REMEMBERING WHAT MATTERS FOR LONG-HORIZON ROBOT WORLD MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Autoregressive robot video world models based on discrete visual tokens remain unreliable over long rollout horizons. Retaining the entire generated history does not necessarily improve long-horizon prediction quality. We investigate a possible *memory-role conflict* in Full Context decoding: persistent scene information, recent visual–action states, and stale generated history compete for attention despite their different predictive roles. This perspective motivates **ReCAP** (Receding Context with Anchor Preservation), a training-free context-scheduling method for long-horizon robot video world models. **ReCAP** keeps the pretrained tokenizer, world model, and visual decoder frozen. At inference time, it preserves the initial scene anchor and a contiguous window of recent complete visual–action blocks while evicting older generated history. The schedule preserves within-block structure and temporal order, requires no additional modules or parameter updates, and bounds active-context growth independently of rollout length. We evaluate **ReCAP** on RT-1, BridgeV2, LIBERO-90, and CALVIN using frozen autoregressive world models. Across the evaluated environments and rollout horizons, **ReCAP** improves visual fidelity, action-conditioned consistency, and robot structure preservation relative to the corresponding baselines. On BridgeV2, it reduces LPIPS by 63.3%. In the system benchmark at $H = 64$, it reduces the logical KV footprint by 73.3% and p50 latency by 6.51%. Attention analyses show increased attention to retained recent states, while history interventions associate larger amounts of stale context with higher prediction error. These observations are consistent with the memory-role-conflict hypothesis, although they do not isolate it from context-length and position-handling effects. Our results support inference-time context scheduling as a practical approach to improving long-horizon robot video prediction and computational efficiency without retraining. Code, implementation details, and additional qualitative examples are available here.

1 INTRODUCTION

Robot video world models predict how visual observations evolve under actions, providing a reusable substrate for planning, policy evaluation, and synthetic experience generation (Finn et al., 2016; Ebert et al., 2018; Ha & Schmidhuber, 2018; Hafner et al., 2023). Progress in discrete visual representation learning (van den Oord et al., 2017; Mentzer et al., 2024) and autoregressive modeling (Vaswani et al., 2017; Yan et al., 2021) underpins models that express observations and actions within a unified causal token sequence (Micheli et al., 2023; Wu et al., 2024). Related advances span interactive environments and robot video prediction (Bruce et al., 2024; Ren et al., 2025; Wu et al., 2025; 2026). Yet robot world models must do more than generate plausible short clips: they must preserve scene identity, robot morphology, object state, and action responsiveness over long horizons. These properties often deteriorate during autoregressive generation, limiting their use in long-horizon visual reasoning, action planning, and robot control.

We use *Full Context* to denote the baseline that retains the initial context, action conditions, and all generated visual states throughout each evaluated rollout. This strategy appears natural because additional history should, in principle, provide more evidence for future prediction. Our experiments reveal the opposite behavior. As shown by the gray curve in Fig. 1(a), the attention assigned to the most recent visual state steadily decreases as the rollout grows, even though distant context and early generated states remain accessible. The resulting predictions are initially plausible but progressively

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.

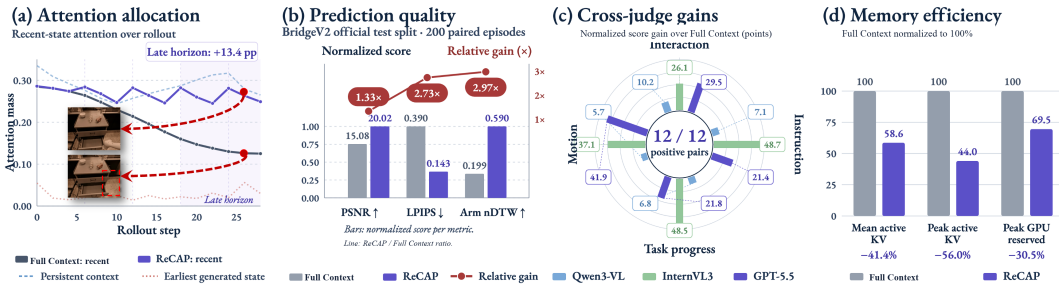

Figure 1: **`ReCAP` *improves long-horizon prediction and memory efficiency.*** (a) `ReCAP` preserves attention to recent visual states. (b) It improves prediction quality on BridgeV2. (c) Three VLM judges score `ReCAP` higher across semantic criteria. (d) It reduces KV-cache and GPU-memory usage. Panels use dedicated samples and extrapolation horizons and are therefore not directly comparable to the fixed-horizon evaluations in Section 5.

exhibit structural omission, weakened action response, and motion drift. Meanwhile, the active context and KV cache grow monotonically with the rollout horizon. This leads to the central question:

> **Key question:** *How should an autoregressive robot video world model organize its inference context to remain dynamically informative and memory efficient over long horizons?*

We hypothesize that this failure partly reflects the different roles of rollout history: persistent scene context, recent visual–action states, and older generated history. Full Context places these tokens in a shared attention pool; as history grows, the relative attention to recent states can decline. We call this possible interference *memory-role conflict*. The traces in Fig. 1(a) motivate this interpretation, but do not by themselves establish causality. Attention sinks and streaming inference have been studied in language models (Xiao et al., 2024; Gu et al., 2025a; Barbero et al., 2025); robot video prediction additionally requires preserving the structure and alignment of visual–action blocks.

Existing long-context and KV-cache methods primarily identify important individual tokens or compress a language-model cache (Zhang et al., 2023; Liu et al., 2023b; Ge et al., 2024; Li et al., 2024). Directly transferring token-level eviction to robot video rollouts is non-trivial because a physical time step is represented by a structured block of visual and action tokens; fragmenting this block can disrupt spatial coherence and causal alignment. Recent long-context video models also recognize that nearby frames and distant context play different roles (Gu et al., 2025b), but typically address this distinction through architectural design or training. We instead ask whether the inference context of an already trained world model can be reorganized without modifying its parameters. This shift turns long-horizon rollout from a problem of retaining *more* history into one of retaining the *right* history for predicting subsequent visual states.

We introduce `ReCAP` (Receding Context with Anchor Preservation), a training-free context scheduler. It preserves the initial anchor and a contiguous window of recent complete visual–action blocks, evicting older generated blocks after each update. This retains within-frame structure, action alignment, and temporal order while bounding active context independently of rollout length. The tokenizer, world model, and decoder remain frozen. Unlike methods requiring long-context training or architectural changes (Gu et al., 2025b), `ReCAP` changes only the inference schedule.

Figure 1 summarizes the resulting gains under its dedicated overview protocols. Context reorganization repeatedly restores the influence of recent visual states, yielding a 13.4-percentage-point advantage over Full Context at the late extrapolation horizon (Fig. 1a). On its 200-pair BridgeV2 evaluation split, `ReCAP` improves perceptual fidelity, pixel quality, and robot-trajectory consistency, reducing LPIPS by 63.3% (Fig. 1b). On 200 paired RT-1 episodes, Qwen3-VL, InternVL3, and GPT-5.5 yield 12/12 positive judge–criterion score differences (Fig. 1c; Appendix D). Under the corresponding efficiency profile, `ReCAP` reduces mean and peak active KV-cache usage by 41.4% and 56.0%, respectively, and lowers peak reserved GPU memory by 30.5% (Fig. 1d), with near-neutral wall-clock overhead. These overview results use different samples, extrapolation lengths, and aggregation rules from the fixed-horizon analyses in Section 5; their numerical values should therefore not be compared directly. Across RT-1, BridgeV2, LIBERO-90, and CALVIN, the main experiments nevertheless show that the improvements persist across environments and rollout horizons.

History interventions associate additional stale context with increased prediction error, although length and position effects remain confounded. A separate prefix-length study tests scene/history budget allocation (Appendix E), while human comparison and rubric sensitivity analyses assess the semantic evaluation (Appendices D.6–D.7). Our contributions are:

1. Empirical diagnosis. We identify memory-role conflict as a recurring failure mode of Full Context decoding in long-horizon robot video world models, and connect it to recent-state attention decay, structural drift, and unbounded context growth. ***2. Training-free context scheduling.*** We introduce `ReCAP`, which preserves a persistent context anchor and recent visual–action states while evicting stale generated history, without retraining or architectural modification. ***3. Multi-level validation.*** We demonstrate consistent improvements in prediction quality, semantic consistency, and memory efficiency across robot datasets, supported by controlled interventions, attention analysis, and ablations of the individual context components.

2 Related Work

Robot Video World Models. World models predict action-conditioned dynamics for planning and evaluation (Ha & Schmidhuber, 2018; Hafner et al., 2023). Early robot video models predict future pixels (Finn et al., 2016; Ebert et al., 2018), while VQ-VAE and VideoGPT develop discrete visual modeling (van den Oord et al., 2017; Yan et al., 2021). iVideoGPT interleaves visual and action tokens (Wu et al., 2024); VideoWorld learns from videos using autoregressive modeling (Ren et al., 2025). Genie supports latent-action-conditioned interaction (Bruce et al., 2024), whereas RoboDreamer uses compositional diffusion for robot imagination (Zhou et al., 2024). RLVR-World and RoboAlign-R1 study reinforcement learning and reward alignment (Wu et al., 2025; 2026). RoboAlign-R1 also refreshes context with training-free Sliding Window Re-encoding (SWR), decoding and re-encoding predicted frames. `ReCAP` instead preserves the initial scene anchor and schedules complete visual–action blocks in token space without re-encoding.

Long-Horizon Autoregressive Video Modeling. Autoregressive world models use generated states as context for subsequent video prediction (Bruce et al., 2024; Wu et al., 2024; Ren et al., 2025). Yet train–inference mismatch compounds errors over time, degrading scene identity, object structure, motion, and action responsiveness (Bengio et al., 2015; Huang et al., 2025; Zhang et al., 2025). Retaining complete history also increases attention and KV-cache costs without guaranteeing better predictions (Gu et al., 2025b; Po et al., 2025). Hierarchical context, context packing, and state-space memory balance local and distant information through specialized architectures or training (Gu et al., 2025b; Zhang et al., 2025; Po et al., 2025). In contrast, `ReCAP` reorganizes the existing visual–action context of a frozen robot world model at inference time.

Memory and Context Management. H2O, Scissorhands, FastGen, and SnapKV select or compress language-model KV caches by token importance (Zhang et al., 2023; Liu et al., 2023b; Ge et al., 2024; Li et al., 2024). StreamingLLM preserves initial and recent tokens (Xiao et al., 2024), while attention-sink analyses explain the role of early tokens (Gu et al., 2025a; Barbero et al., 2025). Transformer-XL, Compressive Transformer, and Recurrent Memory Transformer reuse or compress hidden states across segments (Dai et al., 2019; Rae et al., 2020; Bulatov et al., 2022).

These approaches mainly operate on language tokens or activations. Robot rollouts instead require complete visual–action blocks, since fragmenting a physical step can disrupt spatial coherence and action alignment. Unlike streaming language models, `ReCAP` preserves a semantic scene anchor together with recent dynamics and schedules complete blocks without architectural modification or additional training of the underlying world model.

3 Problem Formulation

We consider an action-conditioned robot video world model that receives an initial observation sequence $\mathbf{o}_{1:C}$ and a future action sequence $\mathbf{a}_{C:C+T-1}$, and generates future observations $\hat{\mathbf{o}}_{C+1:C+T}$. The visual tokenizer $\mathcal{E}_v$ and action tokenizer $\mathcal{E}_a$ map each observation and action to $\mathbf{z}_t = \mathcal{E}_v(\mathbf{o}_t)$ and $\mathbf{u}_t = \mathcal{E}_a(\mathbf{a}_t)$, respectively. We concatenate the aligned representations into a visual–action block $B_t = [\mathbf{z}_t; \mathbf{u}_t]$. A context policy $\mathcal{S}$ constructs the active inference context $C_t = \mathcal{S}(B_{1:t})$ from the history. The frozen autoregressive world model predicts the next visual representation, and the visual

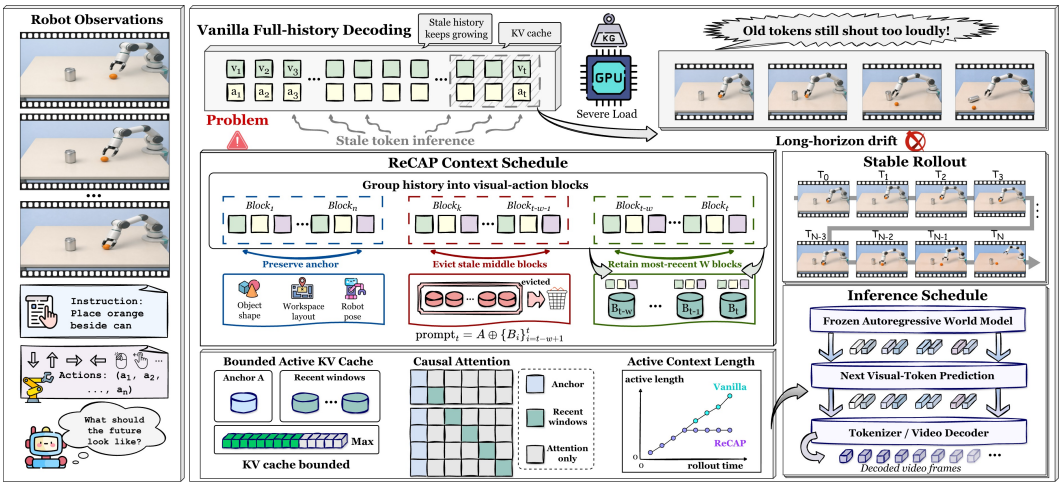

Figure 2: ***Overview of*** **ReCAP.** Full Context retains all generated states, with growing context cost and increasing prediction drift in the evaluated rollouts. **ReCAP** groups history into complete visual–action blocks, preserves the initial scene anchor and the most recent blocks, and evicts stale middle history. This training-free schedule keeps the active KV cache bounded while maintaining stable predictions throughout the evaluated autoregressive rollouts.

decoder reconstructs the corresponding frame:

$$\hat{\mathbf{z}}_{t+1} \sim p_\theta(\cdot \mid C_t), \qquad \hat{\mathbf{o}}_{t+1} = \mathcal{D}(\hat{\mathbf{z}}_{t+1}; C_0). \tag{1}$$

We seek a context policy that preserves accurate long-horizon prediction under a finite context budget while keeping all model parameters fixed:

$$\mathcal{S}^* = \arg\min_{\mathcal{S}} \mathbb{E}\left[\sum_{t=C}^{C+T-1} \ell(\hat{\mathbf{o}}_{t+1}, \mathbf{o}_{t+1})\right], \qquad \text{s.t.} \quad |C_t| \le M, \quad \theta \text{ fixed.} \tag{2}$$

Here, C_0 is the preserved visual context, ℓ measures prediction error, and M is the input-context budget, excluding the next frame's output tokens. This objective requires $\mathcal{S}$ to construct a compact and informative context from the visual–action history, balancing long-horizon prediction quality with the memory and computational costs of inference.

4 Methodology

Overview. Figure 2 illustrates **ReCAP**, a training-free context scheduler for pretrained robot video world models. At step t, Full Context uses $P_t^{\text{full}} = A \oplus B_{1:t}$, whereas **ReCAP** reorganizes this prompt by function: ❶ it preserves the scene anchor A from the real initial observations; ❷ it retains the most recent W complete visual–action blocks $\mathcal{R}_t = B_{\max(1,t-W+1):t}$; and ❸ it evicts stale history $O_t = B_{1:t-W}$ from the active context and KV cache. It therefore forms $P_t^{\text{ReCAP}} = A \oplus \mathcal{R}_t$. The scheduler operates blockwise in token space without changing model parameters or introducing additional pixel-space decoding and re-encoding of generated frames.

4.1 ReCAP Context Scheduling

Given the real initial observations $\mathbf{o}_{1:C}$, we retain their scene context C_0 together with the initial dynamics and action tokens as the fixed anchor $A = C_0 \oplus \mathbf{z}_0 \oplus \mathbf{u}_0$. Its visual representations come from real observations and contain no autoregressive prediction errors. The scene context provides stable information about robot morphology, object identity, and workspace layout. **ReCAP** preserves A throughout the rollout as a persistent reference for future prediction.

