ParaVT: Taming the Tool Prior Paradox for Parallel Tool Use in Agentic Video Reinforcement Learning

Zuhao Yang[1,2,7] **Kaichen Zhang**[1,3,7] **Sudong Wang**[1,4] **Keming Wu**[1,5,7] **Zhongyu Yang**[6]
Bo Li[7] **Xiaojuan Qi**[3] **Shijian Lu**[2,✉] **Xingxuan Li**[1,✉] **Lidong Bing**[1]

[1]Apodex [2]NTU [3]HKU [4]HKUST(GZ) [5]THU [6]LZU [7]LMMs-Lab

https://evolvinglmms-lab.github.io/ParaVT/

Abstract

Training large multimodal models (LMMs) via reinforcement learning (RL) to natively invoke video-processing tools (*e.g.,* cropping) has become a promising route to long-video understanding. However, most existing agentic Video-LMMs dispatch tool calls sequentially (*i.e.,* one per turn). Sequential tool calling can propagate localization errors, accumulate irrelevant context, and require an additional main-model generation for each successive crop. We introduce **ParaVT**, a multi-agent RL framework for **Para**llel **V**ideo **T**ool calling, dispatching multiple time-window crops in a single turn, allowing evidence from multiple temporal windows to be examined concurrently. Yet applying standard Group Relative Policy Optimization (GRPO) to ParaVT reveals an obstacle we term the *Tool Prior Paradox*: inherited tool-use priors appear to support exploration while competing with the output format learned during supervised fine-tuning. Standard RL can also settle on tool skipping. A comparison with a model with weaker tool-use priors shows stable formatting but no tool calls during RL, consistent with a trade-off between format stability and tool exploration. We propose **PARA-GRPO** (**P**arseability-**A**nchored and **R**atio-g**A**ted **GRPO**), which augments standard RL with two complementary mechanisms: (i) a *format reward* that encourages tag closure and penalizes unclosed reasoning blocks, and (ii) a *per-prompt randomization of the overview-frame budget* to create a reward advantage for tool use when the overview provides insufficient evidence. Across six long-video understanding benchmarks, ParaVT improves over the Qwen3-VL baseline by $+7.9\%$ on average, with PARA-GRPO raising training-time format reward from 0.13 to a peak of 0.64. These findings highlight the need to account for inherited tool-use priors when designing RL methods for long-video understanding.

1 Introduction

Long-video understanding spans question answering and temporal grounding [59], with recent work adopting an *agentic video reasoning* paradigm. To answer "Which player took the decisive volley in this ninety-minute soccer match?", a large multimodal model (LMM) is post-trained to invoke video-processing tools via supervised fine-tuning (SFT) on customized tool-use traces followed by reinforcement learning (RL) with verifiable rewards [10, 23, 32, 39, 60, 63, 65]. For example, LongVT [60] trains interleaved multimodal chains of tool-thought through cold-start SFT, agentic RL, and agentic reinforcement fine-tuning, instilling a global-to-local loop that skims the match and zooms into the few seconds of evidence until the answer is grounded. These methods, however,

Work done during Z.Y., K.Z., S.W., and K.W.'s internships at Apodex. This work was supported by Apodex, which provided the compute, storage, and engineering infrastructure for the reported experiments. ✉ Corresponding Authors.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

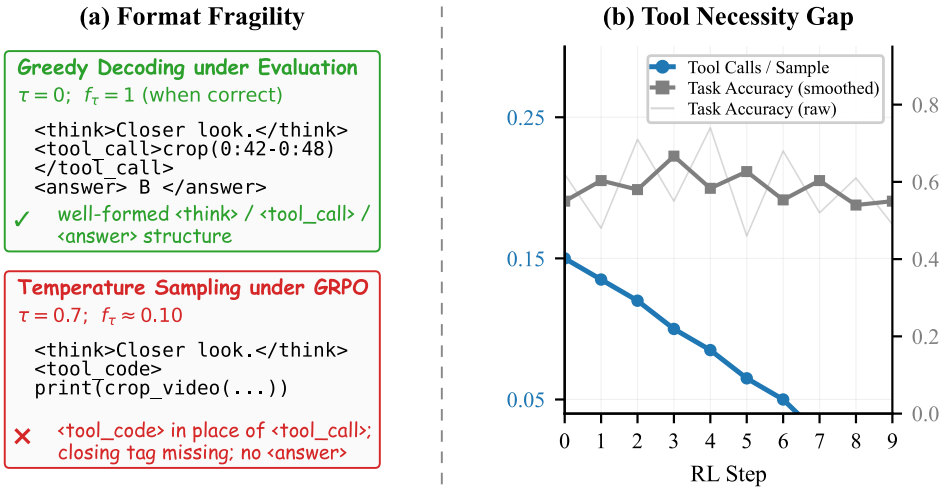


Figure 1: **Two Failure Modes of the Tool Prior Paradox.** *(a) Format Fragility*: rollouts are well-formed under greedy decoding (sampling temperature τ=0, format reward $\approx$ 1). Under temperature sampling within vanilla GRPO (τ=0.7), the policy reverts to the alternative `<tool_code>` tag in place of `<tool_call>`, often drops closing tags, and stops emitting `<answer>` altogether (mean format reward $f_\tau \approx 0.1$). *(b) Tool Necessity Gap*: tool-call count drops to near-zero within 7 steps while task accuracy oscillates between 0.45 and 0.74, as the policy converges on the shortcut of skipping tools.

dispatch tool calls sequentially across turns (*i.e.,* one tool call per turn), with successive tool outputs accumulating in a single context window. This paradigm is brittle along three dimensions (Figure 3a). First, a single mis-localized crop propagates errors with no peer to correct it. Second, multi-turn accumulation aggregates context corruption, and long-context models use information in long inputs unevenly [26]. Third, each crop requires another main-model generation over the accumulated context, so inference cost grows with the number of turns.

To this end, we introduce **ParaVT**, a multi-agent RL framework for **Para**llel **V**ideo **T**ool calling (Figure 3b). Within ParaVT, a main agent issues multiple temporal-window crops in a single turn and aggregates the returned evidence for decision-making. During RL rollouts, these crops are processed in parallel by sub-agents that share the policy's weights, each returning a text summary of its assigned window. The main agent can cross-check these summaries, reducing reliance on any single crop.

A natural choice for end-to-end ParaVT training is Group Relative Policy Optimization (GRPO) [18] on top of a tool-native cold-started Qwen3-VL [1] checkpoint. However, vanilla GRPO exhibits two coupled training-time failures. The first is *Format Fragility* (Figure 1a): the SFT-learned `<think>`/`<tool_call>`/`<answer>` format is reliable under greedy decoding but, within a few vanilla-GRPO steps under temperature sampling, the policy reverts to the alternative `<tool_code>` schema. This is a shallow override of the SFT format reminiscent of the Superficial Alignment Hypothesis [71], compounded by the competing *inherited tool-use priors*: the probability mass on tool-call continuations already present in the base checkpoint before our SFT stage that resurfaces under RL-time temperature. As a result, malformed rollouts cannot be parsed into rewardable tool calls, so the GRPO advantage signal is computed over a corrupted

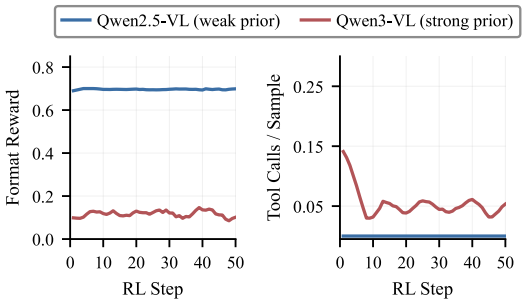


Figure 2: **Cross-Model Evidence for the Tool Prior Paradox under Vanilla GRPO.** Qwen3-VL-8B (stronger tool prior) explores tool use but collapses on format, while Qwen2.5-VL-7B (weaker tool prior) stays format-perfect yet emits zero tool calls.

trajectory population before any tool-use credit can be assigned. The second is *Tool Necessity Gap* (Figure 1b): when uniformly sampled overview frames suffice to answer many prompts directly, rewards need not favor tool-calling over tool-skipping rollouts. GRPO then provides little consistent task-reward incentive for tool use, and the policy can converge to the shortcut of skipping tools [40].

To probe the role of inherited tool-use priors, we replicate the same setup on Qwen2.5-VL [37] (with much weaker tool priors than Qwen3-VL) under identical hyperparameters (Figure 2): its format stays near-perfect, yet RL elicits no tool calls. This cross-model contrast suggests a trade-off: stronger tool-use priors may support exploration while competing with the cold-started output format, whereas the weaker-prior model retains its format but does not explore tools. The comparison does not isolate prior strength from other model differences. We term this observed tension the *Tool Prior Paradox*. This brings us to the central question of this work: *for tool-native LMMs, does the inherited tool-use prior help or hurt tool use after RL?*

We propose **PARA-GRPO** (**P**arseability-**A**nchored and **R**atio-g**A**ted **GRPO**) with *Exploration Anchoring* and *nFrames Gating* to tame the Tool Prior Paradox (Section 3.2). Exploration Anchoring combines a format reward that encourages tag closure and penalizes unclosed reasoning blocks with a Constrained Generation hook that fixes the opening reasoning tag. Together they anchor rollout parseability without restricting reasoning content or tool-call sequences. Both mechanisms adapt practices popularized by DeepSeek-R1 [18]: its `<think>` reasoning template and rule-based format rewards. We adopted this variant after a standard format reward alone failed to keep rollouts parseable. nFrames Gating tackles the reward-signal side: randomizing the overview-frame budget per prompt makes some prompts harder to answer without tools, creating opportunities for a within-group reward contrast between tool-calling and tool-skipping rollouts. The two design choices are complementary. Anchoring preserves parseability so that tool-use credit can be assigned, while gating changes the evidence available before a tool call. The combined recipe achieves higher format reward than either single-component variant while retaining tool use (Section 4.3). Empirically, PARA-GRPO raises training-time format reward from 0.13 to a peak of 0.64 and improves the agentic-setting Qwen3-VL baseline on every tested benchmark (Section 4.2).

Our contributions are fourfold. **(i)** We introduce ParaVT, a framework for parallel video-tool calling via agentic RL, trained on a self-curated 97K-sample multi-task SFT corpus and a separate 4.4K-sample RL corpus covering video question answering (QA) and temporal grounding. *Code, data, and model weights are publicly available.* **(ii)** We identify the Tool Prior Paradox, decompose it into Format Fragility and Tool Necessity Gap, and support the diagnosis with a cross-model contrast on a weak-prior LMM. **(iii)** We propose PARA-GRPO, which composes nFrames Gating, a per-prompt randomization of the overview-frame budget that restores a reward contrast between calling and skipping the tool, with Exploration Anchoring to tackle the Tool Necessity Gap and Format Fragility together. **(iv)** We conduct extensive comparisons with existing methods on six long-video benchmarks and systematic ablations of PARA-GRPO's key design choices, demonstrating the effectiveness of ParaVT.

2 Related Work

Long-Video Understanding. RL-based long-video understanding spans three broad directions: (i) *tool-free RL* [12, 25, 48, 49, 52, 64] optimizes textual reasoning without external tool calls; (ii) *multi-agent and orchestrated RL* jointly optimizes cooperating policy agents [4] or trains a planner to query specialized perception agents [27, 45]; (iii) *single-LMM adaptive perception and tool-augmented RL* revisits visual evidence through additional sampling or tool calls: LongVT [60] (sequential `crop_video` calls), VITAL [65] (on-demand frame sampling over multiple rounds), Zoom-Zero [39] (a single coarse-to-fine zoom-in pass), Conan [32] (an identify-reason-act loop over frames), VideoZoomer [10] (iterative `<video_zoom>` calls), LoVe-R1 [14] (step-decoupled iterative zoom-in), SAGE [23] (a JSON tool-action schema), VideoChat-R1.5 [57] (iterative re-perception of predicted regions), and Video-o3 [63] (multi-hop clue seeking), together with methods that select segments or frames over successive turns [19, 34, 55]. Our work differs in two respects: (1) we train a tool-native LMM to dispatch several temporal crops per turn, combining serial context expansions; (2) we identify and address the Tool Prior Paradox, a failure mode of tool-native RL that, to our knowledge, prior work has not characterized.

