a single arm moves back and forth. The blocks do not move in either row.

What pretraining adds. Pretraining on public data benefits REKI most out of distribution, and in particular under the RoboTwin randomisations (Appendix Tables 1 and 2). In distribution, it adds 5.7 and 3.6 points on RoboTwin and 1.9 on LIBERO. Out of distribution, it adds 10.2 points on Clean2Rand and 3.8 on LIBERO-Plus, and the gain grows where REKI without pretraining fails most, with 24.6 points under clutter and 13.3 under the Hard condition against 8.6 on Easy. On LIBERO-Plus, the gain is likewise largest under perturbed initial robot states (16.4), although it reverses under changed camera viewpoints, where the success drops from 89.9 to 79.7. Qwen-RobotManip, pretrained on in-house data, shows the same pattern with larger gains, 16.2 points on Clean2Rand and 10.7 on LIBERO-Plus, while without pretraining REKI is ahead of it on both benchmarks.

In distribution. On LIBERO, REKI reaches 97.9. Most models lie between 97.6 and 99.4, so the benchmark leaves little room to separate them. On RoboTwin, where the models are finetuned and evaluated on both clean and randomised scenes, REKI reaches 84.9 and 84.2. This is close to the 4B models pretrained on public data or trained without pretraining (85.1 to 88.7) and ahead of $\pi_{0.5}$ (82.7 and 76.8), but below Qwen-RobotManip (93.4 and 92.5), which relies on additional synthetic data.

6.2 Ablation

Memory window. Figure 5a varies the memory window of REKI at finetuning and at evaluation, on LIBERO-Plus after finetuning on LIBERO and on RoboTwin Clean. Finetuned and evaluated with three frames ($3 \rightarrow 3$), REKI reaches 86.6 on LIBERO-Plus and 73.4 on RoboTwin, against 84.1 and 65.1 without memory ($1 \rightarrow 1$). On RoboTwin, both models run on the same seeds, and the gain is spread over the tasks. Of the 231 episodes that only one of the two models solves, the model with memory solves 157, and it is ahead on 27 of the 50 tasks and behind on 8. Most of the gain already comes with the first past frame ($3 \rightarrow 2$). As every frame of a training window is supervised (Eq. (6)), the model finetuned with memory also learns to act without it. Evaluated without memory ($3 \rightarrow 1$), it performs close to the model finetuned at $K=1$ (84.2 and 67.7), so the gain comes mainly from reading the past frames at inference, not from training on windows.

Oscillation. Figure 5b counts the episodes in which an arm moves back and forth between two poses at every new chunk (Fig. 6). Fewer than 5% of these episodes succeed. With the same weights, removing the past frames at evaluation raises their share from 4.1% to 14.8%, and one past frame already brings it down to 4.6%. Trained without memory, REKI oscillates in 18.6% of the episodes, and in half of those that fail. The difference is largest in two-arm tasks (Fig. 5c).

Why memory helps. A policy without memory maps the current observation to the next chunk. If two poses A and B each look like demonstration states that continue towards the other, the chunk predicted at A leads to B, and the chunk predicted at B leads back to A. We attribute the oscillation to this form of perceptual aliasing, under which memoryless policies are known to fail (Whitehead & Ballard, 1991; Singh et al., 1994; McCallum, 1996). A short history resolves the ambiguity, since a window that has just come from B does not match demonstrations that move towards B. In two-arm tasks, the most frequent cycle is an arm swap right after a release (65 against 1 of 1000 episodes, top row of Fig. 6). Memory does not rescue all these episodes: where the policy without memory oscillates, the policy with memory succeeds on 45% of the seeds.

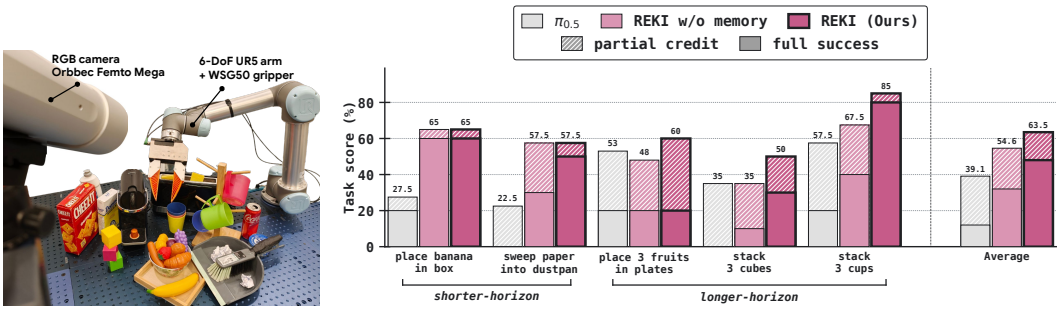


Figure 7: **Real-robot experiments.** Left: the UR5 arm with a WSG50 gripper, observed by a fixed Orbbec Femto Mega RGB camera, with the task objects and distractors. Right: task score of REKI with and without memory and of $\pi_{0.5}$ (Physical Intelligence et al., 2025) on two shorter- and three longer-horizon tasks, over 10 episodes each. Solid bars give full successes, hatched bars partial credit.

7 Real robot experiments

Experimental Setup. We use a 6-DoF UR5 robotic arm with a Weiss WSG50 gripper and UMI-style parallel jaws. The action loop runs at 15 Hz. The robot operates over a 1 m × 1 m tabletop workspace observed by a fixed Orbbec Femto Mega RGB camera, see Fig. 7, left. Observations include an RGB image, and a 7D absolute joint state. Actions are specified as 7D absolute joint state.

Tasks and Protocol. We evaluate performance across 5 manipulation tasks, and report task success rates and partial subtask success rates. Tasks span diverse action primitives, including pick, place, stack, and sweep. The tasks and partial success scores are defined in Appendix G. We collect 50 training episodes per task with a Meta Quest controller, and evaluate on 10 test episodes per task, with randomised initial object placements and distractors in each episode.

Baselines. We finetune the pretrained checkpoints of REKI and $\pi_{0.5}$ on the 5 tasks for 3,000 optimiser steps, with a global batch of 360 samples. Both models use a learning rate of 5×10^{-5}, held constant after warmup. The action chunk has 50 steps (3.3 s) for $\pi_{0.5}$, fixed by its pretraining, and 30 steps (2 s) for REKI. In both models, only the last four vision encoder blocks are trained. The backbone is trained at $0.1\times$ the learning rate of the action expert. REKI is finetuned with a memory window of $K=3$ and without memory ($K=1$). $\pi_{0.5}$ has no memory.

Results. Across the 5 tasks, REKI outperforms $\pi_{0.5}$ by 7 to 37.5 percentage points (Fig. 7, right). Removing the memory leaves performance on the shorter-horizon tasks mainly unchanged (65 vs. 65 and 57.5 vs. 57.5), but consistently degrades it on the longer-horizon tasks, where REKI outperforms its memoryless variant by 12 to 17.5 percentage points, supporting that memory matters most when tasks require tracking progress over many steps. Qualitative examples are shown in Fig. 9.

8 Discussion

We gave the action expert of a VLA a short memory: a few register tokens summarise the encoded observations of each frame and persist in the expert's cache, so that the VLM still encodes each frame only once. The memory is trained in a single batched pass over the window and adds less than 1% to the GPU time of an action chunk. It improves success both in simulation and in real-world experiments, where the gain is largest on the longer-horizon tasks. We observe that most of the gain comes from the first past frame, which provides the action expert with the direction of motion of the robot. Our analysis links this gain to a drop in oscillations between two poses, which we attribute to perceptual aliasing (§6.2). Out of distribution, the resulting 2.2B model is ahead of every other model pretrained on public data (Fig. 4), and pretraining mainly improves its robustness. Pretraining also explains much of the remaining gap to models trained on in-house data: REKI and Qwen-RobotManip reach similar results without pretraining, and Qwen-RobotManip only takes the lead once both models are pretrained. Much of the performance of current VLAs thus comes from pretraining, which cannot be reproduced when neither the data nor the pretraining pipeline is public. We therefore release the full pipeline, from data curation to pretraining, together with the code and the pretrained and finetuned models. We discuss the limitations of this work in Appendix A.