At generation step t, the model has produced visual–action blocks $B_{1:t}$, where $B_i = [\mathbf{z}_i; \mathbf{u}_i]$. **ReCAP** defines the most recent W blocks as the recent trajectory $\mathcal{R}_t = B_{\max(1,t-W+1):t}$ and the preceding blocks as stale history $O_t = B_{1:t-W}$; when $t \le W$, $O_t = \varnothing$. The recent trajectory describes the current robot state, object changes, and action evolution, whereas older generated history may offer

diminishing predictive value as its temporal distance grows. **ReCAP** therefore removes O_t from the active context and constructs the next inference prompt:

$$P_t^{\mathrm{ReCAP}} = \underbrace{A}_{\text{persistent scene anchor}} \oplus \underbrace{B_{\max(1,t-W+1):t}}_{\mathcal{R}_t:\text{ recent trajectory}}, \qquad B_i \coloneqq [\mathbf{z}_i; \mathbf{u}_i]. \tag{3}$$

where $\oplus$ concatenates token sequences in their original temporal order. This context retains stable global scene information and locally relevant dynamics while excluding generated blocks with indices $i \le t - W$. In the evaluated 1,931-token configuration, A contains 1,280 scene tokens plus an initial 93-token dynamics–action block, followed by $W = 6$ recent 93-token blocks. Figure 8 in Appendix A.1 shows this layout and the next-frame output.

ReCAP updates the recent window online. After the model produces a new block B_{t+1}, **ReCAP** appends it to the current trajectory and retains only the last W blocks according to $\mathcal{R}_{t+1} = \mathrm{Tail}_W(\mathcal{R}_t \oplus B_{t+1})$. Once the window reaches its maximum size, each new block evicts the oldest block in the window. This process schedules complete visual–action blocks and thus preserves within-frame visual structure, state–action alignment, and the original causal order. **ReCAP** operates directly in token and KV-cache spaces and never decodes generated tokens into images for re-encoding.

4.2 Bounded-Context Inference

At generation step t, **ReCAP** uses $P_t^{\mathrm{ReCAP}} = A \oplus \mathcal{R}_t$ as the active prompt and predicts the next visual state with the original causal factorization:

$$\hat{z}_{t+1,j} \sim p_\theta\big(\cdot \mid A, \mathcal{R}_t, \hat{\mathbf{z}}_{t+1,<j}\big), \qquad j = 1, \dots, n_v, \tag{4}$$

where n_v denotes the number of visual tokens per observation. After predicting all visual tokens, the model combines them with the aligned action tokens to form $B_{t+1} = [\hat{\mathbf{z}}_{t+1}; \mathbf{u}_{t+1}]$.

The cache retains entries associated with the selected anchor and recent blocks. Let $\mathrm{KV}_t[X]$ denote entries for block X in the active cache at step t, including their dependence on preceding context:

$$\mathrm{KV}_t^{\mathrm{ReCAP}} = \mathrm{KV}_t[A] \oplus \bigoplus_{i=\max(1,t-W+1)}^{t} \mathrm{KV}_t[B_i]. \tag{5}$$

For backend β with position indices $\pi_{t+1}^{(\beta)}$, the update is

$$\mathrm{KV}_{t+1}^{\mathrm{ReCAP}} = \mathrm{Update}_\beta\Big(\mathrm{KV}_t^{\mathrm{ReCAP}}, P_{t+1}^{\mathrm{ReCAP}}; \pi_{t+1}^{(\beta)}\Big). \tag{6}$$

Token selection alone does not imply unchanged cached activations. The BridgeV2 reference backend rebuilds KV from the retained prompt and rebases positions after eviction (Appendix B.6); cache management is separate from the block-selection algorithm.

Let n_A and n_B denote the token counts of the anchor and one visual–action block, respectively. Full Context and **ReCAP** use the following numbers of input tokens:

$$N_t^{\mathrm{full}} = n_A + t n_B, \qquad N_t^{\mathrm{ReCAP}} = n_A + \min(t, W) n_B \le n_A + W n_B. \tag{7}$$

For a Transformer with L layers, hidden dimension d, and b bytes per cached element, the corresponding footprint for the selected KV entries satisfies

$$M_{\mathrm{KV}}^{\mathrm{ReCAP}}(t) = 2Ldb\, N_t^{\mathrm{ReCAP}} \le 2Ldb\,(n_A + W n_B). \tag{8}$$

Once $t \ge W$, both the active prompt and KV cache remain bounded independently of the total rollout length. **ReCAP** preserves the original causal attention, model parameters, and architecture, and introduces no pixel-space decoding or re-encoding step. Algorithm 1 provides the complete inference procedure in the appendix, including the per-step context update.

5 Experiments

Research questions. We organize our experiments around four research questions:

RQ1: **Effectiveness and generality.** Does `ReCAP` consistently improve prediction quality across datasets and the different evaluated autoregressive model backbones?

RQ2: **Long-horizon robustness.** How effectively does `ReCAP` mitigate degradation over increasingly long autoregressive rollouts beyond the initial observation window?

RQ3: **Mechanism and components.** Why does `ReCAP` work, and how do the scene anchor, recent trajectory, and stale-history removal contribute?

RQ4: **Efficiency and scalability.** How does `ReCAP` trade prediction quality against context size, KV-cache memory, and inference cost?

5.1 Experimental Setup

▶ **Datasets and backbones.** We evaluate BridgeV2 (Walke et al., 2023), RT-1 (Brohan et al., 2023), LIBERO-90 (Liu et al., 2023a), and CALVIN (Mees et al., 2022) with RLVR-World (Wu et al., 2025), iVideoGPT (Wu et al., 2024), and RoboAlign-R1 (Wu et al., 2026). Appendices B.4–B.6 document dataset adaptation of the 127.1M RLVR-World transformer and its tokenizer, including splits, action mappings, training curves, and checkpoint selection. Adaptation precedes the scheduling comparison: each paired test uses the same frozen checkpoint. "Training-free" refers to applying `ReCAP`.

▶ **Evaluation metrics.** We evaluate generation quality from three complementary perspectives. PSNR, SSIM (Wang et al., 2004), and LPIPS (Zhang et al., 2018) measure visual fidelity. Arm Score, Shape IoU, and normalized dynamic time warping (nDTW) measure foreground shape and trajectory consistency (Appendix F). Qwen3-VL (Bai et al., 2025), InternVL3 (Zhu et al., 2025), and GPT-5.5 (OpenAI, 2026) further assess interaction quality, instruction following, task progress, and the naturalness of robot motion in generated videos.

▶ **Evaluation protocol and implementation.** Paired rollouts share initial observations, actions, and generation seeds. For the dedicated 200-pair RT-1 VLM study, judges score the same 16 sampled frames per video with a shared four-criterion rubric and hidden method identities (Appendix D, including the complete prompt). Horizon-AUC and component comparisons report paired bootstrap 95% intervals. The efficiency benchmark uses vLLM in BF16; the additional BridgeV2 study uses a separate FP32 PyTorch/Transformers backend. Appendix B details the hardware and implementation settings used for generation and resource measurements.

5.2 Overall Performance and Generality (RQ1)

Table 1: ***Context scheduling across GPT-style AR robot video world models.*** Comparisons pair identical episodes, sampling seeds, and action sequences. RoboAlign-R1 retains SWR in its baseline; RLVR-World and iVideoGPT use Full Context. Δ is the mean paired improvement. RQ1 uses maximum test horizons; later experiments use fixed horizons. Better values are in **bold**.

Context Strategy	BridgeV2			RT-1			LIBERO-90			CALVIN		
	LPIPS↓	nDTW↑	VLM↑	LPIPS↓	nDTW↑	VLM↑	LPIPS↓	nDTW↑	VLM↑	LPIPS↓	nDTW↑	VLM↑
RLVR-World												
Full Context	0.390	0.199	64.8	0.251	0.276	28.1	0.269	0.551	58.9	0.287	0.528	56.4
`ReCAP` **(Ours)**	**0.143**	**0.590**	**72.6**	**0.201**	**0.558**	**71.9**	**0.217**	**0.631**	**67.8**	**0.234**	**0.607**	**65.1**
Δ	−0.247	+0.391	+7.8	−0.050	+0.282	+43.8	−0.052	+0.080	+8.9	−0.053	+0.079	+8.7
iVideoGPT												
Full Context	0.254	0.587	61.3	0.267	0.561	58.6	0.283	0.529	55.7	0.301	0.506	53.2
`ReCAP` **(Ours)**	**0.203**	**0.665**	**70.1**	**0.215**	**0.639**	**67.6**	**0.231**	**0.609**	**64.5**	**0.248**	**0.586**	**61.9**
Δ	−0.051	+0.078	+8.8	−0.052	+0.078	+9.0	−0.052	+0.080	+8.8	−0.053	+0.080	+8.7
RoboAlign-R1												
SWR	0.226	0.634	67.2	0.239	0.608	64.5	0.255	0.576	61.8	0.272	0.554	59.1
`ReCAP` **(Ours)**	**0.178**	**0.709**	**75.8**	**0.190**	**0.683**	**73.1**	**0.206**	**0.654**	**70.2**	**0.222**	**0.634**	**67.4**
Δ	−0.048	+0.075	+8.6	−0.049	+0.075	+8.6	−0.049	+0.078	+8.4	−0.050	+0.080	+8.3

Quantitative results. Table 1 shows that `ReCAP` improves over the corresponding baselines in all 36 dataset–backbone–metric combinations. With RLVR-World on BridgeV2, `ReCAP` reduces LPIPS from 0.390 to 0.143 and improves nDTW from 0.199 to 0.590. On RT-1, it increases nDTW and the VLM score by 0.282 and 43.8, respectively. The consistent gains with iVideoGPT and RoboAlign-R1 further demonstrate its generality across models and environments. An additional 80-pair human comparison favors `ReCAP` in 42 cases, with 14 ties and 24 Full Context wins. Three-way

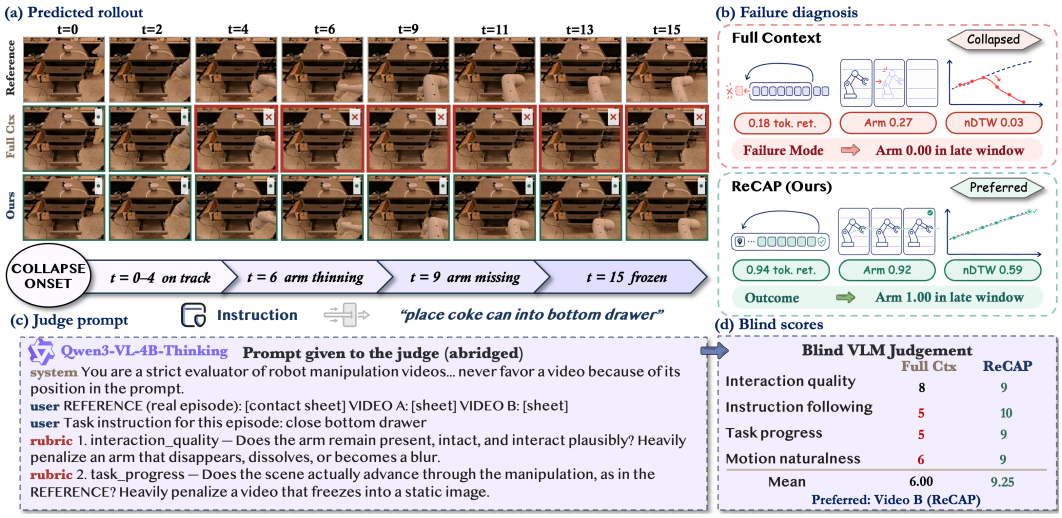

Figure 3: ***Qualitative case study.*** `ReCAP` preserves the robot arm throughout the rollout, whereas Full Context collapses; the blind VLM judge prefers `ReCAP`. The displayed A/B prompt differs from the per-video RT-1 protocol in Appendix D.

VLM agreement with human majorities is 66.3–72.5% under the original protocol and 72.5–77.5% under the combined revision, whose tie rates approach the human rate. The aggregate preference persists across tested variants, including removal of the static penalty (Appendices D.6–D.7). These descriptive results provide limited human corroboration.

Qualitative results. Figure 3 qualitatively illustrates the long-horizon improvement. Full Context begins to thin the robot arm at $t = 6$, loses it at $t = 9$, and freezes by $t = 15$, whereas `ReCAP` preserves the robot structure and task progress under the same initial observation, action sequence, and sampling seed. Accordingly, the Arm Score increases from 0.27 to 0.92, nDTW from 0.03 to 0.59, and the blind VLM mean score from 6.00 to 9.25. Appendix C adds RT-1, CALVIN, and LIBERO-90 rollouts and selected high-gain BridgeV2 examples with 32 predicted frames; these illustrate behavior rather than population-average gains.

5.3 Long-Horizon Robustness (RQ2)

We evaluate whether `ReCAP` mitigates error accumulation during long autoregressive rollouts. We generate 64 frames from identical initial observations and ground-truth action sequences, and evaluate prediction quality at $H = \{8, 16, 32, 48, 64\}$.

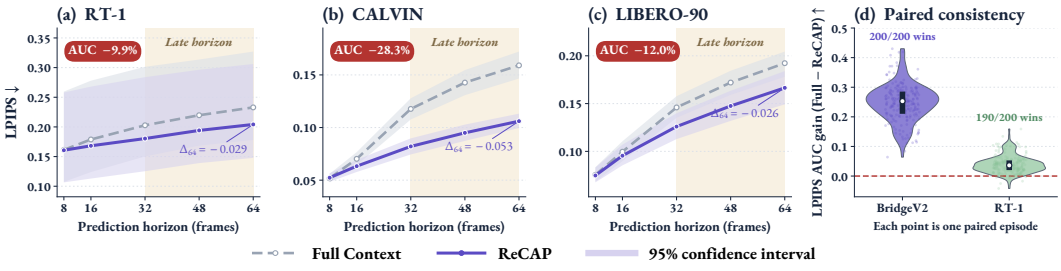

Figure 4: `ReCAP` ***improves long-horizon robustness.*** (a–c) LPIPS across prediction horizons on three datasets; shaded bands denote 95% confidence intervals. (d) Paired LPIPS AUC gains on independent BridgeV2 and RT-1 evaluation sets, where positive values favor `ReCAP`.

Quantitative results. Figure 4(a–c) shows a small performance gap at short horizons and an increasingly clear advantage for `ReCAP` as the rollout grows. `ReCAP` reduces the LPIPS curve AUC by 9.9%, 28.3%, and 12.0% on RT-1, CALVIN, and LIBERO-90, respectively. At $H = 64$, it lowers LPIPS from 0.233, 0.159, and 0.192 to 0.204, 0.106, and 0.166, while improving SSIM on all three datasets. Table 2 further shows that every 95% confidence interval excludes zero. Figure 4(d) uses

Table 2: ***Long-horizon prediction quality under paired rollouts.*** AUC gains are reported with 95% confidence intervals; positive values favor `ReCAP`. Results at $H = 64$ are shown as Full Context → `ReCAP`, and win rates are reported for LPIPS/SSIM.

Dataset	LPIPS AUC Gain ↑	SSIM AUC Gain ↑	H=64 LPIPS ↓	H=64 SSIM ↑	Win Rate
RT-1	+0.0202 [0.0136, 0.0270]	+0.0207 [0.0131, 0.0286]	0.233 → **0.204**	0.703 → **0.732**	100%/100%
CALVIN	+0.0328 [0.0245, 0.0407]	+0.0222 [0.0159, 0.0287]	0.159 → **0.106**	0.833 → **0.867**	100%/100%
LIBERO-90	+0.0173 [0.0120, 0.0229]	+0.0192 [0.0145, 0.0241]	0.192 → **0.166**	0.798 → **0.826**	100%/100%

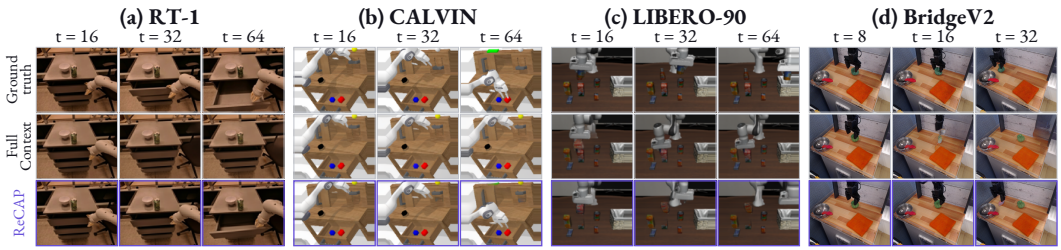

Figure 5: ***Qualitative comparisons across four datasets.*** Selected cases from Appendix C, with three prediction times per dataset: $t = 16, 32, 64$ for RT-1, CALVIN, and LIBERO-90; $t = 8, 16, 32$ for the shorter BridgeV2 protocol. Rows show ground truth, Full Context, and `ReCAP`.

separate BridgeV2 and RT-1 paired evaluation sets rather than the test sets in panels (a–c), and its gains remain concentrated above zero.

