SIMPLESEARCH-VL: A SIMPLE RECIPE FOR EFFICIENT MULTIMODAL AGENTIC SEARCH

Anonymous authors
Paper under double-blind review

ABSTRACT

We present **SimpleSearch-VL**, an efficient, reliable, and practical framework for multimodal agentic search. Its core idea is to improve the agent's own search-and-verification process rather than scaling data, tools, or auxiliary model components. For efficiency, *Factorized Adaptive Rollout* (FAR) improves sampling efficiency by forming more informative training groups while using redundant samples to mitigate long-tail latency and expose hard samples. For reliability, SimpleSearch-VL performs *evidence-verified reasoning*, explicitly using chain-of-thought verification to assess the relevance of retrieved visual and textual cues to the original context. For practicality, SimpleSearch-VL keeps a lightweight tool interface and performs webpage self-summary within the agent, requiring no additional external dependencies. With only 5K supervised tool-interleaved trajectories and 2K RL data, SimpleSearch-VL improves Qwen3-VL agentic baselines by 15.8 and 16.0 average points for the 8B and 30B-A3B variants, respectively. The SimpleSearch-VL-30B-A3B model further achieves performance competitive with agentic Gemini-3-Pro.

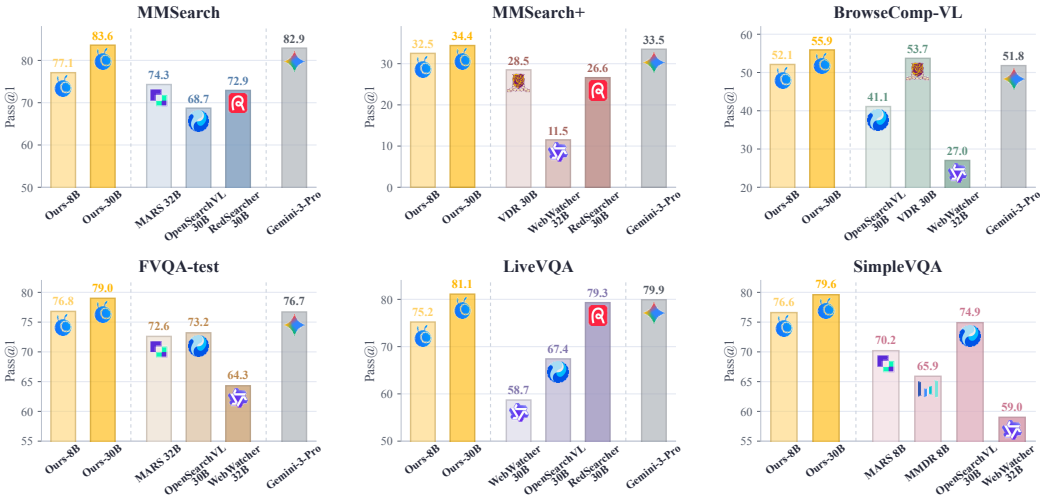


Fig. 1: **Performance comparison with representative multimodal deep search agents.** SimpleSearch-VL-8B outperforms most open-source 30B-scale multimodal deep search agents, while SimpleSearch-VL-30B-A3B achieves performance competitive with agentic Gemini-3-Pro.

1 INTRODUCTION

Agentic search extends large language models from passive response generation to active evidence acquisition. In the language-only setting, recent Deep Search agents have shown that models can answer knowledge-intensive questions by issuing web-search queries, visiting webpages, coding, and synthesizing evidence across multiple sources (Team et al., 2025; MiroMind Team et al., 2025; Du et al., 2026b; Li et al., 2026b). Recent multimodal search agents extend this paradigm to knowledge-intensive visual reasoning by equipping MLLMs with external visual lookup and visual tool use (Wu et al., 2025; Geng et al., 2025; Peng et al., 2026; Hong et al., 2025; Chng et al., 2025; Li et al.,

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

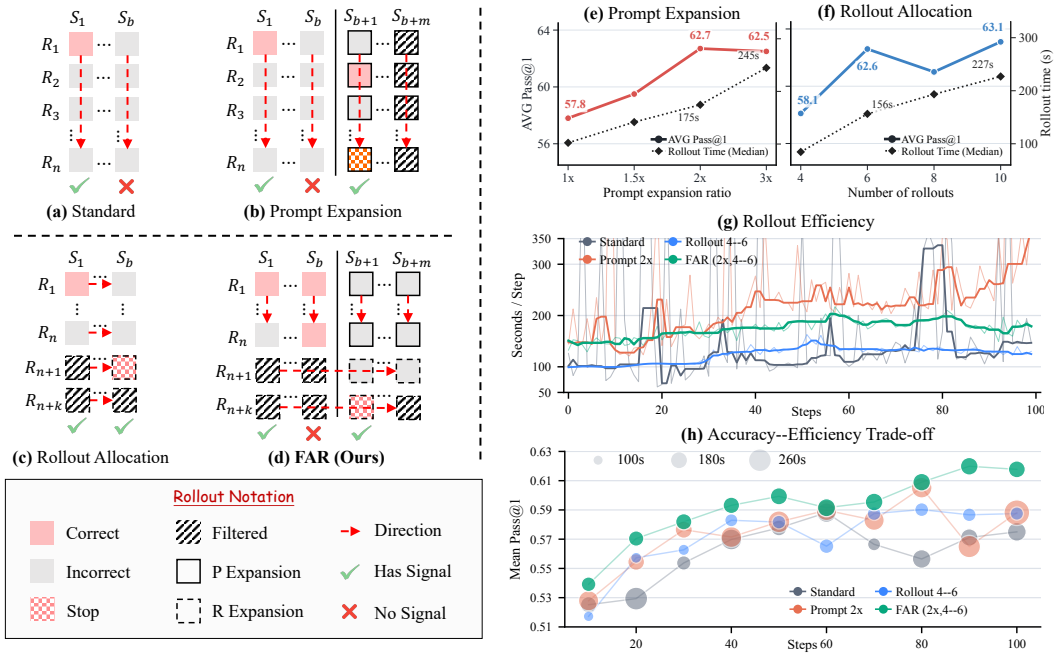


Fig. 2: **Factorized Adaptive Rollout and its empirical motivation.** (a)–(d) contrast standard sampling, Prompt Expansion, Rollout Allocation, and FAR, which allocates additional attempts to hard groups while dynamically masking redundant tail rollouts. (e)–(f) show that expanding either budget can improve accuracy, but uniform expansion increases median rollout time and yields diminishing returns. (g)–(h) compare rollout time and the resulting accuracy–efficiency trade-off across rollout strategies.

2026a). This setting is increasingly important for benchmarks that require external, up-to-date, or entity-specific visual knowledge (Chen et al., 2023; Jiang et al.; Fu et al., 2025; Cheng et al., 2025).

However, multimodal search agents remain difficult to make **efficient**, **reliable**, and **practical**. **Efficiency** is constrained by long-tail rollout generation: each trajectory may involve multi-turn external search and webpage visits, so a small number of samples can consume most rollout-generation time. Fixed rollout budgets further cannot adaptively terminate sampling for prompts that already provide useful reward variation or reallocate attempts to hard prompts that still lack such signal. **Reliability** requires evidence that is grounded and checkable across sources and modalities, since retrieved text, webpages, or image-search titles and URLs can appear plausible yet remain unsupported or mismatched. **Practicality** favors reproducible systems that avoid complex tool orchestration and unnecessary model dependencies. SimpleSearch-VL addresses these challenges with factorized adaptive rollout, evidence-verified reasoning, and self-summarized visit.

Agentic rollout efficiency. Group-based RL estimates relative advantages from multiple responses to each prompt (Shao et al., 2024), motivating dynamic prompt sampling (Yu et al., 2026; Zheng et al., 2026; Hu et al., 2025), adaptive response allocation (Zhang et al., 2025; 2026b; Nguyen et al., 2026), and partial or asynchronous execution (Zhou et al., 2025; Qu et al., 2025; Sheng et al., 2026). Multimodal search makes uniform scaling particularly costly because each rollout may invoke multiple external tools. Our scaling study in Fig. 2(e)–(f) reveals a common trade-off: expanding either the prompt pool or the per-prompt rollout budget improves accuracy, but also increases rollout cost, with diminishing gains under further expansion. These observations motivate a central design question: how can we retain the benefits of both expansion dimensions while allocating additional computation only where it is needed? Rather than applying a fixed expansion uniformly, we seek to combine broader prompt coverage with sample-adaptive rollout allocation, using observed outcomes to determine which groups warrant further attempts. Factorized Adaptive Rollout (FAR) implements this principle by treating both expansion factors as capacity ceilings and dynamically filtering remaining attempts once a group yields an answer-correct trajectory. Breadth-first scheduling and near-completion termination further reduce tool-induced tail latency (Fig. 2(a)–(d)). Together,

these mechanisms balance exploration across prompts, additional sampling within unresolved groups, and realized rollout cost, improving the performance–efficiency trade-off (Fig. 2(g)–(h)); complete diagnostics are provided in Appendix A.5.

Evidence verification. Search tools provide agents with external candidate information, but strong agentic search also requires the model to select evidence that is relevant, supported, and usable for the current question. Text-based search agents can often treat retrieved passages as directly inspectable evidence (Team et al., 2025; Jin et al., 2025). In multimodal search, evidence verification is more delicate because a reverse-image-search candidate may be internally coherent while still failing to match the queried image or region. As illustrated in the middle panel of Fig. 3, a returned candidate can provide a plausible thumbnail, title, and source URL for a visually similar but different object. Although recent multimodal agents improve tool use through supervised trajectories or RL (Wu et al., 2025; Geng et al., 2025; Hong et al., 2025), this visual-to-source validation step is often left implicit. SimpleSearch-VL exposes it directly: the `image_search` tool returns the matched thumbnail, webpage title, and source URL, enabling the model to verify whether the candidate matches the queried visual content before using its title, URL, or webpage evidence. Since the thumbnail is small and used only for consistency checking, it adds little token overhead while making retrieved visual evidence directly checkable.

Tool-interface simplicity. Recent multimodal search agents often implement tool use as coupled pipelines, such as binding search to webpage visiting or chaining reverse image search to subsequent browsing (Chng et al., 2025; Zeng et al., 2026). While effective, these designs can collect broad evidence bundles before the model has decided what information is needed, increasing inference cost and reducing the agent's control over subsequent evidence seeking. Some systems also rely on an external webpage summarizer, which introduces an additional model dependency during training and evaluation (Wu et al., 2025; Chng et al., 2025). SimpleSearch-VL instead exposes decoupled evidence actions: at each step, the model decides which links to visit and summarize, and for multi-image inputs, which image and region should serve as the reverse-image-search query. As shown in the right panel of Fig. 3, each webpage visit is followed by goal-conditioned self-summary, where the agent extracts query-relevant information without calling a separate summarization model. This keeps webpage understanding inside the agent while reducing tool orchestration and model dependencies.

Together, these designs make SimpleSearch-VL an **efficient**, **verifiable**, and **practical** framework for multimodal agentic search. It is also data-efficient and low-cost to train: instantiated with Qwen3-VL (Bai et al., 2025), SimpleSearch-VL learns strong search behavior from 5K supervised tool-interleaved trajectories and 2K RL data, with each variant trained on 8 H200 GPUs in about one day. Despite this modest training scale, the 8B and 30B-A3B variants improve their corresponding Qwen3-VL agentic baselines by 15.8 and 16.0 average points, and the 30B-A3B model reaches performance competitive with agentic Gemini-3-Pro on shared benchmarks.

Our main contributions are summarized as follows:

- We present SimpleSearch-VL, an efficient, verifiable, and practical framework for training multimodal search agents with a lightweight tool interface and reduced dependence on external models.
- We propose Factorized Adaptive Rollout, a signal-aware allocation strategy that expands prompt coverage while using redundant samples to mitigate long-tail stalls and expose hard samples.
- We introduce visual evidence verification with thumbnail-aware reverse image search, enabling the agent to check whether retrieved evidence matches the queried visual content before using it.
- We validate SimpleSearch-VL on six multimodal search benchmarks, where the 8B and 30B-A3B variants improve agentic baselines by 15.8 and 16.0 average points.

2 SIMPLESEARCH-VL

2.1 PRELIMINARIES AND OVERVIEW

Given a visual question Q and input images $\mathcal{I} = \{I_m\}_{m=0}^{M-1}$, SimpleSearch-VL casts multimodal search as a Markov decision process. As illustrated in Fig. 3, the agent alternates between reasoning, tool use, evidence verification, and answer generation. $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R})$:

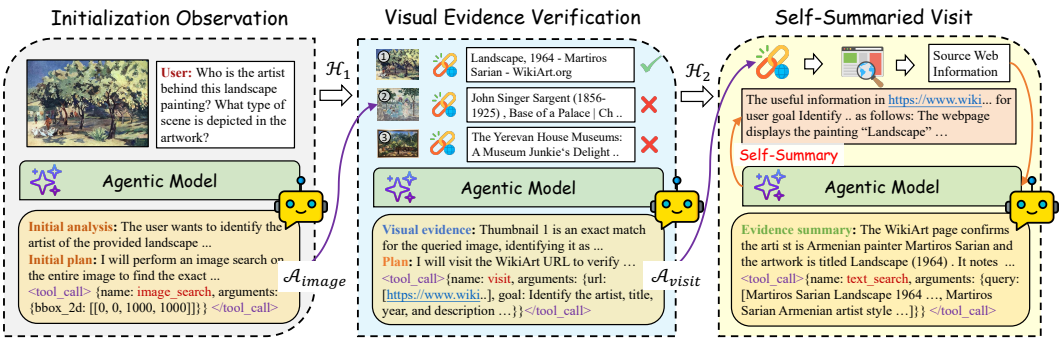


Fig. 3: **Agentic search process of SimpleSearch-VL.** The model alternates between reasoning, tool calls, evidence verification, and final answer generation.

