LOGT: Logically Think with Images for Visual Search

Yanjun Fu[1,2,*] **Quanzeng You**[1,†] **Jiadong Guo**[1] **Yujie Lu**[1] **Sai Charitha Akula**[1]
Haiping Wu[1] **Miao Liu**[1,†] **Sanghamitra Dutta**[2,†] **Lu Yuan**[1,§]

Abstract

Thinking with images empowers vision-language models (VLMs) to use an external ZOOM-IN tool for fine-grained visual search. Recent works employ supervised fine-tuning (SFT) and reinforcement learning (RL) with tool-specific designs to enable such capabilities. However, achieving logical reasoning alongside accurate and efficient tool use remains a significant challenge. To address this, we shift the focus away from increasing RL complexity and instead identify SFT data quality as a key bottleneck. We then propose LOGT (short for **Log**ically **T**hink with Images), a fully automated SFT data engine that explicitly enforces logic-chain-guided reasoning. LOGT generates an SFT dataset featuring challenging, visually grounded questions and structured tool-use traces with human-like logic. We validate our approach by fine-tuning two base models, Qwen2.5-VL-7B and InternVL3.5-8B, on the dataset and evaluating them across seven benchmark test sets. We show that LOGT-SFT based on Qwen2.5-VL-7B outperforms the state-of-the-art trained with both SFT and RL, despite using 3.1$\times$ less data and $>$120$\times$ fewer GPU hours. Furthermore, applying a simple RL recipe without tool-specific design on LOGT-SFT yields a model that surpasses prior methods with complex RL setups by a substantial 9.59-point margin. In addition to strong performance gains, our model exhibits higher tool-use accuracy and efficiency, as well as more structured and logically consistent reasoning traces, directly validating the effectiveness of our approach.

1 Introduction

Vision-Language Models (VLMs) [1, 2] integrate visual and textual information by aligning high-level visual cues with text embeddings via visual encoders and projection layers. However, this alignment process introduces irreversible information loss, causing VLMs to struggle with visually intensive tasks that require fine-grained visual search [3]. Even when augmented with text-based chain-of-thought (CoT) reasoning [4, 5], such limitations persist, as textual reasoning cannot recover missing visual details. Therefore, recent works [6, 7] introduce thinking with images, enabling VLMs to invoke external tools to extract additional visual information and interleave it with textual reasoning. In this context, we investigate how VLMs can more effectively leverage the widely studied ZOOM-IN tool [7–12] for visual search.

Existing efforts approach this problem through post-training, typically combining supervised fine-tuning (SFT) for cold start with reinforcement learning (RL) driven by outcome-based and tool-related rewards. While these methods have shown promising results, we find that the improvement largely stems from increased familiarity with the tool format gained during post-training. In contrast, enabling VLMs to logically reason about when and how to invoke the tool for accurate region-of-interest identification and efficient inference remains a fundamental challenge [13] (see analysis in Section 4.5). While RL with finer-grained step-by-step process supervision using process reward models (PRMs) [14] has shown early promise in improving reasoning aided by non-visual tools,

* Work was done at Meta † Equal advising § Project lead [1] Meta Superintelligence Labs (MSL)
[2] University of Maryland Correspondence to: {yanjunfu@umd.edu, qyou@meta.com}

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

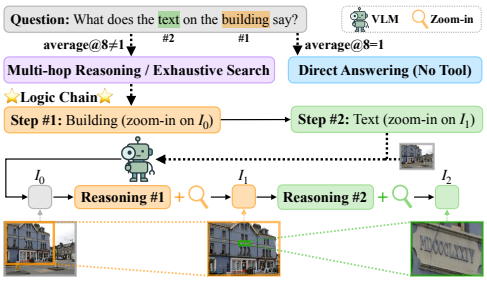


Figure 1: **Main idea.** We generate reasoning traces based on human-like strategies including (i) *direct answering* for simple questions, (ii) *multi-hop reasoning* by focusing on different regions, and (iii) *exhaustive search* for more difficult cases. For easy questions where the base model's average@8 is 1, we synthesize direct answering traces without ZOOM-IN. Otherwise, we first generate *logic chains* that specify key tool-use steps and then use them to guide the VLM in synthesizing tool-use reasoning traces.

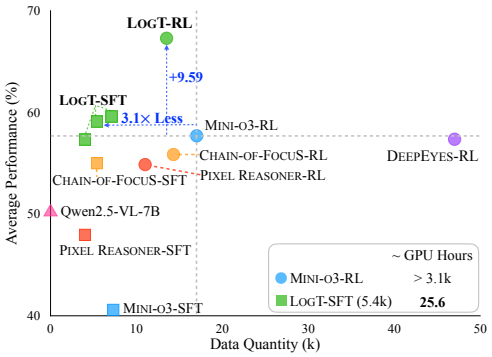


Figure 2: **Result highlights.** By applying SFT to a Qwen2.5-VL-7B model using 5.4k to 7.1k of our synthesized data, LOGT-SFT surpasses the best-performing MINI-O3-RL on average across seven test sets, while using up to 3.1× less data and >120× fewer GPU hours. With a simple RL setup without any tool-specific design, LOGT-RL further outperforms baselines by 9.59.

PRMs for training VLMs to think with images and use ZOOM-IN effectively and efficiently remain underexplored, largely due to the limited scalability and high cost of training dedicated PRMs.

Therefore, we shift the focus from designing more complex RL to improving SFT data quality, which we identify as *a key bottleneck*. We propose LOGT, a novel data engine that generates SFT data to teach VLMs to **Log**ically **T**hink with Images for visual search. Importantly, LOGT is fully automated and relies solely on open-source models to enhance the quality of SFT data. Each synthesized SFT sample includes a visual search question-answer (QA) pair, and the corresponding tool-use reasoning traces. Accordingly, we design LOGT to improve the quality of both QAs and reasoning traces.

For QA quality, we enrich the VLM's input with detailed visual contexts in addition to the full image, enabling the synthesis of more challenging questions that target fine-grained visual details. We then apply rigorous filtering to retain only high-quality QA pairs. For reasoning trace quality, we first construct logic chains that mimic human problem-solving strategies (*e.g.*, multi-hop reasoning in Figure 1). These logic chains serve as skeletons and outline the key steps needed to solve each visual search task. We then use them to guide the VLM in generating structured and coherent reasoning traces with appropriate tool calls.

To demonstrate the effectiveness of LOGT, we train Qwen2.5-VL-7B [15] and InternVL3.5-8B [16] on our synthesized SFT dataset comprising 7.1k high-quality entries. As shown in Figure 2, using 3.1× less data and >120× fewer GPU hours, our SFT-only model (LOGT-SFT based on Qwen2.5-VL-7B) outperforms the state-of-the-art (SOTA) open-source model trained by both SFT and RL across seven test sets. We further apply off-the-shelf RL with straightforward outcome-based rewards, and the resulting model outperforms prior methods by at least 9.59 points, despite their reliance on more complex, tool-specific RL strategies. In addition, both our SFT and RL checkpoints achieve better tool-call accuracy and efficiency. A human evaluation further shows that our method produces more coherent and higher-quality reasoning traces.

In summary, our contributions are threefold:

- We identify SFT data quality as a key bottleneck and introduce LOGT, a novel data synthesis strategy that generates high-quality SFT data with challenging QA pairs and coherent, human-like reasoning traces. This strategy produces a dataset with 7.1k entries.
- We show that training Qwen2.5-VL-7B solely on our SFT data outperforms prior SOTA models that use both SFT and RL, while also achieving significantly higher data and training efficiency, highlighting the benefit of investing more compute and effort in SFT data quality.
- We further improve LOGT-SFT with a simple RL setup without any tool-specific design, underscoring the importance of SFT quality. The resulting model also achieves higher tool-call accuracy, greater tool-call efficiency, and more logically coherent reasoning traces.