Qualitative results. Figure 5 compares matched rollouts on all four datasets. These selected examples illustrate better preservation of robot structure and scene evolution under `ReCAP`. BridgeV2 uses the separate 32-frame protocol in Appendix B.6; the displayed cases do not estimate average prediction quality across the full evaluation population.

5.4 Mechanism and Components (RQ3)

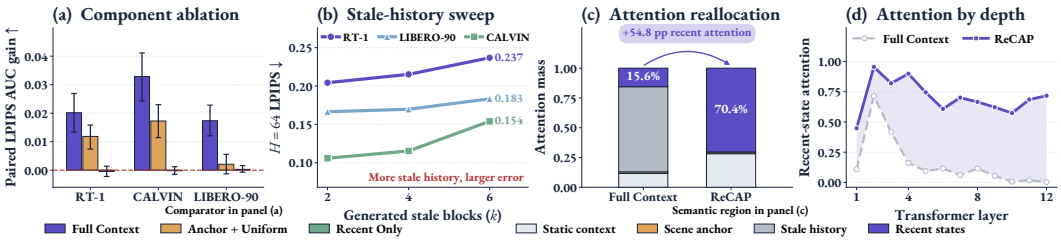

Figure 6: ***Mechanism analysis of*** `ReCAP`**.** (a) Component ablation with 95% bootstrap confidence intervals. (b) H=64 LPIPS after reinserting k stale blocks. (c) Layer-averaged semantic attention and (d) recent-state attention by layer on CALVIN C06 at H=64.

At a fixed budget, *Anchor + Uniform* retains the scene anchor and uniformly samples history; *Recent Only* retains recent states without the anchor.

Component ablation. In Figure 6(a), paired LPIPS AUC improves over Full Context by 0.0202, 0.0328, and 0.0173 on RT-1, CALVIN, and LIBERO-90, and over Anchor + Uniform by 0.0118, 0.0173, and 0.0021. This favors contiguous recent states over uniform history sampling. Gains over Recent Only remain within ±0.0005, with confidence intervals including zero: the anchor adds no measurable LPIPS AUC benefit in this comparison.

Scene-prefix allocation. Using the final 20k LIBERO-90 checkpoint on 450 held-out episodes and five seeds, expanding the scene prefix from 4 to 1,280 tokens under a 1,931-token cap shortens the recent window from 19 to 6 complete blocks. Late LPIPS improves from 0.208 to 0.203 and nDTW from 0.619 to 0.628, with paired 95% CIs $[-0.007, -0.003]$ and $[+0.005, +0.013]$ for the differences. This tests joint scene/history allocation using different metrics from the AUC ablation above. Appendix E reports effective lengths and all comparisons; the shared cap does not give identical counts of tokens actually retained at inference.

Stale-history intervention. Reinserting $k = 2$ to 6 stale blocks monotonically raises H=64 LPIPS from 0.204 to 0.237 on RT-1, 0.106 to 0.154 on CALVIN, and 0.166 to 0.183 on LIBERO-90 (Fig. 6b). Reinsertion also lengthens the context, so it does not isolate stale content from effects induced by context length and token positions.

Attention analysis. On CALVIN C06 at H=64, Full Context assigns 71.5% of attention to stale history and 15.6% to recent states. Under `ReCAP`, recent-state attention is 70.4%, rising in all 12 layers, while anchor attention changes from 1.37% to 1.45% (Fig. 6c–d). Removing tokens also changes attention normalization. Moreover, the 1,931-token inference budget matches the adaptation sequence length (Appendix B). These observations support the memory-role-conflict hypothesis but do not separate it from length, position, and cache-update effects.

5.5 Efficiency and Scalability (RQ4)

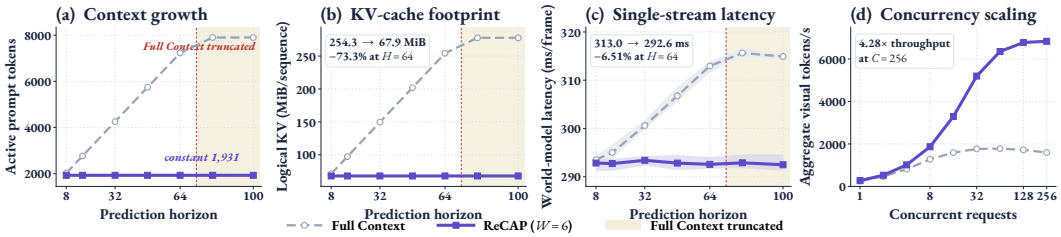

Figure 7: `ReCAP` ***bounds context cost and improves concurrent throughput.*** (a) Prompt tokens, (b) logical BF16 KV, (c) p50 world-model latency, and (d) concurrent throughput. Shading marks Full Context truncation; bands in (c) show the range across three datasets.

Efficiency protocol. We use one dedicated H20 to run a persistent vLLM 0.13.0 BF16 engine in eager/`-O0` mode, with `max_model_len`=8,192 and 80 output tokens reserved per frame. `ReCAP` uses W=6. Automatic prefix caching is disabled here and evaluated separately. Each single-stream measurement uses two warm-up and five measured runs with tokenizer-produced prompts. Latency covers world-model generation only; logical KV excludes allocator overhead and the storage used by activations and runtime workspace.

Table 3: ***System efficiency summary.*** H=64 values are macro averages; throughput uses C=256.

Metric	Full	`ReCAP`	Change
Tokens ↓	7,232	**1,931**	−73.3%
Logical KV (MiB) ↓	254.25	**67.89**	−73.3%
p50 (ms) ↓	312.95	**292.57**	−6.51%
C=256 throughput ↑	1,596	**6,834**	4.28×

Bounded cost. At H=64, `ReCAP` reduces the active prompt from 7,232 to 1,931 tokens and logical KV from 254.25 to 67.89 MiB, a 73.3% reduction (Table 3). It also lowers p50 latency from 312.95 to 292.57 ms/frame (−6.51%). Beginning after rollout step 72, Full Context is truncated to at most 7,900 prompt tokens; at H=100, `ReCAP` remains bounded at 1,931 tokens (Figure 7(a–c)).

Concurrent scaling. The throughput advantage grows from 1.07× at one request to 2.94× at 32 requests and 4.28× at 256 requests (Figure 7(d)). Neither method fails within the tested range; thus, 256 denotes the largest tested concurrency rather than a hard capacity limit.

6 Conclusion

We presented `ReCAP`, a training-free context scheduler that preserves the initial anchor and recent complete visual–action blocks. Across evaluated datasets and backbones, it improves prediction over the corresponding baselines. Prefix-allocation and human–VLM analyses provide complementary evidence under their respective protocols. History and attention analyses are consistent with memory-role conflict, while context-length and backend effects remain unresolved. At H = 64, `ReCAP` reduces logical KV use by 73.3% and achieves 4.28× throughput at 256 concurrent requests. These results support efficient video prediction through context scheduling. Future work should isolate content, position, and cache effects with matched controls, and test adaptive context allocation in closed-loop robot tasks that require repeated prediction and action selection.

AI USE STATEMENT

Generative AI tools assist with drafting, translation, language editing, code preparation and debugging, checking mathematical descriptions against implementations, and preparing figures and document layouts. They also support discussion of experimental methodology and interpretation of results. Separately, vision–language models serve as experimental evaluators; Appendix D documents their inputs, prompts, preprocessing, aggregation, and limitations. The authors take responsibility for the final text, claims, code, and other artifacts, including material prepared with AI assistance.

REPRODUCIBILITY STATEMENT

Appendices A and B specify the inference algorithm, token organization, runtime environment, and adaptation settings. Appendices D and F define the evaluation protocols and metrics. The anonymous code package provides reference implementations, configurations, evaluation scripts, and qualitative examples. Appendix G describes its coverage and separate checkpoint and data dependencies.

ETHICS STATEMENT

This work studies offline robot-video prediction using existing datasets. Improved prediction or evaluator scores do not establish safe closed-loop robot control. Human ratings are provided by authors or members of our research group and complement, rather than independently validate, the model-based evaluation. Data and pretrained models remain subject to their original terms. Appendix G discusses evaluation scope, limitations, and responsible use.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025. URL https://arxiv.org/abs/2511.21631.

Federico Barbero, Álvaro Arroyo, Xiangming Gu, Christos Perivolaropoulos, Michael Bronstein, Petar Veličković, and Razvan Pascanu. Why do LLMs attend to the first token? *arXiv preprint arXiv:2504.02732*, 2025. URL https://arxiv.org/abs/2504.02732.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*, pp. 1171–1179, 2015. URL https://proceedings.neurips.cc/paper/2015/hash/e995f98d56967d946471af29d7bf99f1-Abstract.html.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alexander Herzog, Jasmine Hsu, Julian Ibarz, Brian Ichter, Alex Irpan, Tomas Jackson, Sally Jesmonth, Nikhil Joshi, Ryan Julian, Dmitry Kalashnikov, Yuheng Kuang, Isabel Leal, Kuang-Huei Lee, Sergey Levine, Yao Lu, Utsav Malla, Deeksha Manjunath, Igor Mordatch, Ofir Nachum, Carolina Parada, Jodilyn Peralta, Emily Perez, Karl Pertsch, Jornell Quiambao, Kanishka Rao, Michael S Ryoo, Grecia Salazar, Pannag R Sanketi, Kevin Sayed, Jaspiar Singh, Sumedh Sontakke, Austin Stone, Clayton Tan, Huong Tran, Vincent Vanhoucke, Steve Vega, Quan H Vuong, Fei Xia, Ted Xiao, Peng Xu, Sichun Xu, Tianhe Yu, and Brianna Zitkovich. RT-1: Robotics transformer for real-world control at scale. In *Proceedings of Robotics: Science and Systems*, 2023. doi: 10.15607/RSS.2023.XIX.025. URL https://roboticsproceedings.org/rss19/p025.html.

Jake Bruce, Michael D Dennis, Ashley Edwards, Jack Parker-Holder, Yuge Shi, Edward Hughes, Matthew Lai, Aditi Mavalankar, Richie Steigerwald, Chris Apps, Yusuf Aytar, Sarah Maria Elisabeth Bechtle, Feryal Behbahani, Stephanie C.Y. Chan, Nicolas Heess, Lucy Gonzalez, Simon Osindero, Sherjil Ozair, Scott Reed, Jingwei Zhang, Konrad Zolna, Jeff Clune, Nando De Freitas, Satinder Singh, and Tim Rocktäschel. Genie: Generative interactive environments. In *Forty-first International Conference on Machine Learning*, 2024. URL https://proceedings.mlr.press/v235/bruce24a.html.

Aydar Bulatov, Yuri Kuratov, and Mikhail S. Burtsev. Recurrent memory transformer. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/47e288629a6996a17ce50b90a056a0e1-Abstract.html.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 2978–2988, Florence, Italy, 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1285. URL https://aclanthology.org/P19-1285/.

Frederik Ebert, Chelsea Finn, Sudeep Dasari, Annie Xie, Alex Lee, and Sergey Levine. Visual foresight: Model-based deep reinforcement learning for vision-based robotic control. *arXiv preprint arXiv:1812.00568*, 2018. URL https://arxiv.org/abs/1812.00568.

Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12873–12883, 2021. URL https://openaccess.thecvf.com/content/CVPR2021/html/Esser_Taming_Transformers_for_High-Resolution_Image_Synthesis_CVPR_2021_paper.html.

Chelsea Finn, Ian Goodfellow, and Sergey Levine. Unsupervised learning for physical interaction through video prediction. In *Advances in neural information processing systems*, volume 29, 2016. URL https://proceedings.neurips.cc/paper/2016/hash/d9d4f495e875a2e075a1a4a6e1b9770f-Abstract.html.

Suyu Ge, Yunan Zhang, Liyuan Liu, Minjia Zhang, Jiawei Han, and Jianfeng Gao. Model tells you what to discard: Adaptive KV cache compression for LLMs. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/639a9a172c044fbb64175b5fad42e9a5-Abstract-Conference.html.

Xiangming Gu, Tianyu Pang, Chao Du, Qian Liu, Fengzhuo Zhang, Cunxiao Du, Ye Wang, and Min Lin. When attention sink emerges in language models: An empirical view. In *International Conference on Learning Representations (ICLR)*, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/f1b04face60081b689ba740d39ea8f37-Abstract-Conference.html.

Yuchao Gu, Weijia Mao, and Mike Zheng Shou. Long-context autoregressive video modeling with next-frame prediction. *arXiv preprint arXiv:2503.19325*, 2025b. URL https://arxiv.org/abs/2503.19325.

David Ha and Jürgen Schmidhuber. World models. *arXiv preprint arXiv:1803.10122*, 2018. URL https://arxiv.org/abs/1803.10122.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023. URL https://arxiv.org/abs/2301.04104.

Xun Huang, Zhengqi Li, Guande He, Mingyuan Zhou, and Eli Shechtman. Self forcing: Bridging the train-test gap in autoregressive video diffusion. In *Advances in Neural Information Processing Systems*, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/f4823f831af67a3ef15e41a85434422a-Abstract-Conference.html.

Yuhong Li, Yingbing Huang, Bowen Yang, Bharat Venkitesh, Acyr Locatelli, Hanchen Ye, Tianle Cai, Patrick Lewis, and Deming Chen. SnapKV: LLM knows what you are looking for before generation. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/28ab418242603e0f7323e54185d19bde-Abstract-Conference.html.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023a. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/8c3c666820ea055a77726d66fc7d447f-Abstract-Datasets_and_Benchmarks.html.

Zichang Liu, Aditya Desai, Fangshuo Liao, Weitao Wang, Victor Xie, Zhaozhuo Xu, Anastasios Kyrillidis, and Anshumali Shrivastava. Scissorhands: Exploiting the persistence of importance hypothesis for LLM KV cache compression at test time. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023b. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/a452a7c6c463e4ae8fbdc614c6e983e6-Abstract.html.

Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A benchmark for language-conditioned policy learning for long-horizon robot manipulation tasks. *IEEE Robotics and Automation Letters*, 7(3):7327–7334, 2022. URL https://calvin.cs.uni-freiburg.de/.

Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: VQ-VAE made simple. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/e2dd53601de57c773343a7cdf09fae1c-Abstract-Conference.html.

Vincent Micheli, Eloi Alonso, and François Fleuret. Transformers are sample-efficient world models. In *International Conference on Learning Representations (ICLR)*, 2023. URL https://fleuret.org/papers/micheli-et-al-iclr2023.pdf.

OpenAI. GPT-5.5 system card. https://openai.com/index/gpt-5-5-system-card/, April 2026. Accessed: 2026-08-06.

Ryan Po, Yotam Nitzan, Richard Zhang, Berlin Chen, Tri Dao, Eli Shechtman, Gordon Wetzstein, and Xun Huang. Long-context state-space video world models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8733–8744, 2025. URL https://openaccess.thecvf.com/content/ICCV2025/html/Po_Long-Context_State-Space_Video_World_Models_ICCV_2025_paper.html.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*, 2020. URL https://iclr.cc/virtual_2020/poster_SylKikSYDH.html.

Zhongwei Ren, Yunchao Wei, Xun Guo, Yao Zhao, Bingyi Kang, Jiashi Feng, and Xiaojie Jin. VideoWorld: Exploring knowledge learning from unlabeled videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 29029–29039, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Ren_VideoWorld_Exploring_Knowledge_Learning_from_Unlabeled_Videos_CVPR_2025_paper.html.

Yu Shang, Zhuohang Li, Yiding Ma, Weikang Su, Xin Jin, Ziyou Wang, Lei Jin, Xin Zhang, Yinzhou Tang, Haisheng Su, Chen Gao, Wei Wu, Xihui Liu, Dhruv Shah, Zhaoxiang Zhang, Zhibo Chen, Jun Zhu, Yonghong Tian, Tat-Seng Chua, Wenwu Zhu, and Yong Li. WorldArena: A unified benchmark for evaluating perception and functional utility of embodied world models. *arXiv preprint arXiv:2602.08971*, 2026. URL https://arxiv.org/abs/2602.08971.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*,

2017. URL https://proceedings.neurips.cc/paper/2017/hash/7a98af17e63a0ac09ce2e96d03992fbc-Abstract.html.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/3f5ee243547dee91fbd053c1c4a845aa-Abstract.html.