Parallel and Multi-Agent Tool Use. Executing independent tool calls concurrently is established for text agents, from compiler-style parallel function calling [24] to RL-trained parallel search queries and reasoning threads [33, 69]. With RL, MATPO [31] trains planner and worker roles inside one large language model (LLM) through role-specific prompts. WideSeek-R1 [56] jointly optimizes a lead agent and parallel sub-agents that share one LLM, and Kimi K2.5 [44] trains an orchestrator whose parallel sub-agents are frozen. For video, systems built on off-the-shelf models process clips or frame subsets in parallel [8, 11, 35] or coordinate several LMM agents [3, 58]. MACF [6] trains segment-level agents with a supervised curriculum. Concurrent with our work, ReTool-Video [28] trains a planner with GRPO that may issue several independent actions per round over a large tool library. ParaVT trains parallel temporal-crop dispatch in a tool-native LMM, with sub-agents sharing the current policy's weights but excluded from the RL loss.

Format Stability and Tool Use in RL. In agentic RL, format stability is a precondition for tool-use learning: only parseable rollouts can be credited for their tool calls. Rule-based format rewards, popularized by DeepSeek-R1 [18], are standard in open recipes, either as per-tag counts [21] or as structural terms combined with semantic rewards, where format collapse under semantic pressure has also been reported [2]. More generally, constrained decoding restricts generation to a grammar [53]. The Superficial Alignment Hypothesis characterizes post-training as learning style and format [71], though this hypothesis remains contested [38]. In safety alignment, distributional changes can concentrate in the first few output tokens [36]. Our Format Fragility is analogous but specific to tool-native LMMs under RL-time temperature sampling: the SFT-learned `<tool_call>` tag reverts to the alternative `<tool_code>` tag under RL rollouts, fragmenting the structural-boundary distribution. PRISM inserts on-policy distillation between SFT and RL, using mixture-of-experts perception and reasoning feedback to reduce distributional drift [50]. ParaVT instead intervenes during RL, leaving the SFT-to-RL handoff unchanged. At the token level, RL-induced policy shifts concentrate on a sparse subset of high-divergence tokens [30]. To encourage exploration on tokens that drive correct outcomes, prior work relaxes the Kullback–Leibler penalty on those tokens [46]. Exploration Anchoring differs from these gradient-side interventions: it fixes the opening reasoning tag and adds a sequence-level reward that checks only structural boundary tags. Related efforts shape when a tool is worth calling. AdaTooler-V [47] rescales rewards by an offline estimate of each sample's tool benefit. Pixel Reasoner [41] adds a curiosity bonus against the reluctance to adopt new visual operations. FrameMind [17] assigns each rollout in a group a different frame-count and resolution configuration. nFrames Gating instead randomizes the overview-frame budget per prompt, so that the call/skip reward contrast arises from the training distribution.

3 Method

3.1 ParaVT: Parallel Video Tool Calling for Long-Video Understanding

ParaVT consists of three design elements: a parallel-dispatch architecture (Section 3.1.1), a two-stage training pipeline (Section 3.1.2), and a self-curated multi-task dataset (Section 3.1.3).

3.1.1 Framework Design

A common paradigm for tool-augmented long-video understanding lets the LMM decide *when* and *where* in the video to look more closely by issuing a `crop_video(start, end)` function call that returns the requested temporal segment with densely resampled frames for further inspection (Figure 3a). Existing realizations of this design [10, 32, 60, 65] request additional visual evidence *sequentially*: one tool action per turn, with the returned frames re-injected into the running context before the next turn begins.

ParaVT re-organizes the same loop as a single-turn divide-and-conquer step (Figure 3b). Within a single turn, the main agent emits K *parallel* `<tool_call>` invocations on selected temporal windows, each dispatched to one of K independent sub-agents that share weights with the main agent, a multi-agent-in-one-model design in which roles differ only by prompt [31]. In RL rollouts, a sub-agent is a fresh single-turn call of the current policy under a fixed, query-agnostic summarization prompt that sees only the frames of its assigned window. Each sub-agent processes only its assigned window and returns a textual summary rather than resampled frames. The K calls run concurrently. Each summary

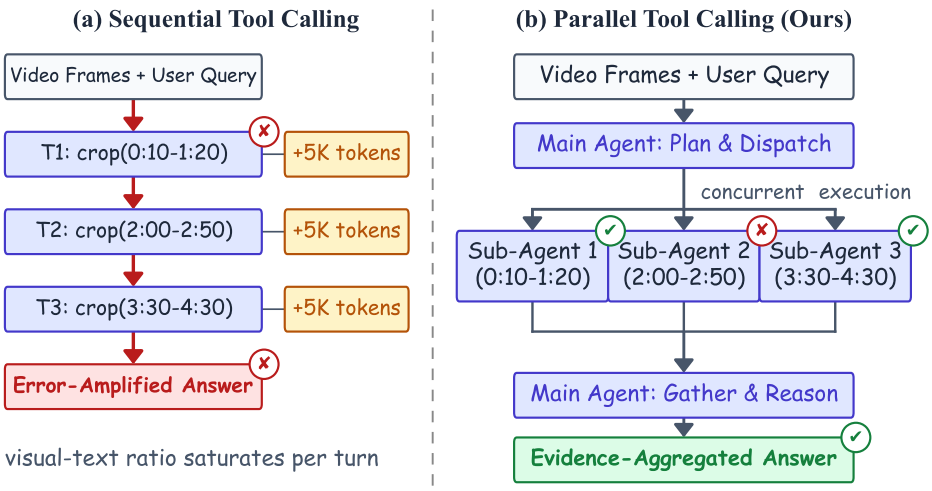


Figure 3: **Framework Comparison.** *(a) Sequential Tool Calling*: successive turns re-include the full context, accumulating visual-token overhead. A single mis-localized crop (✗) propagates errors with no peer to correct it, yielding an error-amplified answer. *(b) Parallel Tool Calling (Ours)*: one main agent dispatches K tool calls concurrently to K independent sub-agents (shown for $K{=}3$). Mis-localized peers (✗) can be cross-checked against correct ones (✓), yielding an evidence-aggregated answer.

is truncated to a fixed length and appended to the main agent's context as a `<tool_response>` block, on which the main agent reasons to generate the final `<answer>`.

This single-turn parallel dispatch yields three concrete advantages over the sequential paradigm. *(i) Peer-Correctable Evidence.* The main agent cross-checks K independent window summaries, allowing peers to correct a mis-localized crop before the error propagates. *(ii) Controlled Context Growth.* All K crops of a turn enter the context at once. The accumulated context is therefore re-encoded once per turn rather than once per crop. In RL rollouts, text summaries further limit the extension to a small constant per crop, while returning frames would add K visual-token blocks per turn. *(iii) Bounded Inference Latency.* Issuing the K crops in one turn replaces K sequential main-model generations with one. During RL rollouts, the K sub-agents run concurrently, so the tool-using portion is bounded by the slowest sub-agent rather than by their sum.

3.1.2 Training Strategy

Cold-Start SFT with Parallel Tool Traces. The base LMM (*i.e.,* Qwen3-VL-8B-Instruct [1]) supports the `<tool_call>` schema, but it is not trained on the single-turn K-crop format that ParaVT requires. Without supervised exposure to parallel traces, probe RL runs from the base checkpoint fail to produce parseable rollouts (Section F), and the GRPO advantage signal collapses before any tool-use credit can be assigned, consistent with the observation that native tool-calling abilities are too weak for direct RL training [60]. Therefore, we conduct an SFT cold start on the base model with the parallel-tool corpus and select an early checkpoint as the RL initialization based on training-time format stability under temperature sampling. The two-stage SFT-then-RL pipeline is a common recipe for open multimodal-reasoning systems [20, 66]. ParaVT specializes it to parallel video-tool calling with the corpus described in Section 3.1.3 and the reward design in Section 3.2.

Agentic RL with Verifiable Rewards. Starting from the cold-started checkpoint, we conduct GRPO with two verifiable reward terms: an accuracy term against the ground-truth answer and a format term over the `<think>`/`<tool_call>`/`<answer>` schema. For each prompt, GRPO samples $G{=}8$ rollouts and updates the policy by their group-normalized advantage. Rewards and advantages are computed on main-agent tokens only, and sub-agent generations are excluded from the loss, so RL optimizes when to dispatch and how to aggregate, while sub-agents change only through the

shared weights. Vanilla GRPO at this stage exposes the Format Fragility and Tool Necessity Gap introduced in Section 1. PARA-GRPO addresses these failures (Section 3.2).

3.1.3 Data Curation

SFT Split. The SFT corpus contains 97K samples spanning four task families (full per-source breakdown in Table 5): general video QA (50K from LLaVA-Video-178K [68]), long-video reasoning chains (5K from LongVideo-Reason [7]), temporal grounding (12K Charades-STA [16] direct grounding + 6K Charades-STA-converted traces with parallel tool calls), and self-curated 22.5K parallel-tool traces. The mix preserves general video understanding while giving the model concentrated supervision on the parallel multi-tool schema. Tool-using samples account for 30% of the corpus. We chose this fraction after an earlier larger mix (212K total at 14% tool) yielded weaker downstream tool-calling than this smaller, tool-richer mix (Section B).

The parallel-tool traces are drawn from three sources: 15K LongVT [60] tool-using rollouts, 5K Gemini-2.5-Flash [9] distillations of LongVT prompts, and 2.5K multi-segment grounding samples from MUSEG [29]. The first two sources emit one `crop_video` call per assistant turn with resampled video frames re-injected into the next turn's context, a *sequential* format that does not exhibit the single-turn K-call schema we want ParaVT to learn. We traverse each sequential trace and merge adjacent crops whose target windows do not overlap and whose tool responses do not cross-reference each other (*e.g.,* "inspect 00:30–00:50" followed by "inspect 02:10–02:25" on independent visual evidence). Calls that fail this independence check, such as a refinement crop conditioning on its predecessor, remain sequential. Each tool's visual response is then replaced by a textual summary of the segment, aligning the SFT data with the RL sub-agent's text-summary output format and keeping context length manageable when several crops appear in the same response.

RL Split. The RL corpus aggregates 4,406 samples from three task families: 1,606 open-ended QA from filtered LongVT [60] RL data, 1,600 multiple-choice questions (MCQ) from the VideoR1 [12] RL pool, and 1,200 temporal video grounding (TVG) queries from the Charades-STA [16] training set. The SFT and RL corpora share some Charades training videos. Before training begins, we apply an offline difficulty filter inspired by DAPO [62]: we exclude open-ended prompts whose ground-truth answers exceed 15 words and prompts with unanimously unsuccessful rollouts under the cold-started policy. These are heuristic exclusions, rather than a guarantee of zero advantage under future policies.

3.2 PARA-GRPO: Parseability-Anchored and Ratio-Gated GRPO

Vanilla GRPO faces two obstacles to learning tool use in ParaVT. *Format Fragility* manifests as tool-tag reversion or unclosed structural tags. Malformed rollouts can disrupt tool execution and the assignment of tool-use credit. The *Tool Necessity Gap* persists even on parseable rollouts: when overview frames already suffice to answer a prompt, task rewards need not consistently favor tool-calling over tool-skipping rollouts.

PARA-GRPO addresses these obstacles with two complementary components. *Exploration Anchoring* stabilizes structural boundaries through constrained generation and rewards for tag closure. *nFrames Gating* randomizes the per-prompt overview-frame budget so that some GRPO groups contain a reward contrast between tool-calling and tool-skipping rollouts. Anchoring preserves parseability, while gating supplies an incentive to use tools when overview frames are insufficient. Increasing temperature can encourage tool exploration but destabilize tag closure. Temperature adjustment does not replace the reward contrast that nFrames Gating is designed to create.