2 Related Work

Think with images. Thinking with images extends text-only CoT reasoning by allowing VLMs to interleave textual reasoning with visual information extracted via external tools from either the input image or intermediate tool outputs. Some methods [6, 17] adopt a training-free, modular agent-based design, allowing highly powerful VLMs (*e.g.*, GPT-4o [18]) to invoke visual tools during inference. In contrast, other approaches, including OpenAI's o3 [7, 19–21], rely on post-training to incentivize VLMs with tool-use capabilities.

Among prior work, ZOOM-IN is one of the most widely used visual tools [7–12, 22–24], showing consistent improvements in visual search [3]. Most methods employ a two-stage post-training pipeline of SFT followed by RL. SFT primarily teaches models to invoke ZOOM-IN and conform to its calling format, which is useful because current VLMs exhibit limited zero-shot tool-use ability. RL then uses outcome-based rewards to further encourage tool usage and format compliance. However, these approaches still face fundamental challenges in training VLMs to reason logically with images and to call ZOOM-IN both accurately [24] and efficiently [13]. Li *et al*. [12] improve tool-call efficiency through designing complex RL rewards and rollout strategies, but at the trade-off of substantial RL compute overhead. RL with process supervision could potentially mitigate these limitations. However, this often requires PRMs [14]. To the best of our knowledge, PRMs tailored for thinking with images remain underexplored, and training such specialized PRMs that meet our requirements is both challenging and very intensive.

SFT data for visual reasoning. Building SFT datasets for thinking with images requires reasoning traces that interleave text with images returned by tool calls. This is challenging because most existing SFT datasets for visual reasoning [25–29] contain only textual reasoning, without intermediate visual evidence. A few datasets [30, 31] do provide reasoning traces that interleave texts and images using referenced bounding boxes. However, they typically use low-resolution images and include only a single bounding box per trace, making them inadequate for high-resolution and fine-grained visual search tasks requiring multi-step tool use.

Recent works on thinking with images [8, 9, 11, 23] release their own SFT datasets. These studies typically generate visual search questions by directly prompting VLMs with full input images. Consequently, the synthesized questions often center on easy-to-search objects. Furthermore, the reasoning traces are usually generated by directly prompting VLMs or agents without careful quality control, and they often lack the human-like logical structure for solving complex problems, as presented in our approach. We also note that, despite the release of datasets and checkpoints in prior work, the data generation engines themselves receive less attention. In contrast, we focus on the data generation strategy itself.

3 Our Data Engine: LOGT

We propose a novel data synthesis framework, LOGT, for generating an SFT dataset with high-quality visual search QA pairs and reasoning traces. This dataset is used to enhance VLMs' ability to use ZOOM-IN for fine-grained visual search. Our goal is to train VLMs not only to invoke ZOOM-IN in the correct format over one or multiple turns, but also to logically think with images and to use ZOOM-IN accurately and efficiently to identify regions of interest.

Figure 3 illustrates the process for generating a single-object QA and its corresponding tool-use reasoning trace. Given an image I_0 and two object-centric crops that provide contextual (I_C) and focused (I_F) views of the target object, the QA generator produces candidate QA pairs, which are then filtered using automatic quality checks. We then pass the generated QAs along with a synthesized caption of I_0 to a logic chain generator, which produces a multi-hop chain L_M and an exhaustive search chain (L_E). One of these chains is selected and used to guide the reasoning trace generator to construct a full trace that interleaves textual reasoning with ZOOM-IN actions. Each ZOOM-IN call extracts a region from I_0 or a previous crop, resulting in crops denoted as I_1 through I_3.

While ensuring high data quality, LOGT is *fully automated* and *accessible*, relying solely on open-source models without needing highly powerful models or complex visual agents. In the following sections, we detail the key components of (i) QA pair generation (Section 3.1) and (ii) reasoning trace synthesis (Section 3.2).

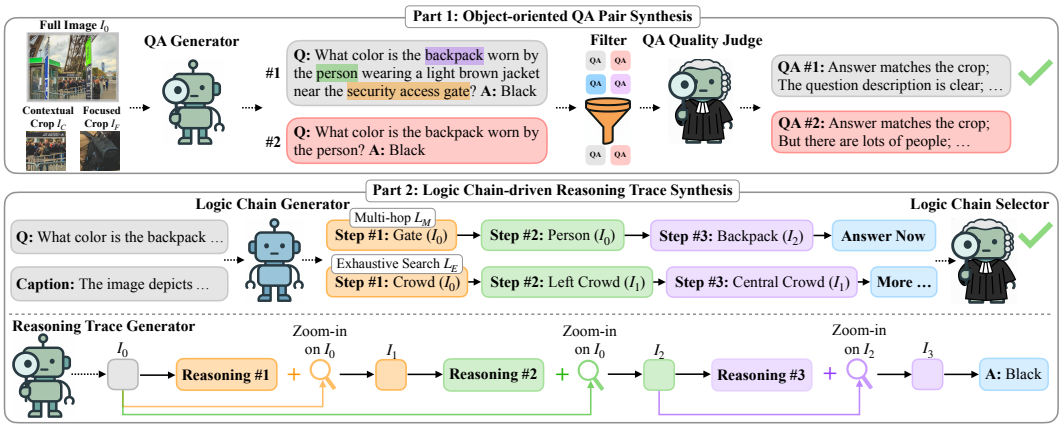


Figure 3: **A detailed overview of LogT SFT data synthesis engine.** Part 1 (top; Section 3.1) synthesizes visual search QA pairs using the full image (I_0) along with rich visual contexts (I_C and I_F, top left) related to randomly selected target objects. After generating candidate QAs, we apply a simple heuristic filter and use a VLM-as-a-Judge to ensure quality. Part 2 (bottom; Section 3.2) generates reasoning traces for the selected QA. For easy questions, we directly generate traces without tool calls (omitted here). For other questions, we first create logic chains (L_M and L_E) based on two human-like strategies. A logic chain selector then chooses the more appropriate one, which is subsequently used to guide the synthesis of multi-turn reasoning traces with tool use.

3.1 Object-oriented QA Pair Synthesis.

Our QA pair synthesis strategy consists of two steps: (i) generating challenging QA pairs using a QA generator, and (ii) performing quality control through a filter and a QA quality judge. Specifically, we use Qwen2.5-VL-72B [15] as the generator and a different open-sourced VLM with 106B parameters, as the judge.

As preparation, we source high-resolution images from the SA1B dataset [32], similar to PIXEL REASONER [8] and CHAIN-OF-FOCUS [9]. For each image, SA1B provides bounding boxes representing unnamed objects.

Enhance QA generation with object-oriented visual contexts. We generate visual search QA pairs in an *object-oriented* manner, with each focusing on one or two specific objects within the input image I_0. To avoid bias toward easily identifiable objects and increase question difficulty, we first filter out bounding boxes that cover more than 1% of the image area. From the remaining boxes, we randomly select one box for single-object questions, or a pair of boxes for relation-based questions. In addition to the full image, we prepare two more crops related to each bounding box: (i) a contextual crop I_C that is several times larger than the bounding box to capture surrounding visual context, and (ii) a focused crop I_F that exactly matches the bounding box to capture fine-grained object details. This results in 3 images for single-object and 5 for relation-based questions. We provide the QA generator with these images, detailed generation rules, a taxonomy of permissible question types, and in-context examples to produce two candidate QA pairs. The QA generation prompt is shown in Figure 7.