Homer Walke, Kevin Black, Abraham Lee, Moo Jin Kim, Max Du, Chongyi Zheng, Tony Zhao, Philippe Hansen-Estruch, Quan Vuong, Andre He, Vivek Myers, Kuan Fang, Chelsea Finn, and Sergey Levine. BridgeData V2: A dataset for robot learning at scale. *arXiv preprint arXiv:2308.12952*, 2023. URL https://arxiv.org/abs/2308.12952.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing*, 13(4):600–612, 2004.

Hao Wu, Yuqi Li, Yuan Gao, Fan Xu, Fan Zhang, Kun Wang, Penghao Zhao, Qiufeng Wang, Yizhou Zhao, Weiyan Wang, Yingli Tian, Xian Wu, and Xiaomeng Huang. RoboAlign-R1: Distilled multimodal reward alignment for robot video world models. *arXiv preprint arXiv:2605.03821*, 2026. URL https://arxiv.org/abs/2605.03821.

Jialong Wu, Shaofeng Yin, Ningya Feng, Xu He, Dong Li, Jianye Hao, and Mingsheng Long. iVideoGPT: Interactive VideoGPTs are scalable world models. In *Advances in Neural Information Processing Systems*, volume 37, pp. 68082–68119, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/7dbb5bfab324e3b86af9bd0df15498dd-Abstract-Conference.html.

Jialong Wu, Shaofeng Yin, Ningya Feng, and Mingsheng Long. RLVR-World: Training world models with reinforcement learning. *arXiv preprint arXiv:2505.13934*, 2025. URL https://arxiv.org/abs/2505.13934.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. Efficient streaming language models with attention sinks. In *International Conference on Learning Representations (ICLR)*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/5e5fd18f863cbe6d8ae392a93fd271c9-Abstract-Conference.html.

Wilson Yan, Yunzhi Zhang, Pieter Abbeel, and Aravind Srinivas. VideoGPT: Video generation using VQ-VAE and transformers. *arXiv preprint arXiv:2104.10157*, 2021. URL https://arxiv.org/abs/2104.10157.

Lvmin Zhang, Shengqu Cai, Muyang Li, Gordon Wetzstein, and Maneesh Agrawala. Frame context packing and drift prevention in next-frame-prediction video diffusion models. In *Advances in Neural Information Processing Systems*, 2025. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/2bde8fef08f7ebe42b584266cbcfc909-Abstract-Conference.html.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 586–595, 2018.

Zhenyu Zhang, Ying Sheng, Tianyi Zhou, Tianlong Chen, Lianmin Zheng, Ruisi Cai, Zhao Song, Yuandong Tian, Christopher Ré, Clark Barrett, Zhangyang Wang, and Beidi Chen. H2O: Heavy-hitter oracle for efficient generative inference of large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/6ceefa7b15572587b78ecfcebb2827f8-Abstract.html.

Siyuan Zhou, Yilun Du, Jiaben Chen, Yandong Li, Dit-Yan Yeung, and Chuang Gan. RoboDreamer: Learning compositional world models for robot imagination. *arXiv preprint arXiv:2404.12377*, 2024. URL https://arxiv.org/abs/2404.12377.

Jinguo Zhu, Weiyun Wang, Zhe Chen, Zhaoyang Liu, Shenglong Ye, Lixin Gu, Hao Tian, Yuchen Duan, Weijie Su, Jie Shao, Zhangwei Gao, Erfei Cui, Xuehui Wang, Yue Cao, Yangzhou Liu, Xingguang Wei, Hongjie Zhang, Haomin Wang, Weiye Xu, Hao Li, Jiahao Wang, Nianchen Deng, Songze Li, Yinan He, Tan Jiang, Jiapeng Luo, Yi Wang, Conghui He, Botian Shi, Xingcheng Zhang, Wenqi Shao, Junjun He, Yingtong Xiong, Wenwen Qu, Peng Sun, Penglong Jiao, Han Lv, Lijun Wu, Kaipeng Zhang, Huipeng Deng, Jiaye Ge, Kai Chen, Limin Wang, Min Dou, Lewei Lu, Xizhou Zhu, Tong Lu, Dahua Lin, Yu Qiao, Jifeng Dai, and Wenhai Wang. InternVL3: Exploring advanced training and test-time recipes for open-source multimodal models. *arXiv preprint arXiv:2504.10479*, 2025. URL https://arxiv.org/abs/2504.10479.

A RECAP INFERENCE ALGORITHM

A.1 TOKEN ORGANIZATION

Figure 8 shows the evaluated 1,931-token input: a 1,373-token fixed prefix A_0 (the anchor A in Equation 3) and six recent visual–action blocks. The next frame's 80 output tokens are excluded from this input budget.

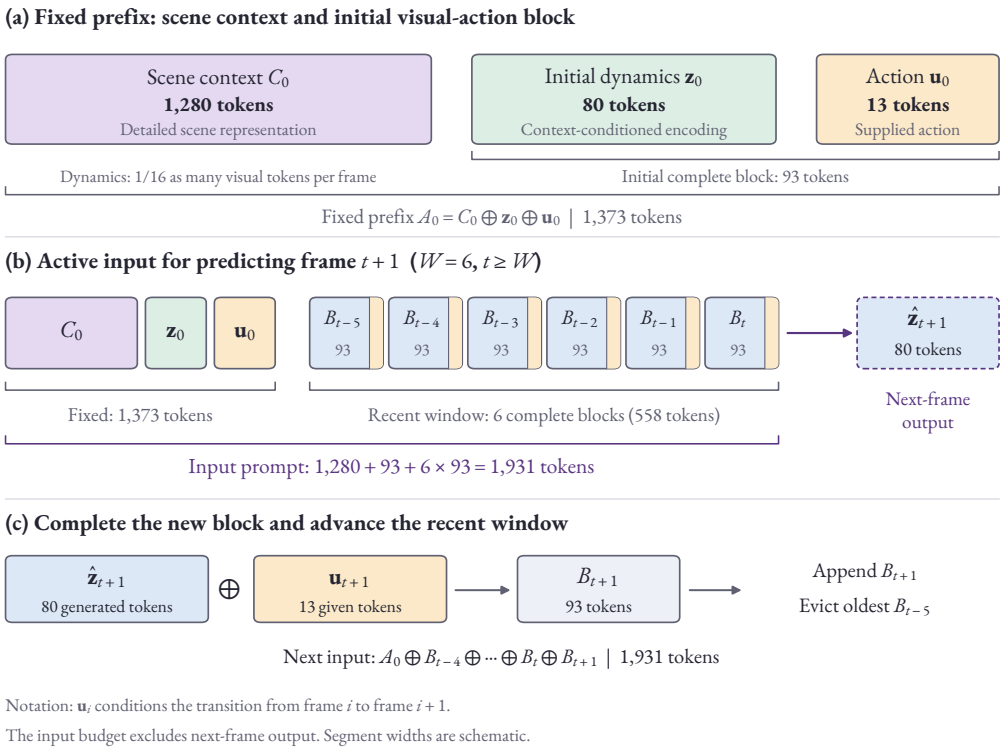

Figure 8: **Token organization for the 1,931-token ReCAP configuration.** (a) The fixed prefix comprises 1,280 scene-context tokens, 80 initial dynamics tokens, and 13 supplied action tokens, totaling 1,373 tokens. Each context-conditioned dynamics representation uses one-sixteenth as many visual tokens as the context representation. (b) Six recent complete visual–action blocks contribute 558 tokens, giving a 1,931-token input before next-frame generation. The next 80 visual tokens are outputs and are excluded from this input budget. (c) Before a further prediction, the generated visual tokens are paired with the next supplied action to complete a 93-token block, and the oldest recent block is evicted. Here, action $\mathbf{u}_i$ conditions the transition from frame i to frame $i+1$. Segment widths are schematic.

A.2 Inference Procedure

Algorithm 1 uses frozen model and decoder interfaces. The `anchor` is the complete fixed prefix A_0, including the initial action block; subsequent action blocks follow their aligned visual blocks. The decoder uses the preserved visual context C_0.

Algorithm 1 `ReCAP` inference

```python
def recap(anchor, actions, model, decoder, W, n_visual):
    # actions[t] follows predicted visual block t + 1.
    # anchor already includes the initial action block.
    assert W >= 1
    recent = []
    rollout = []

    for action in actions:
        # Keep the fixed anchor and recent complete blocks.
        prompt = list(anchor)
        for block in recent:
            prompt.extend(block)

        # Autoregressively sample one frame's visual tokens.
        visual = []
        for _ in range(n_visual):
            token = model.sample_next(prompt + visual)
            visual.append(token)

        rollout.append(decoder.decode(visual, context=anchor))
        block = visual + list(action)
        recent.append(block)
        recent = recent[-W:]  # Evict the oldest whole block.

    return rollout
```

Predicted visual tokens enter the context directly; decoded frames are outputs and are never re-encoded. The algorithm specifies token selection and generation order. Backend-specific KV updates, including cache rebuilding and position rebasing after eviction in the BridgeV2 implementation, are described in Appendix B.6.

B Implementation and Systems Details

B.1 Hardware Environment

Model execution uses a server with eight NVIDIA H20 GPUs (95.6 GiB each) and two AMD EPYC 9K84 96-core CPUs. The RQ4 systems benchmark reserves one H20 exclusively; Ray handles job scheduling only.

B.2 Software Environment

Table 4: ***Software environment used for model execution and evaluation.*** Versions marked "metrics" refer to the separate metric-evaluation environment.

Component	Version
Python	3.13.11
PyTorch / torchvision	2.9.1 / 0.24.1+d801a34
CUDA (PyTorch build) / cuDNN	12.9 / 9.1.6.0
NCCL	2.27.3
Transformers	4.57.1; 4.57.1.post1 (metrics)
vLLM	0.13.0
Ray	2.51.1 (scheduling only)
Accelerate	1.9.0
PEFT	0.16.0; 0.17.1 (metrics)
safetensors / flashinfer	0.7.0 / 0.5.3
lpips / piqa	0.1.4 (VGG) / 1.3.2
NumPy / SciPy	1.26.4 / 1.17.1
Pillow / imageio	12.0.0 / 2.37.0
matplotlib	3.11.1 (rendering only)

B.3 World-Model Serving and Context Limits

The adapted world models share the RLVR-World RT-1 architecture: 12 layers, hidden size 768, 12 attention heads, feed-forward size 3,072, vocabulary size 9,008, and maximum context length 8,192 (127.1M parameters). For the efficiency benchmark, we use one persistent vLLM 0.13.0 engine in BF16 with eager execution (optimization level `-O0`, CUDA graphs disabled). We reserve 80 tokens for each frame's visual output, cap the Full Context prompt at 7,900 tokens, and treat it as truncated after rollout step 72.

For **`ReCAP`**, the scene context contains 1,280 tokens and the fixed prefix also retains the initial 93-token visual–action block (Figure 8). Each recent block contains 93 tokens, and the recent window uses $W = 6$. Its active prompt is consequently bounded by

$$1280 + 93(1 + W) = 1{,}931 \text{ tokens}. \tag{9}$$

Automatic prefix caching is disabled for the main efficiency results. We evaluate the APC-enabled configuration separately as a deployment-sensitivity experiment.

We report the analytical logical BF16 KV footprint rather than allocated GPU memory. For the evaluated model, the per-token footprint is

$$2 \times 12 \times 12 \times 64 \times 2 = 36{,}864 \text{ bytes/token}, \tag{10}$$

where the factors correspond to key/value tensors, Transformer layers, KV heads, head dimension, and bytes per BF16 element. This calculation excludes KV-block rounding, allocator metadata, activations, and runtime workspace.

B.4 CALVIN Adaptation and Training Details

Dataset and preprocessing. We use the official CALVIN D-to-D split (`task_D_D`) (Mees et al., 2022): 31 training rollouts (512,077 time steps) and four validation rollouts (99,022). Each rollout is partitioned into non-overlapping clips of at most 128 frames, discarding remainders shorter than eight frames. This yields 4,013 training clips (512,072 frames) and 776 validation clips (99,022 frames). Training samples contiguous eight-frame segments at stride one, without crossing rollout boundaries.

Static-camera images (200×200) are resized to 256×256, padded by replicating 32 pixels at each horizontal edge to obtain 256×320, and scaled to $[0, 1]$. No image augmentation is applied.

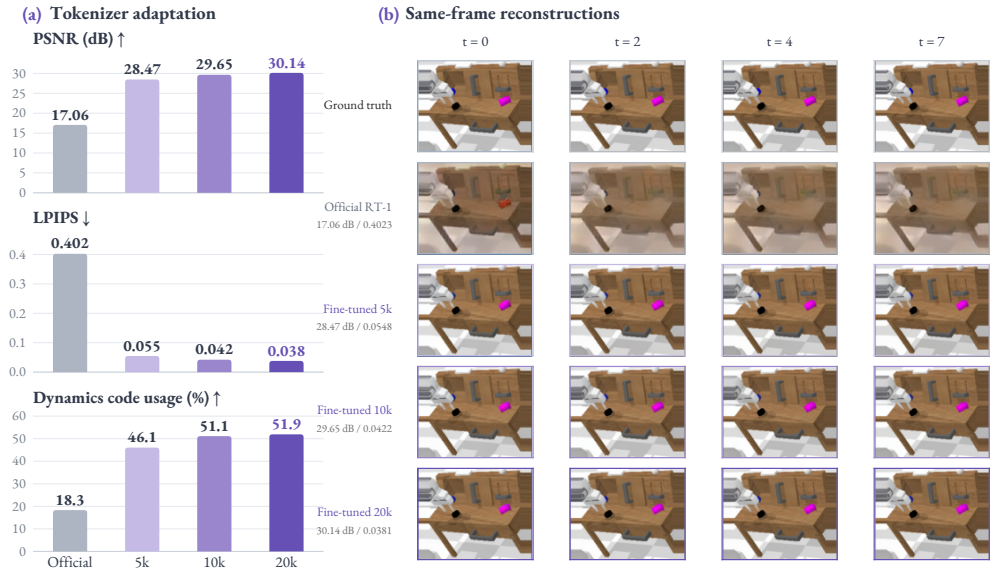

Figure 9: CALVIN tokenizer adaptation. (a) Reconstruction PSNR, LPIPS, and dynamics-code utilization measured on the same fixed set of 32 validation clips (256 frames) for the released RT-1 tokenizer and the 5k, 10k, and 20k checkpoints. (b) Same-frame reconstructions at $t \in \{0, 2, 4, 7\}$. The adapted tokenizer progressively removes the color shift and object disappearance observed with the unadapted RT-1 tokenizer.

Action conversion. The seven-dimensional normalized relative action is mapped to 13 slots: translation in slots 4–6, rotation in 10–12, gripper in 3, and zeros in 0–2 and 7–9 (zero-based indexing). Each dimension uses 256 uniform bins, with ranges estimated from training actions only and expanded by a 2% margin: $[-1.04, 1.04]$ for active dimensions and $[-1, 1]$ for unused slots. Each visual–action block thus contains 80 dynamics tokens and 13 action tokens.

Stage 1: visual-tokenizer adaptation. We adapt the released RLVR-World RT-1 compressive FSQ tokenizer (Wu et al., 2025). It uses a 32×40 context grid (1,280 tokens), an 8×10 dynamics grid (80 tokens), and 4,375 codes per stream. Eight-frame segments contain one context frame and seven dynamics frames. Table 5 gives optimization settings; AdamW uses $(\beta_1, \beta_2) = (0.9, 0.999)$ and $\epsilon = 10^{-8}$, with seed 20260903. The objective combines unit-weight L1 and LPIPS terms with context/dynamics FSQ commitment losses. Context and dynamics contributions are weighted by temporal occupancy ($1/8$ and $7/8$). Adversarial training is disabled.

We evaluate checkpoints on 32 fixed, non-overlapping validation clips (eight per rollout; 256 frames). The selected 20k tokenizer improves PSNR, SSIM, and LPIPS from 17.064 dB, 0.7520, and 0.4023 to 30.138 dB, 0.9457, and 0.0381; dynamics-code utilization rises from 18.3% to 51.9% (Figure 9).

Stage 2: action-conditioned world-model adaptation. We freeze the 20k tokenizer and adapt the released RT-1 world model using causal next-token cross-entropy. Each eight-frame training sequence contains 1,280 context tokens and seven 93-token visual–action blocks (1,931 tokens). Optimization settings are listed in Table 5; AdamW coefficients match Stage 1. Online validation uses two mini-batches per evaluation.