- **State** $s_t \in \mathcal{S}$**:** The state contains the question, images, and accumulated interaction history $\mathcal{H}_t$:

$$s_t = (Q, \mathcal{I}, \mathcal{H}_t), \qquad \mathcal{H}_t = \{(a_1, o_1), \ldots, (a_{t-1}, o_{t-1})\}. \tag{1}$$

  Here a_i is a previous action and o_i is the corresponding tool observation. Conditioning on $\mathcal{H}_t$ makes the state contain all information needed for the next decision.
- **Action** $a_t \in \mathcal{A}$**:** The policy first generates `<thinking>` and then emits exactly one action:

$$a_t \sim \pi_\theta(\cdot \mid s_t), \qquad \mathcal{A} = \{\mathcal{A}_{\text{text}}, \mathcal{A}_{\text{image}}, \mathcal{A}_{\text{visit}}, \mathcal{A}_{\text{answer}}\}. \tag{2}$$

  These actions correspond to text search, reverse image search over selected regions, webpage visit, and final answer generation.
- **Transition** $\mathcal{T}$**:** If a_t is a tool call, $\mathcal{T}$ returns an observation $o_t = \mathcal{T}(s_t, a_t)$ and appends (a_t, o_t) to the history. Text search returns snippets and candidate URLs; `image_search` returns matched thumbnails, webpage titles, and source URLs from reverse image search over queried regions; visit returns a goal-conditioned self-summary of the selected webpage. If $a_t \in \mathcal{A}_{\text{answer}}$, the trajectory terminates.
- **Reward** $\mathcal{R}$**:** During RL, each completed rollout $\tau = (Q, \mathcal{I}, \mathcal{H}_T, a_T)$ receives

$$r(\tau) = 0.5\,\mathbf{1}_{\text{format}}(\tau) + \mathbf{1}_{\text{answer}}(\tau). \tag{3}$$

  Here $\mathbf{1}_{\text{format}}(\tau), \mathbf{1}_{\text{answer}}(\tau) \in \{0, 1\}$ indicate whether τ satisfies the one-action protocol and produces a correct final answer, respectively. The format indicator is one only when every interaction turn is well formed. The answer indicator first uses exact matching and falls back to an LLM judge when exact matching is inconclusive.

2.2 Factorized Adaptive Rollout

Budget factorization. FAR turns fixed-size rollout generation into signal-aware budget allocation for multimodal agentic RL. As illustrated in Fig. 2, it factorizes the budget into two directions: *Prompt Expansion* increases the candidate prompt-group pool, while *Rollout Allocation* allocates extra rollout slots horizontally across groups at the same rollout depth. Appendix Fig. 7 gives the detailed order. FAR first fills the base vertical rollouts for each candidate group, then advances expanded slots by depth across groups, so hard groups receive additional attempts while groups that already provide answer-level variation stop consuming redundant tail slots.

Signal evaluation. For rollout scheduling, FAR uses the answer indicator, rather than the total scalar reward, to identify whether a prompt provides a relative training signal. Let x denote a prompt and let $\mathcal{G}_K(x) = \{\tau_1, \ldots, \tau_K\}$ be its completed rollout group. We mark each rollout by answer correctness and define useful answer-level reward variation as mixed answer-correct and answer-incorrect outcomes:

$$y_i = \mathbf{1}_{\text{answer}}(\tau_i), \qquad \text{Sig}(x; \mathcal{G}_K) = \mathbf{1}\left\{0 < \sum_{i=1}^{K} y_i < K\right\}. \tag{4}$$

Only groups satisfying Eq. 4 are retained for the policy update. Groups with all correct or all incorrect final answers are filtered from the update because they do not provide within-group answer variation.

Rollout filtering. The same signal definition controls which expanded slots remain active. Rollout slots are defined upfront rather than generated by two separate stages. Each candidate group has $n_{\min}$ base slots and at most $n_{\max}$ total slots. After the first k observed rollouts of a group, the next expanded slot remains active only if all observed rollouts are answer-incorrect:

$$\alpha_k(x) = \mathbf{1}\left\{\sum_{i=1}^{k} y_i = 0\right\}, \qquad n_{\min} \le k < n_{\max}, \tag{5}$$

where $\alpha_k(x) = 0$ masks later expanded slots for that group. This "expand-only-all-wrong" rule spends extra attempts on hard groups that still lack an answer-correct trajectory, and filters redundant rollout tails as soon as such a trajectory is observed. The horizontal direction in Fig. 2(c) and the ordered indices in Appendix Fig. 7 show that FAR expands rollouts breadth-first across eligible groups, rather than spending the extra budget deeply on one group at a time. The rightmost panel of Appendix Fig. 7 illustrates this filtering process: after the parallel generation unit returns an answer-correct sample, subsequent allocation slots from the same group are masked before they enter rollout generation.

Stopping rule. A group is counted as valid only after it has completed at least $n_{\min}$ rollouts and satisfies Eq. 4. FAR stops rollout generation when either enough valid groups have been collected for the policy update, or the accounted rollout slots reach a near-complete threshold. The latter counts both completed slots and slots skipped by the rollout mask, and is set to 0.99 of all pre-allocated slots by default. Because FAR creates redundant candidate slots, this condition prevents a training step from waiting for the last few slow or stalled tool-interleaved trajectories.

Rollout selection. After rollout generation, FAR filters groups by Eq. 4. Since a retained group can complete more rollouts than the update budget requires, FAR selects at most $n_{\min}$ trajectories as the update subset $\mathcal{U}(x)$. The default strategy preserves reward diversity and coarse behavior diversity by preferring trajectories with different rewards, reasoning rounds, and expanded tool-call counts. Non-selected rollouts are filtered from the gradient update but still contribute to the full-group outcome estimate below. Let $\mathcal{U}$ denote the union of selected trajectories in a minibatch.

Policy update. FAR uses an RLOO-style clipped surrogate over the adaptively collected rollouts. It computes the leave-one-out baseline from the full completed group, so expanded rollouts can improve the outcome estimate even when only a compact subset enters the update. Because rollout continuation and update-subset construction depend on observed outcomes, we do not claim that this objective is an unbiased estimator of the fixed-group RLOO objective. Here $r(\cdot)$ denotes the reward defined in Eq. 3:

$$\mu_x = \frac{1}{K}\sum_{j=1}^{K} r(\tau_j), \qquad \hat{A}_i = r(\tau_i) - \frac{1}{K-1}\sum_{j\neq i} r(\tau_j) = \frac{K}{K-1}(r(\tau_i) - \mu_x), \quad \tau_i \in \mathcal{U}(x). \tag{6}$$

The scalar advantage is applied only to policy-generated tokens; tool-observation tokens are masked. The policy is optimized with the clipped objective

$$\mathcal{J}(\theta) = \mathbb{E}\left[\frac{1}{|\mathcal{U}|}\sum_{\tau_i\in\mathcal{U}}\frac{1}{|\Omega_i|}\sum_{t\in\Omega_i}\min\left(\rho_{i,t}(\theta)\hat{A}_i,\ \mathrm{clip}(\rho_{i,t}(\theta), 1-\epsilon_{\text{low}}, 1+\epsilon_{\text{high}})\hat{A}_i\right)\right], \tag{7}$$

where Ω_i denotes policy-generated token positions and $\rho_{i,t}$ is the token-level importance ratio. The lower and upper clipping ranges are written separately to match the asymmetric clipping used in implementation.

2.3 Evidence-Verified Reasoning

Evidence verification. SimpleSearch-VL makes retrieved evidence directly checkable inside the model context. Text-search results provide candidate snippets, titles, and URLs for page-level verification, while region-level reverse image search returns thumbnails together with the associated title and URL, allowing the model to compare the visual match against the queried region before using the source. As illustrated in the middle panel of Fig. 3, the agent is trained to follow this explicit chain: inspect candidate evidence, verify whether it supports the current hypothesis, and then choose whether to search, visit, or answer. During SFT data construction, we preserve the original tool calls

and observations but rewrite assistant reasoning to expose these verification decisions; details are given in Appendix C.

Region-level cache. Under Google's API pricing, a reverse-image-search request costs $3\times$ as much as a text-search request, making repeated visual queries a substantial component of rollout cost. Unlike text queries, region-level image queries may differ slightly in their predicted boxes while referring to the same visual instance, so exact-key caching would treat semantically equivalent crops as distinct. We therefore use a region-equivalence cache. Text search and webpage visit are indexed by SHA-256 hashes of the normalized query and URL, respectively. For an image with content hash h and query box b, we reuse the result associated with the most overlapping cached box when

$$b^\star = \arg\max_{\tilde{b}\in\mathcal{C}(h)} \mathrm{IoU}(b,\tilde{b}), \qquad \mathrm{IoU}(b,b^\star) \geq \tau_{\mathrm{cache}}, \tag{8}$$

where $\mathcal{C}(h)$ is the set of cached boxes for the image and $\tau_{\mathrm{cache}} = 0.7$. Otherwise, a new reverse-image-search result is added to the cache. This criterion preserves distinctions between different regions while consolidating small localization variations around the same instance, thereby amortizing image-search cost over repeated rollouts.

2.4 Self-Summarized Visit

SimpleSearch-VL replaces the external webpage summarizer with self-summary by the policy model itself during both training and evaluation. Given selected URLs and an extraction goal, `visit` fetches and normalizes the webpages, and the policy model produces a concise goal-conditioned summary that is appended to the next Markov state in Eq. 1. This avoids deploying an additional summarization service and allows webpage-summary behavior to adapt together with the agentic policy. Table 3 shows that self-summary improves accuracy over an external summarizer and, under a matched two-node budget, also reduces inference time by eliminating the separate summary service.

3 Experiments

3.1 Implementation Details

Training Data. Our supervised fine-tuning data are generated directly with the same agentic search workflow used for RL and evaluation. Specifically, `Qwen3-VL-235B-A22B-Instruct` autonomously produces multi-turn tool-interleaved trajectories, turning the construction of complex search trajectories into a scalable rollout process once the workflow is defined. We then use `gemini-3.1-pro` to audit each trajectory and rewrite only the assistant reasoning, filtering unsupported answers, low-value searches, and weak evidence chains while preserving the original tool calls and observations. The final training pool contains **5K** evidence-aware SFT trajectories, and RL uses **2K** RL data from the same task distribution. For FVQA and LiveVQA, both stages draw examples exclusively from the official training splits; their official test splits are reserved for evaluation, and no test instance is used for SFT or RL. Appendix C provides the detailed construction and quality-control criteria.

Benchmarks We evaluate on six benchmarks that stress different aspects of multimodal web search and visual knowledge reasoning: MMSearch (Jiang et al.), MMSearch+ (Jiang et al.), BrowseComp-VL (Geng et al., 2025), FVQA (Wang et al., 2017), LiveVQA (Fu et al., 2025), and SimpleVQA (Cheng et al., 2025). Together, these benchmarks cover visual entity recognition, open-world information seeking, evidence-grounded web retrieval, multi-hop reasoning.

Tool Definitions. SimpleSearch-VL uses a minimal tool interface for multimodal evidence acquisition: text search, region-level reverse image search, and webpage visit with goal-conditioned self-summary. Complete tool declarations, including each tool's input signature and returned evidence fields, are provided in Appendix B, Table 7.

Training Details. SimpleSearch-VL is instantiated with *Qwen3-VL-8B-Instruct* and *Qwen3-VL-30B-A3B-Instruct* (Bai et al., 2025), using ms-swift (Zhao et al., 2024) for agentic SFT and rLLM (Tan et al., 2025) for agentic RL. On 8 H200 GPUs, the 8B model takes about 2 hours for SFT and 16 hours for RL; the 30B-A3B model takes about 3 hours for SFT and 24 hours for RL. More training configurations are provided in Appendix A.

Table 1: Main results on multimodal agentic search benchmarks. **Bold** and underline mark the best and second-best reported score in each column. "–" denotes unreported results. † indicates results reported on a random subset rather than the full evaluation split.