QA filtering and selection. We first apply a simple rule-based filter to eliminate QAs that violate output format requirements. Then we evaluate the remaining pairs using the QA quality judge to determine whether one or both conform to our quality criteria, including answer correctness, question unambiguity, *etc*. If neither QA candidate passes, we discard both; if both are valid, the judge selects the better one. To mitigate positional bias of the judge [33], we run the evaluation twice with the QA order reversed. We err on the side of caution by accepting a QA pair only if it is consistently validated across both evaluations.

Compared to prior work, our method produces more challenging and higher-quality questions. We define average@8 as the accuracy obtained by randomly sampling responses eight times at a temperature of 0.8. As shown in Figure 11, only 24% of our synthesized questions are perfectly answered by Qwen2.5-VL-7B without tool use, *i.e.*, average@8 = 1, substantially fewer than in existing datasets.

3.2 Logic Chain-driven Reasoning Trace Synthesis.

In visual QA tasks, humans typically adopt one of the following reasoning strategies. We leverage these strategies to ensure the logical coherence of generated reasoning traces and to minimize unnecessary tool usage: (i) **Direct answering:** When the target object is prominent and the question is straightforward, humans can answer directly without intermediate reasoning. (ii) **Multi-hop reasoning:** For questions requiring indirect visual cues, like "What color is the backpack worn by the person?", humans identify intermediate visual anchors (*e.g.*, the person), then refine their focus (*e.g.*, the backpack), reasoning through multiple "hops". (iii) **Exhaustive search:** When the target is difficult to locate and the question lacks clear visual references, such as asking to find a person in a crowd, humans may divide the scene into several parts and search systematically.

For easy questions where Qwen2.5-VL-7B achieves an average@8 of 100% even without requiring tools, we adopt the direct answering strategy to generate a more tool-efficient reasoning trace, consisting of a single turn without invoking the ZOOM-IN tool.

Otherwise, we simulate the remaining two strategies and synthesize reasoning traces that involve one or more tool calls. This process includes: (i) abstracting these strategies into *logic chains* using a logic chain generator, where each chain consists of a sequence of key steps that specify where to focus and which image to zoom into, (ii) selecting the most appropriate logic chain via a logic chain selector, and (iii) leveraging the selected logic chain as structured guidance for the reasoning trace generator to produce coherent tool-use reasoning traces. While an ideal system would rely on a single VLM to perform all three parts, current open-sourced VLMs often lack the instruction-following reliability and fine-grained reasoning precision required for high-quality logic chain generation. To address this, we use GPT-OSS-120B [34] as the logic chain generator, and the VLM serving as the QA quality judge for both logic chain selection and reasoning trace generation.

Logic chain generation and selection. Given an image I_0 and a QA, we first supply the logic chain generator with the question and a VLM-generated image caption as necessary visual context. It then outputs two types of logic chains: one for multi-hop reasoning (L_M) and one for exhaustive search (L_E). Each chain comprises sequential steps that identify a target object or region (*e.g.*, "person" or "left crowd") and determine which image to zoom into (*e.g.*, the full image I_0 or a crop returned from a previous ZOOM-IN call, such as I_1 or I_2 in Figure 3). The logic chain generator must reason over spatial and contextual dependencies. For example, in Figure 3, it needs to decide whether to zoom in gradually, such as identifying the backpack from a crop of the person, or to access multiple objects in parallel, such as finding the security gate and the person.

Once both logic chains are generated, we pass them along with the full image and question to the logic chain selector. We use it to choose a more appropriate chain based on task complexity and tool call efficiency.

Reasoning trace synthesis. Next, we prompt the reasoning trace generator to follow the selected logic chain step-by-step, producing a reasoning thought and a corresponding ZOOM-IN action at each turn until reaching the final answer. The prompt used for reasoning trace synthesis is shown in Figure 8. Finally, we build an SFT dataset with 7.1k high-quality QAs and reasoning traces.

We present example multi-hop reasoning traces in Figure 12 and an example exhaustive search reasoning trace in Figure 9. Tool-use statistics of our synthesized SFT dataset are provided in Section A.3 and visualized in Figure 10. To qualitatively demonstrate that LOGT produces higher-quality reasoning traces, we present a case study comparing our traces with those from prior work in Section A.4. Finally, we analyze the computational cost of our data engine in Section A.5.

4 Experiments

4.1 Experimental Setup

Baselines. We compare our method against a broad set of baselines: (i) **VLMs without ZOOM-IN**, including GPT-4o [18], and open-source models such as Qwen2.5-VL-7B and -72B [15]. (ii) **VLMs with ZOOM-IN**, including o3 [7], and several open-source models post-trained for ZOOM-IN, such as PIXEL REASONER [8], DEEPEYES [10], CHAIN-OF-FOCUS [9], MINI-O3 [11], and Li *et al.* [12]. All open-source baselines here use Qwen2.5-VL-7B as the base model and rely on ZOOM-IN as the sole image-interaction tool. They all apply SFT for cold-start training before RL, except

Table 1: **Performance comparison between our models and baselines.** We evaluate the pass@1 accuracy of various models on seven test sets across four benchmarks, and use MME-RW-Lite to denote the out-of-domain benchmark MME-RealWorld-Lite. As checkpoints from Li *et al*. [12] are not publicly available yet, we report evaluation results directly from the original paper and leave unavailable results blank. For open-sourced SFT and RL models, the best and second-best results are marked in **bold** and underlined, respectively. The green rows refer to our models. Using only 5.4k–7.1k samples from the SFT dataset synthesized by LOGT, the resulting LOGT-SFT outperforms existing baselines, including those trained with both SFT and RL. After applying an off-the-shelf RL recipe without any tool-specific design (see Section 4.2), LOGT-RL surpasses all open-sourced baselines by at least 9.59 points.

	Data		V^*	VisualProbe			HR-Bench		MME-RW-Lite	Avg.
	SFT	RL		Easy	Medium	Hard	4K	8K		
GPT-4o	-	-	66.00	47.50	15.40	11.20	59.00	55.50	46.40	43.00
o3	-	-	95.70	69.50	47.01	32.08	-	-	-	-
Open-sourced Models w/o Tools										
Qwen2.5-VL-7B	-	-	79.58	41.81	23.13	19.81	72.38	66.25	48.46	50.20
Qwen2.5-VL-72B	-	-	79.06	39.72	32.09	28.30	76.25	73.00	50.13	54.08
Open-sourced SFT Models with Tools										
PIXEL REASONER-SFT	4k	0	74.87	49.65	18.66	20.75	64.75	59.88	47.06	47.95
CHAIN-OF-FOCUS-SFT	5.4k	0	83.77	53.90	27.61	34.91	71.50	65.25	48.05	55.00
MINI-O3-SFT	7.3k	0	64.40	32.62	28.73	14.15	53.75	51.75	38.56	40.57
LOGT-SFT	4k	0	85.34	**61.70**	34.70	36.79	71.50	67.25	43.93	57.32
LOGT-SFT	5.4k	0	85.86	60.99	38.06	37.74	**72.25**	**70.62**	**48.15**	59.10
LOGT-SFT	7.1k	0	**87.43**	60.99	**41.04**	**39.62**	72.00	70.12	45.96	**59.59**
Open-sourced RL Models with Tools										
PIXEL REASONER-RL	4k	7k	81.68	58.87	24.25	26.42	73.00	67.38	52.42	54.86
CHAIN-OF-FOCUS-RL	5.4k	8.9k	83.25	55.32	28.36	36.79	72.50	65.88	48.83	55.85
DEEPEYES-RL	0	47k	83.25	62.41	27.99	30.19	74.62	69.50	**53.47**	57.35
Li *et al*. [12]	4k	7k	85.86	-	-	-	73.40	66.60	-	-
MINI-O3-RL	7.3k	9.7k	89.53	51.77	45.15	36.79	71.25	66.62	42.73	57.69
LOGT-RL	7.1k	6.4k	**92.15**	**70.92**	**52.24**	**51.89**	**76.12**	**74.62**	53.00	**67.28**
Δ vs. SOTA	-	-	**+2.62**	**+8.51**	**+7.09**	**+15.10**	**+1.50**	**+5.12**	**-0.47**	**+9.59**
Δ vs. Base Model	-	-	**+12.57**	**+29.11**	**+29.11**	**+32.08**	**+3.74**	**+8.37**	**+4.54**	**+17.08**

for DEEPEYES. Notably, all adopt tool-specific RL strategies. Additional baselines are evaluated in Table 2 and discussed in Section B.3.