3.2.1 Exploration Anchoring

Changing the SFT target tag did not resolve Format Fragility in our probe. Because the main runs revert from `<tool_call>` to `<tool_code>`, we also tested SFT directly on `<tool_code>`. The resulting policy still emits `<tool_call>` more often than its SFT-trained tag during RL (Section I.3). This bidirectional reversion indicates that choosing the alternative tag does not eliminate the competing tool representations. We therefore retain the native `<tool_call>` tag and target structural-boundary stability during RL.

Exploration Anchoring uses two mechanisms to restore the closing </think> and </answer> tags where structural collapse concentrates.

Constrained Generation. At the entry and exit of the response, two minimal interventions reinforce what SFT has already taught reliably. A *Think Prefix* pins the first tokens of every response to <think>\n, ruling out blind direct answers and tool calls without restricting what the model reasons about. It follows the <think> template convention of DeepSeek-R1 [18]. A complementary *Answer Suffix* term in the format reward credits the presence of a final <answer> block even when intermediate structure is imperfect, so policies that recover into a well-formed answer are not penalized for exploration along the way.

Selective Anchoring. To address failures to close structural tags, we add an anchoring term to the format reward:

$$\begin{aligned} R_{\text{anchor}}(y) = \alpha\,\mathbf{1}\big[\text{reasoning block closed}\big] - \gamma\,\mathbf{1}\big[\texttt{<think>}\text{ opened but never closed}\big] \\ + \beta\,\mathbf{1}\big[\text{full }\texttt{<think>}\rightarrow\texttt{</think>}\rightarrow\texttt{<answer>}\text{ flow}\big], \end{aligned} \tag{1}$$

where R_{anchor} is a sequence-level scalar computed from string checks on the completed rollout y. It contributes to the total reward, whose group-normalized advantage GRPO broadcasts to all trained main-agent tokens, changing rollout preference rather than assigning token-level credit. Unlike equal-credit tag-count rewards [21], the anchor penalizes an opened but unclosed <think>. The triplet (α, β, γ) and the outer scaling λ_{anchor} inside $R_{\text{fmt}} = R_{\text{base}} + \lambda_{\text{anchor}} R_{\text{anchor}}$ govern how aggressively the anchor pulls the policy toward parseability.

Constrained Generation and Selective Anchoring address different parts of the response: the former fixes the entry and credits the final answer, the latter rewards closure of internal boundaries. Neither restricts the reasoning or tool-call content that lives between them.

3.2.2 nFrames Gating

Anchoring restores parseable rollouts, but a reward advantage for tool use is still needed. With a generous overview budget, both tool-calling and tool-skipping rollouts may reach the correct answer. When task rewards do not systematically favor tool-calling rollouts within a prompt group, GRPO provides no consistent task-reward incentive for tool use.

nFrames Gating creates the missing gap by randomizing the overview-frame budget per prompt:

$$n \sim \text{Uniform}\big(\{4, 8, 16, 32, 64\}\big), \tag{2}$$

where n is the number of overview frames seen by all $G{=}8$ rollouts in the GRPO group for that prompt. Reduced budgets ($n{<}64$) push part of the visual evidence outside the overview. Rollouts that recover that evidence through `crop_video` can then receive higher task rewards than those relying on the truncated overview. The largest budget ($n{=}64$) preserves the easy regime in which direct answering is sufficient when warranted. Each training step therefore samples a mixture of budget-bound and budget-free prompts, so a controllable fraction of prompts exhibits a non-trivial reward contrast between tool-calling and tool-skipping rollouts, while on prompts where the full budget already suffices, the policy is free to skip tools without penalty. Setting this fraction too low leaves the gating signal too sparse for GRPO to learn from. Setting it too high crowds out the easy regime and induces over-calling.

3.2.3 Reward Modeling

Let x denote a prompt (user query paired with the video input), y a rollout, and a^* the ground-truth answer. The composite reward sums three terms:

$$R(x, y) = R_{\text{acc}}(y, a^*) + \lambda_{\text{fmt}}\, R_{\text{fmt}}(y) + R_{\text{tool}}(y). \tag{3}$$

R_{acc} scores the rollout against the ground truth using a task-appropriate metric (exact match for MCQ, temporal intersection over union (IoU) for grounding, token-level F1 for open-ended QA). R_{fmt} scores structural compliance and embeds R_{anchor} from Equation (1), including the penalty for unclosed tags. R_{tool} rewards well-formed tool invocations.

4 Experiments

4.1 Implementation Details

Training. We initialize from Qwen3-VL-8B-Instruct [1] and SFT-cold-start on a 97K multi-task corpus. An early checkpoint is selected for RL initialization based on training-time format stability (selection details in Section B). RL uses a separately constructed $4{,}406$-sample set after offline difficulty filtering inspired by DAPO [62]. We sample $G{=}8$ rollouts at $\tau{=}0.7$, set the anchor weight to $\lambda_{\text{anchor}}{=}0.5$, and decode up to 16 frames per sub-agent crop. Training leverages AReaL [15] on a node of 8 NVIDIA GPUs (80 GB+ VRAM each), with 7 allocated to Fully Sharded Data Parallel (FSDP) training and 1 to SGLang rollout serving. Full hyperparameters are listed in Table 6 of Section B.

Evaluation. We evaluate on six long-video benchmarks under a unified 64-frame overview protocol, reporting MCQ accuracy on VideoMME [13], LongVideoBench [54], LVBench [51], MLVU [72], and MMVU [70], and mean intersection over union (mIoU) on Charades-STA [16]. Table 1 groups open-source baselines by their training paradigm into three settings: *direct-answer* for instruct backbones with no native thinking pattern, *reasoning-enhanced* for models optimized for explicit reasoning, and *tool-augmented* for models with tool-use or adaptive re-perception mechanisms. GPT-4o and Gemini 1.5 Pro are included as proprietary reference rows using published benchmark scores. We select each baseline's prompt class according to its training paradigm. These settings can differ from those used in the original model report. We restrict our evaluation to natively post-trained single-LMM methods, excluding agent frameworks [3, 27, 58, 61, 67] to keep the comparison fair. These systems combine separately configured perception or reasoning components, so their reported results are not directly matched to our single-policy evaluation protocol.

4.2 Main Results

Table 1: **Performance Comparison with Existing Video-LMMs.** The best result is in **bold**; underline marks ParaVT's value when it is not the benchmark-wise best. $*$ marks cells withheld due to benchmark–training-data overlap. † marks an evaluation whose native tool-call schema could not be reconciled with the Charades-STA grounding-output format under our unified protocol, so the resulting outputs are not measurable with mIoU.

Model	VideoMME w/o sub	VideoMME w/ sub	LongVideo-Bench	LVBench	MLVU	MMVU	Charades-STA test
Proprietary Video-LMMs: best-setting numbers from official reports (not under our unified protocol)							
GPT-4o [22]	71.9	77.2	66.7	34.7	64.6	66.7	—
Gemini 1.5 Pro [42]	75.0	81.3	64.4	33.1	74.3	65.8	—
Open Instruct Video-LMMs: direct-answer setting (native single-pass prompt)							
Qwen2.5-VL-7B [37]	55.7	64.5	46.4	32.2	47.8	65.4	31.6
Eagle2.5-8B [5]	57.6	63.8	54.5	37.9	52.9	58.2	—
Open Reasoning Video-LMMs: reasoning-enhanced setting (<think>→<answer>)							
Video-R1-7B [12]	57.6	66.0	57.4	36.9	61.6	61.3	25.4
VideoChat-R1-7B [25]	50.4	58.2	49.2	23.8	58.7	65.0	31.5
VideoRFT-7B [48]	58.5	65.6	55.1	38.0	44.9	42.7	18.7
Time-R1-7B [52]	58.9	66.2	56.0	38.2	60.5	63.4	34.7
ReWatch-R1-7B [64]	58.8	65.0	53.6	38.5	60.1	59.8	20.2
Video-Thinker-7B [49]	61.9	65.3	56.0	$*$	**65.2**	64.5	29.0
Open Agentic Video-LMMs: tool use or adaptive visual re-perception							
Qwen3-VL-8B [1]	59.9	68.4	52.2	33.1	58.3	68.0	49.3
Conan-7B [32]	55.5	62.8	54.5	38.2	59.2	64.0	25.4
LongVT-RFT-7B [60]	59.5	66.0	54.7	37.9	59.4	63.4	23.4
SAGE-7B [23]	44.1	52.4	37.4	31.8	49.7	55.7	28.9
VideoZoomer-7B [10]	45.3	48.3	39.6	22.9	46.2	61.6	†
VideoChat-R1.5-7B [57]	51.5	59.3	49.3	34.3	60.7	62.3	—
ParaVT-8B (Ours)	**62.1**	**69.4**	**60.4**	**39.8**	65.0	**68.6**	**50.1**

Table 1 compares ParaVT with existing 7–8B models. ParaVT leads the comparable open-source baselines on six of seven evaluation splits and improves on Qwen3-VL-8B across all splits. Relative

gains reach $+15.7\%$ on LongVideoBench, $+20.2\%$ on LVBench, and $+11.5\%$ on MLVU, averaging $+7.9\%$ across the seven splits. It reaches 50.1 mIoU on Charades-STA and exceeds the GPT-4o reference on LVBench and MMVU.

At inference, 94.1% of $11{,}460$ evaluated samples issue multiple crops within one turn (Table 2). Parallel dispatch also uses fewer prompt tokens and main-model calls than sequential dispatch, with mean latency decreasing from 80.1 to 48.5 s (Table 3).

Table 2: **Tool Use at Inference.** Greedy decoding with a 64-frame overview. Tool use and parallel dispatch are the fractions of samples issuing at least one crop and at least two crops in one turn, respectively. Crops/turn is averaged over crop-emitting turns. MMVU uses a $1{,}000$-question inference set.

Benchmark	N	**Tool use**	**Crops/turn**	**Parallel**
VideoMME w/o sub	2,700	100.0%	3.53	97.7%
VideoMME w/ sub	2,700	99.6%	3.55	96.4%
LongVideoBench	1,337	99.9%	2.76	82.8%
LVBench	1,549	99.9%	2.96	83.8%
MLVU	2,174	100.0%	3.59	98.3%
MMVU	1,000	100.0%	4.22	99.8%

Table 3: **Inference Cost.** Per-query tokens, main-model calls and latency (mean/p95) at 64-way client concurrency. Peak memory is per device.

Dispatch	**Prompt tok.**	**Decode tok.**	**Calls**	**Latency (s)**	**Memory (GB)**
Sequential	136,502	519	4.23	80.1/168.3	185
Parallel	77,380	267	2.40	48.5/93.8	165

4.3 Ablation Studies

Table 4: **Ablation Studies.** Each row reports mean training-time format reward f_τ at sampling temperature $\tau{=}0.7$ and mean number of training-time tool calls per rollout κ. Both are statistics of the RL training stream, not of inference. The best result is in **bold**; underline marks ParaVT's value when it is not the block-wise best. Rows shaded gray mark the full recipe. The PARA-GRPO-only row applies PARA-GRPO to the base model without the cold start. Block C compares inference-time dispatch modes on the same trained checkpoint. Thus, f_τ and κ are identical across its rows and reported as "-".