Model	MMSearch	MMSearch+	BC-VL	FVQA	LiveVQA	SimpleVQA
Direct Answer						
Gemini-3-Pro (Google DeepMind, 2025)	65.9	26.4	41.4	58.9	51.1	–
Kimi-K2.5 (Team, 2026)	65.6	9.7	27.6	59.6	57.3	–
Qwen3-VL-8B (Bai et al., 2025)	15.2	3.2	25.1	28.0	41.0	42.9
Qwen3-VL-30B-A3B (Bai et al., 2025)	18.7	3.2	29.6	34.7	42.7	53.2
Agentic Workflow						
Gemini-3-Pro (Google DeepMind, 2025)	82.9	33.5	51.8	76.7	79.9	–
Kimi-K2.5 (Team, 2026)	76.6	27.8	50.3	76.5	76.6	–
Qwen3-VL-8B (Bai et al., 2025)	62.0	13.5	36.6	65.3	54.6	63.4
Qwen3-VL-30B-A3B (Bai et al., 2025)	63.2	14.1	44.6	67.4	61.9	66.6
Multimodal Search Agents						
MMSearch-R1-7B (Wu et al., 2025)	53.8	–	19.1	58.4	48.4	57.4
DeepEyesV2-7B (Hong et al., 2025)	63.7	9.5	24.8	60.6	58.0	59.4
SenseNova-MARS-8B (Chng et al., 2025)	67.8	–	–	67.1	56.2	70.2
MM-DeepResearch-8B (Yao et al., 2026)	67.8	–	37.9	69.2	65.0	65.9
Vision-DeepResearch-8B (Huang et al., 2026)	69.6	20.4	42.6	64.7†	76.7†	–
POINTS-Seeker-8B (Liu et al., 2026)	70.8	25.2	44.4	71.2	77.7	68.8
OpenSearch-VL-8B (Chen et al., 2026)	64.5	–	37.6	71.5	59.6	71.6
Visual-Seeker-8B (Zhang et al., 2026a)	72.2	27.3	47.6	–	–	–
SimpleSearch-VL-8B (ours)	77.1	32.5	52.1	76.8	75.2	76.6
WebWatcher-32B (Geng et al., 2025)	55.3	11.5	27.0	64.3	58.7	59.0
SenseNova-MARS-32B (Chng et al., 2025)	74.3	–	–	72.6	–	–
Vision-DeepResearch-30B-A3B (Huang et al., 2026)	69.6	28.5	53.7	74.2†	77.6†	–
OpenSearch-VL-30B-A3B (Chen et al., 2026)	68.7	–	41.1	73.2	67.4	74.9
RedSearcher-30B-A3B (Chu et al., 2026)	72.9	26.6	**57.2**	–	79.3	–
SimpleSearch-VL-30B-A3B (ours)	**83.6**	**34.4**	55.9	**79.0**	**81.1**	**79.6**

3.2 Main Results

Table 1 reports the main evaluation results. Compared with the corresponding untuned Qwen3-VL (Bai et al., 2025) agentic baselines, SimpleSearch-VL improves the average score by **15.8** points for the 8B model and **16.0** points for the 30B-A3B model on the shared evaluation benchmarks. Notably, SimpleSearch-VL-8B also outperforms the larger OpenSearch-VL-30B-A3B (Chen et al., 2026) on all five shared benchmarks, with gains of 8.4 points on MMSearch, 11.0 on BrowseComp-VL, 3.6 on FVQA, 7.8 on LiveVQA, and 1.7 on SimpleVQA. This cross-scale comparison suggests that the training and tool-use design can be more important than increasing model size alone. SimpleSearch-VL-30B-A3B is also competitive with stronger proprietary agentic systems. Compared with agentic Gemini-3-Pro (Comanici et al., 2025), it is higher on five benchmarks.

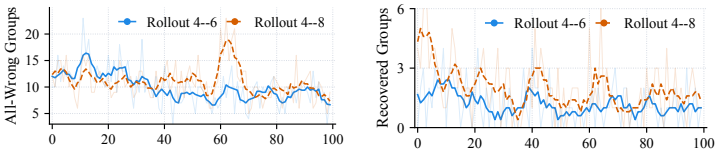
3.3 Ablation Studies

Unless otherwise specified, ablations are conducted on Qwen3-VL-8B-Instruct (Bai et al., 2025) under the direct-RL setting: the model is trained for 100 steps on 914 FVQA-train (Wang et al., 2017) samples with a batch size of 64 and $n = 4$ rollouts. For evaluation, all benchmarks except MMSearch (Jiang et al.) use a sampled subset of 100 examples. In the rollout analysis, *signal rate* is the fraction of prompt groups with mixed correct and incorrect rollouts.

3.3.1 Dynamic Rollout Filtering and FAR Analysis.

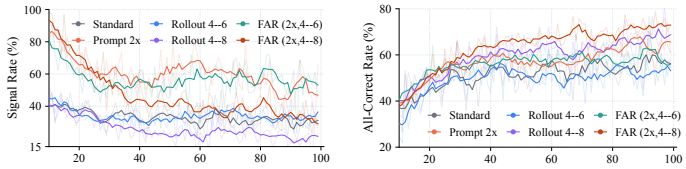
The scaling study in Fig. 2(e)–(f) isolates the two budget dimensions behind FAR. Prompt Expansion increases the number of candidate prompt groups, whereas Rollout Allocation increases the number of attempts available within each group. Increasing either budget can improve accuracy, but the marginal return varies across prompt groups and does not grow proportionally with uniform expansion. A fixed larger budget also raises rollout cost and exposes the training step to more slow or stalled tool-interleaved trajectories. We therefore focus here on mechanisms that realize expansion selectively at the sample level: recovery of hard groups, dynamic filtering of redundant tails, breadth-first scheduling, and near-complete termination.

Additional Attempts	Recovered	Rate
Rollout 4–6	117	11.7%
Rollout 4–8	195	17.7%



(a) Hard-group recovery: initially all-wrong groups can become informative under additional attempts.

Method	Step(s)	AVG	MMS	BC-VL	FVQA
Standard	192	56.6	66.8	35.0	68.0
Rollout 4–6	118	56.8	65.4	37.0	68.0
Rollout 4–8	134	60.4	67.2	41.0	73.0
Prompt 2×	273	59.7	67.2	43.0	69.0
FAR (2×,4–6)	165	62.0	73.0	45.0	68.0
FAR (2×,4–8)	187	62.8	71.3	45.0	72.0



(b) FAR: rollout cost and evaluation scores (left), signal rate (middle), and all-correct rate (right).

Fig. 4: **Dynamic rollout filtering and FAR analysis.** (a) Extra attempts recover groups whose base rollouts are all incorrect. (b) FAR combines expanded prompt coverage with dynamically filtered within-prompt attempts, improving the accuracy–time trade-off over fixed and single-axis variants.

Table 2: **Evidence verification ablation.** "Direct" refers to training using the distillation trajectory directly. "w/o Thumbnail" removes image-search thumbnails from the agent loop during inference.

Method	AVG	MMSearch	BC-VL	FVQA	LiveVQA
Direct SFT	61.0	71.9	36.0	72.0	64.0
Direct RL	64.9 +3.9	73.6 +1.7	46.0 +10.0	71.0 -1.0	69.0 +5.0
Evidence-Aware SFT	64.4 +3.4	74.4 +2.5	48.0 +12.0	70.0 -2.0	65.0 +1.0
w/o Thumbnail	61.9 -2.5	69.7 -4.7	44.0 -4.0	70.0 0.0	64.0 -1.0
Evidence-Aware RL	69.8 +5.4	77.2 +2.8	59.0 +11.0	76.0 +6.0	67.0 +2.0
w/o Thumbnail	66.1 -3.7	71.3 -5.9	54.0 -5.0	74.0 -2.0	65.0 -2.0

Recovering hard groups. Fig. 4a examines groups whose first four rollouts are all incorrect and hence provide no answer-level contrast. Allowing up to two additional attempts recovers 117 such groups (11.7%); allowing up to four recovers 195 (17.7%). This result motivates allocating additional attempts to currently all-wrong groups rather than applying a uniformly larger group size to every prompt. FAR implements this allocation breadth-first and masks the remaining expanded slots as soon as an answer-correct rollout appears.

Accuracy–efficiency trade-off. Fig. 4b compares the combined design with fixed and single-axis alternatives. Prompt expansion alone improves AVG but increases the rollout step from 192 to 273 seconds. FAR instead uses the expanded candidate budget as slack for dynamic filtering and near-complete termination. With a 2× prompt pool and a 4–8 rollout range, it improves AVG from 56.6 to **62.8** while taking 187 seconds per step, comparable to the 192-second Standard setting. The declining signal rate and rising all-correct rate provide complementary evidence that more groups become solved during training. Overall, the gain comes from selectively spending additional attempts on hard groups, not from unconditionally generating every expanded rollout.

3.3.2 Visual Evidence Verification

Table 2 compares direct training on trajectories distilled from the Qwen3-VL-235B-A22B model with training on trajectories augmented by textual and visual evidence-aware reasoning. Under the SFT-only setting, evidence-aware training improves the average score by **3.4** points. Applying RL on top of the evidence-aware SFT model further improves the average score by **5.4** points, suggesting that explicit evidence verification provides a stronger initialization for policy optimization. We further remove visual thumbnails from the image-search observations during evaluation. This consistently causes an average drop of about **3** points for both SFT and RL models, showing that thumbnails are not merely auxiliary metadata: *they provide the visual cue needed to verify whether a reverse-image-search result matches the queried entity.* Qualitative cases, including Fig. 3, show that without thumbnail-based verification, the model can incorrectly rely on visually mismatched image-search results. This verification signal introduces negligible overhead, since each thumbnail typically contains fewer than 0.1M pixels, which is minor compared with the global context.

Table 3: **Self-summary versus a dedicated summarizer.** All search agents use Qwen3-VL-8B. The dedicated-summarizer baseline deploys another Qwen3-VL-8B solely for webpage summarization, whereas self-summary uses the search agent itself. Time is measured over the same evaluation workload; parenthesized changes are relative to the dedicated-summarizer baseline.

Summary Provider	Serving Layout	Time ↓	AVG	MMSearch	BC-VL	FVQA	LiveVQA
Dedicated summarizer	1 search agent + 1 summary node	501s	67.9	76.6	56.0	71.0	68.0
Search agent (self-summary)	1 search-agent node	609s	69.8 +1.9	77.2	59.0	76.0	67.0
Search agent (self-summary)	2 search-agent nodes	**358s** -28.5%	69.7 +1.8	77.8	58.0	76.0	67.0

3.3.3 Self-Summarized Visit

Self-summarized visit lets the search agent generate its own goal-conditioned webpage summaries, avoiding a separately deployed summarization model. Table 3 separates two comparisons. **Summary quality.** Holding the search policy and the number of search-agent replicas fixed (one Qwen3-VL-8B agent), replacing the dedicated summarizer with self-summary improves AVG from 67.9 to 69.8 (**+1.9**). This comparison isolates who produces the webpage summary; it is not a latency comparison because the dedicated-summarizer system uses an extra node. **Matched serving budget.** With the total budget fixed at two Qwen3-VL-8B nodes, assigning both nodes to search-agent replicas—each performing self-summary—is better than assigning one node to search and one to summarization: evaluation time decreases from 501s to 358s (**28.5% lower**, or **1.40× faster**) while AVG increases from 67.9 to 69.7 (**+1.8**).

3.4 Training Cost Comparison

Table 4 compares training scale with OpenSearch-VL. SimpleSearch-VL uses only 5K SFT and 2K RL trajectories—approximately 7× and 4× **fewer**, respectively. Its 8B model completes both stages in 18 hours on eight H200 GPUs, whereas OpenSearch-VL reports substantially larger, multi-day H20 configurations. Despite this smaller budget, SimpleSearch-VL improves the four-benchmark average Pass@1 from 58.3 to **70.3** at 8B (**+12.0**) and from 62.6 to **74.9** at 30B-A3B (**+12.3**), demonstrating that the gains do not rely on scaling training data or cluster size.

Table 4: **Training cost comparison with OpenSearch-VL.** Data counts training trajectories; cost is reported as GPU count × wall-clock hours without cross-GPU normalization. Avg. is the mean Pass@1 on MMSearch, BC-VL, FVQA, and LiveVQA.

Method	Model	SFT Data	RL Data	SFT Cost	RL Cost	Avg.
OpenSearch-VL (Chen et al., 2026)	8B dense	36K	8K	256 H20 × 48h	64 H20 × 240h	58.3
OpenSearch-VL (Chen et al., 2026)	30B-A3B	36K	8K	256 H20 × 96h	64 H20 × 240h	62.6
SimpleSearch-VL	8B dense	5K	2K	8 H200 × 2 h	8 H200 × 16 h	**70.3**
SimpleSearch-VL	30B-A3B	5K	2K	8 H200 × 3 h	8 H200 × 24 h	**74.9**

4 Conclusion

We presented SimpleSearch-VL, a streamlined framework for efficient, verifiable, and practical multimodal agentic search. We study the training efficiency of multimodal search agents in terms of both data requirements and computational cost. Factorized Adaptive Rollout directs prompt- and response-level capacity toward unresolved groups while suppressing redundant samples and long-tail rollouts; evidence-verified reasoning grounds reverse-image-search results through thumbnail inspection; and decoupled tools with goal-conditioned self-summary give the policy direct control over evidence acquisition without a separate summarization model. Across six benchmarks, SimpleSearch-VL improves the corresponding Qwen3-VL agentic baselines by 15.8 and 16.0 average points at 8B and 30B-A3B, respectively, using only 5K SFT trajectories and 2K RL training samples. Policy SFT and RL for each model are completed on a single node with eight H200 GPUs. These results demonstrate that SimpleSearch-VL achieves strong multimodal search performance with limited training data and modest computational requirements for policy optimization.

AI USE STATEMENT

In this work, generative AI tools were used solely to assist with manuscript preparation, including language editing and improvements to grammar, clarity, and phrasing. They were not used to generate research ideas or hypotheses, design the methodology or experiments, implement or debug research code, analyze data, produce experimental results, or draw scientific conclusions. Other disclosure categories are not applicable to this work. All AI-assisted text was carefully reviewed and edited by the authors to ensure its accuracy, consistency with the underlying research, and faithful representation of the intended claims. The authors take full responsibility for the final content of this work, including all text and claims produced with the assistance of generative AI.

ETHICS STATEMENT

This work does not involve human subjects, personally identifiable information, or the collection of sensitive data. We are not aware of any specific ethical concerns arising from the methodology or experimental setup presented in this work. All datasets and resources used in our experiments are employed in accordance with their respective licenses and intended research purposes.

REPRODUCIBILITY STATEMENT

We have taken several steps to support reproducibility. Sections 2.1–2.4 describe the proposed method, and Section 3 specifies the experimental protocol and evaluation procedures. Appendix A reports the training configurations and hyperparameters; Appendix A.3 describes the training-data composition and processing statistics; and Appendices B–D.4 provide the agent workflow, tool interfaces, and prompts used for training and evaluation. Together, these details are intended to enable independent reproduction of the results reported in this work.

REFERENCES

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

Shuang Chen, Kaituo Feng, Hangting Chen, Wenxuan Huang, Dasen Dai, Quanxin Shou, Yunlong Lin, Xiangyu Yue, Shenghua Gao, and Tianyu Pang. Opensearch-vl: An open recipe for frontier multimodal search agents. *arXiv preprint arXiv:2605.05185*, 2026.