Benchmarks. We evaluate our method and baselines on the following benchmarks: (i) **Fine-grained visual search on high-resolution images**: This includes V^* [3], VisualProbe [11], and HR-Bench [35]. VisualProbe is officially divided into three difficulty levels: easy, medium, and hard. HR-Bench consists of two resolution-based subsets: 4K and 8K. (ii) **Out-of-domain visual perception and reasoning beyond visual search:** We use MME-RealWorld-Lite [36], which differs substantially from the LOGT SFT dataset in both the visual content and the types of tasks involved. This makes it well-suited for assessing models' generalization to out-of-domain visual reasoning tasks beyond visual search. MME-RealWorld-Lite is a comprehensive benchmark with 1,919 questions covering 43 task types, evaluating general perception and reasoning capabilities of VLMs across diverse scenarios such as OCR, remote sensing, diagram understanding, *etc*. In total, our evaluation covers seven test sets, accounting for multiple official splits in VisualProbe and HR-Bench.

Evaluation details. We use greedy decoding for all evaluated models[1] and report pass@1 accuracy to ensure reproducibility. For all open-sourced models without built-in tool usage, we apply CoT prompting using the instruction "Please reason step by step." For o3, we report the V^* result from its model card and evaluate VisualProbe using medium reasoning effort with the code interpreter tool. Additional evaluation details are provided in Section B.1.

[1] PIXEL REASONER and CHAIN-OF-FOCUS pad or enlarge model-predicted bounding boxes before applying the ZOOM-IN operation. We argue that altering bounding boxes can artificially inflate grounding and tool call accuracy, while increasing inference cost due to additional visual tokens. To ensure a fair comparison, we disable all bounding-box modifications and evaluate using the model-predicted boxes directly.

4.2 Implementation Details

SFT implementation. Following prior works, we use Qwen2.5-VL-7B as the base model. We train it on the SFT dataset synthesized by LOGT for 3 epochs using a learning rate of 1e-5 and an effective batch size of 128. The resulting model is LOGT-SFT.

RL implementation. We apply off-the-shelf RL to LOGT-SFT fine-tuned on 7.1k samples, to investigate how improved cold-start affects post-RL performance. The resulting model is LOGT-RL. Our reward function consists of only two terms: a correctness reward (r_{correct}) and a formatting reward (r_{format}). The correctness reward assigns $r_{\text{correct}} = 1$ if the model's predicted answer matches the ground-truth, and 0 otherwise. The formatting reward assigns $r_{\text{format}} = 0.5$ if the model follows the correct format, and 0 otherwise. The total reward is $r = r_{\text{correct}} + r_{\text{format}}$.

For the RL algorithm, we slightly modify standard Group Relative Policy Optimization (GRPO) [37] to encourage exploration. Specifically, we remove KL regularization and adopt the clipping-higher strategy recommended by Yu *et al.* [38] for agentic RL to set the upper clipping ratio to 0.315. We sample 8 rollouts per input, and adopt a constant learning rate of 1e-6. We use veRL [39] for implementation and run RL for 100 steps.

We source RL training data from MINI-O3 [11]. Additional details on our RL implementation are provided in Section B.2.

4.3 Main SFT Results

In this section, we report SFT results using Qwen2.5-VL-7B as the base model. We also fine-tune InternVL3.5-8B [16] on our SFT dataset and present those results in Table 3 of Section B.4.

LOGT-SFT outperforms all baselines, including those trained with both SFT and RL, while being highly data- and training-efficient. As shown in Table 1, the LOGT-SFT model trained on 7.1k synthetic samples generated by our method, surpasses all open-source baselines, including those post-trained with both SFT and RL. Compared to the strongest SFT baseline, CHAIN-OF-FOCUS-SFT, our model achieves higher performance across 6 out of 7 test sets, with an average gain of 4.59 points. Against the best-performing baseline, MINI-O3-RL, LOGT-SFT delivers a 1.9-point improvement.

Moreover, even when the number of training samples is reduced from 7.1k to 5.4k, LOGT-SFT still outperforms all baselines. In comparison, DEEPEYES-RL uses 8.7× more data, and MINI-O3-RL uses 3.1× more. Training Qwen2.5-VL-7B on 5.4k SFT samples takes only 3.2 hours on 8×A100 GPUs, making it >120× more compute-efficient than the best RL baseline, MINI-O3-RL, in terms of GPU hours. These results underscore the effectiveness of our data engine in producing high-quality SFT training data.

LOGT-SFT significantly outperforms SFT baselines under equal data budgets. For fair comparison, we evaluate LOGT-SFT under matched or lower data budgets. Specifically, we randomly sample 4k and 5.4k subsets from our synthetic dataset, matching the sizes used by PIXEL REASONER and CHAIN-OF-FOCUS, and then fine-tune Qwen2.5-VL-7B on these subsets. Under equal data constraints, Table 1 shows that LOGT-SFT outperforms PIXEL REASONER-SFT by 9.37 points and CHAIN-OF-FOCUS-SFT by 4.1 points. With a slightly smaller dataset with 7.1k samples, LOGT-SFT surpasses MINI-O3-SFT by a large margin of 19.02 points. These results demonstrate the superior quality of our synthetic data generated by our framework.

4.4 Main RL Results

Applying off-the-shelf RL on top of LOGT-SFT improves performance and generalization, outperforming all baselines by a wide margin. In Table 1, we show that applying off-the-shelf RL to the LOGT-SFT model further leads to a notable improvement of 7.69 points. Remarkably, LOGT-RL surpasses all RL baselines by at least 9.59 points, even though those baselines use more sophisticated and tool-specific RL designs. Moreover, it enhances the model's generalization ability, achieving performance comparable to the current SOTA on the out-of-domain benchmark MME-RealWorld-Lite. These results highlight that the benefits of high-quality SFT data generated by our method are further amplified when combined with RL.

High-quality SFT data from our framework significantly improves the RL performance ceiling. To evaluate the impact of our SFT data engine on the RL performance ceiling, we apply the same

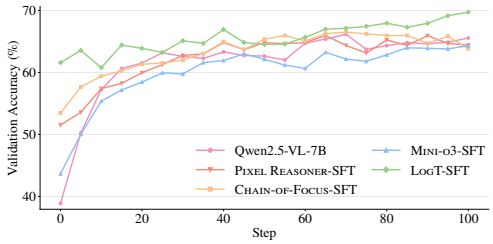


Figure 4: **Training dynamics after applying our simple RL procedure to the baselines and LogT-SFT.** During RL, our model's validation accuracy steadily improves, whereas other SFT baselines perform on par with, or worse than zero-RL. LogT greatly improves the RL performance ceiling.