Setting	f_τ	κ	**VideoMME** w/o sub	**VideoMME** w/ sub	**LongVideo-Bench**	**LVBench**	**MLVU**	**MMVU**
(A) Training Stage								
Qwen3-VL-8B	0.03	0.45	59.9	68.4	52.2	33.1	58.3	68.0
+ SFT Cold-Start	0.13	2.50	60.7	69.0	58.0	39.1	63.7	67.7
+ SFT Cold-Start + GRPO	0.13	0.02	62.0	68.6	57.5	39.3	64.5	67.5
+ PARA-GRPO only (no cold start)	—	—	62.0	**69.4**	58.6	39.3	64.7	68.5
+ SFT Cold-Start + PARA-GRPO	0.41	0.21	**62.1**	**69.4**	**60.4**	**39.8**	**65.0**	**68.6**
(B) Component Effectiveness								
Qwen3-VL-8B + Cold-Start + GRPO	0.13	0.02	62.0	68.6	57.5	39.3	64.5	67.5
+ Exploration Anchoring	0.35	0.19	61.7	68.7	59.6	39.3	64.1	67.2
+ nFrames Gating	0.10	1.36	61.3	68.7	58.4	39.1	63.6	65.3
Full PARA-GRPO	0.41	0.21	**62.1**	**69.4**	**60.4**	**39.8**	**65.0**	**68.6**
PARA-GRPO − Tool Reward R_{tool}	0.33	0.04	61.9	68.5	57.2	38.7	64.3	67.4
PARA-GRPO − Penalty Term γ	0.36	0.27	61.6	69.0	58.9	38.8	64.5	67.7
(C) Dispatch Mode								
Sequential Tool Calling	-	-	61.4	68.8	57.7	37.5	64.1	66.7
Parallel Tool Calling	-	-	**62.1**	**69.4**	**60.4**	**39.8**	**65.0**	**68.6**

Training Stage. As shown in Block A of Table 4, the cold-started checkpoint over-uses tools ($\kappa{=}2.50$) by directly imitating tool-using demonstrations from SFT traces. Vanilla GRPO swings to

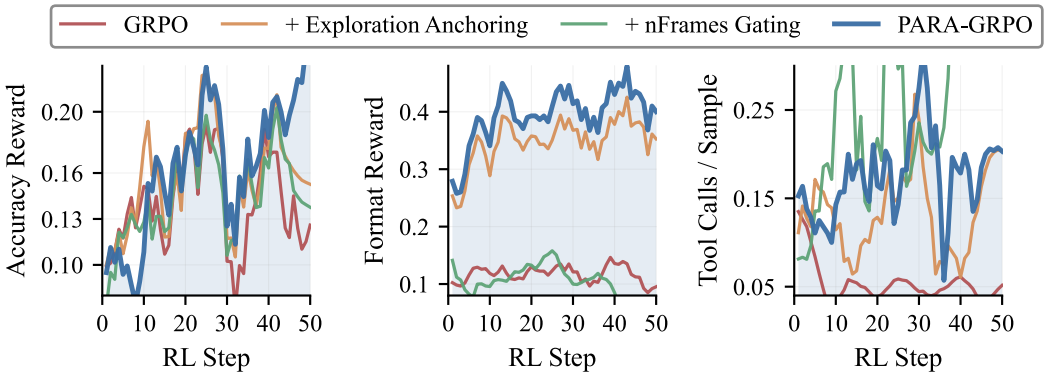


Figure 4: **Training Dynamics across PARA-GRPO Components.** Vanilla GRPO (red) stays flat at $f_\tau \approx 0.13$ while κ collapses to near zero. Exploration Anchoring (orange) lifts f_τ but keeps κ moderate. nFrames Gating (green) pushes κ off-chart while leaving f_τ low. The full PARA-GRPO (blue) achieves higher format reward while retaining moderate tool use.

the opposite extreme (κ=0.02) by skipping tools within 7 steps under the reward shortcut (Figure 1). PARA-GRPO resolves both extremes, reaching the highest training-time mean format reward in Block A (f_τ=0.41, κ=0.21). It strictly improves on vanilla GRPO across all six evaluation splits, with the largest gains on LongVideoBench and MMVU. The format reward is a shaped score whose scale can change with the ablated reward terms. Separate tag-closure rates are reported in Table 9.

Component Effectiveness. Block B shows complementary effects. Exploration Anchoring reaches (f_τ, κ)=$(0.35, 0.19)$, while nFrames Gating reaches $(0.10, 1.36)$. The full recipe reaches $(0.41, 0.21)$, with higher format reward than either single component and higher accuracy on all six evaluation splits. Removing R_{tool} reduces κ from 0.21 to 0.04 and f_τ from 0.41 to 0.33. Removing the unclosed-tag penalty γ reduces f_τ to 0.36 and costs up to 1.5 pt on LongVideoBench. Tool utility is not determined by call count. We therefore interpret κ jointly with format reward and task accuracy.

Training Dynamics. Figure 4 visualizes the same variant comparison during RL. Vanilla GRPO never recovers either metric: f_τ stays flat near 0.13 and κ collapses to near zero within 7 steps, leaving the policy in the format-shortcut regime as introduced in Section 1. Used in isolation, *Exploration Anchoring* restores format ($f_\tau \to 0.35$) while keeping tool use moderate (κ=0.19). *nFrames Gating* alone pushes tool calls aggressively (κ off-chart toward 1.36) but leaves format stuck near 0.10. The full recipe combines stronger format reward with moderate tool use: f_τ rises past step 45 to a peak (0.64) that neither single component attains, while κ stays at 0.21.

Dispatch Mode. Block C compares parallel and sequential dispatch on the same trained checkpoint. Parallel dispatch improves accuracy on every tested benchmark without retraining, with the largest gains on LongVideoBench and LVBench.

5 Conclusion

We present **ParaVT**, a multi-agent RL framework that dispatches parallel temporal crops in one turn for long-video reasoning, enabling peer-correctable evidence aggregation with fewer main-model generations. PARA-GRPO addresses the format instability and tool skipping observed in the Tool Prior Paradox by combining an anchor reward with per-prompt frame-budget randomization to make tool calls reward-relevant. Supported by a self-curated 97K-sample multi-task SFT corpus and a separate 4,406-sample RL split spanning open-ended QA, multiple-choice, and temporal grounding, ParaVT outperforms existing open-source 7–8B baselines on six of seven long-video evaluation splits. These results support combining format anchoring with tool incentives on the tested backbone and motivate further study as tool capabilities become increasingly internalized in modern base LMMs [1, 43].

Acknowledgments and Disclosure of Funding

This work was supported by Apodex, which provided the compute, storage, and engineering infrastructure for the reported experiments. The authors declare no additional competing interests.

References

[1] Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, et al. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[2] Ayoub Ben Chaliah and Hela Dellagi. Datarus-r1: An adaptive multi-step reasoning llm for automated data analysis. *arXiv preprint arXiv:2508.13382*, 2025.

[3] Boyu Chen, Zhengrong Yue, Siran Chen, Zikang Wang, Yang Liu, Peng Li, and Yali Wang. Lvagent: Long video understanding by multi-round dynamical collaboration of mllm agents. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 20237–20246. IEEE, 2025.

[4] Boyu Chen, Zikang Wang, Zhengrong Yue, Kainan Yan, Chenyun Yu, Yi Huang, Zijun Liu, Yafei Wen, Xiaoxin Chen, Yang Liu, et al. Videochat-m1: Collaborative policy planning for video understanding via multi-agent reinforcement learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 33772–33783, 2026.

[5] Guo Chen, Zhiqi Li, Shihao Wang, Jindong Jiang, Yicheng Liu, Lidong Lu, De-An Huang, Wonmin Byeon, Matthieu Le, Max Ehrlich, et al. Eagle 2.5: Boosting long-context post-training for frontier vision-language models. *Advances in Neural Information Processing Systems*, 38: 91077–91100, 2026.

[6] Kerui Chen, Jinglu Wang, Jianrong Zhang, Ming Li, Yan Lu, and Hehe Fan. Scaling video understanding via compact latent multi-agent collaboration. *arXiv preprint arXiv:2605.00444*, 2026.

[7] Yukang Chen, Wei Huang, Baifeng Shi, Qinghao Hu, Hanrong Ye, Ligeng Zhu, Zhijian Liu, Pavlo Molchanov, Jan Kautz, Xiaojuan Qi, et al. Scaling rl to long videos. *Advances in Neural Information Processing Systems*, 38:172842–172870, 2026.

[8] Hyungjin Chung, Hyelin Nam, Jiyeon Kim, Hyojun Go, Byeongjun Park, Junho Kim, Joonseok Lee, Seongsu Ha, and Byung-Hoon Kim. Video parallel scaling: Aggregating diverse frame subsets for videollms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8972–8982, 2026.

[9] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025.

[10] Yang Ding, Xin Lai, Yizhen Zhang, Wei Li, Ruihang Chu, and Yujiu Yang. Videozoomer: Reinforcement-learned temporal focusing for long video reasoning. In *International Conference on Learning Representations*, volume 2026, pages 20087–20111, 2026.

[11] Mohamed Eltahir, Ali Habibullah, Yazan Alshoibi, Lama Ayash, Tanveer Hussain, and Naeemullah Khan. Videoatlas: Navigating long-form video in logarithmic compute. *arXiv preprint arXiv:2603.17948*, 2026.

[12] Kaituo Feng, Kaixiong Gong, Bohao Li, Zonghao Guo, Yibing Wang, Tianshuo Peng, Junfei Wu, Xiaoying Zhang, Benyou Wang, and Xiangyu Yue. Video-r1: Reinforcing video reasoning in mllms. *Advances in Neural Information Processing Systems*, 38:99114–99137, 2026.

[13] Chaoyou Fu, Yuhan Dai, Yongdong Luo, Lei Li, Shuhuai Ren, Renrui Zhang, Zihan Wang, Chenyu Zhou, Yunhang Shen, Mengdan Zhang, et al. Video-mme: The first-ever comprehensive evaluation benchmark of multi-modal llms in video analysis. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 24108–24118. IEEE, 2025.

[14] Shenghao Fu, Qize Yang, Yuan-Ming Li, Xihan Wei, Xiaohua Xie, and Wei-Shi Zheng. Love-r1: Advancing long video understanding with an adaptive zoom-in mechanism via multi-step reasoning. *arXiv preprint arXiv:2509.24786*, 2025.

[15] Wei Fu, Jiaxuan Gao, Xujie Shen, Chen Zhu, Zhiyu Mei, Chuyi He, Shusheng Xu, Guo Wei, Jun Mei, Jiashu Wang, et al. Areal: A large-scale asynchronous reinforcement learning system for language reasoning. *Advances in Neural Information Processing Systems*, 38:36256–36282, 2026.

[16] Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 5277–5285. IEEE, 2017.

[17] Haonan Ge, Yiwei Wang, Kai-Wei Chang, Hang Wu, and Yujun Cai. Framemind: Frame-interleaved video reasoning via reinforcement learning. *arXiv preprint arXiv:2509.24008*, 2025.

[18] Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

[19] Zefeng He, Xiaoye Qu, Yafu Li, Siyuan Huang, Daizong Liu, and Yu Cheng. Framethinker: Learning to think with long videos via multi-turn frame spotlighting. In *International Conference on Learning Representations*, volume 2026, pages 74904–74933, 2026.

[20] Wenxuan Huang, Bohan Jia, Shaosheng Cao, Zheyu Ye, Zhe Xu, Yao Hu, Shaohui Lin, et al. Vision-r1: Incentivizing reasoning capability in multimodal large language models. In *International Conference on Learning Representations*, volume 2026, pages 63794–63812, 2026.

[21] Hugging Face. Open r1: A fully open reproduction of deepseek-r1. `https://github.com/huggingface/open-r1`, January 2025.

[22] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024.

[23] Jitesh Jain, Jialuo Li, Zixian Ma, Jieyu Zhang, Chris Dongjoo Kim, Sangho Lee, Rohun Tripathi, Tanmay Gupta, Christopher Clark, Humphrey Shi, et al. Sage: Training smart any-horizon agents for long video reasoning with reinforcement learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 41478–41488, 2026.

[24] Sehoon Kim, Suhong Moon, Ryan Tabrizi, Nicholas Lee, Michael W Mahoney, Kurt Keutzer, and Amir Gholami. An llm compiler for parallel function calling. *arXiv preprint arXiv:2312.04511*, 2023.

[25] Xinhao Li, Ziang Yan, Desen Meng, Lu Dong, Xiangyu Zeng, Yinan He, Yali Wang, Yu Qiao, Yi Wang, and Limin Wang. Videochat-r1: Enhancing spatio-temporal perception via reinforcement fine-tuning. *arXiv preprint arXiv:2504.06958*, 2025.

[26] Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the association for computational linguistics*, 12:157–173, 2024.

[27] Runtao Liu, Ziyi Liu, Jiaqi Tang, Yue Ma, Renjie Pi, Jipeng Zhang, and Qifeng Chen. Longvideoagent: Multi-agent reasoning with long videos. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 40404–40416, 2026.