Yang Chen, Hexiang Hu, Yi Luan, Haitian Sun, Soravit Changpinyo, Alan Ritter, and Ming-Wei Chang. Can pre-trained vision and language models answer visual information-seeking questions? *arXiv preprint arXiv:2302.11713*, 2023.

Xianfu Cheng, Wei Zhang, Shiwei Zhang, Jian Yang, Xiangyuan Guan, Xianjie Wu, Xiang Li, Ge Zhang, Jiaheng Liu, Yuying Mai, et al. Simplevqa: Multimodal factuality evaluation for multimodal large language models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4637–4646, 2025.

Yong Xien Chng, Tao Hu, Wenwen Tong, Xueheng Li, Jiandong Chen, Haojia Yu, Jiefan Lu, Hewei Guo, Hanming Deng, Chengjun Xie, et al. Sensenova-mars: Empowering multimodal agentic reasoning and search via reinforcement learning. *arXiv preprint arXiv:2512.24330*, 2025.

Zheng Chu, Xiao Wang, Jack Hong, Huiming Fan, Yuqi Huang, Yue Yang, Guohai Xu, Chenxiao Zhao, Cheng Xiang, Shengchao Hu, et al. Redsearcher: A scalable and cost-efficient framework for long-horizon search agents. *arXiv preprint arXiv:2602.14234*, 2026.

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.

Guanting Dong, Hangyu Mao, Kai Ma, Licheng Bao, Yifei Chen, Zhongyuan Wang, Zhongxia Chen, Jiazhen Du, Huiyang Wang, Fuzheng Zhang, Guorui Zhou, Yutao Zhu, Ji-Rong Wen, and Zhicheng Dou. Agentic reinforced policy optimization. *arXiv preprint arXiv:2507.19849*, 2025.

Yifan Du, Zikang Liu, Jinbiao Peng, Jie Wu, Junyi Li, Jinyang Li, Wayne Xin Zhao, and Ji-Rong Wen. Towards long-horizon agentic multimodal search. *arXiv preprint arXiv:2604.12890*, 2026a.

Yuwen Du, Rui Ye, Shuo Tang, Xinyu Zhu, Yijun Lu, Yuzhu Cai, and Siheng Chen. Openseeker: Democratizing frontier search agents by fully open-sourcing training data. *arXiv preprint arXiv:2603.15594*, 2026b.

Mingyang Fu, Yuyang Peng, Benlin Liu, Yao Wan, and Dongping Chen. Livevqa: Live visual knowledge seeking. *arXiv preprint arXiv:2504.05288*, 2025.

Xinyu Geng, Peng Xia, Zhen Zhang, Xinyu Wang, Qiuchen Wang, Ruixue Ding, Chenxi Wang, Jialong Wu, Yida Zhao, Kuan Li, et al. Webwatcher: Breaking new frontier of vision-language deep research agent. *arXiv preprint arXiv:2508.05748*, 2025.

Google DeepMind. Gemini 3 pro model card. `https://storage.googleapis.com/deepmind-media/Model-Cards/Gemini-3-Pro-Model-Card.pdf`, 2025.

Jack Hong, Chenxiao Zhao, ChengLin Zhu, Weiheng Lu, Guohai Xu, and Xing Yu. Deepeyesv2: Toward agentic multimodal model. *arXiv preprint arXiv:2511.05271*, 2025.

Zengjie Hu, Jiantao Qiu, Tianyi Bai, Haojin Yang, Binhang Yuan, Qi Jing, Conghui He, and Wentao Zhang. Vade: Variance-aware dynamic sampling via online sample-level difficulty estimation for multimodal rl. *arXiv preprint arXiv:2511.18902*, 2025.

Wenxuan Huang, Yu Zeng, Qiuchen Wang, Zhen Fang, Shaosheng Cao, Zheng Chu, Qingyu Yin, Shuang Chen, Zhenfei Yin, Lin Chen, et al. Vision-deepresearch: Incentivizing deepresearch capability in multimodal large language models. *arXiv preprint arXiv:2601.22060*, 2026.

Dongzhi Jiang, Renrui Zhang, Ziyu Guo, Yanmin Wu, Pengshuo Qiu, Pan Lu, Zehui Chen, Guanglu Song, Peng Gao, Yu Liu, et al. Mmsearch: Unveiling the potential of large models as multi-modal search engines. In *The Thirteenth International Conference on Learning Representations*.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. Search-r1: Training llms to reason and leverage search engines with reinforcement learning, 2025. URL `https://arxiv.org/abs/2503.09516`.

Thanh-Long V Le, Myeongho Jeon, Kim Vu, Viet Lai, and Eunho Yang. No prompt left behind: Exploiting zero-variance prompts in llm reinforcement learning via entropy-guided advantage shaping. *arXiv preprint arXiv:2509.21880*, 2025.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33: 9459–9474, 2020.

Guankai Li, Jiabin Chen, Yi Xu, Xichen Zhang, and Yuan Lu. Hypereyes: Dual-grained efficiency-aware reinforcement learning for parallel multimodal search agents. *arXiv preprint arXiv:2605.07177*, 2026a.

Zhuofeng Li, Dongfu Jiang, Xueguang Ma, Haoxiang Zhang, Ping Nie, Yuyu Zhang, Kai Zou, Jianwen Xie, Yu Zhang, and Wenhu Chen. OpenResearcher: A Fully Open Pipeline for Long-Horizon Deep Research Trajectory Synthesis. *arXiv preprint arXiv:2603.20278*, 2026b.

Yikun Liu, Yuan Liu, Le Tian, Xiao Zhou, Jiangchao Yao, Yanfeng Wang, and Weidi Xie. Pointsseeker: Towards training a multimodal agentic search model from scratch. *arXiv preprint arXiv:2604.14029*, 2026.

MiroMind Team, Song Bai, Lidong Bing, Carson Chen, Guanzheng Chen, Yuntao Chen, et al. Mirothinker: Pushing the performance boundaries of open-source research agents via model, context, and interactive scaling. *arXiv preprint arXiv:2511.11793*, 2025.

Kartik Narayan, Yang Xu, Tian Cao, Kavya Nerella, Vishal M Patel, Navid Shiee, Peter Grasch, Chao Jia, Yinfei Yang, and Zhe Gan. Deepmmsearch-r1: Empowering multimodal llms in multimodal web search. *arXiv preprint arXiv:2510.12801*, 2025.

Hieu Trung Nguyen, Bao Nguyen, Wenao Ma, Yuzhi Zhao, Ruifeng She, and Viet Anh Nguyen. Adaptive rollout allocation for online reinforcement learning with verifiable rewards. *arXiv preprint arXiv:2602.01601*, 2026.

Xiangyu Peng, Can Qin, An Yan, Xinyi Yang, Zeyuan Chen, Ran Xu, and Chien-Sheng Wu. Mtagent: An open recipe for multimodal deep search agents. *arXiv preprint arXiv:2604.06376*, 2026.

Zekai Qu, Yinxu Pan, Ao Sun, Chaojun Xiao, and Xu Han. Copris: Efficient and stable reinforcement learning via concurrency-controlled partial rollout with importance sampling. *arXiv preprint arXiv:2511.05589*, 2025.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, YK Li, Yang Wu, et al. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

Guangming Sheng, Yuxuan Tong, Borui Wan, Wang Zhang, Chaobo Jia, Xibin Wu, Yuqi Wu, Xiang Li, Chi Zhang, Yanghua Peng, et al. Laminar: A scalable asynchronous rl post-training framework. In *Proceedings of the 21st European Conference on Computer Systems*, pp. 400–422, 2026.

Rohan Surana, Gagan Mundada, Xunyi Jiang, Chuhan Wang, Zhenwei Tang, Difan Jiao, Zihan Huang, Yuxin Xiong, Junda Wu, Sheldon Yu, et al. Generate, filter, control, replay: A comprehensive survey of rollout strategies for llm reinforcement learning. *arXiv preprint arXiv:2605.02913*, 2026.

Sijun Tan, Michael Luo, Colin Cai, Tarun Venkat, Kyle Montgomery, Aaron Hao, Tianhao Wu, Arnav Balyan, Manan Roongta, Chenguang Wang, Li Erran Li, Raluca Ada Popa, and Ion Stoica. rllm: A framework for post-training language agents, 2025. Notion Blog.

Kimi Team. Kimi k2.5: Visual agentic intelligence, 2026. URL `https://arxiv.org/abs/2602.02276`.

Tongyi DeepResearch Team, Baixuan Li, Bo Zhang, Dingchu Zhang, Fei Huang, Guangyu Li, Guoxin Chen, Huifeng Yin, Jialong Wu, Jingren Zhou, et al. Tongyi deepresearch technical report. *arXiv preprint arXiv:2510.24701*, 2025.

Peng Wang, Qi Wu, Chunhua Shen, Anthony Dick, and Anton Van Den Hengel. Fvqa: Fact-based visual question answering. *IEEE transactions on pattern analysis and machine intelligence*, 40 (10):2413–2427, 2017.

Jinming Wu, Zihao Deng, Wei Li, Yiding Liu, Bo You, Bo Li, Zejun Ma, and Ziwei Liu. Mmsearch-r1: Incentivizing lmms to search. *arXiv preprint arXiv:2506.20670*, 2025.

Bokang Yang, Xinyi Sun, Kaituo Feng, Xingping Dong, Dongming Wu, and Xiangyu Yue. From web to pixels: Bringing agentic search into visual perception. *arXiv preprint arXiv:2605.12497*, 2026.

Huanjin Yao, Qixiang Yin, Min Yang, Ziwang Zhao, Yibo Wang, Haotian Luo, Jingyi Zhang, and Jiaxing Huang. Mm-deepresearch: A simple and effective multimodal agentic search baseline. *arXiv preprint arXiv:2603.01050*, 2026.

Fangda Ye, Zhifei Xie, Yuxin Hu, Yihang Yin, Shurui Huang, Shikai Dong, Jianzhu Bao, and Shuicheng Yan. Deep-reporter: Deep research for grounded multimodal long-form generation. *arXiv preprint arXiv:2604.10741*, 2026.

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.

Shi Yu, Chaoyue Tang, Bokai Xu, Junbo Cui, Junhao Ran, Yukun Yan, Zhenghao Liu, Shuo Wang, Xu Han, Zhiyuan Liu, et al. Visrag: Vision-based retrieval-augmented generation on multi-modality documents. *arXiv preprint arXiv:2410.10594*, 2024.

Yu Zeng, Wenxuan Huang, Zhen Fang, Shuang Chen, Yufan Shen, Yishuo Cai, Xiaoman Wang, Zhenfei Yin, Lin Chen, Zehui Chen, Shiting Huang, Yiming Zhao, Yao Hu, Philip Torr, Wanli Ouyang, and Shaosheng Cao. Vision-deepresearch benchmark: Rethinking visual and textual search for multimodal large language models. *preprint*, 2026.

Yuheng Zhang, Wenlin Yao, Changlong Yu, Yao Liu, Qingyu Yin, Bing Yin, Hyokun Yun, and Lihong Li. Improving sampling efficiency in rlvr through adaptive rollout and response reuse. *arXiv preprint arXiv:2509.25808*, 2025.

Zhengbo Zhang, Changtao Miao, Jinbo Su, Zhaowen Zhou, Chunxia Zhang, Xukai Wang, Ruiqi Liu, Kaiyuan Zheng, Jiansheng Cai, Bo Zhang, Zhe Li, Shiming Xiang, and Ying Yan. Visual-seeker: Towards visual-native multimodal agentic search via active visual reasoning, 2026a. URL `https://arxiv.org/abs/2606.15231`.

Zhi Zhang, Zhen Han, Costas Mavromatis, Qi Zhu, Yunyi Zhang, Sheng Guan, Dingmin Wang, Xiong Zhou, Shuai Wang, Soji Adeshina, et al. Train less, learn more: Adaptive efficient rollout optimization for group-based reinforcement learning. *arXiv preprint arXiv:2602.14338*, 2026b.

Yuze Zhao, Jintao Huang, Jinghan Hu, Xingjun Wang, Yunlin Mao, Daoze Zhang, Zeyinzi Jiang, Zhikai Wu, Baole Ai, Ang Wang, Wenmeng Zhou, and Yingda Chen. Swift:a scalable lightweight infrastructure for fine-tuning, 2024. URL https://arxiv.org/abs/2408.05517.

Haizhong Zheng, Yang Zhou, Brian Bartoldson, Bhavya Kailkhura, Fan Lai, Jiawei Zhao, and Beidi Chen. Act only when it pays: Efficient reinforcement learning for llm reasoning via selective rollouts. *Advances in Neural Information Processing Systems*, 38:124321–124346, 2026.

Yuzhen Zhou, Jiajun Li, Yusheng Su, Gowtham Ramesh, Zilin Zhu, Xiang Long, Chenyang Zhao, Jin Pan, Xiaodong Yu, Ze Wang, et al. April: Active partial rollouts in reinforcement learning to tame long-tail generation. *arXiv preprint arXiv:2509.18521*, 2025.

APPENDIX

APPENDIX CONTENTS

A Implementation Details **15**
A.1 SFT Training Configuration 16
A.2 RL Training Configuration 16
A.3 Training Data Composition 16
A.4 FAR Sampling Details 17
A.5 Rollout Scaling Analysis 17

B Agent Workflow and Tool Interface **18**

C Evidence-Aware Reasoning Data Construction **19**

D Prompt Templates **20**
D.1 Agent System Prompt 20
D.2 Tool Prompt and Schemas 20
D.3 Self-Summary Prompt 23
D.4 Judge Prompt 24

E Related Work **24**
E.1 Rollout Sampling for Reinforcement Learning 24
E.2 Multimodal Deep Search Agents 25

F Additional Experiments **25**
F.1 Harness Comparison. 25
F.2 Round/Tool Usage Analysis 26

G Discussion **27**

H Visualization **27**
H.1 Standard Single-Image Cases 27
H.2 Single-Image Case without Image Search 27
H.3 Multi-Frame Cases 27
H.4 Multi-Region Case 31
H.5 Failure Cases 31

A IMPLEMENTATION DETAILS

This appendix reports the implementation details of SimpleSearch-VL. We summarize the training data and optimization settings used in our experiments, while omitting framework defaults that are not essential for reproduction.