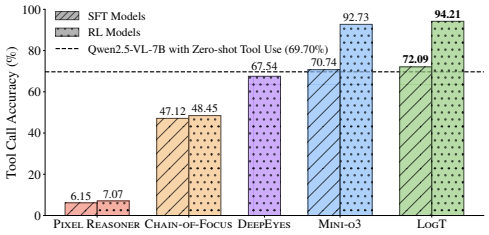


Figure 5: **Tool call accuracy.** We report tool call accuracy on V^* for the base model and all post-trained models. The base model is prompted to use Zoom-in in a zero-shot setting. Our LogT-SFT and LogT-RL models achieve the highest Zoom-in accuracy.

RL recipe from Section 4.2 to Qwen2.5-VL-7B and other existing SFT baselines [8, 9, 11]. The validation accuracy dynamics in Figure 4 show that initializing RL with LogT-SFT leads to much higher performance than applying RL directly to the base model (zero-RL). In contrast, RL training on prior SFT baselines quickly plateaus after initial gains and often performs no better, or even worse, than zero-RL after 100 steps. These results indicate that SFT data produced by LogT greatly improve the initial policy and substantially raise the RL performance ceiling.

4.5 Analysis on Tool Use and Reasoning Quality

Beyond numerical gains, we show that our data engine enables VLMs to think logically with images and use Zoom-in accurately and efficiently. While existing works open-source their training datasets, the SFT data curation pipelines and associated metadata remain unavailable, making apples-to-apples comparisons difficult. Hence, we focus our analysis on the behavior of the post-trained models. We further show that our model can generalize to novel tools in a zero-shot manner in Section B.6.

Analysis on tool call accuracy. We evaluate the tool accuracy of a model response by assessing how well it localizes the intended region(s) of interest via Zoom-in. We first compute the Intersection over Ground-truth (IoG) [40] for each Zoom-in call, defined as $\text{IoG} = |B \cap G|/|G|$, where B is the bounding box used in the Zoom-in call and G is the corresponding ground-truth box. Next, we consider a response as calling the tool accurately if *at least one* Zoom-in invocation achieves $\text{IoG} \geq 0.8$, as multiple calls may be issued while answering a question.

Finally, we define a model's overall tool call accuracy as the percentage of tool-requiring responses that include at least one accurate Zoom-in call. We compute this metric using V^*, the only benchmark that provides ground-truth bounding boxes G for target regions.

Figure 5 shows that LogT-SFT achieves 72.09% tool call accuracy, outperforming both the base model and other SFT baselines. LogT-RL further achieves the highest accuracy of 94.21%. These results show that the SFT data synthesized by LogT effectively teaches VLMs to reason and invoke tools accurately. Interestingly, most SFT- and RL-trained baselines underperform the base model that invokes Zoom-in in a zero-shot manner, except for Mini-o3. Comparing SFT models and their RL-tuned counterparts, we find that while RL improves tool call accuracy, the quality of SFT is critical for achieving strong tool call accuracy after RL.

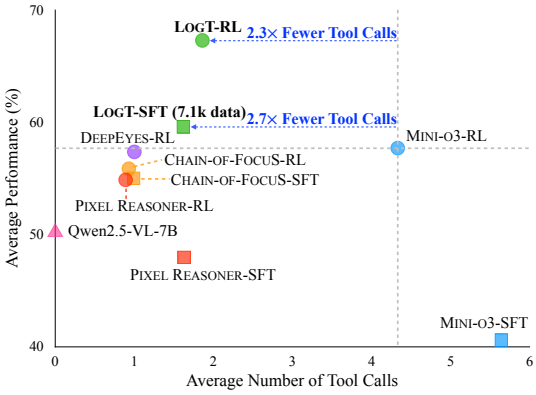


Figure 6: **Tool call efficiency.** We report the average number of tool calls per question across the seven test sets. Despite achieving better performance compared to prior SOTA, LogT-SFT model uses 2.7× fewer tool calls, and LogT-RL uses 2.3× fewer.

Analysis on tool call efficiency. We evaluate tool call efficiency by measuring the average number of ZOOM-IN calls used to solve a question across all test sets. As shown in Figure 6, the best-performing baseline, MINI-O3-RL, uses an average of 4.3 ZOOM-IN calls per task. In comparison, while delivering higher performance, LOGT-SFT and LOGT-RL achieve significantly better efficiency, requiring 2.7× and 2.3× fewer calls, respectively. Although other baselines use fewer ZOOM-IN calls, their performance is considerably lower. These results suggest that the SFT data synthesized by LOGT enables VLMs to reason about the necessary steps for solving questions, striking a superior balance between tool call efficiency and overall performance.

Human evaluation on reasoning quality and logical coherence. To assess whether the SFT data synthesized by LOGT enables VLMs to reason logically with images beyond achieving strong benchmark performance, we conduct a human evaluation comparing LOGT-RL with the prior SOTA (MINI-O3-RL) based on overall benchmark scores and tool-use accuracy. We randomly sample 100 questions from V^* and ask three human annotators to compare the responses produced by both models. For each question, annotators select the preferred response based on three criteria: (i) **Faithfulness:** Does the model answer the question correctly without hallucinating non-existent visual content? (ii) **Logical coherence:** Does the reasoning follow a consistent and logical flow? (iii) **Tool efficiency:** Does the model arrive at the answer using ZOOM-IN efficiently? On average, LOGT-RL is preferred in 59% of the cases, with a tie in 14%, indicating the effectiveness of our data generation approach. Detailed instructions and per-annotator breakdowns are provided in Section B.5. Besides human study, we also present a qualitative case study in Section B.7.

4.6 Ablation Study

Ablation on question difficulty. To investigate the impact of question difficulty on SFT training, we split the 7.1k-sample dataset into an easier half and a harder half based on Qwen2.5-VL-7B's average@8. As shown in Table 4 (Appendix), the VLM trained on the harder split achieves better performance on V^* (+2.09) and VisualProbe-Hard (+4.72), while showing only a slight drop on the easy and medium subsets of VisualProbe. These highlight the importance of synthesizing more challenging questions, as described in Section 3.1, to help the model acquire the ability to tackle more complex tasks.

Ablation on logic chain-driven reasoning trace synthesis. To examine the impact of explicitly generating logic chains for each question to guide reasoning trace synthesis, we compare our method to an alternative that uses only fixed in-context examples based on multi-hop and exhaustive search strategies. As shown in Table 5 in the Appendix, incorporating logic chains improves overall performance after SFT and yields a substantial gain of 2.87 points after RL. These results suggest that logic chain-driven synthesis is essential for creating higher-quality SFT data that effectively incentivizes logical reasoning and tool-use capabilities in VLMs.

Ablation on models used for data synthesis. To verify that our data engine's effectiveness stems from the framework design rather than specific large models, we replace all models used in Section 3 with much smaller variants: Qwen3-VL-8B [41], Qwen2.5-VL-32B [15], and GPT-OSS-20B [34], respectively. We then re-synthesize 4k SFT samples, train the base model with SFT, and evaluate it on V^*, VisualProbe, and HR-Bench. As shown in Table 6 in the Appendix, performance remains nearly unchanged compared with using the original models for synthesizing 4k SFT data.

Using these smaller models, generating 7.1k SFT samples requires about 120 GPU hours (30 hours on 4×A100), making it over 25× compute cheaper than the RL training of MINI-O3.

5 Discussion

5.1 Connection to Cognitive Science

The reasoning strategies in Section 3.2 have close parallels in findings from cognitive science: (i) **Direct answering.** Feature-integration theory [42] proposes that humans can detect simple visual features in parallel with no attention limits, enabling an immediate impression of recognizable objects. This is analogous to a VLM directly answering a visual question without invoking tools to inspect finer details. (ii) **Multi-hop.** Research on natural vision [43] suggests that visual information is acquired actively, in a task-dependent and coarse-to-fine manner: observers first attend to a promising contextual region and then inspect task-relevant details. This process closely parallels the multi-hop

reasoning strategy used in our work. (iii) **Exhaustive search.** Guided-search models [44] propose that visual search is efficient when target features, scene context, or prior knowledge prioritize a small subset of locations. When these signals are weak, attention must cover more candidates, resulting in a more systematic search. This corresponds to the exhaustive-search strategy used in our work.