All CALVIN rollouts use the final 20k checkpoint, following the specified training budget. Monitored validation loss decreases from 2.3010 to 0.7809 (minimum 0.7611 near 19k), and perplexity from 9.9837 to 2.1835. Figure 10 shows the training curves and a held-out open-loop rollout.

B.5 LIBERO-90 Adaptation and Training Details

Dataset and task-level split. LIBERO-90 (Liu et al., 2023a) contains 90 tasks with 50 demonstrations each. We lexicographically sort the task files and hold out nine evenly spaced tasks, yielding 81 training tasks (4,050 episodes; 596,637 time steps) and nine validation tasks (450 episodes; 72,406

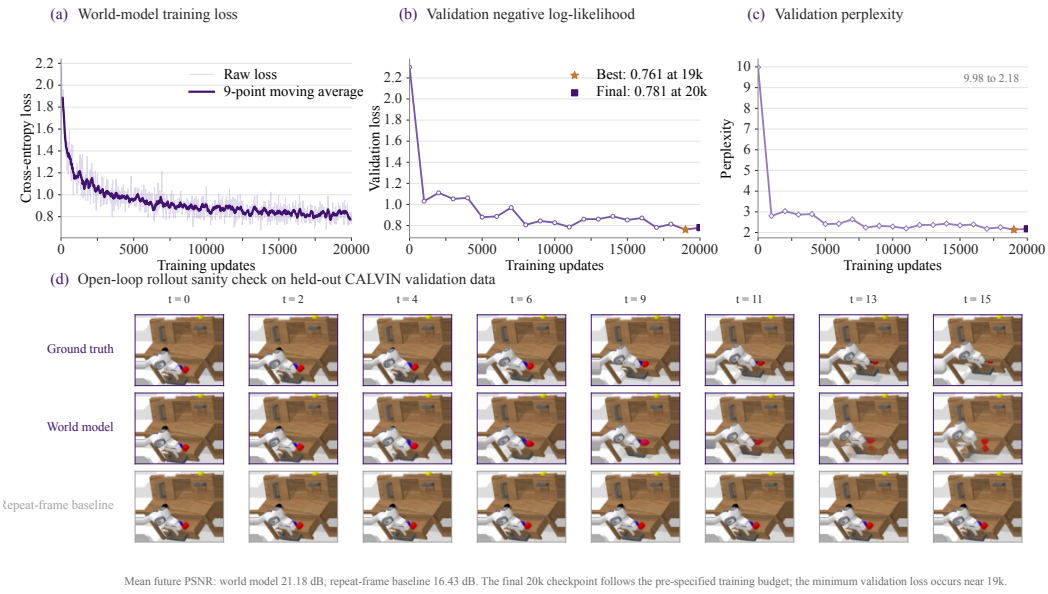

Figure 10: CALVIN world-model adaptation. (a) Raw training loss and its nine-point moving average. (b) Validation negative log-likelihood evaluated every 1,000 updates; the star marks the minimum near 19k and the square marks the pre-specified final 20k checkpoint. (c) Validation perplexity. (d) A 16-frame action-conditioned open-loop rollout on held-out CALVIN validation data. The model receives the first frame and ground-truth actions, then autoregressively feeds back predicted visual tokens. It achieves 21.18 dB mean future PSNR, compared with 16.43 dB for repeating the first frame.

Table 5: CALVIN adaptation and training hyperparameters. Batch size denotes the global batch across eight H20 GPUs.

Setting	Tokenizer adaptation	World-model adaptation
Initialization	RLVR-World RT-1 tokenizer	RLVR-World RT-1 world model
Trainable component	Compressive tokenizer	Autoregressive transformer
Training updates	20,000	20,000
Segment / context frames	8 / 1	8 / 1
Global batch size	8	8
Optimizer	AdamW	AdamW
Peak learning rate	5×10^{-6}	1×10^{-5}
LR schedule	Cosine	Constant with warmup
Warmup updates	1,000	5,000
Weight decay	0	0
Gradient clipping	1.0	1.0
Precision	bfloat16	bfloat16
Image augmentation	None	None
Checkpoint / validation interval	1,000	1,000
GPUs	8× NVIDIA H20	8× NVIDIA H20
Wall-clock time	2 h 08 min	33 min 13 s

time steps). Demonstrations remain intact; training samples contiguous eight-frame segments at stride one.

We use the static `agentview_rgb` camera. Native 128×128 images are bilinearly resized to 256×256, edge-padded to 256×320 as in CALVIN, and scaled to $[0, 1]$, without augmentation.

Action conversion. We use the CALVIN seven-to-13-slot mapping and 256 uniform bins per dimension. Ranges are fitted on training actions only, with a 2% margin: $[-1.04, 1.04]$ for the

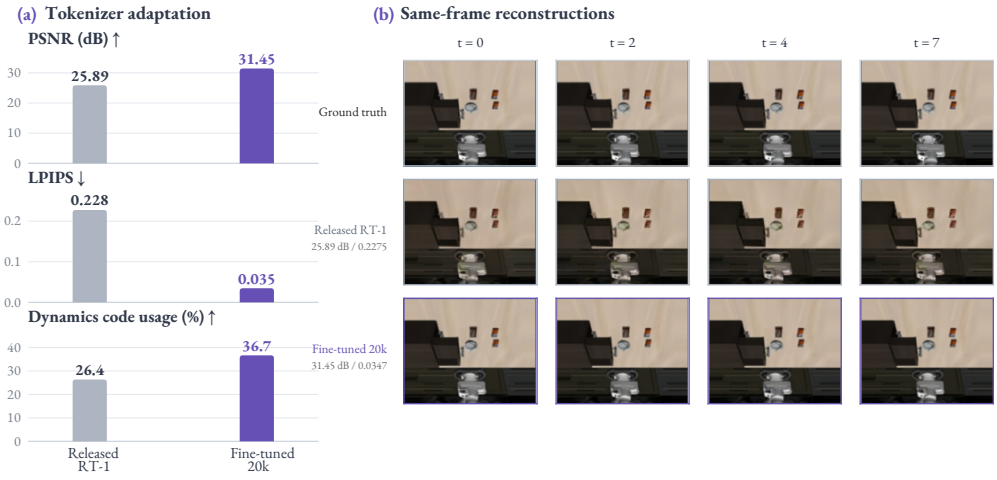

Figure 11: LIBERO-90 tokenizer adaptation. (a) Reconstruction PSNR, LPIPS, and dynamics-code utilization for the released RT-1 tokenizer and the final 20k LIBERO tokenizer, evaluated on the identical fixed set of 32 held-out clips (256 frames). (b) Same-frame reconstructions at $t \in \{0, 2, 4, 7\}$. Adaptation substantially reduces color shift and preserves robot and object structure.

gripper, $[-0.975, 0.975]$ for translation, $[-0.35394, 0.38929]$ for roll, $[-0.39, 0.39]$ for pitch and yaw, and $[-1, 1]$ for unused slots.

Stage 1: visual-tokenizer adaptation. Tokenizer initialization, representation, loss, optimizer coefficients, and seed match CALVIN (Appendix B.4); Table 6 lists the training settings. We compare the released tokenizer and the final 20k checkpoint on 32 fixed, non-overlapping clips from eight held-out episodes (256 frames). PSNR, SSIM, and LPIPS improve from 25.8858 dB, 0.9062, and 0.2275 to 31.4548 dB, 0.9669, and 0.0347, while dynamics-code utilization rises from 26.35% to 36.66% (Figure 11). The final checkpoint follows the specified training budget; intermediate images from random validation batches are not used for checkpoint selection.

Stage 2: action-conditioned world-model adaptation. We freeze the 20k tokenizer and adapt the RT-1 world model using the same 1,931-token sequences, causal objective, and optimization recipe as CALVIN (Table 6), with seed 20260903. Online validation uses two mini-batches per evaluation. Monitored loss decreases from 1.7984 to 0.8089 and perplexity from 6.0401 to 2.2453; the minimum loss is 0.7482 near 17k (perplexity 2.1133). All LIBERO experiments use the final 20k checkpoint specified by the training budget.

On a held-out high-motion episode, the model receives one observed frame and recorded future actions. Mean future PSNR is 19.549 dB for a 16-frame open-loop rollout, versus 15.411 dB for repeating the observed frame; at 32 frames, the values are 15.193 and 13.719 dB. Figure 12 shows the training curves and 16-frame rollout.

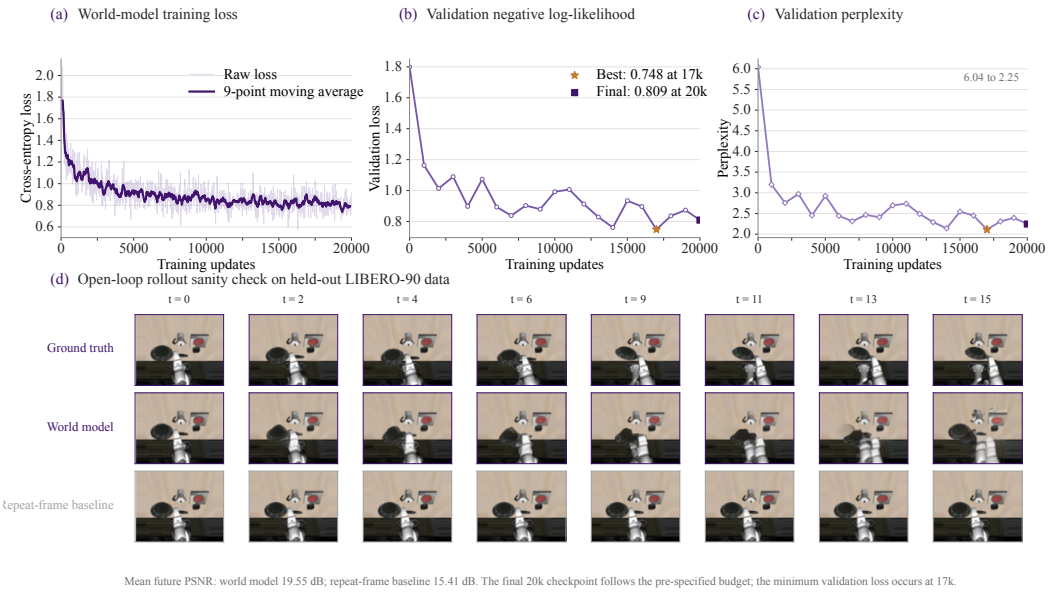

Figure 12: LIBERO-90 world-model adaptation. (a) Raw training loss and its nine-point moving average. (b) Validation negative log-likelihood evaluated every 1,000 updates; the star marks the minimum near 17k and the square marks the pre-specified final 20k checkpoint. (c) Validation perplexity. (d) A 16-frame action-conditioned visual open-loop rollout on a held-out high-motion LIBERO-90 episode. The model observes the first frame, receives ground-truth future actions, and autoregressively feeds back its visual predictions.

Table 6: LIBERO-90 adaptation and training hyperparameters. Batch size denotes the global batch across eight H20 GPUs.

Setting	Tokenizer adaptation	World-model adaptation
Initialization	RLVR-World RT-1 tokenizer	RLVR-World RT-1 world model
Trainable component	Compressive tokenizer	Autoregressive transformer
Training updates	20,000	20,000
Segment / context frames	8 / 1	8 / 1
Global batch size	8	8
Optimizer	AdamW	AdamW
Peak learning rate	5×10^{-6}	1×10^{-5}
LR schedule	Cosine	Constant with warmup
Warmup updates	1,000	5,000
Weight decay	0	0
Gradient clipping	1.0	1.0
Precision	bfloat16	bfloat16
Image augmentation	None	None
Checkpoint / validation interval	1,000	1,000
GPUs	8× NVIDIA H20	8× NVIDIA H20
Wall-clock time	2 h 07 min 55 s	32 min 33 s

B.6 BridgeV2 Adaptation and Short-Horizon Evaluation

Data and action representation. We preserve the train/test split of BridgeV2 TFDS 0.1.0 (Walke et al., 2023): 25,460 training episodes (864,292 time steps) and 3,475 validation episodes (118,603 time steps), with mean length approximately 34 frames. Training samples contiguous eight-frame segments at stride one within episodes. Images use the native 256×320 geometry and the same 1,280-token context, 80-token dynamics representation, and 4,375-code FSQ streams as CALVIN.

Bridge's action mapping differs from CALVIN and LIBERO: translation occupies slots 0–2, rotation 3–5, gripper 6, zeros 7–11, and termination 12. Each dimension uses 256 bins with ranges fitted on training actions only, preserving the 93-token visual–action block.

For the additional qualitative study, we take the first 24 validation files in lexicographic order with at least 33 frames and retain their first 33 frames: one observation and 32 prediction targets. This candidate pool is separate from the main-text BridgeV2 evaluation sets.

Stage 1: tokenizer adaptation. We fine-tune the released RLVR-World RT-1 tokenizer (Wu et al., 2025) with an adversarial reconstruction objective. Table 7 lists the optimization settings. Both AdamW optimizers use $(\beta_1, \beta_2) = (0.9, 0.999)$ and $\epsilon = 10^{-8}$; gradient checkpointing is enabled and the seed is 20260906. The final checkpoint is shared step 60,000, reached in approximately 18 h 49 min.

Before shared step 10,000, only the tokenizer is updated. Thereafter, generator and discriminator branches alternate after groups of four accumulation micro-batches. Each learning-rate schedule advances only with its corresponding optimizer. Thus, the 60k shared counter does not denote 60,000 generator updates.

Augmentation and reconstruction objective. Tokenizer training uses clip-consistent color jitter: brightness, contrast, and saturation factors in $[0.9, 1.1]$, and hue shifts in $[-0.05, 0.05]$. Random resized crops use area scale $[0.8, 1.0]$ and aspect ratio $[1.0, 1.4]$, with output size 256×320. The same sampled transform is applied across each clip. The online validation loader inherits these augmentation settings; the fixed-subset reconstruction evaluation below instead uses unchanged input frames.

Let x_c denote the context frame and $x_{1:7}$ the dynamics frames, with reconstructions $\hat{x}$. For this run, the generator objective is

$$\begin{aligned}\mathcal{L}_G = {} & \mathcal{L}_1(x_c, \hat{x}_c) + \mathcal{L}_1(x_{1:7}, \hat{x}_{1:7}) \\ & + \mathcal{L}_{\text{LPIPS}}(x_c, \hat{x}_c) + \mathcal{L}_{\text{LPIPS}}(x_{1:7}, \hat{x}_{1:7}) \\ & + 0.1\, \mathbb{1}[s \geq 10{,}000]\, \mathcal{L}_{\text{adv}}, \qquad \mathcal{L}_{\text{adv}} = -\mathbb{E}[D(\hat{x})].\end{aligned} \tag{11}$$

Each dynamics term averages over seven frames. Context and dynamics terms have unit weight, without temporal-occupancy balancing. FSQ commitment losses are zero, and the adversarial weight is fixed at 0.1.

Adversarial objective. Following adversarial perceptual tokenization (Esser et al., 2021), a frame-level discriminator processes real and reconstructed context and dynamics images. It uses six stride-two convolutional stages, spectral normalization, LeakyReLU activations, intermediate instance normalization, and a spatial logit head. With reconstructed frames detached, its loss is

$$\mathcal{L}_D = \mathbb{E}[(1 - D(x))_+ + (1 + D(\hat{x}))_+] + \mathcal{R}_{\text{GP}}. \tag{12}$$

The real-input gradient penalty has coefficient 10 and unit target norm, using gradients of summed logits with respect to real dynamics inputs. The discriminator is used only during tokenizer adaptation. Full Context and `ReCAP` share the resulting frozen tokenizer and transformer; we do not separately estimate the effect of adversarial training.

Tokenizer evaluation. The final 60k tokenizer is evaluated on 32 eight-frame clips from the first eight validation episodes (four fixed starts per episode; some clips overlap). Across 256 frame evaluations, PSNR, SSIM, and LPIPS are 27.0667 dB, 0.8336, and 0.1089; dynamics-only values are 26.7168 dB, 0.8286, and 0.1115. Dynamics-code utilization is 65.19%. These measure reconstruction from real frames, not open-loop prediction.

Figure 13 shows the optimization curves. Online validation reconstruction loss decreases from 0.0643 to 0.0279 and LPIPS from 0.3608 to 0.1110 at the last evaluation (step 59,001). These augmented-batch evaluations differ from the unchanged fixed subset above. The generator objective also changes at adversarial activation (step 10k).