AI USE STATEMENT

We used a generative AI tool (Claude, through the Claude Code agent) as a coding assistant to implement the method and to write the training, evaluation and benchmarking code. We also used it to draft and edit parts of the paper text, to write the TikZ source of the figures, and to find, summarise and check related work. We have not used generative AI to propose the method, to formulate hypotheses, to design the experiments or to interpret the results. No synthetic datasets, theoretical results or proofs are part of this work.

We reviewed all AI-assisted work. The authors read and tested the code written with the tool before it was used in an experiment. Every number, architectural detail and experimental setting in the paper was checked against the code and the training logs. Each cited work was checked against the primary paper. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

REKI is trained on public data only. Appendix D lists the robot embodiments and datasets, and Appendices D.2, D.4 and D.5 describe how the robot, egocentric and vision–language data are curated. Appendix C gives the model details and Appendix E the training configuration. Table 4 lists the hyperparameters, and Table 5 the finetuning steps and the number of evaluation rollouts for each benchmark. Appendix B gives the full benchmark results, Appendix C.1 how the inference cost is measured, and Appendix F the training budgets of the compared methods. We will release the code, the data curation pipeline and the pretrained and finetuned models. We are not licensed to redistribute the pretraining data, so the curation pipeline rebuilds it from its public sources.

REFERENCES

Manoj Acharya, Kushal Kafle, and Christopher Kanan. TallyQA: Answering complex counting questions. In *Proc. AAAI Conf. Artif. Intell.*, 2019.

Abhinav Agarwal, Adam Wei, Taylan Kargin, Michael Zeng, Cole Becker, Arif Kerem Dayı, Pablo Parrilo, Asuman Ozdaglar, and Russ Tedrake. Training and evaluating diffusion policies with long context lengths. *arXiv:2606.16447*, 2026.

Arthur Allshire, Himanshu Gaurav Singh, Ritvik Singh, Adam Rashid, Hongsuk Choi, David McAllister, Justin Yu, Yiyuan Chen, Huang Huang, Pieter Abbeel, et al. Scalable behavior cloning with open data, training, and evaluation. *arXiv:2606.27375*, 2026.

Xiang An, Yin Xie, Kaicheng Yang, Wenkang Zhang, Xiuwei Zhao, Zheng Cheng, Yirui Wang, Songcen Xu, Changrui Chen, Chunsheng Wu, et al. LLaVA-OneVision-1.5: Fully open framework for democratized multimodal training. *arXiv:2509.23661*, 2025.

Homanga Bharadhwaj, Jay Vakil, Mohit Sharma, Abhinav Gupta, Shubham Tulsiani, and Vikash Kumar. RoboAgent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking. In *Proc. IEEE Int. Conf. Robot. Autom.*, 2024.

Ali Furkan Biten, Ruben Tito, Andres Mafla, Lluis Gomez, Marçal Rusinol, Ernest Valveny, C. V. Jawahar, and Dimosthenis Karatzas. Scene text visual question answering. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019.

Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, et al. GR00T N1: An open foundation model for generalist humanoid robots. *arXiv:2503.14734*, 2025.

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

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. RT-1: Robotics transformer for real-world control at scale. In *Robotics: Science and Systems*, 2023.

Qingwen Bu, Jisong Cai, Li Chen, Xiuqi Cui, Yan Ding, Siyuan Feng, Xindong He, Xu Huang, et al. AgiBot World Colosseo: A large-scale manipulation platform for scalable and intelligent embodied systems. In *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2025.

Jimmy Carter. TextOCR-GPT4V. Hugging Face dataset, 2024. URL https://huggingface.co/datasets/jimmycarter/textocr-gpt4v.

Guiming Hardy Chen, Shunian Chen, Ruifei Zhang, Junying Chen, Xiangbo Wu, Zhiyi Zhang, Zhihong Chen, Jianquan Li, Xiang Wan, and Benyou Wang. ALLaVA: Harnessing GPT4V-synthesized data for lite vision-language models. *arXiv:2402.11684*, 2024a.

Lin Chen, Jinsong Li, Xiaoyi Dong, Pan Zhang, Conghui He, Jiaqi Wang, Feng Zhao, and Dahua Lin. ShareGPT4V: Improving large multi-modal models with better captions. In *Proc. Eur. Conf. Comput. Vis.*, 2024b.

Ronghan Chen, Yandan Yang, Zuojin Tang, Dongjie Huo, Tong Lin, Haoning Wu, Haoyun Liu, Yuzhi Chen, Lulu Zheng, Botai Yuan, et al. ABot-M0.5: Unified mobility-and-manipulation world action model. *arXiv:2607.00678*, 2026.

Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. RoboTwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv:2506.18088*, 2025.

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

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *Int. J. Robot. Res.*, 44(10-11):1684–1704, 2025.

Chee Kheng Chng, Yuliang Liu, Yipeng Sun, Chun Chet Ng, Canjie Luo, Zihan Ni, ChuanMing Fang, Shuaitao Zhang, Junyu Han, and Errui Ding. ICDAR2019 robust reading challenge on arbitrary-shaped text (RRC-ArT). In *Proc. Int. Conf. Doc. Anal. Recognit.*, 2019.

Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *Int. Conf. Learn. Represent.*, 2024.

Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, et al. Molmo and PixMo: Open weights and open data for state-of-the-art vision-language models. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2025.

Haoquan Fang, Jiafei Duan, Donovan Clay, Sam Wang, Shuo Liu, Weikai Huang, Xiang Fan, Wei-Chuan Tsai, Shirui Chen, Yi Ru Wang, et al. MolmoAct2: Action reasoning models for real-world deployment. *arXiv:2605.02881*, 2026.

Senyu Fei, Siyin Wang, Junhao Shi, Zihao Dai, Jikun Cai, Pengfang Qian, Li Ji, Xinzhe He, Shiduo Zhang, Zhaoye Fei, et al. LIBERO-Plus: In-depth robustness analysis of vision-language-action models. *arXiv:2510.13626*, 2025.

Yihuai Gao, Jeff Jinyun Liu, Shuang Li, and Shuran Song. Gated memory policy: In-context memorization and adaptation. *arXiv:2604.18933*, 2026.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2017.

Jack Hessel, Jena D. Hwang, Jae Sung Park, Rowan Zellers, Chandra Bhagavatula, Anna Rohrbach, Kate Saenko, and Yejin Choi. The abduction of Sherlock Holmes: A dataset for visual abductive reasoning. In *Proc. Eur. Conf. Comput. Vis.*, 2022.

Ryan Hoque, Peide Huang, David J. Yoon, Mouli Sivapurapu, and Jian Zhang. EgoDex: Learning dexterous manipulation from large-scale egocentric video. In *Int. Conf. Learn. Represent.*, 2026.

Chengkai Hou, Kun Wu, Jiaming Liu, Zhengping Che, Di Wu, Fei Liao, Guangrun Li, Jingyang He, Qiuxuan Feng, Zhao Jin, et al. RoboMIND 2.0: A multimodal, bimanual mobile manipulation dataset for generalizable embodied intelligence. *arXiv:2512.24653*, 2025.

Drew A. Hudson and Christopher D. Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019.

Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. BC-Z: Zero-shot task generalization with robotic imitation learning. In *Proc. Conf. Robot Learn.*, pp. 991–1002, 2022.

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

Harsh Jhamtani and Taylor Berg-Kirkpatrick. Learning to describe differences between pairs of similar images. In *Proc. Conf. Empir. Methods Nat. Lang. Process.*, 2018.

Tao Jiang, Tianyuan Yuan, Yicheng Liu, Chenhao Lu, Jianning Cui, Xiao Liu, Shuiqi Cheng, Jiyang Gao, Huazhe Xu, and Hang Zhao. Galaxea open-world dataset and G0 dual-system VLA model. *arXiv:2509.00576*, 2025.