5.2 Relationship to Knowledge Distillation

The model ablation (Table 6) suggests that LOGT does not rely on the original large synthesizing models. Moreover, on V^*, the 7B model trained on 4k reasoning traces synthesized using Qwen2.5-VL-32B outperforms Qwen2.5-VL-32B itself by 4.18 points. Together, these results suggest that synthesizing-model strength alone does not explain the effectiveness of our data engine, although knowledge distillation [45, 46] may still contribute to the gains.

5.3 Limitations

The centrality of ZOOM-IN. In this work, we focus primarily on ZOOM-IN for two primary reasons. (i) **Practical importance.** The zoom-in tool is widely used in visual reasoning tasks. For example, OpenAI's o3 model prominently features this tool in its blog [13]. Moreover, all thinking-with-image baselines in Table 1 rely on zoom-in as their sole image interaction mechanism. (ii) **Technical challenge.** Despite its prevalence, enabling VLMs to accurately and efficiently decide when and how to invoke the zoom-in tool remains challenging (see analysis in Section 4.5). OpenAI's blog [13] similarly identifies "redundant or unnecessary tool calls and image manipulation steps" as a key limitation of o3.

While centered on ZOOM-IN, our approach demonstrates significant transferability. LOGT improves performance on out-of-domain tasks (*e.g.*, MME-RW-Lite in Table 1) and generalizes to unseen tools (Table 8 in Section B.6).

More broadly, LOGT decouples logic chain synthesis from reasoning trace generation, making it a general and extensible framework. This design allows it to be applied to a wide range of structured reasoning tasks, and extending LOGT to broader domains is a promising direction for future work.

Reliance on VLM-as-a-Judge. To ensure scalability and avoid costly human annotation, we build a fully automated data engine that uses VLM-as-a-Judge. Consequently, the synthesized SFT data may contain a small amount of noise, as VLM judges can make mistakes. We view this trade-off as acceptable for two primary reasons. (i) **Annotation noise is unavoidable.** Errors are not unique to automated judging; they are often inevitable even in expensive human annotations [47, 48]. (ii) **Empirical results and manual verification suggest limited noise.** Our method achieves the strongest empirical performance (Table 1) and the highest tool-call accuracy (Figure 5), suggesting that any remaining noise is limited and much lower than in prior work. We further randomly sample 100 examples and manually evaluate errors at each stage of the data engine. As shown in Table 7, the error rate at each stage is low despite the fully automated pipeline.

6 Conclusion

We propose LOGT, a novel data engine for synthesizing high-quality SFT data that empowers VLMs to logically think with images and use ZOOM-IN both effectively and efficiently. The core is to generate challenging QAs and high-quality tool-use reasoning traces, guided by logic chains that emulate human-like visual reasoning strategies. VLMs fine-tuned on our synthetic data outperform existing baselines trained with both SFT and RL. Moreover, applying off-the-shelf RL on top of LOGT-SFT, without any tool-specific design, further substantially improves performance.

Acknowledgment

We would like to thank Hugo Laurençon, Jianwei Yang, Kaiwen Zha, and Xingyi Zhou for their helpful discussions during this project.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katherine Millican, Malcolm Reynolds, et al. Flamingo: a visual language model for few-shot learning. In *Advances in Neural Information Processing Systems*, pages 23716–23736, 2022.

[2] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Advances in Neural Information Processing Systems*, pages 34892–34916, 2023.

[3] Penghao Wu and Saining Xie. V*: Guided visual search as a core mechanism in multimodal llms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13084–13094, 2024.

[4] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models. In *Advances in Neural Information Processing Systems*, pages 24824–24837, 2022.

[5] Ruohong Zhang, Bowen Zhang, Yanghao Li, Haotian Zhang, Zhiqing Sun, Zhe Gan, Yinfei Yang, Ruoming Pang, and Yiming Yang. Improve vision language model chain-of-thought reasoning. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1631–1662, 2025.

[6] Yushi Hu, Weijia Shi, Xingyu Fu, Dan Roth, Mari Ostendorf, Luke Zettlemoyer, Noah A Smith, and Ranjay Krishna. Visual sketchpad: Sketching as a visual chain of thought for multimodal language models. In *Advances in Neural Information Processing Systems*, pages 139348–139379, 2024.

[7] OpenAI. Openai o3, 2025. URL https://openai.com/index/introducing-o3-and-o4-mini/.

[8] Alex Su, Haozhe Wang, Weiming Ren, Fangzhen Lin, and Wenhu Chen. Pixel reasoner: Incentivizing pixel space reasoning via curiosity-driven reinforcement learning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[9] Xintong Zhang, Zhi Gao, Bofei Zhang, Pengxiang Li, Xiaowen Zhang, Yang Liu, Tao Yuan, Yuwei Wu, Yunde Jia, Song-Chun Zhu, and Qing Li. Adaptive chain-of-focus reasoning via dynamic visual search and zooming for efficient vlms. *arXiv preprint arXiv:2505.15436*, 2025.

[10] Ziwei Zheng, Michael Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, Chao Shen, and Xing Yu. Deepeyes: Incentivizing "thinking with images" via reinforcement learning. *arXiv preprint arXiv:2505.14362*, 2025.

[11] Xin Lai, Junyi Li, Wei Li, Tao Liu, Tianjian Li, and Hengshuang Zhao. Mini-o3: Scaling up reasoning patterns and interaction turns for visual search. *arXiv preprint arXiv:2509.07969*, 2025.

[12] Xuchen Li, Xuzhao Li, Jiahui Gao, Renjie Pi, Shiyu Hu, and Wentao Zhang. Look less, reason more: Rollout-guided adaptive pixel-space reasoning. *arXiv preprint arXiv:2510.01681*, 2025.

[13] OpenAI. Thinking with images, 2025. URL https://openai.com/index/thinking-with-images/.

[14] Renhao Li, Jianhong Tu, Yang Su, Yantao Liu, Fei Huang, Hamid Alinejad-Rokny, Derek F. Wong, Junyang Lin, and Min Yang. Toolrm: Towards agentic tool-use reward modeling. *arXiv preprint arXiv:2510.26167*, 2026.

[15] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025.

[16] Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. Internvl3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025.

[17] Shitian Zhao, Haoquan Zhang, Shaoheng Lin, Ming Li, Qilong Wu, Kaipeng Zhang, and Chen Wei. Pyvision: Agentic vision with dynamic tooling. *arXiv preprint arXiv:2507.07998*, 2025.

[18] 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.

[19] Zeyi Huang, Yuyang Ji, Anirudh Sundara Rajan, Zefan Cai, Wen Xiao, Haohan Wang, Junjie Hu, and Yong Jae Lee. Visualtoolagent (vista): A reinforcement learning framework for visual tool selection. *arXiv preprint arXiv: 2505.20289*, 2025.

[20] Jiaqi Huang, Zunnan Xu, Jun Zhou, Ting Liu, Yicheng Xiao, Mingwen Ou, Bowen Ji, Xiu Li, and Kehong Yuan. Sam-r1: Leveraging sam for reward feedback in multimodal segmentation via reinforcement learning. *arXiv preprint arXiv: 2505.22596*, 2025.