[28] Xiao Liu, Nayu Liu, Junnan Zhu, Ruirui Chen, Guohui Xiang, Changjian Wang, Kaiwen Wei, Rongzhen Li, and Jiang Zhong. Retool-video: Recursive tool-using video agents with meta-augmented tool grounding. *arXiv preprint arXiv:2605.13228*, 2026.

[29] Fuwen Luo, Shengfeng Lou, Chi Chen, Ziyue Wang, Chenliang Li, Weizhou Shen, Jiyue Guo, Peng Li, Ming Yan, Ji Zhang, et al. Museg: Reinforcing video temporal understanding via timestamp-aware multi-segment grounding. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 35549–35561, 2026.

[30] Haoming Meng, Kexin Huang, Shaohang Wei, Chiyu Ma, Shuo Yang, Guoyin Wang, Bolin Ding, Jingren Zhou, et al. Sparse but critical: A token-level analysis of distributional shifts in rlvr fine-tuning of llms. In *International Conference on Learning Representations*, volume 2026, pages 124171–124210, 2026.

[31] Zhanfeng Mo, Xingxuan Li, Yuntao Chen, and Lidong Bing. Multi-agent tool-integrated policy optimization. *arXiv preprint arXiv:2510.04678*, 2025.

[32] Kun Ouyang, Yuanxin Liu, Linli Yao, Yishuo Cai, Hao Zhou, Fandong Meng, Jie Zhou, and Xu Sun. Conan: Progressive learning to reason like a detective over multi-scale visual evidence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 41089–41099, 2026.

[33] Jiayi Pan, Xiuyu Li, Long Lian, Charlie Snell, Yifei Zhou, Adam Yala, Trevor Darrell, Kurt Keutzer, and Alane Suhr. Learning adaptive parallel reasoning with language models. *arXiv preprint arXiv:2504.15466*, 2025.

[34] Junwen Pan, Qizhe Zhang, Rui Zhang, Ming Lu, Xin Wan, Yuan Zhang, Chang Liu, and Qi She. Timesearch-r: Adaptive temporal search for long-form video understanding via self-verification reinforcement learning. In *International Conference on Learning Representations*, volume 2026, pages 8376–8405, 2026.

[35] Ziqi Pang and Yu-Xiong Wang. Mr. video:" mapreduce" is the principle for long video understanding. *arXiv preprint arXiv:2504.16082*, 2025.

[36] Xiangyu Qi, Ashwinee Panda, Kaifeng Lyu, Xiao Ma, Subhrajit Roy, Ahmad Beirami, Prateek Mittal, and Peter Henderson. Safety alignment should be made more than just a few tokens deep. In *International Conference on Learning Representations*, volume 2025, pages 54911–54941, 2025.

[37] Qwen Team. Qwen2.5-VL technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[38] Mohit Raghavendra, Vaskar Nath, and Sean Hendryx. Revisiting the superficial alignment hypothesis. *arXiv preprint arXiv:2410.03717*, 2024.

[39] Xiaoqian Shen, Min-Hung Chen, Yu-Chiang Frank Wang, Mohamed Elhoseiny, and Ryo Hachiuma. Zoom-zero: Reinforced coarse-to-fine video understanding via temporal zoom-in. *arXiv preprint arXiv:2512.14273*, 2025.

[40] Joar Skalse, Nikolaus Howe, Dmitrii Krasheninnikov, and David Krueger. Defining and characterizing reward gaming. *Advances in neural information processing systems*, 35:9460–9471, 2022.

[41] Alex Su, Haozhe Wang, Weiming Ren, Fangzhen Lin, and Wenhu Chen. Pixel reasoner: Incentivizing pixel space reasoning via curiosity-driven reinforcement learning. *Advances in Neural Information Processing Systems*, 38:8222–8251, 2026.

[42] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[43] Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, Jianfeng Cai, Xinyuan Cai, Peizhou Cao, Yuxuan Cao, Ziwei Chai, Y Charles, et al. Kimi k3: Open frontier intelligence. *arXiv preprint arXiv:2607.24653*, 2026.

[44] Kimi Team, Tongtong Bai, Yifan Bai, Yiping Bao, SH Cai, Yuan Cao, Ziwei Chai, Y Charles, HS Che, Cheng Chen, et al. Kimi k2. 5: Visual agentic intelligence. *arXiv preprint arXiv:2602.02276*, 2026.

[45] Shulin Tian, Ruiqi Wang, Hongming Guo, Penghao Wu, Yuhao Dong, Xiuying Wang, Jingkang Yang, Hao Zhang, Hongyuan Zhu, and Ziwei Liu. Ego-r1: Chain-of-tool-thought for ultra-long egocentric video reasoning. *arXiv preprint arXiv:2506.13654*, 2025.

[46] Jean Vassoyan, Nathanaël Beau, and Roman Plaud. Ignore the kl penalty! boosting exploration on critical tokens to enhance rl fine-tuning. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 6108–6118, 2025.

[47] Chaoyang Wang, Kaituo Feng, Dongyang Chen, Zhongyu Wang, Zhixun Li, Sicheng Gao, Meng Meng, Xu Zhou, Manyuan Zhang, Yuzhang Shang, et al. Adatooler-v: Adaptive tool-use for images and videos. In *Findings of the Association for Computational Linguistics: ACL 2026*, pages 18058–18072, 2026.

[48] Qi Wang, Yanrui Yu, Ye Yuan, Rui Mao, and Tianfei Zhou. Videorft: Incentivizing video reasoning capability in mllms via reinforced fine-tuning. *Advances in neural information processing systems*, 38:4350–4376, 2026.

[49] Shijian Wang, Jiarui Jin, Xingjian Wang, Linxin Song, Runhao Fu, Hecheng Wang, Zongyuan Ge, Yuan Lu, and Xuelian Cheng. Video-thinker: Sparking" thinking with videos" via reinforcement learning. *arXiv preprint arXiv:2510.23473*, 2025.

[50] Sudong Wang, Weiquan Huang, Xiaomin Yu, Zuhao Yang, Hehai Lin, Keming Wu, Chaojun Xiao, Chen Chen, Wenxuan Wang, Beier Zhu, et al. Beyond sft-to-rl: Pre-alignment via black-box on-policy distillation for multimodal rl. *arXiv preprint arXiv:2604.28123*, 2026.

[51] Weihan Wang, Zehai He, Wenyi Hong, Yean Cheng, Xiaohan Zhang, Ji Qi, Ming Ding, Xiaotao Gu, Shiyu Huang, Bin Xu, et al. Lvbench: An extreme long video understanding benchmark. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 22958–22967. IEEE, 2025.

[52] Ye Wang, Ziheng Wang, Boshen Xu, Yang Du, Kejun Lin, Zihan Xiao, Zihao Yue, Jianzhong Ju, Liang Zhang, Dingyi Yang, et al. Time-r1: Post-training large vision language model for temporal video grounding. *Advances in Neural Information Processing Systems*, 38:83330–83364, 2026.

[53] Brandon T Willard and Rémi Louf. Efficient guided generation for large language models. *arXiv preprint arXiv:2307.09702*, 2023.

[54] Haoning Wu, Dongxu Li, Bei Chen, and Junnan Li. Longvideobench: A benchmark for long-context interleaved video-language understanding. *Advances in Neural Information Processing Systems*, 37:28828–28857, 2024.

[55] Yuan Xie, Tianshui Chen, Zheng Ge, and Lionel Ni. Video-mtr: Reinforced multi-turn reasoning for long video understanding. *arXiv preprint arXiv:2508.20478*, 2025.

[56] Zelai Xu, Zhexuan Xu, Ruize Zhang, Chunyang Zhu, Shi Yu, Weilin Liu, Quanlu Zhang, Wenbo Ding, Chao Yu, and Yu Wang. Wideseek-r1: Exploring width scaling for broad information seeking via multi-agent reinforcement learning. *arXiv preprint arXiv:2602.04634*, 2026.

[57] Ziang Yan, Yinan He, Xinhao Li, Zhengrong Yue, Xiangyu Zeng, Yali Wang, Yu Qiao, Limin Wang, and Yi Wang. Videochat-r1. 5: Visual test-time scaling to reinforce multimodal reasoning by iterative perception. *Advances in Neural Information Processing Systems*, 38:119152–119184, 2026.

[58] Zhongyu Yang, Zuhao Yang, Shuo Zhan, Tan Yue, Wei Pang, and Yingfang Yuan. Svagent: Storyline-guided long video understanding via cross-modal multi-agent collaboration. *arXiv preprint arXiv:2604.05079*, 2026.

[59] Zuhao Yang, Yingchen Yu, Yunqing Zhao, Shijian Lu, and Song Bai. Timeexpert: An expert-guided video llm for video temporal grounding. In *2025 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 24286–24296. IEEE, 2025.

[60] Zuhao Yang, Sudong Wang, Kaichen Zhang, Keming Wu, Sicong Leng, Yifan Zhang, Bo Li, Chengwei Qin, Shijian Lu, Xingxuan Li, et al. Longvt: Incentivizing "thinking with long videos" via native tool calling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 33816–33826, 2026.

[61] Jinhui Ye, Zihan Wang, Haosen Sun, Keshigeyan Chandrasegaran, Zane Durante, Cristobal Eyzaguirre, Yonatan Bisk, Juan Carlos Niebles, Ehsan Adeli, Li Fei-Fei, et al. Re-thinking temporal search for long-form video understanding. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8579–8591. IEEE, 2025.

[62] Qiying Yu, Zheng Zhang, Ruofei Zhu, Yufeng Yuan, Xiaochen Zuo, Yu Yue, Weinan Dai, Tiantian Fan, Gaohong Liu, Lingjun Liu, et al. Dapo: An open-source llm reinforcement learning system at scale. *Advances in Neural Information Processing Systems*, 38:113222–113244, 2026.

[63] Xiangyu Zeng, Zhiqiu Zhang, Yuhan Zhu, Xinhao Li, Zikang Wang, Changlian Ma, Qingyu Zhang, Zizheng Huang, Kun Ouyang, Tianxiang Jiang, et al. Video-o3: Native interleaved clue seeking for long video multi-hop reasoning. *arXiv preprint arXiv:2601.23224*, 2026.

[64] Congzhi Zhang, Zhibin Wang, Yinchao Ma, Jiawei Peng, Yihan Wang, Qiang Zhou, Jun Song, and Bo Zheng. Rewatch-r1: Boosting complex video reasoning in large vision-language models through agentic data synthesis. In *International Conference on Learning Representations*, volume 2026, pages 140958–140985, 2026.

[65] Haoji Zhang, Xin Gu, Jiawen Li, Chixiang Ma, Sule Bai, Chubin Zhang, Bowen Zhang, Zhichao Zhou, Dongliang He, and Yansong Tang. Thinking with videos: Multimodal tool-augmented reinforcement learning for long video reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 32903–32914, 2026.

[66] Kaichen Zhang, Keming Wu, Zuhao Yang, Bo Li, Kairui Hu, Bin Wang, Xingxuan Li, and Lidong Bing. Openmmreasoner: Pushing the frontiers in multimodal reasoning with an open and general recipe. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19276–19286, 2026.

[67] Xiaoyi Zhang, Zhaoyang Jia, Zongyu Guo, Jiahao Li, Bin Li, Houqiang Li, and Yan Lu. Deep video discovery: Agentic search with tool use for long-form video understanding. *Advances in Neural Information Processing Systems*, 38:89863–89895, 2026.

[68] Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Llava-video: Video instruction tuning with synthetic data. *arXiv preprint arXiv:2410.02713*, 2024.

[69] Shu Zhao, Tan Yu, Anbang Xu, Japinder Singh, Aaditya Shukla, and Rama Akkiraju. Parallelsearch: Train your llms to decompose query and search sub-queries in parallel with reinforcement learning. *arXiv preprint arXiv:2508.09303*, 2025.

[70] Yilun Zhao, Haowei Zhang, Lujing Xie, Tongyan Hu, Guo Gan, Yitao Long, Zhiyuan Hu, Weiyuan Chen, Chuhan Li, Zhijian Xu, et al. Mmvu: Measuring expert-level multi-discipline video understanding. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8475–8489. IEEE, 2025.