A.1 SFT Training Configuration

The supervised fine-tuning stage initializes SimpleSearch-VL with executable evidence-aware trajectories. As shown in Table 5, both the 8B dense model and the 30B-A3B MoE model are trained on the same 5,193 trajectories with a 64K-token context window, three epochs, and batch size 64. The two scales use the corresponding Qwen3-VL-Instruct checkpoints as initialization and are trained with ms-swift Megatron SFT on a single H200 node. The 30B-A3B MoE model uses a smaller learning rate and requires a slightly longer SFT run, but both runs finish within a few hours, keeping SFT as a lightweight initialization stage rather than the main source of compute.

Table 5: **SFT training configuration.** Recipe-level settings for the 8B dense and 30B-A3B MoE variants; framework defaults are omitted.

Configuration	**8B**	**30B-A3B**
Base model	`Qwen3-VL-8B-Instruct`	`Qwen3-VL-30B-A3B-Instruct`
Hardware	1 H200 node (8×141GB GPUs)	1 H200 node (8×141GB GPUs)
Training time	∼2 hours	∼3 hours
Learning rate	2.0×10^{-5} with minimum LR 5.0×10^{-7}	1.0×10^{-5} with minimum LR 2.0×10^{-7}
Training data	5,193 evidence-aware SFT trajectories	
Training framework	ms-swift Megatron SFT	
Context length	64K tokens	
Epochs	3	
Batch size	64	

A.2 RL Training Configuration

The reinforcement learning stage further optimizes the SFT-initialized policy in the same agentic search environment used at inference time. As summarized in Table 6, RL uses a 64K-token context window for both model scales and runs for 150 training steps on one H200 node. The training recipe applies Factorized Adaptive Rollout with 2× Prompt Expansion and Rollout Allocation from four to six rollouts, uses a small 1.0×10^{-6} learning rate with cosine decay, and disables KL loss. A large Qwen3-VL judge supplies the answer-level correctness signal used by the reward and rollout filtering. Compared with SFT, RL is the longer stage (∼16 hours for 8B and ∼24 hours for 30B-A3B), but it still fits within a single-node training budget.

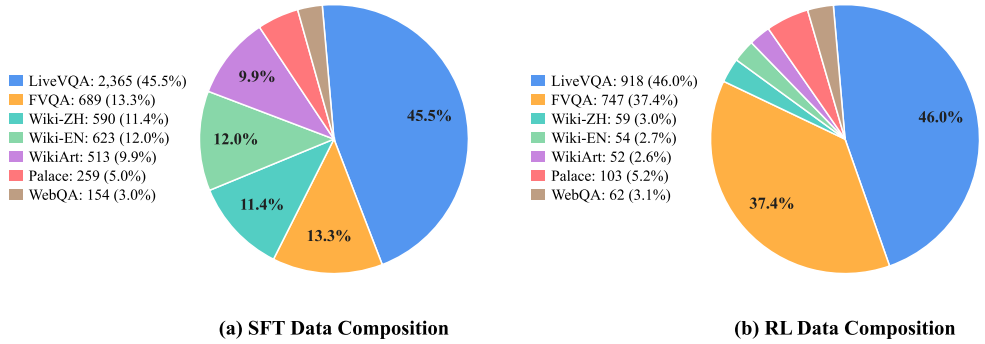
A.3 Training Data Composition

Fig. 5 summarizes the raw data-source composition used by the SFT and RL stages. The final SFT set contains 5,193 evidence-aware trajectories, led by LiveVQA (2,365) and complemented by FVQA (689), Wiki-EN (623), Wiki-ZH (590), WikiArt (513), Palace (259), and WebQA (154). The RL pool contains 1,995 prompts and is intentionally concentrated on the sources that provide reliable answer checking and useful search difficulty: LiveVQA contributes 918 prompts and FVQA contributes 747, while Palace, WebQA, Wiki-ZH, Wiki-EN, and WikiArt provide smaller but diverse long-tail coverage. This mixture gives SFT broad evidence-aware behavior and reserves RL for a cleaner prompt pool with stable binary feedback. For both FVQA and LiveVQA, source examples are restricted to the official training split. Their official test splits are reserved exclusively for evaluation and are never used for trajectory construction, SFT, or RL.

Fig. 6 provides two complementary views of the training data. **Multi-image coverage.** We count multi-image samples by the number of original input images, excluding retrieved visual thumbnails used for evidence verification. Under this definition, the SFT archive contains 366 multi-image trajectories out of 5,193 samples (7.0%), and the RL pool contains 403 multi-image prompts out of 1,995 samples (20.2%). All multi-image samples are from LiveVQA, which supplies the main supervision signal for multi-image search and comparison. **Trajectory length.** The SFT trajectories use 3.20 tool-use rounds on average before the final answer, with most samples concentrated between two and four rounds. This distribution indicates that the SFT data emphasizes concise tool-interleaved reasoning rather than unnecessarily long search chains.

Table 6: **RL training configuration.** Recipe-level settings for reproducing the 8B dense and 30B-A3B MoE runs.

Configuration	8B	30B-A3B
Hardware	1 H200 node (8×141GB GPUs)	1 H200 node (8×141GB GPUs)
Training time	∼16 hours	∼24 hours
Context length	64K tokens	
Training steps	150	
FAR setting	Prompt Expansion with a 2× and Rollout Allocation from 4 to 6 rollouts	
Learning rate	1.0×10^{-6} with cosine decay, 0.05 warmup ratio, and minimum LR 1.0×10^{-7}	
KL loss	Disabled	
Judge model	`Qwen3-VL-235B-A22B-Instruct`	



(a) SFT Data Composition **(b) RL Data Composition**

Fig. 5: **Raw source composition of SFT and RL training data.** (a) SFT trajectory mixture. (b) RL data mixture.

A.4 FAR Sampling Details

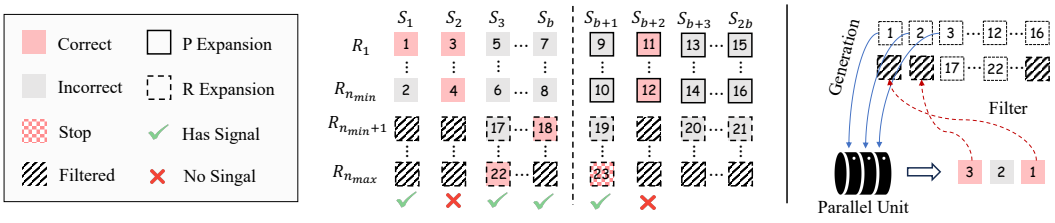


Fig. 7: **FAR sampling details and filtering process.** The numbers 1, 2, 3, and subsequent indices indicate the intuitive sampling order. FAR fills base rollouts first, then allocates expanded rollout slots horizontally across groups. Once an answer-correct rollout appears for a group, later Rollout Allocation slots for that group are filtered, avoiding redundant rollouts.

A.5 Rollout Scaling Analysis

We separately scale the two budget dimensions used by FAR. **Prompt Expansion** increases the number of candidate prompt groups, whereas **Rollout Allocation** increases the number of attempts within each group. Fig. 8 reports the complete per-benchmark scores and training diagnostics. Prompt Expansion reaches its best practical trade-off near 2×: further expansion increases rollout time without improving AVG. Additional within-group attempts can recover useful training signal from harder prompts, but their accuracy gains are non-monotonic and their cost grows at larger budgets. These results motivate adaptive filtering and scheduling rather than uniform expansion along either dimension.

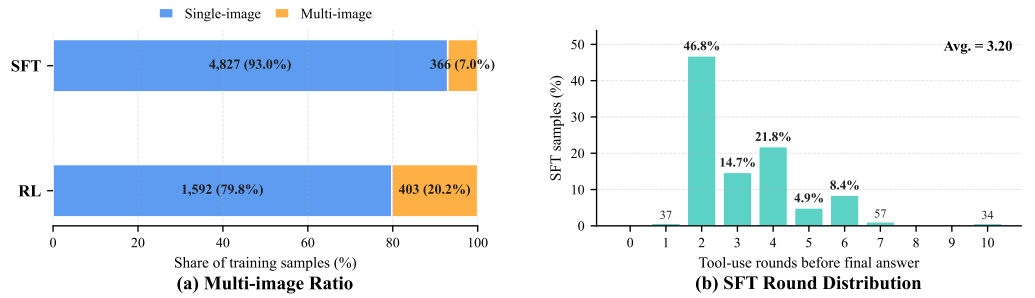
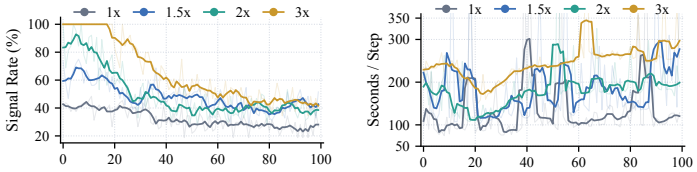


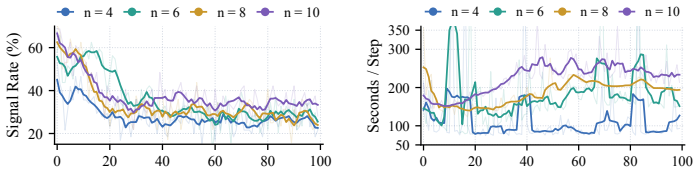
Fig. 6: **Input-image type and SFT round distribution.** (a) Single-image and multi-image shares in SFT and RL data. (b) SFT tool-use round distribution, where a round denotes one tool call before the final answer.

Prompt	AVG	MMS	BC-VL	FVQA	Time†
1×	57.8	70.3	33.0	70.0	136/102
1.5×	59.5	69.6	37.0	72.0	185/142
2×	62.7	72.2	43.0	73.0	181/175
3×	62.5	73.6	43.0	71.0	246/245



(a) Prompt Expansion: evaluation scores, signal rate, and rollout time.

Rollouts	AVG	MMS	BC-VL	FVQA	Time†
$n = 4$	58.1	67.3	40.0	67.0	113/85
$n = 6$	62.6	71.9	47.0	69.0	181/156
$n = 8$	61.0	69.0	46.0	68.0	187/193
$n = 10$	63.1	71.3	48.0	70.0	224/227



(b) Rollout Allocation: evaluation scores, signal rate, and rollout time.

Fig. 8: **Detailed rollout-scaling analysis.** Complete results for Prompt Expansion and Rollout Allocation, including per-benchmark scores, signal rate, and rollout time. †Mean/median seconds per step.

B Agent Workflow and Tool Interface

The agent operates in a tool-interleaved loop. Given a question and one or more input images, the system assigns explicit image identifiers, e.g., `img_idx=0` and `img_idx=1`. At every turn, the model emits a `<thinking>` block and then takes exactly one action: a JSON-formatted tool call or a final `<answer>`. Tool observations are appended to the context, and the loop continues until a valid answer is produced or the configured budget is reached.

Table 7: **Tool declarations in SimpleSearch-VL.** The table lists each tool's role, input signature, and returned evidence fields.

Tool	Role	Input Signature	Output
`text_search`	Search the web for external textual knowledge.	`query`: list of search queries	`title`: webpage title `url`: source webpage `snippet`: short matched context
`image_search`	Perform region-level reverse image search using a selected image region as the visual query.	`regions`: list of {`img_idx`, `bbox_2d`}	`thumbnail`: visual match preview `title`: matched webpage title `url`: source webpage
`visit`	Extract webpages relevant to a goal.	`url`: list of webpages `goal`: extraction goal string	`summary`: goal-conditioned summary

Text Search. The text-search tool is used when the model already has keywords or candidate entities to verify. Its input is a list of search queries, and its output contains webpage titles, URLs, and short matched snippets. The returned snippets provide preliminary evidence; when exact webpage evidence is needed, the model is instructed to call `visit` on the selected URLs.

Image Search. The image-search tool performs region-level reverse image search, using selected image regions as visual queries. Its input is a list of regions, where each region contains an `img_idx` and a normalized $[0, 1000]$ `bbox_2d`; $[0, 0, 1000, 1000]$ denotes the full image. Its output includes the matched thumbnail, webpage title, and source URL. Unlike single-image search interfaces, our schema allows a single batched call to include regions from multiple input images, which is important for multi-image visual questions. The returned thumbnails are used for visual evidence verification before the agent trusts the associated title or URL.

Visit. The visit tool takes a list of URLs and a goal string as input. For each URL, a webpage reader first fetches and normalizes the content into text; the agent then produces a concise summary conditioned on the goal. This keeps webpage understanding inside the model's own generation process while avoiding an additional deployed summary model during training and inference. The exact self-summary prompt is provided in Table 15.

C Evidence-Aware Reasoning Data Construction

We construct SFT trajectories with an evidence-aware data pipeline. The pipeline is designed to produce trajectories where visual observations, image-search thumbnails, web evidence, and final answers are explicitly connected. We focus on the three model-involved stages after candidate pool construction: direct-answer filtering, agentic rollout filtering, and trajectory audit with reasoning rewrite.