[21] Mingyuan Wu, Jingcheng Yang, Jize Jiang, Meitang Li, Kaizhuo Yan, Hanchao Yu, Minjia Zhang, Chengxiang Zhai, and Klara Nahrstedt. Vtool-r1: Vlms learn to think with images via reinforcement learning on multimodal tool use. *arXiv preprint arXiv: 2505.19255*, 2025.

[22] Zhaochen Su, Linjie Li, Mingyang Song, Yunzhuo Hao, Zhengyuan Yang, Jun Zhang, Guanjie Chen, Jiawei Gu, Juntao Li, Xiaoye Qu, et al. Openthinkimg: Learning to think with images via visual tool reinforcement learning. *arXiv preprint arXiv:2505.08617*, 2025.

[23] Yi-Fan Zhang, Xingyu Lu, Shukang Yin, Chaoyou Fu, Wei Chen, Xiao Hu, Bin Wen, Kaiyu Jiang, Changyi Liu, Tianke Zhang, et al. Thyme: Think beyond images. *arXiv preprint arXiv:2508.11630*, 2025.

[24] Zujing Liu, Junwen Pan, Qi She, Yuan Gao, and Guisong Xia. On the faithfulness of visual thinking: Measurement and enhancement. *arXiv preprint arXiv:2510.23482*, 2025.

[25] Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *Advances in Neural Information Processing Systems*, pages 2507–2521, 2022.

[26] Yao Mu, Qinglong Zhang, Mengkang Hu, Wenhai Wang, Mingyu Ding, Jun Jin, Bin Wang, Jifeng Dai, Yu Qiao, and Ping Luo. Embodiedgpt: vision-language pre-training via embodied chain of thought. In *Advances in Neural Information Processing Systems*, pages 25081–25094, 2023.

[27] Guowei Xu, Peng Jin, Ziang Wu, Hao Li, Yibing Song, Lichao Sun, and Li Yuan. Llava-cot: Let vision language models reason step-by-step. *arXiv preprint arXiv:2411.10440*, 2025.

[28] Jiawei Guo, Tianyu Zheng, Yizhi Li, Yuelin Bai, Bo Li, Yubo Wang, King Zhu, Graham Neubig, Wenhu Chen, and Xiang Yue. MAmmoTH-VL: Eliciting multimodal reasoning with instruction tuning at scale. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13869–13920, 2025.

[29] Linzhuang Sun, Hao Liang, Jingxuan Wei, Bihui Yu, Tianpeng Li, Fan Yang, Zenan Zhou, and Wentao Zhang. MM-verify: Enhancing multimodal reasoning with chain-of-thought verification. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14100–14115, 2025.

[30] Hao Shao, Shengju Qian, Han Xiao, Guanglu Song, Zhuofan Zong, Letian Wang, Yu Liu, and Hongsheng Li. Visual cot: Advancing multi-modal language models with a comprehensive dataset and benchmark for chain-of-thought reasoning. In *Advances in Neural Information Processing Systems Datasets and Benchmarks Track*, pages 8612–8642, 2024.

[31] Ang Li, Charles Wang, Deqing Fu, Kaiyu Yue, Zikui Cai, Wang Bill Zhu, Ollie Liu, Peng Guo, Willie Neiswanger, Furong Huang, Tom Goldstein, and Micah Goldblum. Zebra-cot: A dataset for interleaved vision language reasoning. *arXiv preprint arXiv:2507.16746*, 2025.

[32] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4015–4026, 2023.

[33] Lin Shi, Chiyu Ma, Wenhua Liang, Xingjian Diao, Weicheng Ma, and Soroush Vosoughi. Judging the judges: A systematic study of position bias in llm-as-a-judge. *arXiv preprint arXiv:2406.07791*, 2024.

[34] OpenAI. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

[35] Wenbin Wang, Liang Ding, Minyan Zeng, Xiabin Zhou, Li Shen, Yong Luo, Wei Yu, and Dacheng Tao. Divide, conquer and combine: A training-free framework for high-resolution image perception in multimodal large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 7907–7915, 2025.

[36] Yi-Fan Zhang, Huanyu Zhang, Haochen Tian, Chaoyou Fu, Shuangqing Zhang, Junfei Wu, Feng Li, Kun Wang, Qingsong Wen, Zhang Zhang, et al. Mme-realworld: Could your multimodal llm challenge high-resolution real-world scenarios that are difficult for humans? *arXiv preprint arXiv:2408.13257*, 2024.

[37] Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *arXiv preprint arXiv:2402.03300*, 2024.

[38] Zhaochen Yu, Ling Yang, Jiaru Zou, Shuicheng Yan, and Mengdi Wang. Demystifying reinforcement learning in agentic reasoning. *arXiv preprint arXiv:2510.11701*, 2025.

[39] Guangming Sheng, Chi Zhang, Zilingfeng Ye, Xibin Wu, Wang Zhang, Ru Zhang, Yanghua Peng, Haibin Lin, and Chuan Wu. Hybridflow: A flexible and efficient rlhf framework. *arXiv preprint arXiv: 2409.19256*, 2024.

[40] Xinlong Wang, Tete Xiao, Yuning Jiang, Shuai Shao, Jian Sun, and Chunhua Shen. Repulsion loss: Detecting pedestrians in a crowd. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7774–7783, 2018.

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

[42] Anne M. Treisman and Garry Gelade. A feature-integration theory of attention. *Cognitive Psychology*, 12 (1):97–136, 1980.

[43] Rajesh P.N. Rao, Gregory J. Zelinsky, Mary M. Hayhoe, and Dana H. Ballard. Eye movements in iconic visual search. *Vision Research*, 42(11):1447–1463, 2002.

[44] Jeremy M Wolfe and Todd S Horowitz. Five factors that guide attention in visual search. *Nature Human Behaviour*, 1(3):0058, 2017.

[45] Yanjun Fu, Faisal Hamman, and Sanghamitra Dutta. T-SHIRT: Token-selective hierarchical data selection for instruction tuning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[46] Faisal Hamman, Pasan Dissanayake, Yanjun Fu, and Sanghamitra Dutta. Few-shot knowledge distillation of LLMs with counterfactual explanations. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[47] Joshua Vendrow, Edward Vendrow, Sara Beery, and Aleksander Madry. Do large language model benchmarks test reliability? *arXiv preprint arXiv:2502.03461*, 2025.

[48] OpenAI. Why swe-bench verified no longer measures frontier coding capabilities, 2026. URL https://openai.com/index/why-we-no-longer-evaluate-swe-bench-verified/.

[49] Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. Llava-onevision: Easy visual task transfer. *arXiv preprint arXiv:2408.03326*, 2024.

[50] Geng Li, Jinglin Xu, Yunzhen Zhao, and Yuxin Peng. Dyfo: A training-free dynamic focus visual search for enhancing lmms in fine-grained visual understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9098–9108, 2025.

Appendix

Contents

- **Section A**: Additional Details on Our Data Engine: LOGT
  - **Section A.1**: System Prompts for QA Generator and Reasoning Trace Generator
  - **Section A.2**: An Example of Exhaustive Search Reasoning Trace
  - **Section A.3**: Tool Use Statistics in the LOGT SFT Dataset
  - **Section A.4**: Qualitative Case Study of Synthesized Tool-use Traces
  - **Section A.5**: Computational Cost Analysis
- **Section B**: Additional Experiment Setup and Results
  - **Section B.1**: Additional Evaluation Details
  - **Section B.2**: Additional RL Implementation Details
  - **Section B.3**: Performance Comparison Between LOGT and Additional Baselines (Including VISUAL SKETCHPAD, PYVISION, InternVL3.5-8B, LLaVA-OneVision-7B, SEAL, and DYFO)
  - **Section B.4**: SFT Results on an Additional Base Model: InternVL3.5-8B
  - **Section B.5**: Additional Human Evaluation Details
  - **Section B.6**: Zero-Shot Generalization to Unseen Tools
  - **Section B.7**: Qualitative Case Study of Model Responses Generated By Baselines and LOGT-RL

A Additional Details on Our Data Engine: LOGT

A.1 System Prompts for QA Generator and Reasoning Trace Generator

We detailed the QA synthesis in Section 3.1 and the reasoning trace synthesis in Section 3.2. Here, we show part of the system prompt for the QA generator in Figure 7 and part of the system prompt for the reasoning trace generator in Figure 8.