[71] Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36:55006–55021, 2023.

[72] Junjie Zhou, Yan Shu, Bo Zhao, Boya Wu, Zhengyang Liang, Shitao Xiao, Minghao Qin, Xi Yang, Yongping Xiong, Bo Zhang, et al. Mlvu: Benchmarking multi-task long video understanding. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13691–13701. IEEE, 2025.

Appendix

- **Limitations and Broader Impact** (Section A): scope limits and dual-use considerations.
- **Implementation Details** (Section B): hardware, SFT data composition and curation pipeline (sequential→parallel conversion, Gemini-CoT distillation, format/storage), RL data and DAPO-inspired offline difficulty filtering, optimizer and reward coefficients, and token-budget accounting.
- **Prompts and Templates** (Section C): illustrative prompt structures, source-specific SFT layouts, and evaluation prompt classes.
- **Relation to Prior Work** (Section D): component attribution and related reward designs.
- **Porting Checklist** (Section E): adapting the recipe to another tool-native LMM.
- **Rollout Examples** (Section F): three schematic examples illustrating format collapse and its mitigation under PARA-GRPO.
- **Cross-Model Evidence** (Section G): per-tag format closure breakdown (Table 9) and the two-model before/after trajectory (Figure 5), extending Figure 2.
- **Tool Usage Patterns** (Section H): training-time tool-call trajectories under three reward configurations (Figure 6).
- **Negative Results** (Section I): grouped by intervention axis: reward-shape (phase-reward staging, task-aware reward coefficients), data-shape (Pre-RFT, stronger cold-start), and gradient/format-shape (Token-Decoupled GRPO structural mask, bidirectional tag reversion).

A Limitations, Broader Impact, and Future Work

Limitations. *(i)* The RL stage delivers its primary contribution as training-time format and tool-use stability under temperature sampling rather than as a large standalone greedy-eval delta on top of the cold-started checkpoint. Further amplifying the eval-time translation is an open direction. *(ii)* The cross-model evidence for the role of the prior comes from a single Qwen2.5-VL vs. Qwen3-VL contrast (consistent with a causal interpretation but not equivalent to a controlled intervention). The full PARA-GRPO pipeline has only been validated on Qwen3-VL-8B. Its intended scope is LMMs that already carry a usable tool prior: on a weak-prior backbone the format machinery transfers but no tool behavior emerges for it to stabilize. Porting the recipe to other model families requires mapping their reasoning, answer, and tool-call tags (Section E). A controlled sweep over prior strength across families is future work. *(iii)* Only the `crop_video` tool is evaluated. Whether the recipe generalizes to other tool families (text retrieval, scene-graph extraction, audio transcription) is open.

Broader Impact. Agentic long-video understanding lowers the human cost of searching extended footage by content, with applications in accessibility, sports analytics, and archival retrieval. The same capability also reduces the marginal cost of large-scale surveillance over CCTV or body-camera streams, and ParaVT's parallel-tool dispatch amplifies that throughput rather than restraining it. Deployment in such contexts should be paired with explicit consent and transparency frameworks. The PARA-GRPO training recipe is tool-agnostic and could be retargeted to tool families with different safety profiles (*e.g.,* document or person retrieval), so the dual-use surface is broader than the `crop_video` tool we evaluate. We release code, data, and weights to enable independent audit but withhold surveillance-specific finetunes. Downstream users adapting PARA-GRPO to higher-risk tool families should conduct their own impact assessment.

Future Work. *(i)* Scaling PARA-GRPO to larger LMMs (32B–72B) where richer base capabilities may make RL exploration more effective. *(ii)* Extending necessity gating to other agentic settings where tool necessity is not guaranteed, such as retrieval-augmented generation and code execution.

B Implementation Details

Hardware. All experiments are conducted on 2 machines, each with 8× NVIDIA GPUs ($\geq$80 GB VRAM each). For RL training, we use 7 GPUs for FSDP parameter updates and 1 GPU for SGLang inference serving.

Base Model. We use Qwen3-VL-8B-Instruct [1] as the base LMM. Training overview processing is configured at 1 fps with a 64-frame cap. The released evaluation driver instead samples 64 overview frames uniformly. Temporal crops use 1 fps sampling with at most 16 frames.

Training Infrastructure. The AReaL framework pipelines rollout generation (GPU 0, SGLang) with FSDP training (GPUs 1–7). After the first step, rollout wait drops from $\sim$500s to $<$2s due to pipelining. Each training step takes approximately 50 minutes, including $\sim$35 minutes for rollout and $\sim$15 minutes for parameter updates. We set `SGLANG_VLM_CACHE_SIZE_MB=4096` to accommodate 64-frame video embeddings ($\sim$134 MB each).

Multi-Agent Rollouts and Loss Masking. In RL rollouts, every `<tool_call>` parsed from a main-agent turn launches a sub-agent, a fresh single-turn call of the current policy. Its input holds only the frames of the requested window, decoded at 1 fps with at most 16 frames of at most 224×224 pixels, and a fixed, query-agnostic prompt. The prompt asks for a concise summary of the segment. The calls of one turn run concurrently. Each reply is truncated to 512 tokens and appended to the main agent's context as a user-role `<tool_response>` turn. Sub-agent generations are excluded from the loss: the main-agent response receives a sequence-level scalar reward, and its group-normalized advantage weights the policy loss across all trained main-agent tokens. The optimized object is the main agent's dispatch and aggregation behavior, and sub-agents change only through the shared weights. In SFT, summaries are also supplied as user-role context, and multiple `<tool_response>` blocks may share one user turn. Of the 96,768 SFT samples, 29,122 (30.1%) contain `<tool_response>` blocks, all of them in user-role turns. The SFT loss is computed on assistant turns only, so the summaries are consumed as context but never supervised.

Evaluation Harness. Tool-augmented models in Table 1 are evaluated with greedy decoding in a multi-turn harness that allows at most 3 tool turns and requests a final answer after the last one. Each crop returns its frames, decoded as above, directly to the calling model, and ParaVT may issue up to 5 crops per turn. Each baseline keeps its own prompt class and tool schema (Section C).

SFT Configuration. SFT uses the lmms-engine framework with FSDP. We train for 1,500 steps total, with checkpoints at every 100 steps. Learning rate: 2×10^{-5}, batch size: 32, optimizer: AdamW. The cold-started (step 500) checkpoint is selected as the RL initialization based on training-time format stability under temperature sampling (Section 4.3).

SFT Data (97K samples). The SFT training set contains 97K samples from 7 sources:

Table 5: **SFT data composition.** "Tool" indicates whether the sample contains parallel `crop_video` calls.

Dataset	Samples	Tool?	Description
LLaVA-Video-178K (subsampled)	50K	No	General video QA from the VideoR1 training pool
Self-trace (LongVT rollouts)	15K	Yes	Sequential→parallel converted tool traces
Charades-STA	12K	No	Temporal grounding (`[start, end]`)
TVG (Charades-STA)	6K	Yes	Temporal grounding with parallel tool calls
Long video reasoning	5K	No	Multi-step reasoning on 5–60 min videos
Gemini-CoT (distilled)	5K	Yes	Tool traces generated by Gemini-2.5-Flash
MUSEG (multi-segment)	2.5K	Yes	Avg 4.4 parallel `crop_video` calls/sample

Tool-Augmented Fraction. Tool-bearing samples comprise 30% of the SFT mix. This fraction was set by a Plan A vs. Plan B comparison: an earlier Plan A (212K total, 14% tool) produced weaker tool-calling behavior in downstream RL than the current Plan B (97K total, 30% tool), despite having more raw samples. We read this as evidence that the fraction of tool-bearing samples matters more than absolute count once a non-trivial volume of non-tool video QA is present, and we did not re-tune the ratio further.

Sequential→Parallel Conversion. The selftrace, Gemini-CoT, and TVG sources start as sequential single-tool LongVT-style traces (one `<tool_call>` per assistant turn, with cropped frames re-injected into the next turn's context). We convert each trace to a single-turn parallel format by merging consecutive *independent* tool calls into one turn. We treat two adjacent calls as independent

when their target windows do not overlap and the tool responses they consumed contain no cross-reference to one another (typical case: "inspect 00:30–00:50" followed by "inspect 02:10–02:25," both grounded in disjoint visual evidence). Calls that fail this check (for example, a follow-up crop refining the timestamps of a previous response) are kept on their own turn. We then replace each tool's frame response with a textual summary of the segment's visual content, drawn from the LongVT model's existing assistant continuation that consumed those frames. The text-summary substitution serves two purposes: it aligns the SFT data with the RL sub-agent's output format (text, not frames) and it keeps context length manageable when several crops appear in the same response. After conversion, MUSEG remains the only source with consistently many parallel calls per turn (~ 4.4 on average). Other sources average close to one call per turn because most LongVT traces issued only one crop to begin with.

Gemini-CoT Distillation. The 5K Gemini-CoT subset is produced by sampling LongVT-selfQA prompts and generating sequential tool traces with Gemini-2.5-Flash, then running them through the same sequential→parallel conversion above. Two practical issues drove additional steps. First, Gemini's content filter refuses certain video-question pairs. For those pairs, we re-issue the prompt to Qwen3-VL-235B as a fallback distiller and accept its trace if it passes downstream validation. Second, raw model outputs occasionally contain JSON-structural noise (unbalanced braces, prose around the tool call). We run a GPT-4o cleanup pass that re-emits each tool call as a strict JSON block and discards any sample violating `start_time` $<$ `end_time` or with an empty answer field.

Format and Storage. All splits are stored as Parquet files with the `messages` column serialized as a JSON string, sidestepping Arrow's schema requirement when messages have heterogeneous tool-call structure. Chat layouts vary by source: direct Charades-STA examples contain a user query and an assistant answer, whereas tool traces include additional tool-response context. The configured video parameters are `max_pixels` $= 50176$, `fps` $= 1$, and `max_frames` $= 64$.

RL Data (4,406 samples). The RL training set aggregates three task families: 1,606 open-ended QA from filtered LongVT-selfQA-v2 (HACS / Ego4D-NaQ source videos), 1,600 multiple-choice from the VideoR1 pool, and 1,200 temporal-grounding queries from the Charades-STA training split. SFT and RL share some Charades training videos; the Charades test split is held out for evaluation. The open-ended QA pool starts from 1,668 raw samples. An offline difficulty filter inspired by DAPO [62] excludes 55 prompts with ground-truth answers longer than 15 words and 7 prompts with unanimously unsuccessful cold-start rollouts. These exclusions are heuristics based on the cold-started policy, not guarantees of zero advantage under later policies. The filter runs once before RL.

RL Configuration. GRPO training uses the AReaL asynchronous RL framework with the following hyperparameters:

Table 6: **RL training hyperparameters.**

Parameter	Value
Learning rate	2×10^{-6}
Warmup	0 steps
Temperature (τ)	0.7
GRPO group size (G)	8
Batch size	7 (must be divisible by the number of FSDP workers (7))
Max new tokens	2048
KL coefficient	0.01
Clip ratio (ϵ)	0.2
Reward bias	-0.2
FORMAT_WEIGHT	1.0
ANCHOR_WEIGHT	0.5
nFrames Gating set	{4, 8, 16, 32, 64}
Anchoring α, β, γ	0.4, 0.3, 0.3
Necessity bonus	disabled (ablation: negligible effect)
Contrastive reward	disabled (ablation: hurt performance)
Phase reward staging	disabled (ablation: 160 steps, no improvement)
Infrastructure	
Framework	AReaL (async RL, FSDP + SGLang)
GPU allocation	1 GPU (SGLang inference) + 7 GPUs (FSDP training)
SGLang VLM cache	4096 MB (`SGLANG_VLM_CACHE_SIZE_MB`)
SGLang watchdog timeout	600 s
Checkpoint save frequency	every 5 steps

Reward Function Details. Instantiating Equation (3) with the released defaults (λ_{fmt}=1.0, λ_{anchor}=0.5) gives:

$$R(x, y) = R_{\text{acc}}(y, a^*) + R_{\text{fmt}}(y) + R_{\text{tool}}(y), \qquad R_{\text{fmt}}(y) = R_{\text{base}}(y) + 0.5 \cdot R_{\text{anchor}}(y).$$

Under the released default (`ANSWER_SUFFIX` on), the base format reward R_{base} assigns partial credit: $+0.2$ for substantive `<think>` content ($\geq$10 chars), $+0.3$ for `<answer>` tag, $+0.2$ for `</answer>` tag, $+0.3$ for correct think$\rightarrow$tool ordering, and $+0.1$ for balanced tag pairs. The anchoring component R_{anchor} is defined in Equation (1) with (α, β, γ)=$(0.4, 0.3, 0.3)$. This sequence-level scalar contributes to the total reward, whose group-normalized advantage is applied across all trained main-agent tokens. The format reward is a shaped score, not a binary parseability rate.