Yunfan Jiang, Yevgen Chebotar, Ruijie Zheng, Fengyuan Hu, Yunhao Ge, Jimmy Wu, Tianyuan Dai, Scott Reed, Li Fei-Fei, Yuke Zhu, et al. RoboTTT: Context scaling for robot policies. *arXiv:2607.15275*, 2026.

Hazal Karakus. MSCOCO-ControlNet-Canny-Less-Colors. Hugging Face dataset, 2024. URL https://huggingface.co/datasets/hazal-karakus/mscoco-controlnet-canny-less-colors.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. DROID: A large-scale in-the-wild robot manipulation dataset. In *Robotics: Science and Systems*, 2024.

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

Myungkyu Koo, Daewon Choi, Taeyoung Kim, Kyungmin Lee, Changyeon Kim, Younggyo Seo, and Jinwoo Shin. HAMLET: Switch your vision-language-action model into a history-aware policy. In *Int. Conf. Learn. Represent.*, 2026.

LAION. GPT4V-Dataset. Hugging Face dataset, 2023. URL https://huggingface.co/datasets/laion/gpt4v-dataset.

Hugo Laurençon, Léo Tronchon, Matthieu Cord, and Victor Sanh. What matters when building vision-language models? In *Adv. Neural Inf. Process. Syst.*, 2024.

Jason Lee, Jiafei Duan, Haoquan Fang, Yuquan Deng, Shuo Liu, Boyang Li, Bohan Fang, Jieyu Zhang, Yi Ru Wang, Sangho Lee, et al. MolmoAct: Action reasoning models that can reason in space. *arXiv:2508.07917*, 2025.

Paul Lerner, Olivier Ferret, Camille Guinaudeau, Hervé Le Borgne, Romaric Besançon, José G. Moreno, and Jesús Lovón Melgarejo. ViQuAE, a dataset for knowledge-based visual question answering about named entities. In *Proc. Int. ACM SIGIR Conf. Res. Dev. Inf. Retr.*, 2022.

Xiaotong Li, Fan Zhang, Haiwen Diao, Yueze Wang, Xinlong Wang, and Lingyu Duan. DenseFusion-1M: Merging vision experts for comprehensive multimodal perception. In *Adv. Neural Inf. Process. Syst.*, 2024.

Zhuowan Li, Xingrui Wang, Elias Stengel-Eskin, Adam Kortylewski, Wufei Ma, Benjamin Van Durme, and Alan L. Yuille. Super-CLEVR: A virtual benchmark to diagnose domain robustness in visual reasoning. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023.

Shijie Lian, Bin Yu, Xiaopeng Lin, Zhaolong Shen, Laurence Tianruo Yang, Yurun Jin, Haishan Liu, Changti Wu, Hang Yuan, Cong Huang, and Kai Chen. IntentVLA: Short-horizon intent modeling for aliased robot manipulation. *arXiv:2605.14712*, 2026.

Adam Dahlgren Lindström and Savitha Sam Abraham. CLEVR-Math: A dataset for compositional language, visual and mathematical reasoning. *arXiv:2208.05358*, 2022.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. In *Int. Conf. Learn. Represent.*, 2023.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Adv. Neural Inf. Process. Syst.*, 2023a.

Fangyu Liu, Guy Emerson, and Nigel Collier. Visual spatial reasoning. *Trans. Assoc. Comput. Linguist.*, 11:635–651, 2023b.

Fuxiao Liu, Kevin Lin, Linjie Li, Jianfeng Wang, Yaser Yacoob, and Lijuan Wang. Mitigating hallucination in large multi-modal models via robust instruction tuning. In *Int. Conf. Learn. Represent.*, 2024a.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Adv. Neural Inf. Process. Syst.*, 2023c.

Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. RDT-1B: A diffusion foundation model for bimanual manipulation. In *Int. Conf. Learn. Represent.*, 2025.

Songming Liu, Bangguo Li, Kai Ma, Lingxuan Wu, Hengkai Tan, Xiao Ouyang, Hang Su, and Jun Zhu. RDT2: Exploring the scaling limit of UMI data towards zero-shot cross-embodiment generalization. *arXiv:2602.03310*, 2026.

Yuejiang Liu, Jubayer Ibn Hamid, Annie Xie, Yoonho Lee, Maximilian Du, and Chelsea Finn. Bidirectional decoding: Improving action chunking via guided test-time sampling. *arXiv:2408.17355*, 2024b.

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *Adv. Neural Inf. Process. Syst.*, 2022.

Yujie Lu, Dongfu Jiang, Wenhu Chen, William Yang Wang, Yejin Choi, and Bill Yuchen Lin. WildVision: Evaluating vision-language models in the wild with human preferences. In *Adv. Neural Inf. Process. Syst.*, 2024.

Hao Luo, Wanpeng Zhang, Yicheng Feng, Sipeng Zheng, Haiweng Xu, Chaoyi Xu, Ziheng Xi, Yuhui Fu, and Zongqing Lu. Being-H0.7: A latent world-action model from egocentric videos. *arXiv:2605.00078*, 2026.

Run Luo, Haonan Zhang, Longze Chen, Ting-En Lin, Xiong Liu, Yuchuan Wu, Min Yang, Minzheng Wang, Pengpeng Zeng, Lianli Gao, et al. MMEvol: Empowering multimodal large language models with Evol-Instruct. *arXiv:2409.05840*, 2024.

Haoxiang Ma, Junhao Cai, Xiaoxu Xu, Hao Li, Yuyin Yang, Yang Tian, Jiafei Cao, Hongrui Zhu, Zherui Qiu, Zhaxizhuoma, et al. InternVLA-A1.5: Unifying understanding, latent foresight, and action for compositional generalization. *arXiv:2607.04988*, 2026.

Max Sobol Mark, Jacky Liang, Maria Attarian, Chuyuan Fu, Debidatta Dwibedi, Dhruv Shah, and Aviral Kumar. BPP: Long-context robot imitation learning by focusing on key history frames. *arXiv:2602.15010*, 2026.

Andrew Kachites McCallum. *Reinforcement Learning with Selective Perception and Hidden State*. PhD thesis, University of Rochester, 1996.

Anand Mishra, Shashank Shekhar, Ajeet Kumar Singh, and Anirban Chakraborty. OCR-VQA: Visual question answering by reading text in images. In *Proc. Int. Conf. Doc. Anal. Recognit.*, 2019.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, et al. Octo: An open-source generalist robot policy. In *Robotics: Science and Systems*, 2024.

OpenGVLab. ShareGPT-4o, 2024. URL https://sharegpt4o.github.io/.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023.

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

Physical Intelligence, Bo Ai, Ali Amin, Raichelle Aniceto, Ashwin Balakrishna, Greg Balke, Kevin Black, George Bokinsky, Shihao Cao, Thomas Charbonnier, et al. $\pi_{0.7}$: A steerable generalist robotic foundation model with emergent capabilities. *arXiv:2604.15483*, 2026.

Renjie Pi, Jianshu Zhang, Jipeng Zhang, Rui Pan, Zhekai Chen, and Tong Zhang. Image textualization: An automatic framework for creating accurate and detailed image descriptions. *arXiv:2406.07502*, 2024.

Jordi Pont-Tuset, Jasper Uijlings, Soravit Changpinyo, Radu Soricut, and Vittorio Ferrari. Connecting vision and language with localized narratives. In *Proc. Eur. Conf. Comput. Vis.*, 2020.

Ryan Punamiya, Simar Kareer, Zeyi Liu, Josh Citron, Ri-Zhao Qiu, Xiongyi Cai, Alexey Gavryushin, Jiaqi Chen, Davide Liconti, Lawrence Y. Zhu, et al. EgoVerse: An egocentric human dataset for robot learning from around the world. *arXiv:2604.07607*, 2026.

Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, Jiayuan Gu, Bin Zhao, Dong Wang, and Xuelong Li. SpatialVLA: Exploring spatial representations for visual-language-action model. In *Robotics: Science and Systems*, 2025.

Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2009.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

Mengye Ren, Ryan Kiros, and Richard Zemel. Exploring models and data for image question answering. In *Adv. Neural Inf. Process. Syst.*, 2015.

Ropedia. Xperience-10M: A large-scale egocentric multimodal dataset of human experience. Hugging Face dataset, 2026. URL https://huggingface.co/datasets/ropedia-ai/xperience-10m.

Dustin Schwenk, Apoorv Khandelwal, Christopher Clark, Kenneth Marino, and Roozbeh Mottaghi. A-OKVQA: A benchmark for visual question answering using world knowledge. In *Proc. Eur. Conf. Comput. Vis.*, 2022.

Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019.

Hao Shi, Bin Xie, Yingfei Liu, Lin Sun, Fengrong Liu, Tiancai Wang, Erjin Zhou, Haoqiang Fan, Xiangyu Zhang, and Gao Huang. MemoryVLA: Perceptual-cognitive memory in vision-language-action models for robotic manipulation. In *Int. Conf. Learn. Represent.*, 2026.

Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. ALFWorld: Aligning text and embodied environments for interactive learning. In *Int. Conf. Learn. Represent.*, 2021.

Mustafa Shukor, Dana Aubakirova, Francesco Capuano, Pepijn Kooijmans, Steven Palma, Adil Zouitine, Michel Aractingi, Caroline Pascal, Martino Russi, Andres Marafioti, et al. SmolVLA: A vision-language-action model for affordable and efficient robotics. *arXiv:2506.01844*, 2025.

Oleksii Sidorov, Ronghang Hu, Marcus Rohrbach, and Amanpreet Singh. TextCaps: A dataset for image captioning with reading comprehension. In *Proc. Eur. Conf. Comput. Vis.*, 2020.

Chonghao Sima, Katrin Renz, Kashyap Chitta, Li Chen, Hanxue Zhang, Chengen Xie, Jens Beißwenger, Ping Luo, Andreas Geiger, and Hongyang Li. DriveLM: Driving with graph visual question answering. In *Proc. Eur. Conf. Comput. Vis.*, 2024.

Oriane Siméoni, Huy V. Vo, Maximilian Seitzer, Federico Baldassarre, Maxime Oquab, Cijo Jose, Vasil Khalidov, Marc Szafraniec, Seungeun Yi, Michaël Ramamonjisoa, et al. DINOv3. *arXiv:2508.10104*, 2025.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards VQA models that can read. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019.

Satinder P. Singh, Tommi Jaakkola, and Michael I. Jordan. Learning without state-estimation in partially observable Markovian decision processes. In *Proc. Int. Conf. Mach. Learn.*, pp. 284–292, 1994.

Ajay Sridhar, Jennifer Pan, Satvik Sharma, and Chelsea Finn. MemER: Scaling up memory for robot control via experience retrieval. In *Int. Conf. Learn. Represent.*, 2026.

StarVLA Community. StarVLA: A lego-like codebase for vision-language-action model developing. *arXiv:2604.05014*, 2026.

Alane Suhr, Mike Lewis, James Yeh, and Yoav Artzi. A corpus of natural language for visual reasoning. In *Proc. Annu. Meet. Assoc. Comput. Linguist.*, 2017.

Yang Tian, Yuyin Yang, Yiman Xie, Zetao Cai, Xu Shi, Ning Gao, Hangxu Liu, Xuekun Jiang, Zherui Qiu, Feng Yuan, et al. InternData-A1: Pioneering high-fidelity synthetic data for pre-training generalist policy. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2026.

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, et al. Cambrian-1: A fully open, vision-centric exploration of multimodal LLMs. In *Adv. Neural Inf. Process. Syst.*, 2024.

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

Haoqin Tu, Chenhang Cui, Zijun Wang, Yiyang Zhou, Bingchen Zhao, Junlin Han, Wangchunshu Zhou, Huaxiu Yao, and Cihang Xie. How many unicorns are in this image? A safety evaluation benchmark for vision LLMs. *arXiv:2311.16101*, 2023.

Andreas Veit, Tomas Matera, Lukas Neumann, Jiri Matas, and Serge Belongie. COCO-Text: Dataset and benchmark for text detection and recognition in natural images. *arXiv:1601.07140*, 2016.

Homer Rich Walke, Kevin Black, Tony Z. Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. BridgeData V2: A dataset for robot learning at scale. In *Proc. Conf. Robot Learn.*, pp. 1723–1736, 2023.

Junke Wang, Lingchen Meng, Zejia Weng, Bo He, Zuxuan Wu, and Yu-Gang Jiang. To see is to believe: Prompting GPT-4V for better visual instruction tuning. *arXiv:2311.07574*, 2023.

Qiuyue Wang, Mingsheng Li, Jian Guan, Jinhui Ye, Sicheng Xie, Yitao Liu, Junhao Chen, Zhixuan Liang, Jie Zhang, Xintong Hu, et al. Qwen-VLA: Unifying vision-language-action modeling across tasks, environments, and robot embodiments. *arXiv:2605.30280*, 2026.

Xinyu Wang, Yuliang Liu, Chunhua Shen, Chun Chet Ng, Canjie Luo, Lianwen Jin, Chee Seng Chan, Anton van den Hengel, and Liangwei Wang. On the general value of evidence, and bilingual scene-text visual question answering. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020.

Tobias Weyand, Andre Araujo, Bingyi Cao, and Jack Sim. Google Landmarks Dataset v2: A large-scale benchmark for instance-level recognition and retrieval. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020.

Steven D. Whitehead and Dana H. Ballard. Learning to perceive and act by trial and error. *Machine Learning*, 7(1):45–83, 1991.

Luis Wiedmann, Orr Zohar, Amir Mahla, Xiaohan Wang, Rui Li, Thibaud Frere, Leandro von Werra, Aritra Roy Gosthipaty, and Andrés Marafioti. FineVision: Open data is all you need. arXiv:2510.17269, 2025.

Shihan Wu, Xuecheng Liu, Shaoxuan Xie, Pengwei Wang, Xinghang Li, Bowen Yang, Zhe Li, Kai Zhu, Hongyu Wu, Yiheng Liu, et al. RoboCOIN: An open-sourced bimanual robotic data COllection for INtegrated manipulation. *arXiv:2511.17441*, 2025.

Lei Xiao, Jifeng Li, Juntao Gao, Feiyang Ye, Yan Jin, Jingjing Qian, Jing Zhang, Yong Wu, and Xiaoyuan Yu. AVA-VLA: Improving vision-language-action models with active visual attention. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2026.

Guowei Xu, Peng Jin, Ziang Wu, Hao Li, Yibing Song, Lichao Sun, and Li Yuan. LLaVA-CoT: Let vision language models reason step-by-step. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2025.

Zhiyang Xu, Chao Feng, Rulin Shao, Trevor Ashby, Ying Shen, Di Jin, Yu Cheng, Qifan Wang, and Lifu Huang. Vision-Flan: Scaling human-labeled tasks in visual instruction tuning. *arXiv:2402.11690*, 2024.

Kaiyu Yang, Olga Russakovsky, and Jia Deng. SpatialSense: An adversarially crowdsourced benchmark for spatial relation recognition. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2019.

Yandan Yang, Shuang Zeng, Tong Lin, Xinyuan Chang, Dekang Qi, Junjin Xiao, Haoyun Liu, Ronghan Chen, Yuzhi Chen, Dongjie Huo, et al. ABot-M0: VLA foundation model for robotic manipulation with action manifold learning. *arXiv:2602.11236*, 2026.

Haoqi Yuan, Zhixuan Liang, Anzhe Chen, Ye Wang, Haoyang Li, Pei Lin, Yiyang Huang, Zixing Lei, Tong Zhang, Jiazhao Zhang, et al. Qwen-RobotManip technical report: Alignment unlocks scale for robotic manipulation foundation models. *arXiv:2606.17846*, 2026.

Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. RoboPoint: A vision-language model for spatial affordance prediction in robotics. In *Proc. Conf. Robot Learn.*, 2024.

Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training. In *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023.

Bo Zhao, Boya Wu, Muyang He, and Tiejun Huang. SVIT: Scaling up visual instruction tuning. *arXiv:2307.04087*, 2023a.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems*, 2023b.

Linqing Zhong, Yi Liu, Yifei Wei, Ziyu Xiong, Maoqing Yao, Si Liu, and Guanghui Ren. ACoT-VLA: Action chain-of-thought for vision-language-action models. *arXiv:2601.11404*, 2026.

Enshen Zhou, Jingkun An, Cheng Chi, Yi Han, Shanyu Rong, Chi Zhang, Pengwei Wang, Zhongyuan Wang, Tiejun Huang, Lu Sheng, and Shanghang Zhang. RoboRefer: Towards spatial referring with reasoning in vision-language models for robotics. In *Adv. Neural Inf. Process. Syst.*, 2025.

Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. Visual7W: Grounded question answering in images. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2016.

APPENDIX

A Limitations **17**

B Full benchmark results **17**

C Model details **18**

C.1 Inference cost and caching 19

D Data **19**

D.1 Robot embodiments 19

D.2 Data curation 20

D.3 Sampling weights 20

D.4 Egocentric data 21

D.5 Vision-language data 21

E Training configuration **22**

F Training budgets of the compared methods **23**

G Real Robot Experiments **23**

G.1 Tasks and Scoring 23

G.2 Finetuning Details 24

G.3 Qualitative examples of real-robot rollouts 24

A LIMITATIONS

Our memory is short. With $K=3$ frames one second apart, it covers the two preceding seconds, and four on the real robot, where the frames are two seconds apart. Since the benchmarks we use rarely require more, whether the registers can carry longer context, for tasks that depend on events far in the past, remains untested. The memory is also built from the observations alone and does not include past robot states. In distribution, REKI stays below Qwen-RobotManip, which is pretrained on in-house data. The oscillation analysis covers a single benchmark, with a detector whose threshold was fixed on 16 of its 50 tasks, and most of our evaluation is in simulation. Finally, we are not licensed to redistribute the pretraining data, so we release the pipeline that curates it from its public sources, whose availability we do not control.

B FULL BENCHMARK RESULTS

Tables 1 and 2 give the per-condition results summarised in Fig. 4, together with the in-distribution results on LIBERO and RoboTwin. Each LIBERO-Plus column perturbs one axis: the camera viewpoint, the initial robot state, the language instruction, the lighting, the background texture, the sensor noise or the object layout. The Overall column is the success rate over all 10,030 perturbed episodes, so each axis is weighted by its number of episodes (Fei et al., 2025). For the in-distribution RoboTwin results, the models are finetuned on clean and randomised scenes together and evaluated on each (Clean, Rand). For Clean2Rand, the models are finetuned on clean scenes only and evaluated on clean scenes (Easy), under a randomised background, lighting, clutter or table height applied alone, and under all four together (Hard). The baseline numbers are taken from published tables. Qwen-RobotManip (Yuan et al., 2026) provides its own results and those of Qwen-RobotManip-scratch. It also provides the LIBERO and LIBERO-Plus results of StarVLA and Being-H0.7, the LIBERO results of π_0 and $\pi_{0.5}$, the Clean and Rand results of StarVLA and $\pi_{0.5}$, and the Clean2Rand results of every baseline. The Clean2Rand row of Qwen-RobotManip is its joint-space variant. The LIBERO-Plus results of π_0 and $\pi_{0.5}$ are from the zero-shot block of ACoT-VLA (Zhong et al., 2026), which evaluated the released checkpoints. ACoT-VLA, Abot-M0, Abot-M0.5 and InternVLA-A1.5 report their own LIBERO and LIBERO-Plus results, and Abot-M0 also reports its own Clean and Rand results. For Being-H0.7 and Abot-M0.5, the LIBERO-Plus Overall is pooled from the seven published dimensions, because the published total does not match them. No Clean and Rand results are published for GR00T-N1.7.

Table 1: **LIBERO and LIBERO-Plus results.** All models are finetuned on LIBERO; LIBERO-Plus evaluates out-of-distribution robustness across seven perturbation dimensions. Bold: best in each column within each block, grey rows excluded. †Pooled from the seven reported dimensions, which the published total does not match.

Method	#Params		LIBERO	LIBERO-Plus							
	VLM	Expert		Camera	Robot	Language	Light	Backgd	Noise	Layout	Overall
No pretraining											
StarVLA (StarVLA Community, 2026)	4B	0.1B	98.0	52.5	49.8	**88.5**	95.7	95.7	73.0	76.9	74.1
Qwen-RobotManip-scratch (Yuan et al., 2026)	4B	0.2B	98.2	70.4	44.9	88.1	95.8	95.5	84.4	79.1	78.3
ACoT-VLA (Zhong et al., 2026)	2.9B	0.9B	**98.5**	72.6	**82.6**	87.5	**97.7**	**96.5**	87.8	**88.1**	**86.6**
REKI (ours) *scratch*	2B	0.2B	96.0	**89.9**	57.0	81.1	96.0	94.4	**92.2**	75.1	82.8
With pretraining – in-house data											
Qwen-RobotManip (Yuan et al., 2026)	4B	0.2B	99.1	87.2	75.5	85.6	96.6	97.7	97.7	87.3	89.0
π_0 (Black et al., 2024)	2.9B	0.3B	94.4	61.0	40.8	63.5	89.3	84.1	80.1	76.4	69.4
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	2.9B	0.4B	97.6	75.8	79.4	83.3	95.5	95.0	89.6	87.0	85.7
Being-H0.7 (Luo et al., 2026)	3B	–	99.2	82.0	59.0	82.8	**97.8**	90.0	93.5	**88.5**	84.0†
With pretraining – public data only											
Abot-M0 (Yang et al., 2026)	4B	0.2B	98.6	60.4	67.9	86.4	96.2	91.6	86.4	82.6	80.5
Abot-M0.5 (Chen et al., 2026)	5B	–	**99.4**	70.5	**87.4**	**88.6**	94.0	89.7	75.5	85.2	83.7†
InternVLA-A1.5 (Ma et al., 2026)	2B	0.5B	98.9	**83.1**	55.1	86.9	96.4	**98.2**	95.6	85.2	84.8
REKI (ours)	2B	0.2B	97.9	79.7	73.4	86.0	95.9	98.0	**96.0**	83.1	**86.6**

Table 2: **RoboTwin results.** *Left*: in-distribution results for models finetuned on RoboTwin + RoboTwin-Randomized. *Right*: out-of-distribution evaluation on RoboTwin-Clean2Rand, where models are finetuned on the Clean dataset only and tested under various environment randomisations. Bold: best in each column within each block, grey rows excluded.

Method	#Params		RoboTwin		RoboTwin Clean2Rand					
	VLM	Expert	Clean	Rand	Easy	Backgd	Light	Clutter	Height	Hard
No pretraining										
StarVLA (StarVLA Community, 2026)	4B	0.1B	85.7	87.3	58.1	27.1	50.9	24.2	48.4	10.6
Qwen-RobotManip-scratch (Yuan et al., 2026)	4B	0.2B	**88.7**	**88.4**	**71.6**	60.6	**70.7**	24.6	**63.6**	22.6
REKI (ours) *scratch*	2B	0.2B	79.2	80.6	64.8	**63.2**	67.2	**33.4**	60.8	**32.8**
With pretraining – in-house data										
Qwen-RobotManip (Yuan et al., 2026)	4B	0.2B	93.4	92.5	73.2	74.6	68.4	61.3	71.0	62.6
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	2.9B	0.4B	82.7	76.8	73.1	**67.0**	69.2	57.9	67.6	**47.9**
GR00T-N1.7 (Bjorck et al., 2025)	1.8B	1.3B	–	–	43.6	40.4	41.9	27.1	39.0	20.7
With pretraining – public data only										
Abot-M0 (Yang et al., 2026)	4B	0.2B	**86.1**	**85.1**	70.7	56.5	68.8	46.0	56.3	36.0
REKI (ours)	2B	0.2B	84.9	84.2	**73.4**	66.6	**70.7**	**58.0**	**68.4**	46.1

C MODEL DETAILS

Figure 8 shows how the prompt and the camera views are packed into one stream for the VLM of §3.