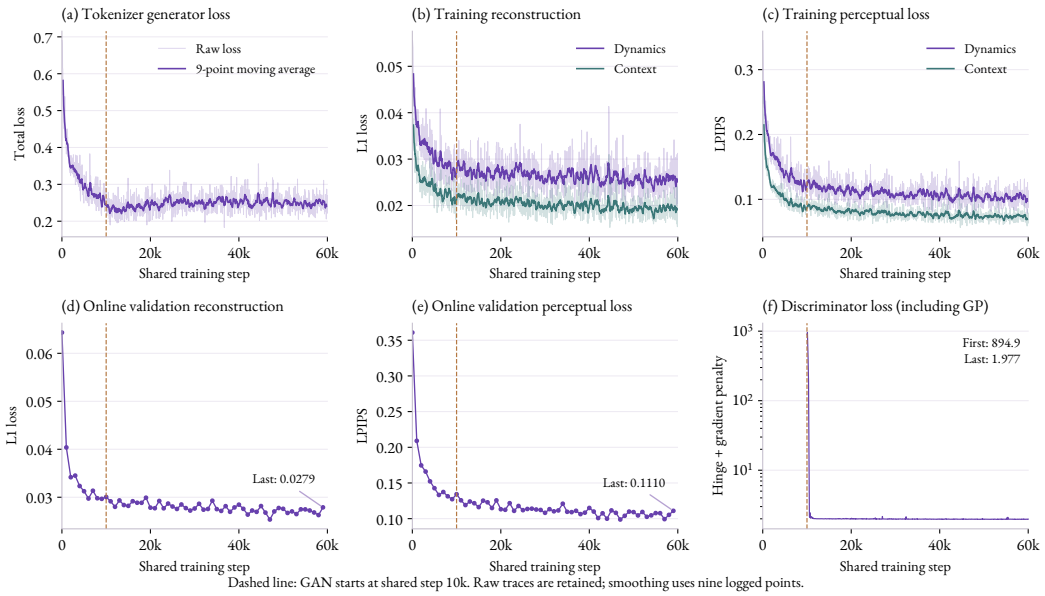

Figure 13: **BridgeV2 tokenizer training.** (a) Total generator loss; (b–c) context and dynamics reconstruction and perceptual losses; (d–e) online validation reconstruction and perceptual losses; (f) discriminator hinge loss plus gradient penalty (logarithmic vertical axis). Dashed lines mark adversarial training activation at shared step 10k. Faint training traces show raw logged values; solid training traces use a nine-point moving average. Validation points are unsmoothed and end at step 59,001. The shared counter includes alternating generator/discriminator updates after GAN activation.

Stage 2: action-conditioned world-model adaptation. We freeze the 60k tokenizer and adapt the released RT-1 transformer (127,101,696 parameters) for 100,000 updates using causal next-token cross-entropy on 1,931-token, eight-frame sequences. Table 7 gives training settings; AdamW coefficients match Stage 1, with seed 20260906 and no image augmentation or adversarial loss. Online validation uses two mini-batches per evaluation, so its fluctuations do not estimate full-validation-set performance.

Training loss decreases from 4.2416 to 1.5072; monitored validation loss from 4.3995 to 1.6072 and perplexity from 81.4104 to 4.9888 (Figure 14). Although validation loss reaches its minimum of 1.5297 near 70k (perplexity 4.6168), rollout comparisons use the final 100k checkpoint specified by the training budget.

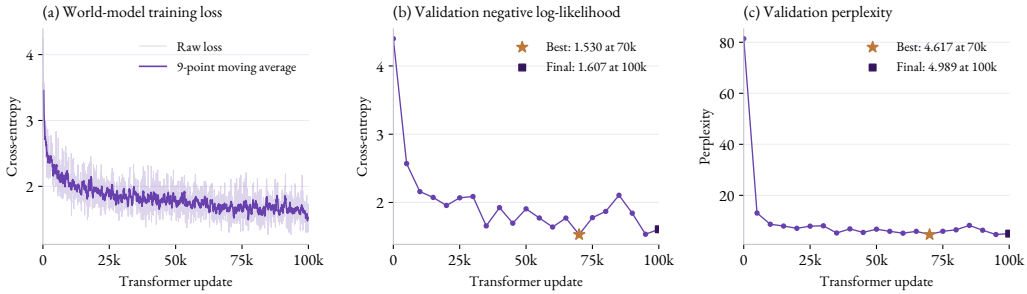

Figure 14: **BridgeV2 world-model training.** (a) Raw causal cross-entropy and its nine-point moving average over 100k transformer updates. (b) Validation negative log-likelihood and (c) validation perplexity. Stars indicate the lowest monitored validation loss near 70k; squares mark the final 100k checkpoint used for the rollout comparisons. All validation points are plotted without smoothing, including the final evaluation at 100,000.

Table 7: BridgeV2 adaptation hyperparameters. Tokenizer steps use the shared generator/discriminator counter; world-model steps are transformer updates.

Setting	Tokenizer adaptation	World-model adaptation
Initialization	RT-1 tokenizer	RT-1 world model
Checkpoint counter	60,000 (shared)	100,000
Segment / context frames	8 / 1	8 / 1
Per-device batch / accumulation	1 / 4	1 / 1
Global batch per optimizer update	32	8
Optimizer / weight decay	AdamW / 0	AdamW / 0
Peak learning rate	5×10^{-5} (G and D)	10^{-5}
LR schedule	Cosine (G), constant (D)	Constant
Configured warmup	5,000 per schedule	5,000
Gradient clipping	1.0	1.0
Precision	BF16; TF32 enabled	BF16
Adversarial start / weight	10,000 / 0.1	Disabled
Discriminator depth / GP weight	6 / 10	–
Image augmentation	Color jitter and crop	None
Checkpoint / validation interval	2,000 / 1,000	5,000 / 5,000
GPUs	8 H20	8 H20

Paired inference protocol. The model observes $\mathbf{o}_0$ and predicts $\mathbf{o}_{1:32}$ from recorded actions. The initial observation supplies both context and the initial dynamics anchor. Full Context and `ReCAP` share the checkpoint, observation, actions, and sampling settings (Table 8). This reference implementation uses PyTorch/Transformers on one H20 in FP32, with TF32 disabled. Full Context reuses its KV cache incrementally. `ReCAP` retains the fixed anchor and six recent complete blocks; after eviction it rebuilds KV and rebases positions, without re-encoding pixels. Its prompt is bounded at 1,931 tokens. The methods produce identical visual tokens for the first seven predictions, before eviction. These timings are separate from the BF16 vLLM efficiency benchmark in RQ4.

Metrics and qualitative selection. We average full-image PSNR, SSIM (11-pixel window, σ = 1.5), and VGG-LPIPS over the 32 predicted frames, excluding the observation. Figure 18 shows the three candidates with the largest relative LPIPS reduction among 24 episodes. Metadata supplies display labels only. These selected-case gains are not full-validation-set estimates; remaining duplication and shape errors also limit conclusions about task success.

Table 8: BridgeV2 settings for the additional $H = 32$ comparisons.

Setting	Value
Tokenizer / world-model checkpoint	60k / 100k
World-model parameters	127.1M
Training GPUs / global batch / segment	8 H20 / 8 / 8 frames
World-model learning rate / precision	10^{-5} / BF16
Observed / predicted frames	1 / 32
Inference precision	FP32, TF32 disabled
Sampling seed / temperature / top-k	20260906 / 1.0 / 100
Recent window / maximum prompt	6 blocks / 1,931 tokens
Candidate pool / displayed examples	24 / 3

C Additional Qualitative Rollouts

Figures 15–18 compare matched rollouts with identical initial observations, recorded actions, and sampling seeds: 64-frame rollouts for RT-1, CALVIN, and LIBERO-90, and 32-frame predictions for BridgeV2 (Appendix B.6). Task labels come from dataset annotations. LIBERO images are vertically flipped for display only; inference and metrics use the original arrays.

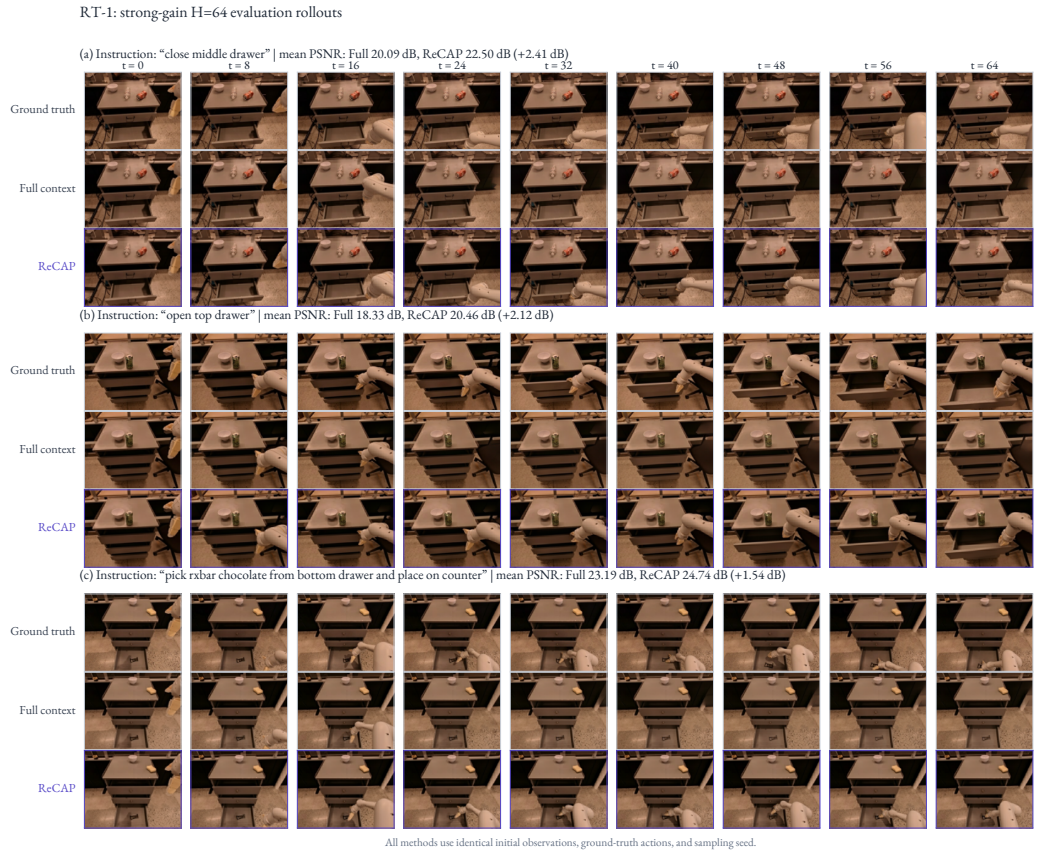

Figure 15: ***Additional RT-1 qualitative comparisons.*** Nine time points are shown for each rollout. Across these three instruction-conditioned examples, `ReCAP` improves mean PSNR over Full Context by +2.41, +2.12, and +1.54 dB, respectively.

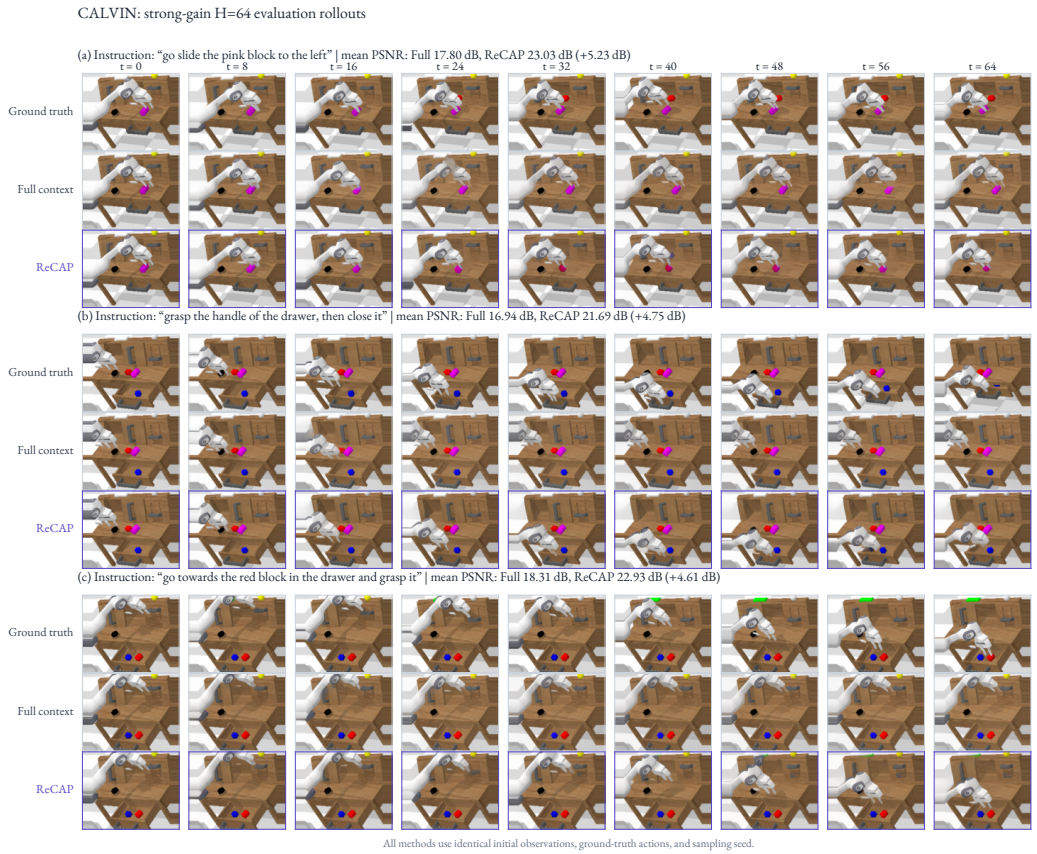

Figure 16: ***Additional CALVIN validation comparisons.*** The examples are drawn from annotated validation trajectories. `ReCAP` improves mean PSNR over Full Context by +5.23, +4.75, and +4.61 dB, respectively.

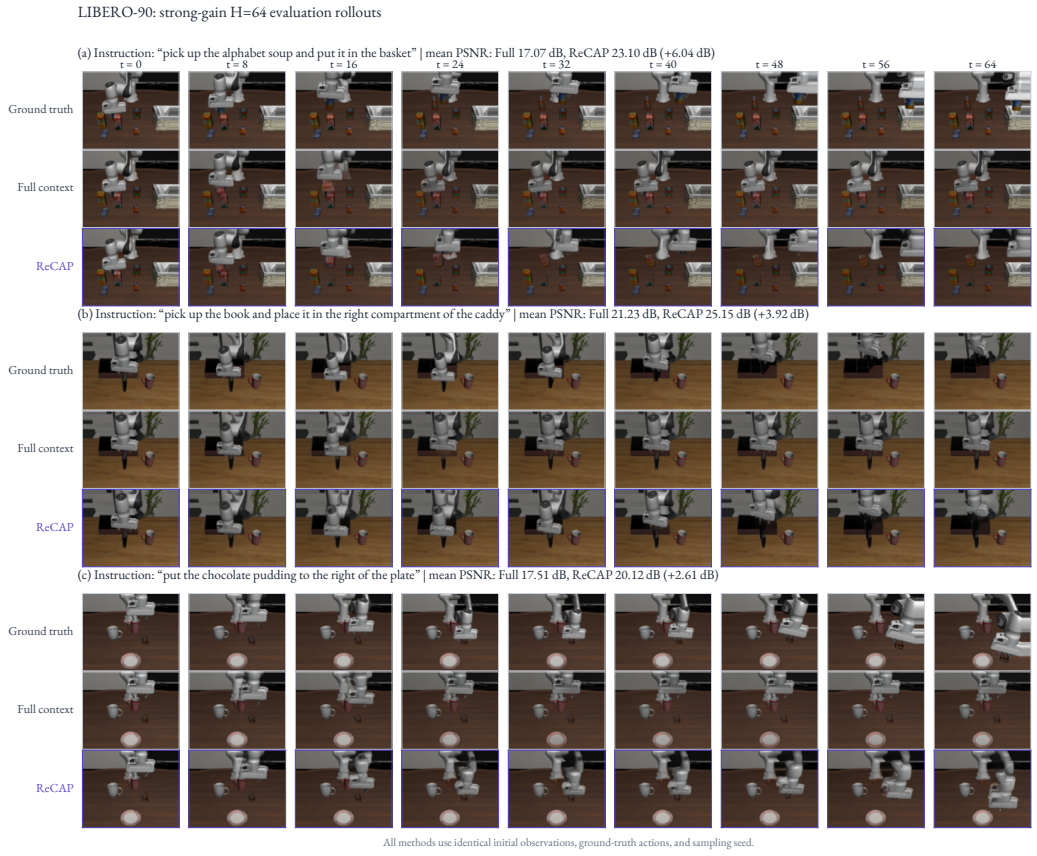

Figure 17: ***Additional LIBERO-90 held-out-task comparisons.*** The three examples use distinct language instructions from the task-disjoint validation split. `ReCAP` improves mean PSNR over Full Context by +6.04, +3.92, and +2.61 dB, respectively. Frames are vertically flipped for canonical display orientation only.