The answer extraction follows a 3-level fallback: (1) content within `<answer>` tags; (2) if no `<answer>` tag, content after `</think>` excluding tool calls; (3) last non-empty line. Early detection of degenerate outputs (responses containing 5+ `<|im_start|>` tokens in under 300 characters) short-circuits to zero reward.

Longer Training and Larger Batches. Table 7 reports the extended-training and larger-batch controls.

Table 7: **RL Training Scale.** Extended training and a larger batch size from the cold-started checkpoint.

Setting	VMME w/o	VMME w/	LongVB	LVBench	MLVU	MMVU
Released checkpoint	62.1	69.4	60.4	39.8	65.0	68.6
Batch 7, 100 steps	62.8	70.2	61.1	39.1	65.5	67.7
Batch 7, 200 steps	62.9	69.8	60.9	39.0	65.3	67.5
Batch 7, 300 steps	62.5	70.9	61.8	41.2	65.6	69.6
Batch 14	62.3	69.2	61.8	40.4	65.2	69.5

Token-Budget Accounting: Parallel vs. Sequential. We distinguish main-agent input-token accounting from end-to-end latency. Consider one batch of K independent crops, with an initial prompt of P tokens and C returned tokens per crop. If each request reprocesses its full context, sequential dispatch requires K request generations followed by a final answer, whereas parallel dispatch requests all crops together and then generates an answer. Ignoring the generated request text, their main-agent input-token totals are approximately

$$T_{\text{seq}}(K) \approx (K+1)P + \frac{K(K+1)}{2}C, \qquad T_{\text{par}}(K) \approx 2P + KC. \tag{4}$$

Here C can represent text summaries during RL or visual tokens in the frame-return evaluation harness. Visual-token counts depend on the processor's spatial and temporal grids, rather than a fixed number per frame. This accounting excludes sub-agent computation and cache reuse; measured latency and memory are reported separately in Table 3.

C Prompts and Templates

The templates below illustrate the reasoning and tool-call structure used in the pipeline. Exact wording and chat layouts can vary by data source and stage.

SFT cold-start system prompt (tool-augmented sources). The following illustrates the tool-augmented prompt structure:

```
You are a video understanding agent.

# Workflow
1. Think inside <think>...</think> about which video
   segments contain the evidence needed to answer.
2. Call tools using <tool_call>...</tool_call> blocks.
   You may issue multiple <tool_call> blocks in one turn
   to inspect different temporal windows in parallel.
3. After receiving <tool_response>, place your final
   answer inside <answer>...</answer>.

# Format
<think>your reasoning here</think>
<tool_call>{"name": "crop_video",
            "arguments": {"video_path": "...",
                          "start_time": ...,
                          "end_time": ...}}</tool_call>
... (more <tool_call> blocks if needed) ...
[After tool responses arrive]
<answer>your final answer</answer>

# Important
- ONLY use <tool_call> with the JSON format above.
- Do NOT use <tool_code>, Python syntax, or any other
  tool format.
- Do NOT call the same temporal window twice.
```

SFT cold-start system prompt (non-tool sources). Non-tool examples use task-appropriate question-answering prompts. The following illustrates reasoning-style supervision; direct grounding examples can instead require only an answer block:

```
You are a helpful video understanding assistant.
First, reason about the question inside <think>...</think>.
Then provide your final answer inside <answer>...</answer>.
```

Evaluation prompts (per baseline class). We select prompt classes according to model training paradigms (Section 4.1). *Direct-answer* (Qwen2.5-VL-Instruct): the question is

asked verbatim, with no system prompt and no required output schema. *Reasoning* (Video-R1, VideoRFT, VideoChat-R1, Video-Thinker, Time-R1, ReWatch-R1): a common `<think>...</think><answer>...</answer>` evaluation template. This does not imply that every reasoning model was trained on identical tags; ReWatch-R1, for example, uses action and observation markers in its original prompt [64]. *Agentic* (Qwen3-VL, Conan, ParaVT): the parallel tool prompt above, which uses Qwen's native `<tool_call>` JSON schema and allows multiple calls per turn. The main-agent/sub-agent role split follows the multi-agent-in-one-model setup of MATPO [31], whose planner prompt instead uses a Model Context Protocol (MCP)-style schema with one tool call per message. The remaining tool-trained baselines use their published native schemas (`<video_zoom>` for VideoZoomer [10], the Context-VLM JSON for SAGE [23], the interleaved multimodal chain-of-tool-thought suffix for LongVT-RFT [60]).

D Relation to Prior Work

Table 8 lists the components of PARA-GRPO that come from prior work. Format rewards and the reasoning template are standard. PARA-GRPO combines an anchor reward with nFrames Gating. In our ablations, the combination improves format reward over either single-component variant while retaining tool use (Section 4.3).

Table 8: **Components of PARA-GRPO and Their Prior Work.**

Prior work	What it contributes	Relation to PARA-GRPO
DeepSeek-R1 [18]	`<think>`/`<answer>` template; rule-based format reward	Adopted (Think Prefix; base format reward R_{base})
Constrained decoding [53]	Grammar-constrained generation over the vocabulary	The Think Prefix is its simplest instance, a fixed prefix
Open-R1 [21]	Equal credit for each reasoning and answer tag	Closest prior to our anchor reward, which also penalizes unclosed reasoning blocks
Datarus-R1 [2]	Structural format reward combined with a semantic reward	Same class as R_{base} in our vanilla GRPO baseline
DAPO [62]	Clip-Higher, dynamic sampling, token-level loss, overlong shaping	Our offline difficulty filter differs from its online sampling (Section 3.1.3)

E Porting Checklist

To apply PARA-GRPO to another tool-native LMM: *(i)* identify the reasoning, answer, and tool-call tags of its chat template and set the Think Prefix to its reasoning opener; *(ii)* point the closure checks of R_{anchor} and the base format reward at those tags; *(iii)* before RL, sample the cold-started policy at the training temperature and record its format reward and tool-call rate, the two quantities PARA-GRPO is meant to keep jointly non-trivial; *(iv)* keep part of the nFrames Gating support below the full overview budget, scaled to the frame limit of the backbone. We regard a port as successful when, relative to vanilla GRPO on the same backbone, the format reward at least doubles, the tool-call rate stays at least half of its pre-RL value, and accuracy does not decrease.

F Rollout Examples

The following schematic examples illustrate the intended tag structure, Format Fragility, and its mitigation. Tool calls are abbreviated as pseudocode; executable calls use the JSON schema documented above. These examples are illustrations rather than verbatim evaluation outputs.

Example 1: SFT Baseline (Greedy Decoding). The first example illustrates the intended structure under greedy decoding:

```
<think>The question asks about... I should examine
the segment from 01:20 to 02:30 more closely.</think>
<tool_call>crop_video("video.mp4", 80, 150)</tool_call>
<tool_call>crop_video("video.mp4", 200, 280)</tool_call>
[tool responses...]
<answer>The person picks up the cup at 01:45.</answer>
```

Example 2: Early RL (Format Collapse). The second illustrates format collapse under temperature sampling (τ=0.7) without PARA-GRPO:

```
<think>Looking at the video, I can see...
[reasoning continues for 500+ tokens]
[no </think> closure]
<tool_code>python
# Switches to an alternative tool format
crop_video("video.mp4", start=80, end=150)
</tool_code>
[no <answer> tag, output truncated]
```

Example 3: PARA-GRPO (Recovered Format). The third illustrates the structure encouraged by PARA-GRPO under sampling:

```
<think>The question requires temporal localization.
With only 8 overview frames, I need to crop specific
segments for detail.</think>
<tool_call>crop_video("video.mp4", 75, 155)</tool_call>
<tool_call>crop_video("video.mp4", 195, 285)</tool_call>
[tool responses...]
<answer>The person picks up the cup around 01:42.</answer>
```

G Cross-Model Evidence

G.1 Per-Tag Format Closure

Table 9 reports the per-tag closure breakdown of the Format Fragility side of the paradox (Section 1). Rows are computed from raw τ=0.7 training-stream rollouts, so they reflect compliance *during* RL exploration. Vanilla GRPO reduces the cold-start-learned closure rates within 9 steps as the policy reward-hacks toward direct answering. PARA-GRPO restores all three rates above the cold-started (step 500) baseline by step 19.

Table 9: **Format Fragility Quantified.** Each row reports the closure rates of the three structural tags in raw τ=0.7 rollouts sampled from training streams. `<think>`: fraction of completions with a properly closed reasoning block. `<tool_call>`: fraction with a closed and JSON-parseable tool-call block, the agentic tag whose collapse most directly disables tool-augmented reasoning. `<answer>`: fraction with a properly bracketed answer block. Vanilla GRPO reduces the cold-start-learned closure rates within 9 steps as the policy reward-hacks toward direct answering. PARA-GRPO restores them via Selective Anchoring, a reward based on tag checks. † marks the full PARA-GRPO recipe.

Configuration (τ=0.7 rollouts)	N	`<think>`	`<tool_call>`	`<answer>`
Cold-start (step 500) (pre-RL baseline)	166	51.8%	47.4%	30.1%
Phase C step 9 (vanilla GRPO, no PARA-GRPO)	56	26.8%	18.2%	19.6%
Constrained only step 9 (no anchor)	56	14.3%	9.5%	10.7%
PARA-GRPO step 9 (anchor active)	56	30.4%	31.7%	28.6%
PARA-GRPO step 19 (anchor stabilized) †	56	**58.9%**	**52.6%**	**41.1%**

G.2 Two-Model Trajectory

Figure 2 in the main body summarizes the cross-model contrast (Qwen2.5-VL vs. Qwen3-VL) at the two endpoints of the prior gradient. Figure 5 provides the complementary before/after view of the same two checkpoints over the full 520-step training horizon.

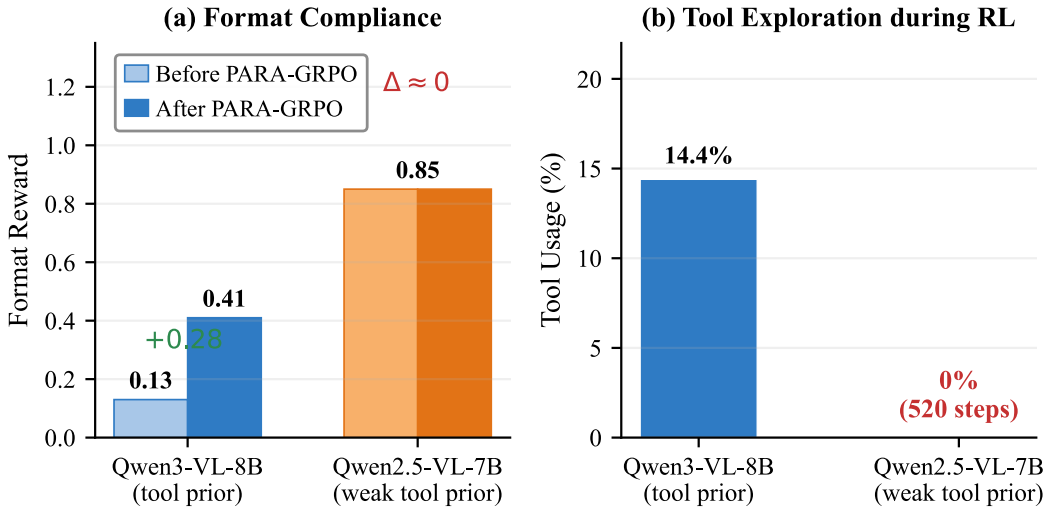


Figure 5: **The Tool Prior Paradox (two-model trajectory).** *(a)* Qwen3-VL's format climbs from 0.13 to 0.41 under PARA-GRPO. Qwen2.5-VL stays near 0.85. *(b)* Qwen3-VL keeps exploring tools (14.4% of rollouts contain a tool call). Qwen2.5-VL emits none over 520 steps. Complements Figure 2.