Prompt The prompt z_t of §3.2 is a structured user turn carrying one `field: value` per line:

```
embodiment: gr1
instruction: grasp the mug
speed: 2
fps: 30
control mode: joint
```

The speed field is the length of the episode in frames divided by 100 and rounded, so that a smaller value marks a shorter demonstration. It is left out at evaluation.

Positions In the expert's self-attention, register i of a frame sits at slot i and action step j at slot $n_r + j$, so that the registers of each frame carry their own temporal encoding. The frame and slot axes reuse the temporal and spatial axes of the VLM's multimodal rotary embedding.

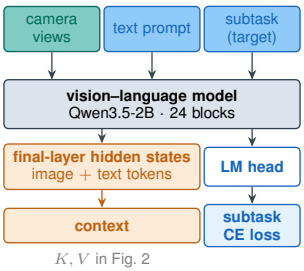


Figure 8: The VLM encoder. It feeds the expert of Fig. 2 the encoded context and the state. A vision–language model encodes the views and the prompt; its final-layer hidden states are the encoded context and its LM head predicts the subtask.

C.1 Inference cost and caching

What is cached. One chunk is an encoder pass, a pass over the register tokens, then N Euler steps over the action tokens. Nothing the expert reads besides the noisy actions depends on the flow time τ, so four quantities are reused across the N steps.

1. The encoded context $\boldsymbol{E}_t$ of Eq. (4).
2. Its projection to keys and values, one pair per expert block.
3. The register tokens $\boldsymbol{M}_t$ of the current frame.
4. The register keys and values of the $K-1$ preceding frames, carried over from earlier chunks.

The first three are built the same way at training and at inference, while the fourth is not: the first registers of a window hold memory from earlier registers at inference, and none at training.

What it costs. Figures 3b and 3c give the measurement. Attending to the registers of one more past frame costs 127 µs of GPU time: 112 in the attention of the action tokens over their cached keys and values, repeated at the N Euler steps, and 15 in the register pass, which reads them once. At $K=3$ that is 0.4 ms on 59.5, against a chunk covering one second of motion. Peak allocated memory is flat at 4.6 GB between $K=1$ and 16. Wall-clock is about 220 ms per chunk at every K, inside the second of motion the chunk covers. The gap to GPU time is the kernel-launch overhead of eager execution, not a cost of the model.

The register pass is flat in K, at 5.6 ms. It runs eagerly and pays for kernel launches over an $n_r \times d_e$ tensor. The chunk is bound by the number of kernels launched rather than by arithmetic, so the caches that pay are the ones removing a traversal, not the ones shrinking a tensor. Caching the context and the current frame's registers saves work repeated over the N Euler steps. The past frames cost only the attention over their stored keys and values.

Protocol. Every measurement is taken on one H100, with two 256×256 cameras, a chunk of 10 actions at 10 Hz and $N=10$ Euler steps. The model is run twice at each K, once for wall-clock timing and once under the CUDA profiler. It is trained and evaluated at $K=3$. The longer windows are used only for these measurements.

D Data

D.1 Robot embodiments

Table 3 lists the robots of the pretraining mixture with their action conventions and source datasets. A robot appears under several conventions when its datasets command it differently, joint angles in one and end-effector deltas in another. The state projection of §3.2 is selected by state layout and control convention, and the action statistics of §5.1 are per dataset, so no source is remapped to another's convention. A state layout is the ordered list of state channels a dataset records, for example the joint angles and gripper opening of each arm, and fixes the input width d_s of its projection.

Table 3: **Robot embodiments in the training mixture.** "EE" = end-effector pose; "J" = joint angles; "Δ" = delta (relative) commands, otherwise absolute; "G" = parallel gripper; "DH" = dexterous hand; "W" = waist/torso; "B" = mobile base. A robot may appear under several conventions across its source datasets.

Robot	Action type	Sources
Single-arm		
WidowX	ΔEE + G	(Walke et al., 2023)
Google Robot	ΔEE + G	(Brohan et al., 2023; Jang et al., 2022)
Franka Panda	ΔEE + G; J + G	(Bharadhwaj et al., 2024; Hou et al., 2025; Khazatsky et al., 2024; Tian et al., 2026)
UR5e	J + G	(Hou et al., 2025)
Dual-arm		
AgileX Cobot Magic	J + G	(Wu et al., 2025; Hou et al., 2025; Liu et al., 2025)
ALOHA	J + G	(Tian et al., 2026)
AgiBot G1	J + G	(Bu et al., 2025; Tian et al., 2026)
Galaxea R1	J + G + W + B	(Jiang et al., 2025; Wu et al., 2025)
YAM	J + G	(Allshire et al., 2026)
Bimanual Panda	ΔEE + G; ΔEE + DH	(Bjorck et al., 2025)
Fourier GR-1	J + DH	(Bjorck et al., 2025)
TienKung	J + G; J + DH	(Hou et al., 2025)
AIRBOT MMK2	J + DH	(Wu et al., 2025)
Unitree G1	J + DH	(Wu et al., 2025)
Kuavo	J + DH	(Wu et al., 2025)
Human		
Real Human	ΔEE + eigengrasp	(Hoque et al., 2026; Punamiya et al., 2026; Ropedia, 2026)

D.2 Data curation

Aggregating data across embodiments, simulators, and collection pipelines introduces heterogeneous noise: sensor outliers, temporal state-action misalignment, and inconsistent conventions across datasets sharing a robot.

- *Channel selection.* We keep only the degrees of freedom the policy commands, dropping derived or uncontrolled channels: forward-kinematic end-effector poses, joint velocities, efforts, IMU readings, and telepresence-parked head, waist, and leg joints.
- *Degenerate episodes.* We discard static and near-zero-variance recordings, including episodes whose follower arm never moves, a failure mode Qwen-RobotManip reports for $\sim 81\%$ of RoboMIND's UR subset.
- *Action representation.* Rotations are harmonised to one Euler parameterisation; other conventions are kept as the source provides them. Each action dimension is mapped to $[-1, 1]$ from its per-dataset 1st and 99th percentiles (Wang et al., 2026), which removes scale differences across embodiments while preserving relative motion within each source.
- *Convention correction.* We fix per-dataset sign conventions that would otherwise break cross-dataset consistency (*e.g.*, a sign-flipped elbow joint in InternData); differing scales and encodings are then absorbed by the per-dataset normalisation.

D.3 Sampling weights

During pretraining, each dataset d is sampled with probability proportional to $F_d^{0.5}\, L_d^{0.5}\, V_d^{0.35}$ (Table 4). F_d is its number of frames, and L_d and V_d are its language and vision diversity, each divided by its median over the datasets. Datasets from the same source are grouped before weighting.

Both diversities are the effective rank of a set of L2-normalised embeddings, $2^{H(p)}$ with $p_i = \sigma_i^2 / \sum_j \sigma_j^2$, where σ_i are the singular values of the embedding matrix. The embeddings are not centred, so that a dataset with two distinct instructions does not collapse to rank one. For the language diversity, the rows are the SigLIP (Zhai et al., 2023) text embeddings of the distinct episode task

strings, each scaled by the square root of the fraction of frames it covers. For the vision diversity, the rows are the DINOv3 (Siméoni et al., 2025) class tokens of every 15th frame of the main external camera, at most 50,000 frames per dataset.

The RoboCOIN and InternData-A1 collections are further weighted by 0.5, and the weights are capped so that no dataset is seen more than four times.

D.4 Egocentric data

Sources The three egocentric sources differ in capture conditions: EgoDex is seated tabletop recordings on Apple Vision Pro, EgoVerse head-mounted rigs, and Xperience in-the-wild moving-camera footage. Frame rates range from 20 to 30 Hz and are left unaltered; the differences are absorbed by the per-dataset quantile normalisation of §5.1.