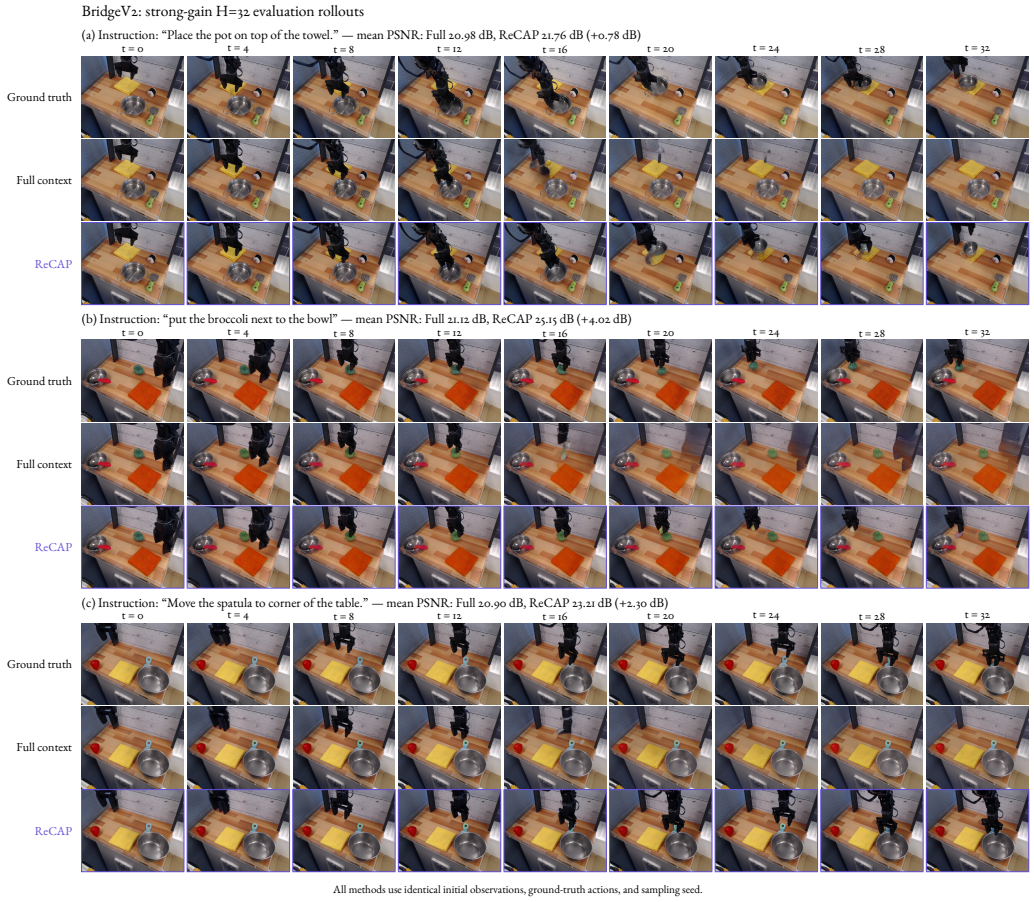

Figure 18: ***Additional BridgeV2 validation comparisons.*** Three high-gain examples selected from a fixed pool of 24 candidates, with nine displayed time points ($t = 0, 4, 8, 12, 16, 20, 24, 28, 32$) over 32 predicted frames. Rows show ground truth, Full Context, and **`ReCAP`**. Mean future PSNR improves by +0.78, +4.02, and +2.30 dB; mean LPIPS decreases by 26.7%, 32.5%, and 23.3%, respectively. Metrics use all 32 predictions, excluding $t = 0$. Task text identifies the episodes and is not a model input.

D VLM Evaluation Protocol

D.1 Evaluation Scope and Inputs

We evaluate 200 held-out RT-1 episodes, each with paired Full Context and `ReCAP` predictions sharing the initial observations, actions, and sampling seed (400 videos per judge). Videos contain 26–113 frames (median 42) at 256×320 resolution and are encoded as H.264 at 25 frames/s before sampling.

Each judge receives one video's sampled frames and the episode instruction, without method identity, the competing prediction, or ground-truth video. Scores are compared afterward across matched episodes. The instruction specifies the evaluation goal; it is not a language input to the world model. An additional 80-pair human comparison and sensitivity analysis appear in Appendices D.6 and D.7.

D.2 Deterministic Frame Sampling

Let N be the decoded MP4 frame count. If $N \leq 16$, all frames are used. Otherwise, the evaluator selects the following zero-based indices in temporal order:

$$\mathcal{I}(N) = \left\{ \left\lfloor \frac{iN}{16} \right\rfloor : i = 0, \ldots, 15 \right\}. \tag{13}$$

The same ordered RGB images at original resolution are shared across all judges before model-specific preprocessing. Sampling is deterministic, with no key-frame detection; the budget is 16 images, whose visual-token counts vary by model. The final video frame is not necessarily included, so the rubric's "last frame" refers to the last sampled image.

D.3 Judge-Specific Preprocessing and Generation

Qwen3-VL-8B. We use the native chat template and smart resize, with sampling disabled and at most 400 output tokens.

InternVL3-8B. Frames are bicubically resized to 448×448, normalized with ImageNet means $(0.485, 0.456, 0.406)$ and standard deviations $(0.229, 0.224, 0.225)$, and converted to BF16. Each frame uses one tile, without dynamic tiling, and is labeled in temporal order. The square resize changes the source aspect ratio.

GPT-5.5. The API receives the same sampled images and rubric with provider-default decoding parameters; repeated responses need not be identical.

D.4 Shared Rubric and Required Output

The fixed prompt requires an integer score from 1 to 10 and an evidence-based reason for each of four criteria, returned as JSON. Judgments must use only visible evidence. The episode `{instruction}` is the sole runtime text substitution. Table 9 summarizes the rubric; Appendix D.10 reproduces the complete prompt.

Mandatory caps and score guidance. The *hallucination check* caps Instruction Following at 4 when human hands replace the expected robotic arm. The *static penalty* caps Motion Naturalness at 4 when the arm shows essentially no meaningful motion or interaction, even if the scene appears smooth. The prompt also encourages use of the full range, intermediate odd scores between even anchors, and distinct scores for distinguishable videos. The latter instruction may affect tie rates.

D.5 Parsing, Normalization, and Paired Comparisons

Qwen3-VL and GPT-5.5 each produce 400 compliant responses. InternVL3 requires offline formatting recovery for 26.5% of responses; scores are extracted from the original output without re-judging. One metric in one truncated InternVL3 response remains missing and is excluded rather than scored zero.

Table 9: ***Shared VLM evaluation rubric.*** Score bands summarize the prompt; the mandatory score caps also apply. Higher values indicate better observed behavior.

Score	Interaction quality	Instruction following	Task progress	Motion naturalness
1–2	Penetration or absent proper contact	Wrong action, object, or scene	No visible progress	Severe jumps or jitter; or a frozen arm
3–4	Contact with implausible physical response	Related behavior with major errors	Motion without meaningful progress	Frequent irregular motion; or tiny, aimless movements
5–6	Mostly plausible, with minor interaction errors	General intent followed with execution errors	Partial completion	Meaningful motion with occasional stutter or kinematic issues
7–8	Realistic contact and object response	Mostly correct, with minor deviations	Mostly completed	Meaningful, smooth motion with minor imperfections
9–10	Visually indistinguishable from real interaction	All instructed elements correctly executed	Fully completed	Meaningful, smooth, physically plausible motion

For a valid raw score $s_{e,m,j,k} \in \{1, \ldots, 10\}$ for episode e, method m, judge j, and criterion k, the normalized score is

$$q_{e,m,j,k} = s_{e,m,j,k}/10. \tag{14}$$

Normalized values range from 0.1 to 1.0; raw integers are retained. We report means and standard deviations by method, judge, and criterion, with comparisons paired by episode and effective sample counts stated where values are missing. All 12 judge–criterion aggregate differences favor **ReCAP**, with Cohen's d from 0.36 to 1.37. These are standardized method effects, distinct from the inter-rater agreement measured below.

D.6 Human–VLM Agreement

Human ratings are provided by authors or members of our research group. On 80 paired rollouts, human majorities favor **ReCAP** in 42 pairs, tie in 14, and favor Full Context in 24 (Table 10). Original-protocol VLM agreement is 66.3–72.5%, with unweighted Cohen's κ = 0.43–0.52. All judges produce fewer ties than humans (Figure 19); this is consistent with, but does not isolate, the prompt's discouragement of identical scores.

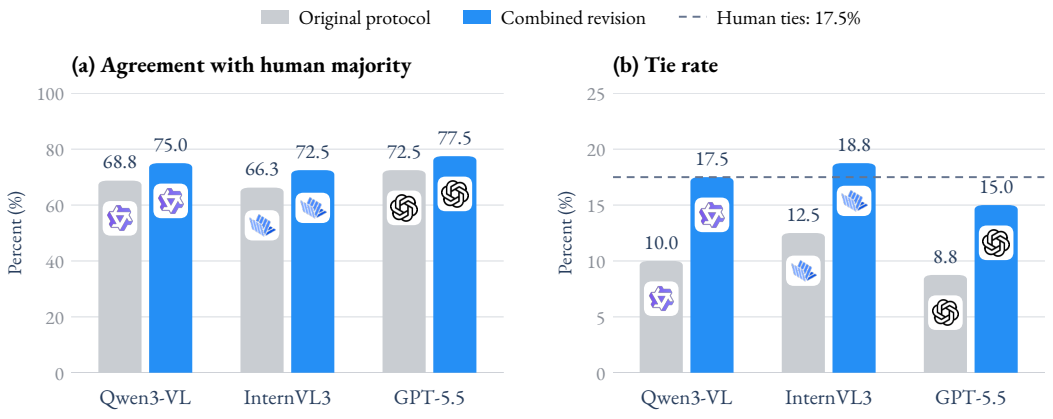

Figure 19: **Human–VLM alignment under two judging protocols.** Bars show pooled fractions on the same 80 pairs: (a) three-way agreement with human-majority labels and (b) tie rates. Gray denotes the original protocol and blue the combined revision. The dashed line in (b) marks the human tie rate of 17.5%. Judge order is fixed across panels.

The combined revision raises observed agreement to 72.5–77.5% and κ to 0.55–0.62. These descriptive comparisons do not establish statistical significance or a ranking of judge quality.

Table 10: **Human–VLM agreement on 80 paired rollouts.** Agreement counts all three labels, including ties. Unweighted Cohen's κ is calculated from the pooled preference counts and observed agreement. The combined revision is described in Appendix D.7.

Protocol	Evaluator	ReCAP wins	Ties	Full Context wins	Agreement with humans	κ
Reference	Human majority	42	14	24	–	–
Original	Qwen3-VL	47	8	25	55/80 (68.8%)	0.46
	InternVL3	45	10	25	53/80 (66.3%)	0.43
	GPT-5.5	49	7	24	58/80 (72.5%)	0.52
Combined	Qwen3-VL	43	14	23	60/80 (75.0%)	0.58
	InternVL3	42	15	23	58/80 (72.5%)	0.55
	GPT-5.5	45	12	23	62/80 (77.5%)	0.62

D.7 Sensitivity to the Judging Protocol

Table 11 compares protocol variants on the same 80 pairs. The combined revision allows equal scores, removes score-band steering, includes both endpoint frames, and halves the static penalty.

Table 11: **Sensitivity of VLM preferences to the scoring protocol.** Each cell gives ReCAP wins / ties / Full Context wins and sums to 80. The original protocol remains the basis of the main VLM results.

Protocol	Qwen3-VL	InternVL3	GPT-5.5
Original	47 / 8 / 25	45 / 10 / 25	49 / 7 / 24
Explicitly allow equal scores	43 / 14 / 23	42 / 15 / 23	45 / 12 / 23
Half-strength static penalty	45 / 11 / 24	43 / 13 / 24	47 / 10 / 23
Remove static penalty	42 / 13 / 25	40 / 15 / 25	44 / 12 / 24
Include first and last frames	46 / 11 / 23	44 / 13 / 23	48 / 9 / 23
Combined revision	43 / 14 / 23	42 / 15 / 23	45 / 12 / 23

All tested judge–protocol combinations favor ReCAP in aggregate, including removal of the static penalty. Combined-revision tie rates (15.0–18.8%) approach the human rate (17.5%). Allowing equal scores alone produces the same marginal counts as the combined revision; this neither implies identical per-pair labels nor isolates benefits from the other changes.

D.8 Evaluation Source and Reproducibility

We adapt the VLM-based evaluation pipeline of WorldArena (Shang et al., 2026) to the three judges and shared rubric described above. The evaluation implementation is publicly available. Per-video scores and original judge responses are retained for verification, including responses that required formatting recovery.

D.9 Limitations of VLM Evaluation

Sparse frames can miss transient errors or late completion, particularly in longer episodes. Shared evidence and rubric also make the judges' errors potentially correlated; the 12 positive differences are not independent replications. The static penalty explicitly penalizes freezing, a common Full Context failure, so Motion Naturalness measures more than smoothness. Sensitivity tests retain the aggregate preference direction, but the 80-pair human comparison offers only limited corroboration. These scores assess visible plausibility and instruction compatibility, not verified physics, closed-loop performance, or real-robot task success.

D.10 Full VLM Evaluation Prompt

The original shared prompt is reproduced below, with only line wrapping and typography adjusted. `{instruction}` is replaced by the episode instruction.