H Tool Usage Patterns

Figure 6 traces training-time tool-call trajectories under three reward configurations (no penalty, no-tool penalty only, full PARA-GRPO).

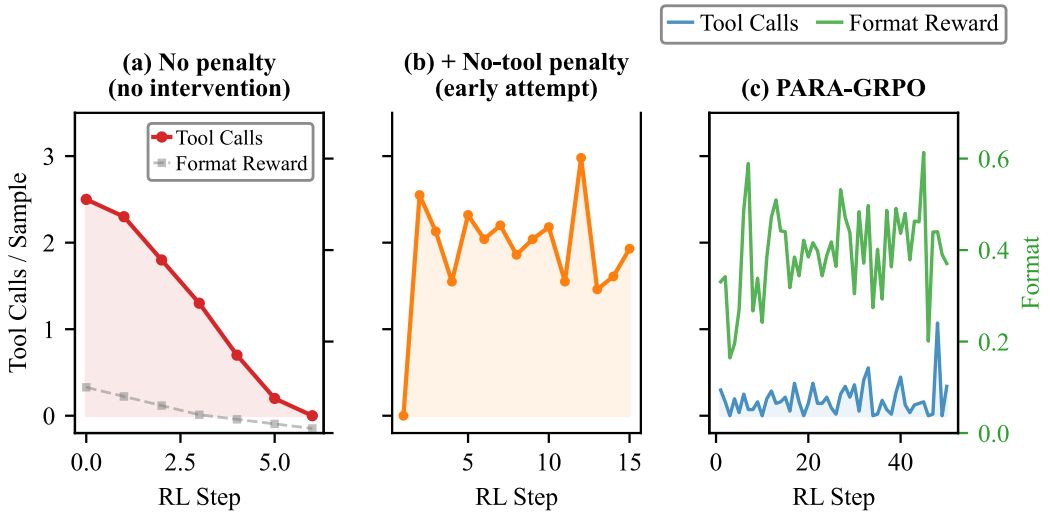


Figure 6: **Training-time tool usage** (rollout averages at τ=0.7, group size 8). *(a)* Early GRPO without intervention: tool usage drops from 2.5 to 0 by step 7 (reward hacking). *(b)* Adding a penalty for skipping tools keeps tool calls but format stays low. *(c)* Under PARA-GRPO, κ stabilizes at 0.1–0.5 while f_τ (green, right axis) climbs to 0.41.

I Negative Results and Failure Modes

We organize the negative results by the axis they intervene on: *reward-shape* (Phase staging, Task-Aware coefficients), *data-shape* (Pre-RFT, Stronger Cold-Start), and *gradient/format-shape* (TD-GRPO mask, Bidirectional tag reversion). Each fails for a distinct reason that further constrains the design space.

I.1 Reward-Shape Interventions

Phase Reward Staging. We first optimized format reward in isolation, planning to introduce accuracy reward once format stabilized. After 160 steps of format-only optimization, f_τ remained at 0.13 with no upward trend, suggesting format and accuracy signals are interdependent: the model needs the accuracy gradient to motivate format learning in the first place.

Task-Aware Reward Coefficients. We added task-aware coefficients ($1.5\times$ for concise MCQ answers, 0.3–$0.4\times$ for verbose ones) on top of PARA-GRPO. Training-time accuracy reward improves from 0.15 to 0.24 over 30 steps and format reward stays comparable at 0.39, but the variant does not outperform base PARA-GRPO on held-out eval: the best checkpoint reaches VideoMME 61.81 (vs. PARA-GRPO's 62.11) and LongVideoBench 58.26 (vs. 60.40). Task-aware shaping improves training signal quality without translating into held-out eval gains, so we keep the simpler unweighted accuracy reward in the default recipe.

I.2 Data-Shape Interventions

Pre-RFT (rejection fine-tuning). We sampled $\tau{=}0.7$ rollouts from the cold-started checkpoint, filtered for format-compliant samples, and mixed them back into SFT training. Subsequent RL from this Pre-RFT init peaked at $f_\tau{=}0.40$, but this refinement degraded the cold-start quality on every downstream metric, so we did not adopt this pre-RL refinement route.

Stronger Cold-Start, Worse RL. Augmenting cold-start data with 12% parallel tool-calling samples (106K vs. 97K) produces a stronger cold-started checkpoint (VideoMME $61.3{\rightarrow}62.3$). RL from this stronger init produces zero tool calls throughout training. Three possible explanations are: *(i)* the stronger model answers correctly without tools even under gating, so the tool-rewarded gradient is averaged out; *(ii)* mixed single/parallel tool patterns in the cold-start data increase Format Fragility; *(iii)* more thorough SFT coverage shifts the policy toward reproducing the cold-start distribution rather than exploring. These coupled effects motivate keeping the cold-start scope to the format schema rather than expanding it into the tool-call distribution itself.

I.3 Gradient and Format-Shape Interventions

Token-Decoupled GRPO (TD-GRPO) Structural Mask. We test a token-decoupled GRPO variant that selectively zeros the policy-gradient contribution of structural tokens (*e.g.,* `<think>`, `<tool_call>`) so that RL only updates semantic content tokens, motivated by token-level analyses of sparse policy changes [30]. After 11 steps, f_τ dropped to 0.11 (*below* baseline 0.13): this masking did not preserve format in the tested run.

Bidirectional Format Reversion. Our main runs use SFT with `<tool_call>`, which is Qwen3-VL's native tool-calling tag: it is present in the tokenizer vocabulary as a single added token (ID 151657) and is the format emitted by the model's default chat template. To probe whether Format Fragility is competition between the SFT tag and inherited tool-format tendencies, we re-run SFT with `<tool_code>` instead: a multi-subword tool-format sequence outside the tokenizer's added vocabulary. The `<tool_code>`-trained model still generates `<tool_call>` in 5.4% of rollouts (despite never seeing it during SFT), while its trained `<tool_code>` appears in only 1.8%. This *bidirectional format reversion* (Figure 7) supports an interpretation of competing tool-format representations rather than a single mismatched tag: neither SFT target eliminates the alternative tag under temperature sampling. The probe does not identify the training stage at which these representations were acquired. The format-substituted model also shows lower total tool emission (7.2% vs. 14.4% of rollouts emit any tool tag), consistent with the probability-mass argument that single-token special tokens are more efficiently reinforced by RL than multi-subword sequences.

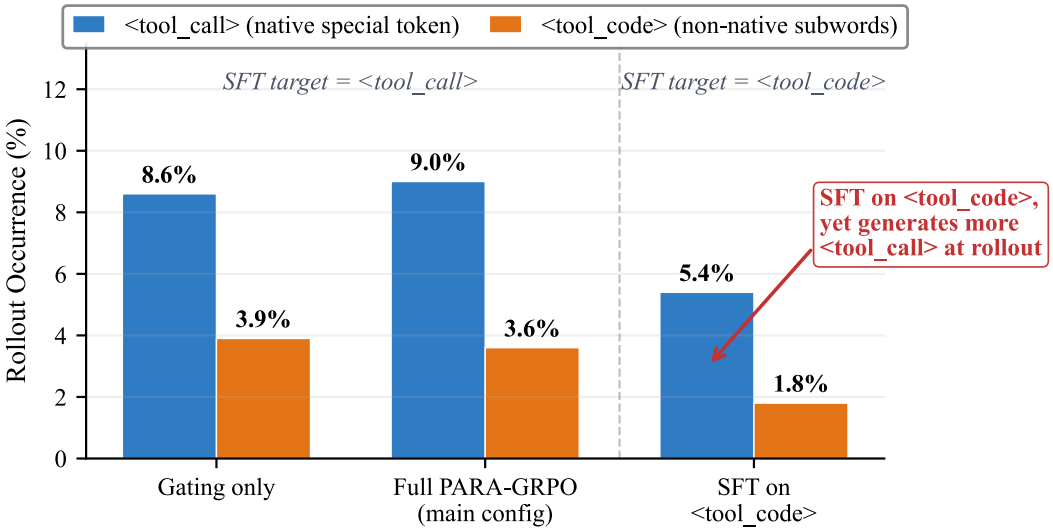


Figure 7: **Bidirectional tag reversion during RL.** SFT with `<tool_call>`: RL also emits `<tool_code>` (3.6–3.9% of rollouts). SFT with `<tool_code>`: RL still emits `<tool_call>` more often (5.4%) than the SFT-trained tag (1.8%). Tag substitution does not remove Format Fragility.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and Section 1 state the four contributions (parallel video tool architecture, identification of the Tool Prior Paradox, the PARA-GRPO framework, and the empirical study). Each is supported by results in Section 4 and the Appendix.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Section A discusses single-model validation of the full PARA-GRPO pipeline (Qwen3-VL-8B only), its scope to backbones with a usable tool prior, single-tool scope (`crop_video`), and the eval-deployment translation.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [N/A]

Justification: The paper is empirical and introduces no formal theorems or proofs. The reward and anchoring formulas in Section 3.2 are definitional rather than theoretical claims.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Section 4.1 specifies the base model, SFT and RL data, training framework (AReaL), and 64-frame agentic eval protocol. Section B lists the full SFT/RL hyperparameters, reward decomposition, infrastructure settings, and the SFT data composition (Table 5).

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: Code (`https://github.com/EvolvingLMMs-Lab/ParaVT`, Apache-2.0), the SFT and RL data (`https://huggingface.co/datasets/ParaVT/ParaVT-Parquet` and `https://huggingface.co/datasets/ParaVT/ParaVT-Source`), and the ParaVT-8B weights (`https://huggingface.co/ParaVT/ParaVT-8B`) are publicly released with instructions to reproduce the main results.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: Training and evaluation details (data splits, optimizer, learning rates, batch size, GRPO group size, temperature, KL coefficient, frame count, eval protocol) are summarized in Section 4.1 and reported in full in Section B (Table 6).

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

Justification: Multi-seed retraining of the full SFT→RL pipeline was beyond our compute budget. We rely on training-time format and tool-use trajectories (Figures 4 and 6) as the primary evidence for the RL contribution, alongside the held-out evaluation in Table 1.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Section B reports the hardware (8 NVIDIA GPUs at $\geq$80 GB VRAM each per RL run, with 7 FSDP + 1 SGLang, with evaluation on nodes of 4 to 8 such GPUs). It also reports the AReaL framework setup and $\sim$50 minutes per RL training step.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work uses public benchmarks and a public base LMM under their original research licenses, and involves no human subjects or private user data. We have reviewed and complied with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.

- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: A Broader Impact paragraph is included in the appendix (Section A).

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: ParaVT is a long-video question-answering model fine-tuned on public benchmark data. The released assets are a video-QA fine-tune and annotations derived from public research datasets. Potential dual-use considerations are discussed in Section A.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: All used assets are cited at first mention.

Guidelines:

- The answer [N/A] means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [Yes]

Justification: The ParaVT-8B weights, the SFT and RL data, and the code are released with documentation (README and model and dataset cards) at the URLs given in the answer on open access. The code is licensed under Apache-2.0.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects. All training and evaluation data come from existing public datasets.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: No human-subjects research is conducted in this paper, so IRB review is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [Yes]

Justification: A large multimodal model (Qwen3-VL-8B) is the research subject of this paper and is the artifact being trained, as detailed in Section B. Gemini-2.5-Flash was additionally used to generate 5K tool-calling traces in the SFT data (Table 5), with Qwen3-VL-235B as a fallback distiller and GPT-4o for a JSON cleanup pass (Section B).

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.