Step 0: Candidate Pool Construction. We first collect candidate visual QA samples from FVQA, LiveVQA, Wiki-EN, Wiki-ZH, WikiArt, Palace, and WebQA. Since RL training requires a reliable correctness signal, we focus on questions with a *determinate and verifiable answer boundary*: the expected answer should be a specific entity, attribute, event, count, date, title, location, relation, or another objective fact. Open-ended or non-factoid queries, such as questions asking for broad descriptions, explanations, opinions, or ambiguous "why/how" reasoning, are removed because many plausible responses could be acceptable and cannot be judged robustly by a binary reward.

Step 1: Direct-Answer Filtering. We use `Qwen3-VL-235B-A22B-Instruct` as both the direct-answer model and the judge. The model first produces a direct answer for each sample. The same model then performs two judgments: whether the prediction matches the reference answer, and whether the question is a clear fact-seeking query with an objective correctness criterion. Samples that can be answered correctly in this direct-answer setting are removed, and samples judged ambiguous, subjective, open-ended, or insufficiently contextualized are also excluded. Only incorrect, valid, fact-seeking samples are kept for agentic rollout. This stage filters out directly answerable examples while preserving questions that require external evidence.

Step 2: Agentic Rollout Filtering. We then generate tool-interleaved trajectories for the retained samples using `Qwen3-VL-235B-A22B-Instruct` as the policy model. The rollout environment provides three tools, `text_search`, `image_search`, and `visit`; image-search observations include visual thumbnails for evidence verification. Sampling is deterministic during data construction ($T = 0$), and each sample is allowed up to 10 model turns. We keep trajectories that end with a valid `<answer>`, follow the required assistant format, are judged answer-correct, and contain substantive tool use. Trajectories that fail format checks, terminate without a supported final answer, or do not use external evidence are rejected.

Step 3: Joint Trajectory Audit and Reasoning Rewrite. We use `gemini-3.1-pro` to perform trajectory audit and reasoning rewrite in a single model call. The auditor rejects trajectories with unsupported final answers, repeated low-value searches, premature or unnecessary tool calls, or weak connections between the visual evidence and the final response. For retained samples, Gemini

rewrites only the assistant reasoning and final answer while preserving the original tool calls, tool responses, and evidence order. The rewrite explicitly requires image-search reasoning to compare retrieved thumbnails with the queried image region before trusting the associated title or URL, making visual evidence verification visible in the SFT trajectory.

D PROMPT TEMPLATES

This section groups the prompt templates used by SimpleSearch-VL into the agent system prompt, tool prompt, self-summary prompt, and judge prompt. The same agent and tool prompts are used during training and inference.

D.1 AGENT SYSTEM PROMPT

Table 8: **Agent system prompt.** The active date is replaced by `{current_date}`.

Agent System Prompt

```
#Role
You are a visual research assistant.
Given a question, your task is to solve the problem **one substep at a time**.

## Guiding Principles
At each turn, you must **either**:
1. Issue **one specific tool** enclosed in <tool_call></tool_call> tags,
2. Or provide the **final answer** enclosed in <answer></answer> tags.

All outputs **must begin with a thought** enclosed in <thinking></thinking> tags,
     explaining your current reasoning and what to do next.

## Output Format (strict):
Always start with <thinking>. Do not output the previous reasoning chain. Then,
    depending on the case, output one of the following:

1. If reasoning continues:
<thinking>Your current reasoning and next plan</thinking>
<tool_call>One precise tool call to assist your reasoning</tool_call>

2. If ready to conclude:
<thinking>Summarize all reasoning and derive the answer</thinking>
<answer>Final answer</answer>
```

D.2 TOOL PROMPT AND SCHEMAS

The tool section is appended after the system prompt and before the current date. It exposes only the active tool schemas and enforces a single valid JSON tool call per assistant turn.

Table 9: **Tool calling prompt.** Function schemas are injected into `{tool_json_lines}`.

Tool Calling Prompt

```
# Tools

You may call exactly one function per assistant turn. Use batched arguments
    inside that single tool call when you need parallel search.

You are provided with function signatures within <tools></tools> XML tags:
<tools>
{tool_json_lines}
```

```
</tools>

For the function call, return one valid JSON object inside <tool_call></tool_call
    >.
The top-level object must have exactly two sibling keys: "name" and "arguments".
Never insert an extra object between "name" and "arguments".

Use these exact JSON shapes:
<tool_call>
{"name": "text_search", "arguments": {"query": ["example query"]}}
</tool_call>
<tool_call>
{"name": "visit", "arguments": {"url": ["https://example.com"], "goal": "find
    relevant evidence"}}
</tool_call>
<tool_call>
{"name": "image_search", "arguments": {"regions": [{"img_idx": 0, "bbox_2d": [0,
    0, 1000, 1000]}]}}
</tool_call>

Current date: {current_date}
```

The active schemas are listed below for readability.

Table 10: `image_search` **schema.** Multi-image region search is supported through `img_idx`.

Tool Definition: image_search

```
Description:
Perform reverse image search for one or more input image regions. Use this to
    identify unfamiliar objects, logos, landmarks, artworks, products, people, or
     scenes. Input images are referred to by img_idx, starting from 0. Each
    region must pair one img_idx with one bbox_2d in 0-1000 scale. Use [0, 0,
    1000, 1000] for a whole image. The tool response labels each searched region
    with the same img_idx.

Arguments:
{
  "regions": [
    {
      "img_idx": number,
      "bbox_2d": [x1, y1, x2, y2]
    }
  ]
}

Constraints:
- 1 to 3 regions per call.
- bbox_2d uses the 0-1000 coordinate system.
- [0, 0, 1000, 1000] denotes the full image.

Example:
<tool_call>
{
  "name": "image_search",
  "arguments": {
    "regions": [
      {"img_idx": 0, "bbox_2d": [0, 0, 1000, 1000]}
    ]
  }
}
</tool_call>
```

Table 11: `text_search` **schema.** Complementary queries can be batched in one call.

Tool Definition: text_search

```
Description:
Perform text-based web searches and return top results. Use when you already have
     specific keywords or facts to look up. Supply complementary queries in a
    single call when useful.

Arguments:
{
  "query": ["search query 1", "search query 2", "..."]
}

Constraints:
- 1 to 3 queries per call.
- The query array must stay inside arguments.

Example:
<tool_call>
{
  "name": "text_search",
  "arguments": {
    "query": ["example query"]
  }
}
</tool_call>
```

Table 12: `visit` **schema.** Webpages are summarized with respect to an extraction goal.

Tool Definition: visit

```
Description:
Visit webpages and extract evidence relevant to your goal. Returns extracted
    evidence, a summary, and related links for further research. Use exact,
    complete URLs from search results or related links. Do not modify, truncate,
    or guess URLs.

Arguments:
{
  "url": ["https://example.com"],
  "goal": "what information to extract from the webpages"
}

Constraints:
- 1 to 3 URLs per call by default; the maximum can be configured at runtime.
- URLs must be exact and complete.
- goal should specify the evidence needed for answering the question.

Example:
<tool_call>
{
  "name": "visit",
  "arguments": {
    "url": ["https://example.com"],
    "goal": "find relevant evidence"
  }
}
</tool_call>
```

Table 13: **Image-search observation instruction.** Visual thumbnails are checked before using titles or URLs as evidence.

Visual Thumbnail Verification Instruction

```
Reminder: Each image thumbnail corresponds to the result immediately below it.
Compare the thumbnails with the queried image region first.
Visit all URLs whose thumbnails and titles are visually and semantically
    consistent with the target.
If thumbnails do not match the target region, treat those results as low-
    confidence and do not rely on them.
You can pass multiple URLs in one visit call.
```

Table 14: **Multi-image image-search example.** One call can search localized regions from different input images.

Multi-Image Search Example

```
Input images are indexed from 0. A single image_search call may include regions
    from multiple input images:

<tool_call>
{
  "name": "image_search",
  "arguments": {
    "regions": [
      {"img_idx": 0, "bbox_2d": [0, 0, 1000, 1000]},
      {"img_idx": 1, "bbox_2d": [120, 180, 760, 920]}
    ]
  }
}
</tool_call>

The returned observation keeps the same img_idx for each searched region. If
    thumbnails are returned, compare each thumbnail against the queried region
    before trusting the corresponding title or URL.
```

D.3 Self-Summary Prompt

After `visit` fetches and normalizes a webpage, the policy model itself summarizes the content with respect to the extraction goal.

Table 15: **Self-summary prompt.** The model summarizes webpage content with respect to the extraction goal.

Self-Summary Prompt for visit

```
System:
You are a helpful assistant. Your task is to summarize webpage content concisely
    and factually.

User:
Summarize the following webpage content with respect to the user's goal.

Webpage Content (may be truncated):
{webpage_content}
```

```
User Goal: {goal}

Instructions:
- Summarize the main content of the webpage in no more than five sentences.
- Your summary should cover the overall key points of the page, not just parts
    related to the user's goal.
- If any part of the content is helpful for answering the user's goal, include it
     clearly in the summary.
- If the page contains no information relevant to the goal, state that briefly.
- Your summary should be concise, factual, and informative.
- Return the summary text directly. Do not use JSON, code fences, or special
    formatting.
```

D.4 JUDGE PROMPT

The final training configuration uses a binary accuracy reward. Exact matching is attempted first; otherwise, the fallback LLM judge uses the prompt below.

Table 16: **Accuracy judge prompt.** The fallback judge returns the binary correctness signal used by the reward.

Accuracy Judge Prompt

```
You are an impartial judge evaluating whether a deep research report contains the
     correct answer.

[Question]
{question}

[Correct Answer]
{reference_answer}

[Deep Research Report]
{assistant_answer}

Task: Determine if the deep research report contains the correct answer anywhere
    in its content.

Instructions:
1. Read through the entire research report carefully.
2. Look for the correct answer anywhere in the report. It may be embedded in
    paragraphs, tables, or sections.
3. Check if the information in the report is consistent with the correct answer.
4. The answer does NOT need to be in a specific format or labeled as "final
    answer".
5. Provide your reasoning.
6. Answer with "yes" if the report contains the correct answer, and "no" if it
    does not or contradicts it.

Output format:
correct: [yes/no]
reasoning: [your explanation]
```

E RELATED WORK

E.1 ROLLOUT SAMPLING FOR REINFORCEMENT LEARNING

Rollout generation is a central efficiency bottleneck in RL post-training, from PPO-style on-policy optimization (Schulman et al., 2017) to reasoning-oriented RL (Shao et al., 2024) and further to

agentic RL with multi-turn tool calls (Dong et al., 2025). Recent studies treat rollout as an explicit sampling problem, covering generation, filtering, control, and replay mechanisms (Surana et al., 2026). Dynamic-rollout methods improve useful-signal density by filtering or selecting informative prompt groups, from dynamic sampling (Yu et al., 2026) to pre-rollout selective sampling and variance-aware sample selection (Zheng et al., 2026; Hu et al., 2025); complementary work instead extracts signal from zero-variance prompts rather than discarding them (Le et al., 2025). Partial-rollout systems target wall-clock inefficiency from long-tail generations by interrupting, buffering, and resuming unfinished trajectories (Zhou et al., 2025; Qu et al., 2025), while asynchronous rollout systems further reduce synchronization stalls through trajectory-level scheduling and repacking (Sheng et al., 2026). Adaptive-rollout methods allocate different numbers of responses across prompts according to difficulty, uncertainty, or expected gradient variance (Zhang et al., 2025; 2026b; Nguyen et al., 2026). In contrast, as illustrated in Fig. 2, Factorized Adaptive Rollout (FAR) shifts budget toward hard groups that still lack useful reward variation, while skipping redundant tail rollouts once a training signal has been obtained. Because FAR generates candidate redundancy at the group level, completed alternative rollouts can provide the needed signal instead of waiting for slow or stalled tool-interleaved trajectories.

E.2 Multimodal Deep Search Agents

Multimodal deep search agents tackle knowledge-intensive visual questions by actively gathering and verifying external evidence through multi-turn tool interactions. Compared with standard RAG (Lewis et al., 2020; Yu et al., 2024), they move beyond fixed retrieval by adaptively deciding what to inspect, which modality to query, and when to refine the search. Recently, some methods construct VQA tasks paired with supervised multi-hop web-search trajectories to teach search behavior (Geng et al., 2025; Narayan et al., 2025; Chen et al., 2026; Peng et al., 2026). Another line scales long-horizon search by training agents to decompose complex questions and iteratively gather evidence over extended tool-use trajectories (Huang et al., 2026; Chu et al., 2026; Du et al., 2026a). Related tool-policy work studies how models should invoke and coordinate external tools, including visual operations, retrieval tools, native agentic search behavior, and efficiency-aware parallel search (Hong et al., 2025; Chng et al., 2025; Li et al., 2026a). Other efforts shift the target output or system bottleneck, such as grounded multimodal long-report generation and visual perception (Ye et al., 2026; Yang et al., 2026), or address the growing interaction history through context compression (Liu et al., 2026). Despite these advances, the relevance and verifiability of retrieved evidence remain under-explored, especially when assessing the trustworthiness of visual-search results. To address this limitation, we propose evidence-verified reasoning to improve textual and visual evidence verification, together with a self-summary mechanism that enables the agent to summarize webpages autonomously without relying on external models.

F Additional Experiments

F.1 Harness Comparison.

We further compare different agentic harnesses under a zero-shot setting, where each harness is directly paired with an untuned 8B model. This isolates the effectiveness of the inference-time tool environment from post-training. As shown in Table 17, SimpleSearch-VL improves the average score over Vision-DeepResearch by **2.7** points, validating the benefit of our decoupled tool design. Unlike Vision-DeepResearch, which couples reverse image search, webpage visit, and summary, or SenseNova-MARS, which couples search and visit, our harness lets the model independently plan search queries, select webpages to visit, and choose target entities for reverse image search. OpenSearch-VL adopts a similarly decoupled design, but our harness still improves the average score by **11.7** points.