VLM Evaluation Prompt

```
**VIDEO INSTRUCTION:** {instruction}

You are an expert evaluator for robot interaction videos. You are evaluating videos
    generated for **embodied AI manipulation scenarios**, specifically focusing on
    robotic arms interacting with objects in tabletop environments.
**EVALUATION CONTEXT:**
- Target scenario: Robotic manipulation (e.g., pick-place, push, grasp)
- Expected agent: **Robotic arm/end-effector**, NOT human hands
- Expected environment: Tabletop with objects, typical for robot manipulation tasks
- Expected physics: Realistic robot-object interactions following physical laws
**CRITICAL EVALUATION PRINCIPLES:**
1. Base ALL judgments ONLY on what is visually observable in the sampled frames
2. DO NOT infer information not shown (no assumptions about unseen parts)
3. Evaluate temporal coherence across the sampled frames
4. For instruction following: Compare STRICTLY against the provided text instruction
5. **You MUST use the FULL 1-10 scale**. Avoid defaulting to the middle (5-6) when
    uncertain; instead, pick the odd anchor between the two neighboring even anchors
    that best matches the visual evidence. Two visually similar videos should NOT
    receive identical scores unless they are indistinguishable frame-by-frame.
**EVALUATION DIMENSIONS & SCORING RUBRICS (integer 1..10):**
1. Interaction_Quality (Quality of robot-object interactions)
- Score 1-2: Objects pass through robot or other objects; no proper contact
- Score 3-4: Contact exists but interaction is unrealistic (e.g., sliding without
    friction, incorrect force response)
- Score 5-6: Mostly plausible interactions with minor issues (e.g., slight
    penetration, imperfect grasping)
- Score 7-8: Realistic contact physics (proper friction, force transfer, object
    deformation)
- Score 9-10: Perfect interaction physics; indistinguishable from real robot
    manipulation
2. Instruction_Following (Adherence to given instruction: **VIDEO INSTRUCTION**)
- **HALLUCINATION CHECK**: If the video shows human hands instead of robotic arms,
    score <= 4 immediately
- Score 1-2: Completely different from instruction (wrong action, wrong objects,
    wrong scene)
- Score 3-4: Partially related but major errors (e.g., wrong target object,
    incorrect manipulation type)
- Score 5-6: Follows general intent but with execution errors (e.g., correct action
    sequence but imprecise)
- Score 7-8: Mostly correct with minor deviations (e.g., slight position error,
    extra unnecessary motion)
- Score 9-10: Perfect execution of all specified elements (action, object, scene,
    outcome)
3. Task_Progress (Task completion from first to last frame)
- Score 1-2: No progress; scene at last frame is essentially identical to first
    frame w.r.t. the task
- Score 3-4: Robot initiates motion but no meaningful progress toward the goal state
- Score 5-6: Partial progress (e.g., object approached / grasped but not delivered)
- Score 7-8: Task mostly completed with minor issues (e.g., object placed near but
    not exactly at target)
- Score 9-10: Task fully completed as described by the instruction
4. Motion_Naturalness (Smoothness AND presence of physically plausible arm motion)
- **STATIC PENALTY (MANDATORY)**: This metric evaluates the naturalness of
    *executed* motion, NOT merely the absence of visual artifacts. If the robotic
    arm exhibits essentially no meaningful motion across the sampled frames - i.e.
    the end-effector position barely changes, no interaction with objects is
    initiated, and the scene at the last frame looks almost identical to the first
    frame - you MUST score <= 4, regardless of how smooth or artifact-free the
    (non-)motion appears. A perfectly still or barely-moving arm is NOT natural
    robot motion; it is a failure to act.
```

VLM Evaluation Prompt (continued)

```
- Score 1-2: Severe teleportation, jitter, or discontinuous jumps in end-effector
    position; OR the arm is completely frozen / drifts imperceptibly without
    executing any manipulation.
- Score 3-4: Frequent unnatural motion (visible jitter, non-smooth trajectory); OR
    the arm makes only tiny, aimless movements and clearly fails to attempt the
    instructed motion.
- Score 5-6: The arm clearly executes meaningful motion toward the task, but with
    occasional stutter, unnatural acceleration, or minor kinematic issues.
- Score 7-8: The arm executes meaningful motion smoothly with realistic joint
    kinematics; only very minor imperfections in transitions.
- Score 9-10: The arm executes meaningful motion perfectly smoothly and physically
    plausibly, indistinguishable from a real robot performing the task.
**SPECIFIC ROBOT-RELATED CHECKS:**
- Robotic arm should have mechanical appearance, NOT human limbs
- End-effector (gripper) should maintain consistent form throughout interaction
- Robot motion should show appropriate joint movement and kinematics
- Object manipulation should respect object mass and inertia
- Contact should be maintained appropriately during grasping/lifting
**OUTPUT FORMAT REQUIREMENTS:**
You MUST output a SINGLE, VALID JSON object with EXACTLY four keys:
- 'Interaction_Quality'
- 'Instruction_Following'
- 'Task_Progress'
- 'Motion_Naturalness'
Each value must be an object with exactly two keys:
- '"score"': integer 1-10
- '"reason"': concise explanation citing SPECIFIC visual evidence from frames
**EXAMPLE OUTPUT:**
{
"Interaction_Quality": {
"score": 4,
"reason": "Object slides without friction during pushing; gripper penetrates object
    slightly"
},
"Instruction_Following": {
"score": 2,
"reason": "Video shows human hand instead of robotic arm (hallucination)"
},
"Task_Progress": {
"score": 3,
"reason": "Robot approaches the object but never grasps or moves it"
},
"Motion_Naturalness": {
"score": 3,
"reason": "End-effector position barely changes across frames; no manipulation is
    attempted, so per the STATIC PENALTY this cannot exceed 4 despite the absence of
    jitter"
}
}
**CRITICAL INSTRUCTIONS:**
1. Output ONLY the JSON object, no other text
2. Base scoring on observed visual evidence only
3. For instruction following: Strictly compare with the provided instruction
4. Consider temporal coherence across all sampled frames
5. Penalize hallucinations (e.g., human hands instead of robot) heavily
6. **For Motion_Naturalness, always apply the STATIC PENALTY first**: if the arm
    doesn't meaningfully move, cap the score at 4 before considering smoothness.
7. **Use the full 1-10 range**. Do NOT round every score to 5 or 6. Prefer picking
    the odd anchor (3, 5, 7, 9) when the video is "between" two even levels - this
    is how you avoid an artificially high tie-rate against a similar comparison
    video.
Now evaluate the provided video frames based on the above criteria.
```

E Effect of Scene Prefix Length

E.1 Evaluation Setup and Token Allocation

We evaluate 450 held-out LIBERO-90 episodes with five generation seeds using the final 20k checkpoint (Appendix B.5) and a 1,931-token budget cap. Each configuration retains a scene prefix of $p \in \{4, 16, 64, 1280\}$ tokens, the initial 93-token visual–action block, and a contiguous window of recent complete blocks (Figure 8). At steady state,

$$W(p) = \left\lfloor \frac{1{,}931 - p - 93}{93} \right\rfloor, \qquad N(p) = p + 93 + 93W(p). \tag{15}$$

The unused budget is $1{,}931 - N(p)$: preserving complete blocks yields slightly different effective lengths under the common cap.

Table 12: **Scene-prefix allocation under a 1,931-token budget cap.** Results on 450 held-out LIBERO-90 episodes with five generation seeds are reported as mean ± standard deviation. Token counts describe the steady-state prompt; every row includes the same initial 93-token block. The full scene anchor is highlighted.

Scene prefix (tokens)	Initial block (tokens)	Recent blocks	Recent tokens	Effective tokens	Unused budget	Late LPIPS ↓	nDTW ↑
4	93	19	1,767	1,864	67	0.208 ± 0.003	0.619 ± 0.006
16	93	19	1,767	1,876	55	0.207 ± 0.003	0.621 ± 0.006
64	93	19	1,767	1,924	7	0.205 ± 0.003	0.624 ± 0.005
1,280	**93**	**6**	**558**	**1,931**	**0**	**0.203 ± 0.002**	**0.628 ± 0.005**

E.2 Results and Paired Comparisons

The full scene anchor gives the best measured performance despite retaining six recent blocks versus 19 for shorter prefixes (Table 12). Relative to four scene tokens, it reduces Late LPIPS by 0.005 and increases nDTW by 0.009. All six episode-level paired 95% confidence intervals exclude zero (Table 13); gains over the 64-token prefix are modest.

Table 13: **Episode-level paired comparisons.** Differences are full scene anchor minus the shorter scene prefix; brackets give 95% confidence intervals. Lower Late LPIPS and higher nDTW favor the full anchor.

Comparison	Δ Late LPIPS ↓	95% CI	Δ nDTW ↑	95% CI
Full anchor − 4 tokens	−0.005	[−0.007, −0.003]	+0.009	[+0.005, +0.013]
Full anchor − 16 tokens	−0.004	[−0.006, −0.002]	+0.007	[+0.003, +0.011]
Full anchor − 64 tokens	−0.002	[−0.004, −0.001]	+0.004	[+0.001, +0.007]

Interpretation and scope. This experiment tests joint allocation between scene context and recent history, without isolating specific scene semantics. It complements the LPIPS-AUC ablation in Figure 6(a), where `ReCAP` and Recent Only show no measurable difference under that protocol. Implementations retaining identical tokens, block boundaries, positions, and cache updates realize the same schedule; this comparison distinguishes prefix allocations.

F Evaluation Metric Definitions

F.1 Foreground Masks and Normalized Centroids

Arm Score, Shape IoU, and nDTW use the same foreground extraction in `traj_metrics.py`. Let I_t and $\widehat{I}_t$ denote the reference and predicted RGB frames for $t = 1, \ldots, T$, with height H and width W. Only generated future frames are scored; the two observed input frames are excluded. The inference wrapper clamps RGB values to $[0, 1]$, multiplies by 255, and rounds to uint8 before trajectory evaluation. If sequence lengths differ, the metric function uses their common prefix.

The background and foreground threshold are computed once from the reference future and shared by both sequences:

$$B(p, c) = \text{median}_t\, I_t(p, c), \qquad d_t(p) = \frac{1}{3}\sum_{c=1}^{3} |I_t(p, c) - B(p, c)|, \tag{16}$$

$$\tau = \max\left\{12, \text{Percentile}_{88}\left(\{d_t(p)\}_{t,p}\right)\right\}, \qquad F_t(p) = \mathbf{1}\{d_t(p) > \tau\}. \tag{17}$$

Prediction masks use $\widehat{d}_t(p) = \frac{1}{3}\sum_c |\widehat{I}_t(p, c) - B(p, c)|$ and the same τ. These reference-derived quantities are used only by the evaluator, never by the world model. With SciPy, each binary mask undergoes 3×3 opening and 7×7 closing, followed by selection of the largest connected component (default 2D connectivity). The implementation skips these cleanup operations if SciPy is unavailable; its `scipy_available` output records this distinction.

Let $\widetilde{M}_t$ be the resulting component and $a_t = |\widetilde{M}_t|/(HW)$. Its validity and normalized centroid are

$$v_t = \mathbf{1}\{0.004 \le a_t \le 0.45\}, \qquad c_t = \frac{1}{|\widetilde{M}_t|}\sum_{(y,x)\in\widetilde{M}_t}\left(\frac{y}{H}, \frac{x}{W}\right), \quad v_t = 1. \tag{18}$$

Coordinates are zero-indexed. Distances use Euclidean distance in this height/width-normalized coordinate space, not pixel distance divided by the image diagonal. Invalid centroids are undefined. The stored mask M_t is $\widetilde{M}_t$ when $a_t \le 0.45$ and empty otherwise: a component below 0.004 is retained for IoU bookkeeping but has no valid centroid. The same rules define $\widehat{M}_t$, $\widehat{v}_t$, and $\widehat{c}_t$.

F.2 Arm Score and Shape IoU

Let $\mathcal{V} = \{t : v_t = 1\}$ be the reference-valid frames. Arm Score is the fraction of these frames with a valid predicted component whose centroid is within radius $r = 0.12$ of the reference:

$$\text{ArmScore} = \frac{1}{|\mathcal{V}|}\sum_{t\in\mathcal{V}} \mathbf{1}\{\widehat{v}_t = 1 \text{ and } \|\widehat{c}_t - c_t\|_2 \le 0.12\}. \tag{19}$$

An invalid predicted component contributes zero. Shape IoU averages mask intersection-over-union over the same reference-valid frames:

$$\text{ShapeIoU} = \frac{1}{|\mathcal{V}|}\sum_{t\in\mathcal{V}} \frac{|\widehat{M}_t \cap M_t|}{\max\{1, |\widehat{M}_t \cup M_t|\}}. \tag{20}$$

The denominator uses pixel counts, and an empty union gives zero. Both metrics are undefined (NaN) when $\mathcal{V}$ is empty. The diagnostic `mask_quality` equals $|\mathcal{V}|/T$; it reports reference mask coverage rather than a learned detector's confidence.

F.3 Normalized Dynamic Time Warping

For each coordinate separately, the implementation linearly interpolates all missing centroid positions between valid frames and fills leading/trailing gaps with the nearest valid endpoint. A sequence with no valid centroid has an empty path. Denote the resulting prediction and reference paths by $P = (p_1, \ldots, p_m)$ and $R = (r_1, \ldots, r_n)$. Dynamic time warping uses

$$D_{i,j} = \|p_i - r_j\|_2 + \min\{D_{i-1,j}, D_{i,j-1}, D_{i-1,j-1}\}, \tag{21}$$

$$\text{nDTW}(P, R) = \exp\left(-\frac{D_{m,n}}{0.10\, n}\right), \tag{22}$$

with $D_{0,0} = 0$ and $D_{i,0} = D_{0,j} = +\infty$ for positive indices. When both paths are nonempty, interpolation preserves their full temporal length ($m = n = T$). If only the prediction path is empty, nDTW is zero; if the reference path is empty, the rollout score is undefined. All three scores lie in $[0, 1]$ when defined, with larger values indicating better agreement. The trajectory function stores these rollout scores rounded to four decimals; the comparison summarizer averages available non-NaN rollout scores.

Scope and limitations. These are foreground-based proxies for robot shape and trajectory consistency, not semantic arm segmentation or 3D pose accuracy. Moving objects, shadows, camera motion, and nearly stationary robots can affect the masks; the largest component need not be the arm. Interpolation can also conceal temporary prediction dropouts in nDTW, whereas Arm Score counts them as misses. The metrics therefore complement image-fidelity and human/VLM assessments rather than establish task success on their own.

F.4 Horizon AUC, Paired Gains, and Win Rates

The fixed-horizon evaluator, `evaluate_rq2_dataset.py`, computes a per-frame metric $q_{i,t}$ for rollout i and takes prefix means from the same 64-frame prediction:

$$\overline{q}_i(h) = \frac{1}{h} \sum_{t=1}^{h} q_{i,t}, \qquad (h_1, \ldots, h_5) = (8, 16, 32, 48, 64). \tag{23}$$

The normalized trapezoidal AUC is

$$A_i(q) = \frac{1}{h_5 - h_1} \sum_{j=1}^{4} (h_{j+1} - h_j) \, \frac{\overline{q}_i(h_j) + \overline{q}_i(h_{j+1})}{2}. \tag{24}$$

It integrates the prefix-mean curve over horizons 8–64, not the raw per-frame curve and not an unweighted mean over the five horizons. For a baseline b, the paired gain is $g_i = A_i(q_b) - A_i(q_{\mathrm{ReCAP}})$ for LPIPS and $g_i = A_i(q_{\mathrm{ReCAP}}) - A_i(q_b)$ for SSIM or PSNR. Thus, positive gains always favor `ReCAP`. For N matched valid rollouts,

$$\overline{g} = \frac{1}{N} \sum_{i=1}^{N} g_i, \qquad \mathrm{WinRate} = \frac{100}{N} \sum_{i=1}^{N} \mathbf{1}\{g_i > 0\} \ \%. \tag{25}$$

The denominator counts rollout cases, not frames or horizons; ties are not wins. The fixed-horizon script stores paired gains at seven decimal places and uses 5,000 case-level bootstrap resamples with replacement, initialized with seed 20260903. The 2.5th and 97.5th percentiles of resampled mean gains form its 95% confidence interval. This definition applies to that evaluator; other studies retain their separately specified sampling units and protocols.

G Reproducibility, Transparency, and Responsible Use

G.1 Review Materials and Release Coverage

The anonymous project page links the manuscript, qualitative rollouts, and downloadable source code. The code package contains the ReCAP scheduler, reference inference and metric implementations, archived training recipes, configuration files, and reproduction documentation. Its reference world models use the approximately 127M-parameter RLVR-World-derived architecture across RT-1, CALVIN, LIBERO-90, and BridgeV2. The package does not reproduce the complete expanded-backbone study in the manuscript; its coverage document distinguishes these experiments from the released reference runs. Archived reference results should therefore not be substituted for results from different backbones or evaluation cohorts.

Weights and datasets are separate dependencies and are not bundled in the anonymous snapshot. The asset manifest retains expected file paths, byte counts, and SHA-256 checksums for matched tokenizers, world models, and action ranges. Author-identifying checkpoint distribution links are omitted during review. Running the reference configurations requires these matched assets, the specified data conversion, and a compatible runtime. The presence of training recipes alone does not establish a complete retraining reproduction.

G.2 Protocol Traceability and Fair Comparison

The inference procedure and active-token budget are specified in Appendix A. Hardware, software, cache settings, adaptation schedules, and data preparation appear in Appendix B. Comparisons should retain the corresponding initial observations, action sequences, checkpoints, sampling settings, and evaluation horizons. The shared VLM rubric and judge-specific processing are given in Appendix D; image-derived motion metrics, missing-value handling, and aggregation are defined in Appendix F. Logical KV storage measures the specified tensor footprint, not total allocated GPU memory. Runtime measurements apply to the documented hardware, concurrency, caching, and truncation settings.

G.3 Human Assessment and Limits of the Evidence

The 80-pair human comparison in Appendix D.6 uses ratings from authors or members of our research group. It provides a limited internal assessment of generated videos, rather than an independent external validation or a population-level study of human preferences. The reported agreement is specific to this sample and judging protocol. VLM scores can change with frame sampling, prompts, preprocessing, and tie handling; they complement the quantitative metrics and do not replace task-level validation. Selected qualitative examples illustrate temporal behavior and failure modes, but do not estimate their frequency across the full evaluation distribution.

G.4 Data, Licensing, and Deployment

Existing datasets and pretrained components retain their respective licenses and access conditions; the code license does not grant additional rights to these assets. The review materials preserve upstream attribution and separate software from checkpoint and dataset dependencies. Generated rollouts are predictions, and should not be presented as recordings of executed robot behavior. Image fidelity, motion proxies, and preference judgments do not measure closed-loop task success or certify physical safety. Deployment on robots requires task-specific validation, safety constraints, and appropriate human oversight beyond the offline evidence reported here.