Action target We follow the layout of Qwen-VLA's egocentric target (Wang et al., 2026): per hand, a 6-dimensional wrist motion as translation and axis-angle rotation, and 10 articulation coefficients, 32 dimensions in total. Unlike Qwen-VLA, which expresses the wrist relative to its own initial pose, we express it in the observation camera frame through the rigid transform $T_{\text{cam}}^{-1} T_{\text{wrist}}$. The articulation coefficients are the first 10 principal components ("eigengrasps") of the finger-joint configuration under a PCA basis shared across our three sources. The target is defined relative to the frame that opens each action chunk, so it is composed at load time from the stored world-frame wrist poses and per-frame camera poses rather than precomputed.

State and pacing The egocentric data have no proprioceptive state, and the policy receives a learned null state in its place. Human hands move faster than a teleoperated robot, so the human actions are slowed down in time (Yuan et al., 2026). Waypoint i of a chunk is the hand pose at time $i\rho/f$ after the observation, where f is the frame rate of the recording and $\rho \leq 1$ a per-source pace (Table 4). The pose is interpolated between the two nearest frames. A one-second chunk at $\rho = 0.25$ thus holds a quarter of a second of human motion.

D.5 Vision-language data

REKI is co-trained on vision–language data at 10% of the sampled frames (Table 4). The pool is every FineVision (Wiedmann et al., 2025) dataset except the text-only ones, plus the LLaVA-OneVision-1.5 (An et al., 2025) datasets that FineVision does not already contain, ∼20.2M samples. We keep the datasets covering the tasks most relevant to manipulation: natural-scene objects and attributes, spatial relations, counting, grounding, captioning, scene text, and knowledge and commonsense VQA. Documents, charts, tables, mathematics, code, transcription, GUI grounding, memes and puzzles are dropped. 60 datasets remain, 7.64M samples or 37.8% of the pool. One question–answer pair is drawn per sample and sampling is uniform over samples, so exposure is proportional to the size of a dataset.

The 51 kept from FineVision are `a_okvqa` (Schwenk et al., 2022), `alfworldgpt` (Shridhar et al., 2021), `allava_laion` (Chen et al., 2024a), `allava_vflan` (Chen et al., 2024a), `aokvqa` (Schwenk et al., 2022), `art` (Chng et al., 2019), `cambrian(filtered)_processed` (Tong et al., 2024), `clevr` (Lindström & Abraham, 2022), `clevr_math` (Lindström & Abraham, 2022), `coco_colors` (Karakus, 2024), `cocoqa` (Ren et al., 2015), `cocotext` (Veit et al., 2016), `densefusion_1m` (Li et al., 2024), `drivelm` (Sima et al., 2024), `est_vqa` (Wang et al., 2020), `google_landmarks` (Weyand et al., 2020), `image_textualization(filtered)` (Pi et al., 2024), `indoor_qa` (Quattoni & Torralba, 2009), `laion_gpt4v` (LAION, 2023), `LLaVA_Instruct_150K` (Liu et al., 2023c), `lnqa` (Pont-Tuset et al., 2020), `localized_narratives` (Pont-Tuset et al., 2020), `lrv_normal(filtered)` (Liu et al., 2024a), `lvis_instruct4v` (Wang et al., 2023), `mimic_cgd` (Laurençon et al., 2024), `mmevol` (Luo et al., 2024), `nlvr2` (Suhr et al., 2017), `objects365_qa` (Shao et al., 2019), `ocrvqa` (Mishra et al., 2019), `oodvqa` (Tu et al., 2023), `scienceqa` (Lu et al., 2022), `scienceqa(nona_context)` (Lu et al., 2022), `sharegpt4o` (OpenGVLab, 2024), `sharegpt4v(coco)` (Chen et al., 2024b), `sharegpt4v(knowledge)` (Chen et al., 2024b), `sharegpt4v(llava)` (Chen et al., 2024b), `sharegpt4v(sam)` (Chen et al., 2024b), `sketchyvqa` (Tu et al., 2023),

Table 4: **Training configuration.**

Setting	Value
Encoder	
Vision–language backbone	Qwen3.5-2B (18 gated-DeltaNet + 6 full-attention layers)
Image resolution	native, up to 448×448
Expert conditioning	cross-attention to the final-layer backbone states in every expert block
Trainable components	backbone at $0.1\times$ the expert LR, last 4 vision blocks, expert
Vision–language co-training	10%: FineVision (subset), LLaVA-OneVision-1.5 (subset), RoboPoint, RefSpatial, PixMo
Text loss (subtask, vision–language data)	cross-entropy, weight 0.1, pretraining only
Prompt-field dropout	0.2 per field
Action expert	
Size	10 blocks, 768 hidden, 12 heads, 3072 MLP
Precision	bfloat16, flow heads in fp32
Proprioception	per-layout linear layer with GELU, shared zero-initialised projection, summed into the adaptive-normalisation conditioning
Proprioception dropout	0.2
Registers per frame	32
Window	$K = 3$ frames at a 1 s stride
Action generation	
Action chunk	1 s (round(control freq.) steps)
Objective	flow matching, DoF-balanced
Time-step distribution	Beta
Flow draws per forward pass	8
Data	
Robot corpora	OXE, DROID, AgiBot, RoboMIND, Galaxea, RoboSet, RDT, RoboCOIN, GR00T, InternData, ABC
Egocentric human video	EgoDex, EgoVerse, Xperience
Sampling weight	$\propto$ {#frames}$^{0.5}$ {language diversity}$^{0.5}$ {vision diversity}$^{0.35}$ (Appendix D.3)
Normalisation	quantile (1st / 99th percentile)
Action-speed alignment	EgoDex 0.6, EgoVerse 0.25, Xperience 1 (unchanged)
Vision & augmentation	
Camera dropout	0.15 per camera; views shuffled
Augmentation	colour jitter 0.2, random crop 0.95
Optimisation	
Learning rate	1×10^{-4} (expert), 1×10^{-5} (encoders, see above)
Weight decay	0
Gradient clipping	1.0
Schedule	constant with warmup (5,000 steps)
Weight averaging	EMA, decay 0.999
Gradient checkpointing	✓

`spatialsense` (Yang et al., 2019), `spot_the_diff` (Jhamtani & Berg-Kirkpatrick, 2018), `st_vqa` (Biten et al., 2019), `super_clevr(mathv360k)` (Li et al., 2023), `tallyqa` (Acharya et al., 2019), `textcaps` (Sidorov et al., 2020), `textocr(gpt4v)` (Carter, 2024), `textvqa` (Singh et al., 2019), `vision_flan(filtered)` (Xu et al., 2024), `visual7w` (Zhu et al., 2016), `vqav2` (Goyal et al., 2017), `vsr` (Liu et al., 2023b), and `wildvision` (Lu et al., 2024). The nine kept from LLaVA-OneVision-1.5 are `gqa` (Hudson & Manning, 2019), `llava_cot_100k` (Xu et al., 2025), `llava_wild` (Liu et al., 2023c), `llrv_gpt4v` (Liu et al., 2024a), `sherlock` (Hessel et al., 2022), `svit` (Zhao et al., 2023a), `viquae` (Lerner et al., 2022), `vision_oritented`, and `visual_chat`.

To these we add the object- and free-space-reference splits of RoboPoint (Yuan et al., 2024), the 2D and 3D splits of RefSpatial (Zhou et al., 2025), and PixMo-Points (Deitke et al., 2025) rewritten as pointing and counting questions, all with coordinates on a $[0, 1000]$ grid.

E Training configuration

Table 4 lists the training configuration. Table 5 gives the node count, the number of steps and the settings that differ between pretraining and the three finetuning runs. Settings absent from Table 5 are those of Table 4 in every run.

Table 5: **Per-run setup of REKI** for pretraining and the three finetuning runs, and the number of evaluation rollouts per task. The two RoboTwin runs are finetuned on the Clean demonstrations alone (Clean2Rand) and on Clean and Randomised together (in-distribution).