Table 17: **Zero-shot harness comparison.** AVG is computed over common reported benchmarks; "–" denotes unavailable results. † marks Vision-DeepResearch results on a random 300-sample subset.

Harness	Model	MMSearch	FVQA	LiveVQA	AVG
SenseNova-MARS (Chng et al., 2025)	Qwen3-VL-8B-Instruct	47.4	53.6	39.4	46.8
OpenSearch-VL (Chen et al., 2026)	Qwen3-VL-8B-Instruct	37.4	58.7	50.6	48.9
Vision-DeepResearch (Zeng et al., 2026)	Qwen3-VL-8B-Instruct	52.0	58.7†	63.0†	57.9
MTA-DeepSearch (Peng et al., 2026)	Qwen3-VL-8B-Instruct	57.1	64.2	–	–
SimpleSearch-VL	Qwen3-VL-8B-Instruct	62.0	65.3	54.6	60.6

F.2 Round/Tool Usage Analysis

Fig. 9 compares SFT+RL and Direct RL under the same FAR (2×, 4–6) setting. The SFT-initialized run keeps a longer and more stable search process, with average reasoning rounds changing only from 5.6 to 5.4 between steps 10 and 100. Direct RL starts from shorter trajectories and increases from 3.4 to 4.2 rounds, indicating that it is still learning when to search and verify evidence. The sample-level image-search ratio in Fig. 9a shows a complementary pattern. Direct RL increases image-search invocation on all three reported benchmarks, from 59.1% to 84.2% on MMS, 33.0% to 74.0% on BC-VL, and 74.0% to 93.0% on FVQA. SFT+RL changes more moderately, from 61.4% to 66.1%, 47.0% to 60.0%, and 68.0% to 74.0% on the same benchmarks. This suggests that SFT provides a stronger tool-use prior, whereas Direct RL more readily expands reverse image search as a broadly useful evidence-gathering action.

Method	MMS	BC-VL	FVQA
SFT	62.0	43.0	69.0
SFT+RL$_{step=10}$	61.4	47.0	68.0
SFT+RL$_{step=100}$	66.1	60.0	74.0
Direct RL$_{step=10}$	59.1	33.0	74.0
Direct RL$_{step=100}$	84.2	74.0	93.0

(a) Image-search ratio.



(b) Tool composition.

Fig. 9: **Training-time tool behavior.** Left: image-search usage ratio. Right: tool-call composition and average reasoning rounds during training.

Fig. 10 reports inference rounds and Pass@1 across benchmarks. SimpleSearch-VL uses a moderate number of rounds overall, but adapts its reasoning depth to task difficulty: it uses about seven rounds on the harder MMSearch+ benchmark and about four rounds on MMSearch. This indicates that the trained agent does not follow a fixed search template, but adjusts its search depth to the task.

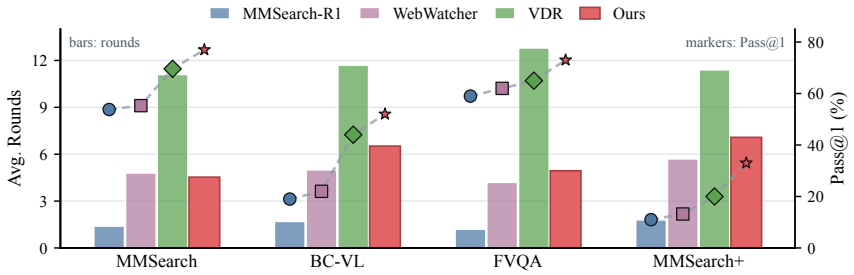


Fig. 10: **Inference rounds and Pass@1.** Bars show the average number of reasoning rounds on each benchmark, while markers and dashed segments report the corresponding Pass@1 scores.

G DISCUSSION

Beyond the main SimpleSearch-VL recipe, our codebase also includes two exploratory training options that were implemented but not used in the final reported system. Although they are not part of the main experimental results, they are useful for understanding how rollout selection and interaction efficiency affect multimodal agentic RL.

Mastered-sample filtering. The optional `mastered_sample_filter` mechanism filters out prompt groups whose recent rollouts are all correct. This is related in spirit to GRESO-style (Zheng et al., 2026) selective rollouts, which move filtering before generation: if a prompt has recently been consistently all-wrong or all-correct, it is likely to remain zero-variance, so generating a full rollout group only to discard it wastes real computation. The difference is that our filter is simpler and more asymmetric. Rather than deciding whether every prompt should be sampled in the next step, it directly removes mastered all-correct prompts so that the remaining rollout budget is spent on harder examples. This design matches our goal of mining difficult samples. In practice, after easy groups were removed, the average number of interaction rounds increased quickly, suggesting that the policy was exposed to harder prompts that required longer tool-use trajectories.

Round-efficiency reward. We also explored a round-efficiency reward that gives an additional bonus to successful trajectories completed with fewer interaction rounds. Concretely, the training loop maintains an online reference for each sample, such as the historical minimum or average number of completion rounds, and rewards a rollout when it solves the sample more efficiently than that reference. This shaping term had the expected behavioral effect: the model learned to shorten successful trajectories and avoid some unnecessary search detours. However, the reduction in rounds did not consistently translate into better final task accuracy. Since multimodal search often benefits from spending extra steps on evidence verification, we leave this reward as an optional engineering setting rather than including it in the main SimpleSearch-VL recipe.

H VISUALIZATION

H.1 STANDARD SINGLE-IMAGE CASES

Fig. 11 and Fig. 12 show standard single-image trajectories. In Fig. 11, the agent crops the left half of a composite news image, retrieves visually matching thumbnails for Don Lemon, and then visits supporting pages before answering. In Fig. 12, reverse image search first grounds the animated scene as Disney's Evil Queen and magic mirror, after which webpage evidence and text search verify that Gal Gadot plays the Evil Queen in the 2025 live-action *Snow White*. These examples show the basic evidence chain used by SimpleSearch-VL: region-level reverse image search proposes an entity, thumbnail consistency filters the proposal, and text or webpage evidence confirms the final answer.

H.2 SINGLE-IMAGE CASE WITHOUT IMAGE SEARCH

Fig. 13 shows a simple case where the model chooses not to use image search. The input image contains a readable platform logo, "fineartamerica", and the question asks about the type of prints made from high-quality cotton/poly canvas on that platform. Since this visual cue is already sufficient to identify the platform, the agent skips reverse image search, directly issues text-search queries about Fine Art America canvas prints, visits the relevant product pages, and verifies that the platform sells canvas art prints made with premium cotton/poly blend canvas, archival inks, and stretcher bars. This example illustrates adaptive tool allocation: when the image already supplies a reliable textual anchor and the problem is straightforward, the model can avoid unnecessary reverse image search and move directly to textual evidence gathering.

H.3 MULTI-FRAME CASES

Fig. 14 and Fig. 15 illustrate multi-frame reasoning. The simulator example first searches one cockpit image and obtains generic aviation evidence, then switches to the second frame, where the CNBC-style visual match identifies a United Airlines pilot-training video. The agent visits and cross-checks this evidence to answer that the facility belongs to United Airlines in Denver. The historical-event

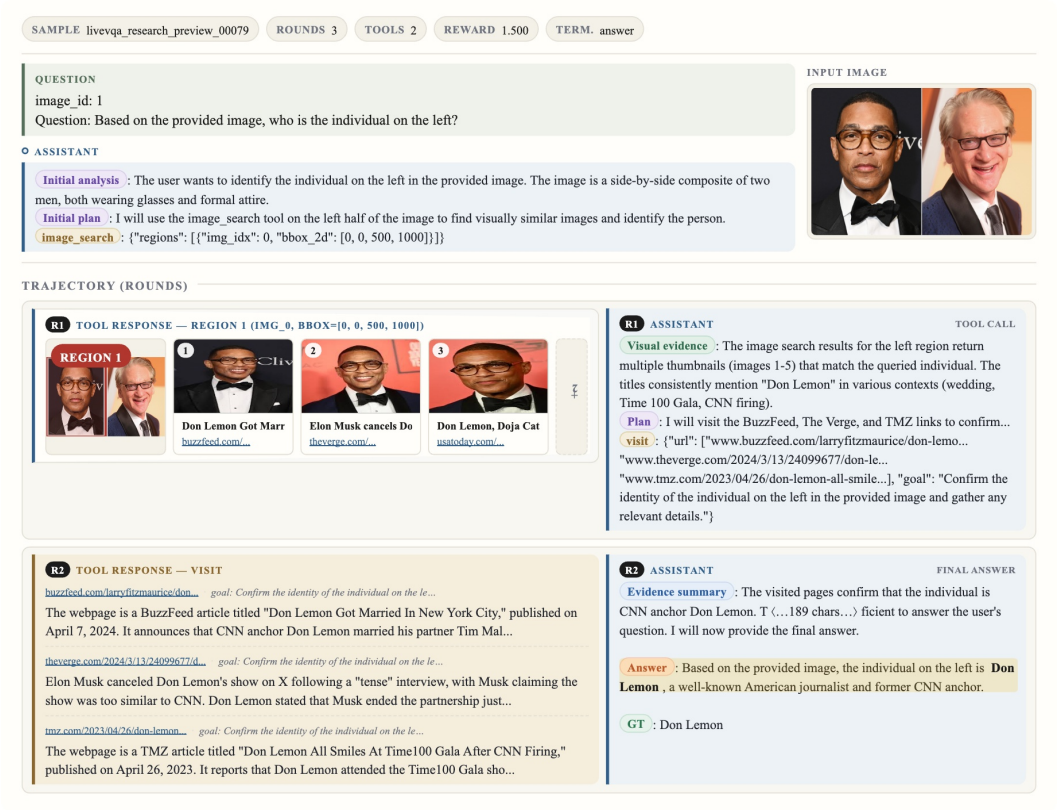


Fig. 11: **Standard single-image case.** The agent searches the left image region, matches retrieved thumbnails and titles to Don Lemon, and verifies the identity with visited webpages before answering.

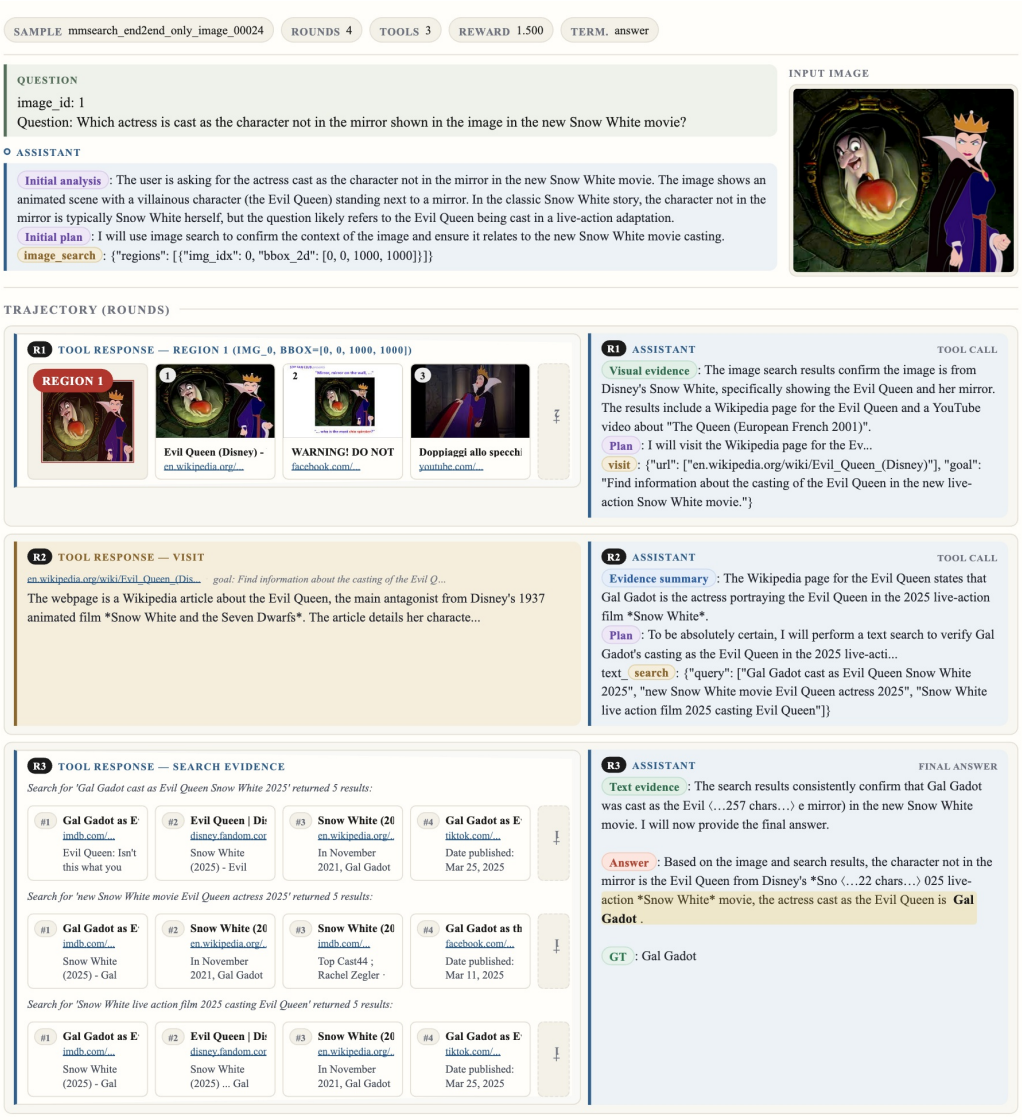


Fig. 12: **Standard single-image case with visual-to-text verification.** The reverse-image-search result grounds the scene as the Evil Queen from *Snow White*; webpage and text evidence then confirm Gal Gadot as the actress in the new film.

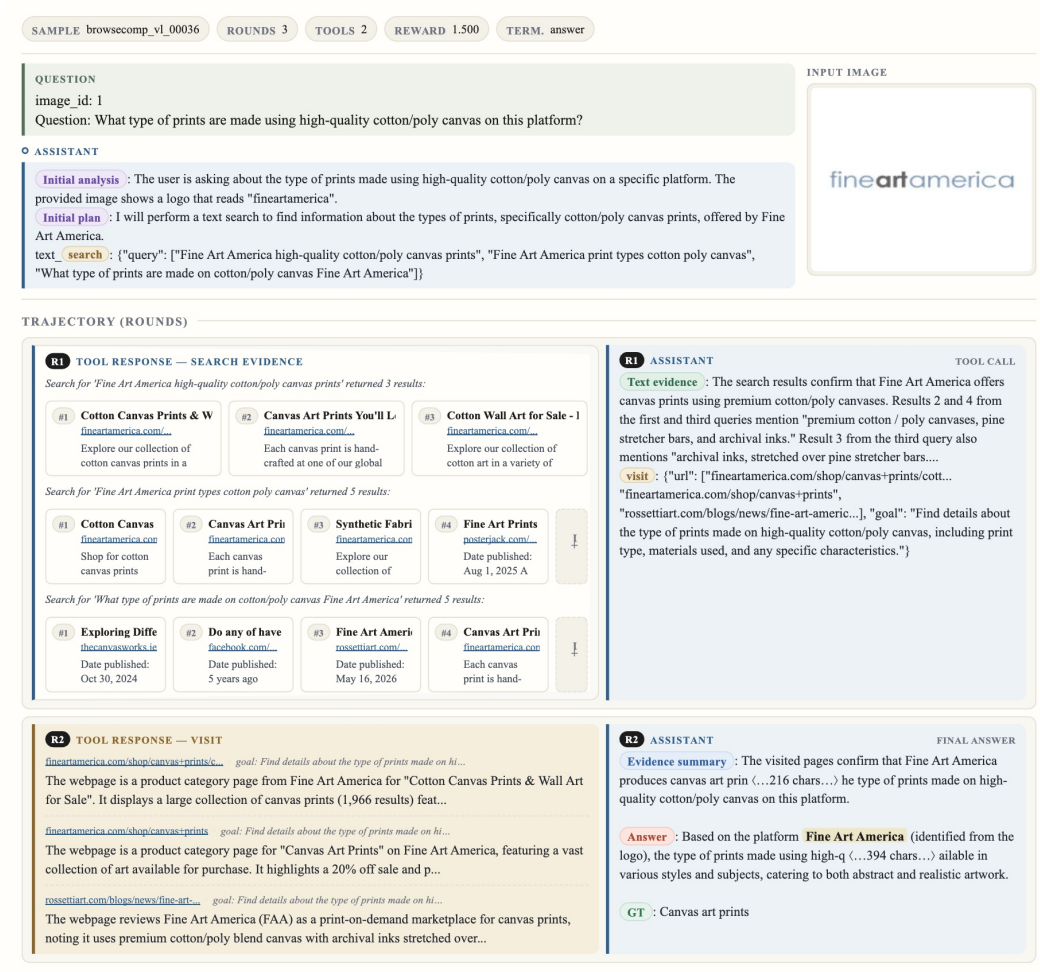


Fig. 13: **Single-image case without image search.** For this straightforward logo-based question, the model does not call image search; it uses the visible Fine Art America text as an anchor and verifies the answer through text search and webpage visiting.

example searches both frames in one visual step: the first image is linked to the 1932 Ford Hunger March, while the second is linked to the 1941 Ford Strikers Riot. The final answer is obtained by verifying both dates and computing the nine-year gap. These trajectories show why the interface exposes image indices explicitly: the model can decide which frame is informative, and can compare evidence across frames when the question requires relational reasoning.

H.4 Multi-Region Case

Fig. 16 shows a trajectory that searches multiple regions within the same image. The full-image query only establishes a broad England-football context, so the agent refines the visual query by cropping the goalkeeper and the second player separately. The first region identifies Jordan Pickford, while the subsequent textual and webpage evidence clarifies that the relevant recall story concerns Jordan Henderson under new England manager Thomas Tuchel. This example demonstrates the role of region-level reverse image search: rather than trusting a whole-image match, the model can isolate entities and then use text evidence to resolve the question-specific relation.

H.5 Failure Cases

Fig. 17 and Fig. 18 show two failure modes. In Fig. 17, the image is only a thematic cue for songwriting, and reverse image search returns generic songwriting pages rather than the intended referent. The agent then follows a plausible but wrong text-search path to Taylor Swift's Songwriters Hall of Fame induction, whereas the benchmark target is Ben Peters. This is an entity-selection failure: the external evidence is internally consistent, but it resolves the wrong hidden target. In Fig. 18, the visible sticker correctly identifies Dan Reeder, and the agent retrieves track-list evidence for *Smithereens* and *little bitty songs*. However, it selects "Gin Tonic" as the overlapping song, while the gold answer is "Fun Campfire Song". This is a fine-grained evidence extraction failure after the visual grounding step has already succeeded.

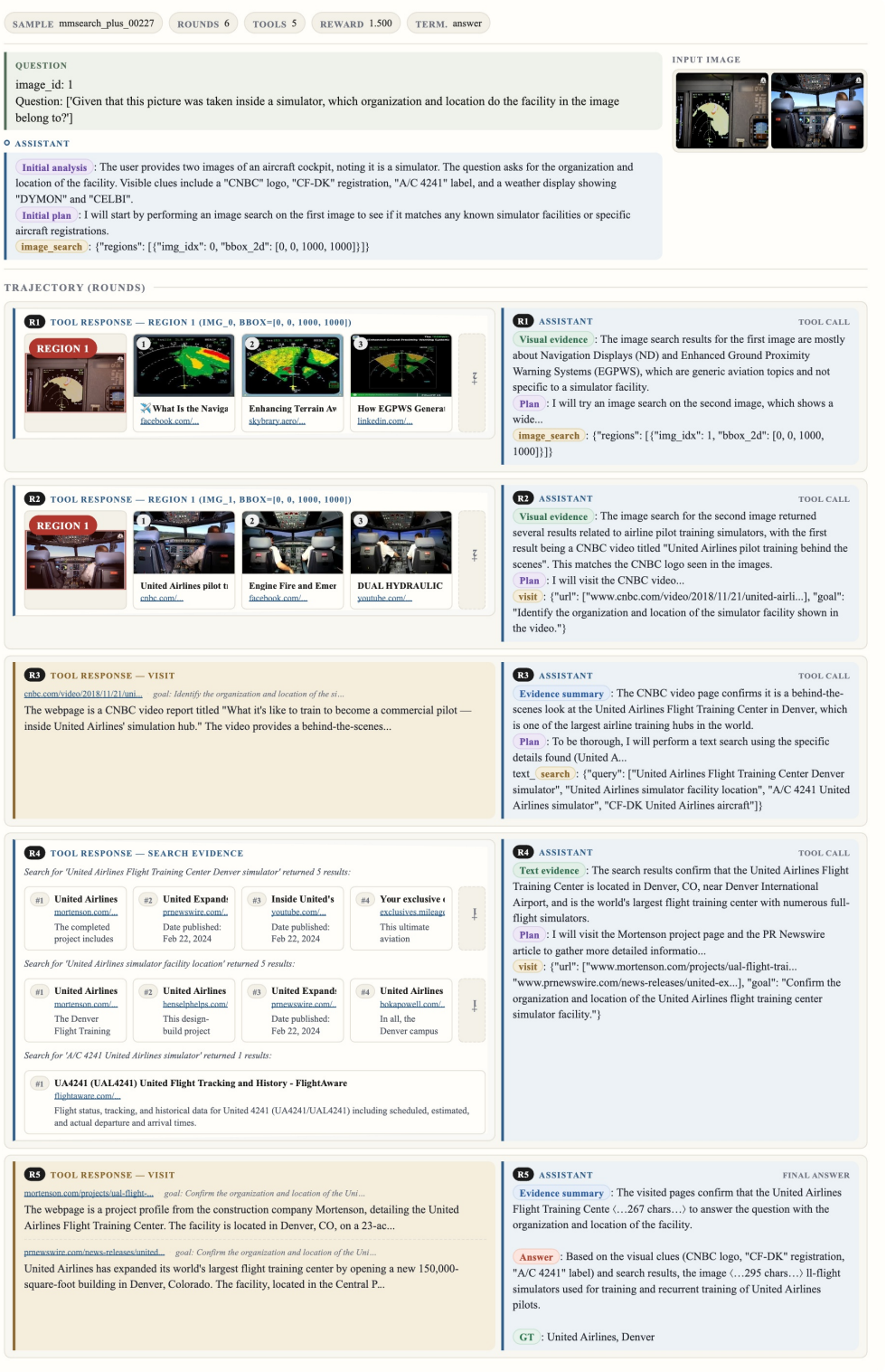


Fig. 14: **Multi-frame case with frame selection.** The first cockpit frame gives generic simulator evidence, while the second frame retrieves a United Airlines training video; the agent then verifies that the facility is the United Airlines Flight Training Center in Denver.



Fig. 15: **Multi-frame case with cross-image comparison.** The agent links the first frame to the 1932 Ford Hunger March and the second frame to the 1941 Ford Strikers Riot, verifies both dates through webpage evidence, and computes the correct nine-year interval.



Fig. 16: **Multi-region case.** After a broad whole-image reverse search, the agent searches localized player regions, uses the goalkeeper match to anchor the England-squad context, and verifies from webpages that Thomas Tuchel recalled Jordan Henderson.



Fig. 17: **Failure case with wrong entity selection.** A generic songwriting image leads the agent to search for a plausible Songwriters Hall of Fame induction and answer Taylor Swift, although the benchmark target is Ben Peters.

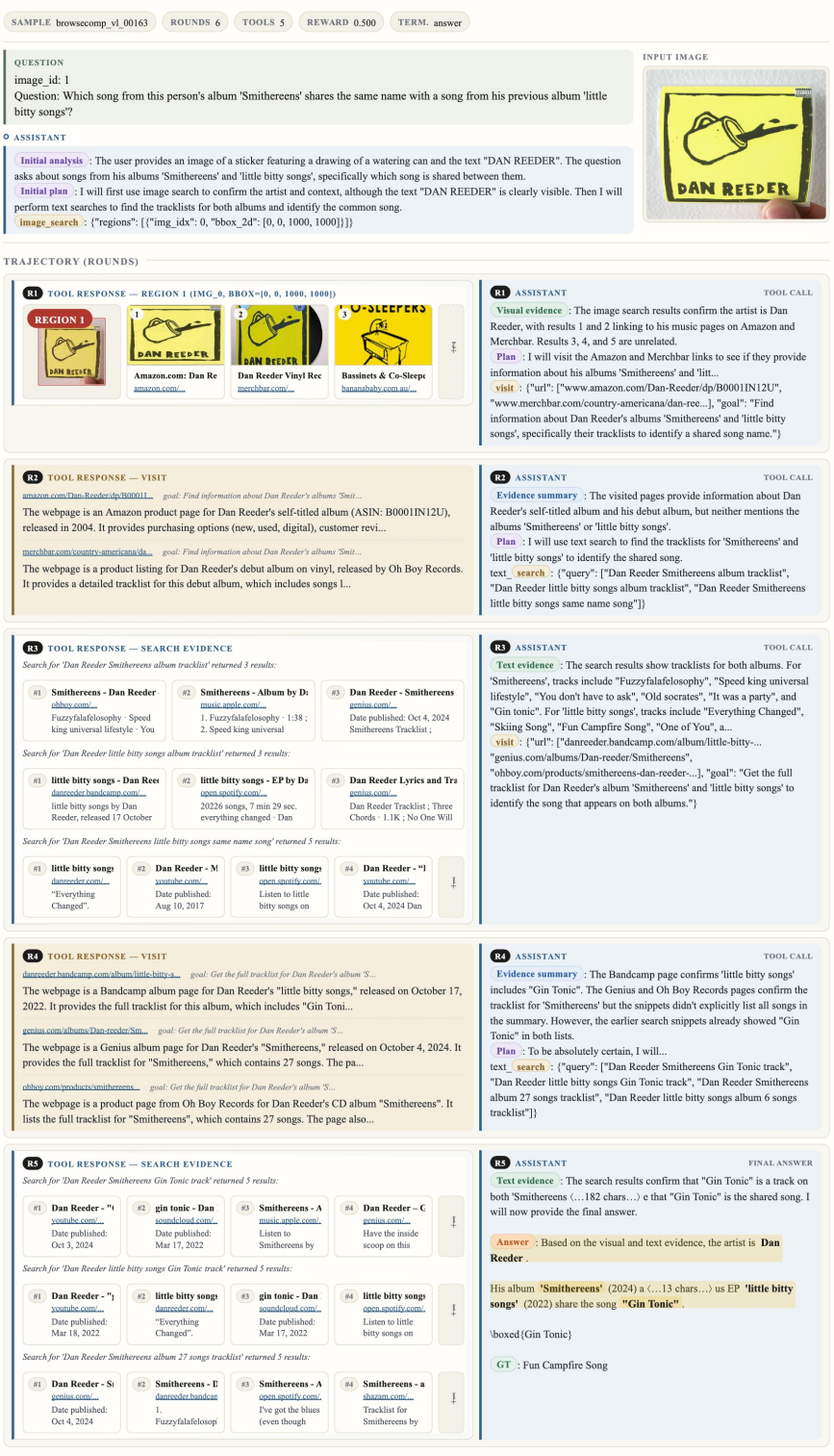


Fig. 18: **Failure case after correct visual grounding.** Reverse image search identifies Dan Reeder, but the later track-list comparison selects "Gin Tonic" instead of the target shared song, "Fun Campfire Song".