	Pretraining	LIBERO	RoboTwin	
			Clean	Clean+Rand
Nodes (8×H100)	4	1	1	3
Steps	100k	20k	30k	30k
Frame sampling (W $\propto \sqrt{N_{\text{frames}}}$)	W	–	–	W
Head-camera dropout	✗	✓	✗	✗
Trained vision encoder (last 4 blocks)	✓	✗	✓	✓
Sagittal mirroring	✗	✓	✓	✓
Evaluation rollouts per task	–	50	20	20

Table 6: **Training budgets** of the compared methods as optimiser steps, global batch size and training samples seen (batch × steps). A sample is one timestep with the action chunk that follows it. None of the methods accumulates gradients. –: not reported; †: read off the released code or checkpoint; ∗: stated in the paper as a maximum; the actual run length is not reported. For REKI, the batch is $K=3$ frames times the number of windows. This is an upper bound, since a batch is trimmed to a token budget. A slash separates the Clean and Clean+Rand RoboTwin runs.

Method	Pretraining			LIBERO			RoboTwin		
	Steps	Batch	Samples	Steps	Batch	Samples	Steps	Batch	Samples
No pretraining									
StarVLA (StarVLA Community, 2026)	-	-	-	30k	128	3.8M	150k	192	28.8M
REKI (ours), scratch	-	-	-	30k	720	21.6M	45k	720 / 2160	32.4M / 97.2M
With pretraining – in-house data									
$\pi_{0.5}$ (Physical Intelligence et al., 2025)	280k + 80k	–	–	30k†	256†	7.7M†	–	–	–
With pretraining – public data only									
Abot-M0 (Yang et al., 2026)	100k	1024	102.4M	40k†	64†	2.6M†	125k†	192†	24.0M†
InternVLA-A1.5 (Ma et al., 2026)	300k + 600k	1024	921.6M	100k†	–	–	100k	384	38.4M
REKI (ours) (Table 5)	100k	2880	288.0M	20k	720	14.4M	30k	720 / 2160	21.6M / 64.8M

F Training budgets of the compared methods

Table 6 collects the training budgets published for the methods compared in Tables 1 and 2. Qwen-RobotManip, its scratch variant and Being-H0.7 state none and are left out. No compared method reports wall-clock time or accelerator-hours, so the success rates of those tables are not matched for training compute.

G Real Robot Experiments

G.1 Tasks and Scoring

We evaluate on five tasks. Partial credit is awarded for each completed subtask, such as a successful grasp or placement. For tasks with an imposed ordering, a penalty is applied when objects are handled in the wrong order. An episode counts as a full success only when every subtask is completed in the correct order. The first two tasks involve a single object and are shorter-horizon. The remaining three require manipulating two or three objects in a prescribed order, so the policy must keep track of which steps have already been completed.

- **place_banana_in_box**: pick the yellow banana and place it in the box. Partial credit: 0.5 for grasp; 0.5 for placement (only 0.25 if partially inside the box).
- **sweep_paper_into_dustpan**: grasp the brush and sweep the paper crumples into the dustpan. Partial credit: 0.5 for grasp; 0.25 once one crumple is inside the dustpan; 0.25 once all crumples are inside.

- **arrange_3_fruits_in_plates**: place the purple grapes in the yellow plate, then the yellow banana in the pink plate, then the red strawberry in the blue plate. Partial credit: $1/6$ per grasp and $1/6$ per placement for each of the three fruits; $-1/6$ for wrong order.
- **stack_3_cubes**: stack the pink cube on the yellow cube, then the green cube on the pink cube. Partial credit: 0.25 per grasp and 0.25 per placement for each of the two cubes; -0.25 for wrong order.
- **stack_3_cups**: stack the pink cup into the yellow cup, then stack the blue cup into the pink cup. Partial credit: 0.25 per grasp and 0.25 per placement for each of the two cups; -0.25 for wrong order.

G.2 Finetuning Details

Table 7: **Finetuning hyperparameters** for the real-robot experiments.

	REKI	$\pi_{0.5}$
Optimiser steps	3,000	3,000
Global batch size	360	360
Learning rate (action expert)	5×10^{-5}	5×10^{-5}
Learning rate (backbone)	5×10^{-6}	5×10^{-6}
LR schedule	warmup, then constant	warmup, then constant
Trainable vision blocks	last 4	last 4
Action chunk size H	30 (2 s)	50 (3.3 s)
Memory window K	$K=3$ frames, 2 s apart ($K=1$ without memory)	–

Both REKI and $\pi_{0.5}$ are initialised from their pretrained checkpoints and finetuned jointly on the five tasks, using the hyperparameters in Table 7. The action chunk has $H=30$ steps (2 s) for REKI and $H=50$ (3.3 s) for $\pi_{0.5}$, which is fixed by its pretraining.

G.3 Qualitative examples of real-robot rollouts

In Fig. 9, we show examples of REKI's trajectories across tasks on the real-robot setup.

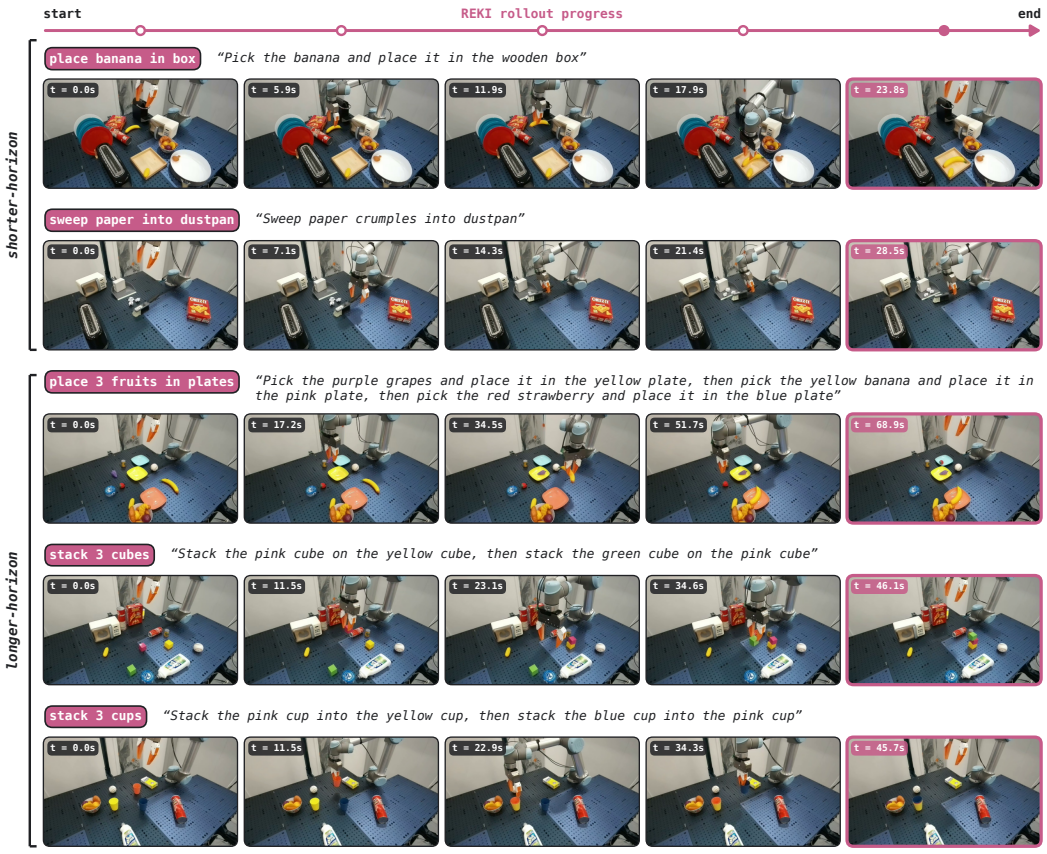


Figure 9: **Qualitative examples of rollouts of REKI on our UR5 robotic arm, when evaluated across 5 manipulation tasks.**