A.2 An Example of Exhaustive Search Reasoning Trace

We provide an additional example of an exhaustive-search logic chain and its corresponding tool-use reasoning trace in Figure 9. In this example, the target object for the given question, *i.e.*, the pink sign held by the person, is particularly difficult to locate. Therefore, the logic chain selector (see Section 3.2) chooses the exhaustive search logic chain. This logic chain instructs the reasoning trace generator to first zoom in on the crowd and then systematically search the scene from left to right. Although the logic chain specifies examining all four regions of the crowd (left, center-left, center-right, and right) because the logic chain generator does not know exactly where the target object is located, the reasoning trace generator successfully identifies the target object after zooming into the second region (center-left crowd). Consequently, it terminates the search early. This is because our prompt requires the reasoning trace generator to stop searching once the target object is found. After identifying the target, it calls ZOOM-IN on the object and then proceeds to generate the final answer.

A.3 Tool Use Statistics in the LOGT SFT Dataset

We present the normalized distribution of ZOOM-IN call counts in Figure 10. The number of tool calls ranges from 0 to 6. Specifically, 94% of the questions in the LOGT SFT dataset are solved using 3 or fewer ZOOM-IN calls.

A.4 Qualitative Case Study of Synthesized Tool-use Traces

We present a qualitative comparison of reasoning trace synthesis strategies used by PIXEL REASONER [8], CHAIN-OF-FOCUS [9], MINI-O3 [11], and LOGT, as shown in Figure 12.

System Prompt for QA Generator

You are a Vision–Language assistant tasked with generating one high–quality question and corresponding answer for each input image. The purpose is to synthesize QA datasets based on visual content from specific regions in the image.

Inputs (per task)
1. The input image, with key regions marked by red boxes and larger surrounding areas marked by blue boxes.
2. Annotations, including:
   – Input image size.
   – One or more bounding boxes (indicating key regions).
   – Larger cropped images that **include** the red bounding boxes.
   – Cropped images that **correspond to** the red bounding boxes.

Your task
1. Analyze the full image, the marked regions, and the cropped images.
2. Determine whether the input image and the provided region(s) provide enough information to generate a high–quality question.
3. If yes, generate one question and its corresponding brief answer.
4. If not, return {"valid": false}.

Rules
1. **Location**
The question must be about the object(s) **within** the provided **red** bounding box(es).
   – **Be very careful**: the target object(s) tied to the answer must lie **within the red bounding box**, not merely nearby and not just within the larger blue box(es).
2. **Type**
Each question must fit one of the following types:
   – "perception_single": attribute of a single object (e.g., color, category, material).
   – "perception_multi": relationship between multiple objects (e.g., larger/smaller, left/right).
   – "counting": count of objects in the **entire image**, but only if those objects appear in at least one bounding box.
   – "ocr": read visible text in the region(s) (e.g., words, numbers, labels).
   – "reasoning": interpret visuals that require math or chart understanding.
(The remaining 5 longer rules are omitted for brevity.)

Example questions by type
(Example questions are omitted for brevity.)

Figure 7: **System prompt for the QA generator.** The remaining rules and in-context examples are omitted for brevity.

PIXEL REASONER synthesizes multi-turn reasoning traces by deliberately inserting ZOOM-IN calls that focus on overly broad or incorrect regions. In Figure 12a, the first two ZOOM-IN calls target completely irrelevant areas. In contrast, LOGT generates a logic chain that guides the reasoning trace generator to first invoke ZOOM-IN on the building, then refine the focus onto the target text.

In Figure 12b, the CHAIN-OF-FOCUS SFT trace redundantly zooms in on the shoes for six consecutive steps, despite correctly identifying their location in the first step. In contrast, LOGT first generates a logic chain that instructs the reasoning trace generator to initially zoom in on the person, then gradually focus on the shoes, and finally answer the question using only two ZOOM-IN calls.

A similar issue occurs with the MINI-O3 SFT trace in Figure 12c: although it initially identifies the correct target, it repeatedly invokes ZOOM-IN and ultimately selects an unrelated region, resulting in a total of four ZOOM-IN calls. By comparison, LOGT guides the reasoning trace generator to directly locate the wheels, with the reasoning trace requiring only a single ZOOM-IN call before answering.

A.5 Computational Cost Analysis

Synthesizing 7.1k samples with LOGT requires approximately 960 GPU hours (5 days on 8 A100 GPUs), making it more than 3× compute cheaper than RL training for the Mini-o3 baseline.

As shown in Section 4.6, synthesis resources used can be substantially reduced by using smaller model variants without compromising performance. With smaller models, the total resource use drops to approximately 120 GPU hours, making it more than 25× compute cheaper than RL training for Mini-o3.

System Prompt for Reasoning Trace Generator

You are a **visual question answering assistant** equipped with a tool called "zoom_in", which crops a specified region from an image.

Input (per task)
1. The input image (image_id: 0).
2. Annotations, including:
   – Ground–truth region(s) shown as red boxes in the input image.
   – Input image size.
   – Cropped image(s) corresponding to the ground–truth region(s).
3. Question.
4. Logic chain describing the sequential regions or relations to inspect.

Your task
1. Your task is to produce a **complete, step–by–step reasoning trace** that visually derives the answer to a question using only what is visible in the provided image.
2. Your reasoning trace must **follow the given logic chain**, though you must never mention the chain itself in your reasoning text.

Chain structure
1. Each chain is a **numbered sequence of steps**.
2. Each step follows the format <short noun phrase> (k):
   – The **short noun phrase** names the region, object, or relation to inspect.
   – The optional **(k)** is a numeric zoom tag indicating which image_id to zoom into.

Rules
You must strictly follow all rules.
1. Simulate visual reasoning from scratch and act as if you only have access to:
   – the **original image** (without ground–truth region(s)),
   – and the **question**.
Do **not** mention annotations, ground–truth region(s), or the chain explicitly. Simulate a realistic, step–by–step visual discovery process.
2.1. Follow the logic chain exactly (multi–hop reasoning chain only).
   – Strictly follow every step in the chain **in order**, no skipping, merging, or reordering.
2.2. Be exhaustive but structured. Search systematically, not randomly (exhaustive–search chain only).
   – Four broad patches max when scanning a large area; examine them in a deterministic order.
   – Fully examine each candidate region before moving on.
   – Stop the search only when you confirm the target visually.
(The remaining 8 longer rules are omitted for brevity.)

Examples
(Example reasoning traces are omitted for brevity.)

Figure 8: **System prompt for the reasoning trace generator.** Each logic-chain entry of the form $\langle$short noun phrase$\rangle(k)$ specifies a target object or region and its source image I_k. Here, I_0 is the original image, and I_k for $k \geq 1$ is the crop returned by the k-th ZOOM-IN call. The reasoning trace generator locates the specified target in I_k, predicts its bounding box, and invokes ZOOM-IN on that image using the predicted box. For example, "Backpack (2)" corresponds to "Backpack (I_2)" in Figure 3 and instructs the generator to zoom in on the backpack within the person crop returned by the second tool call. The remaining rules and in-context examples are omitted for brevity.

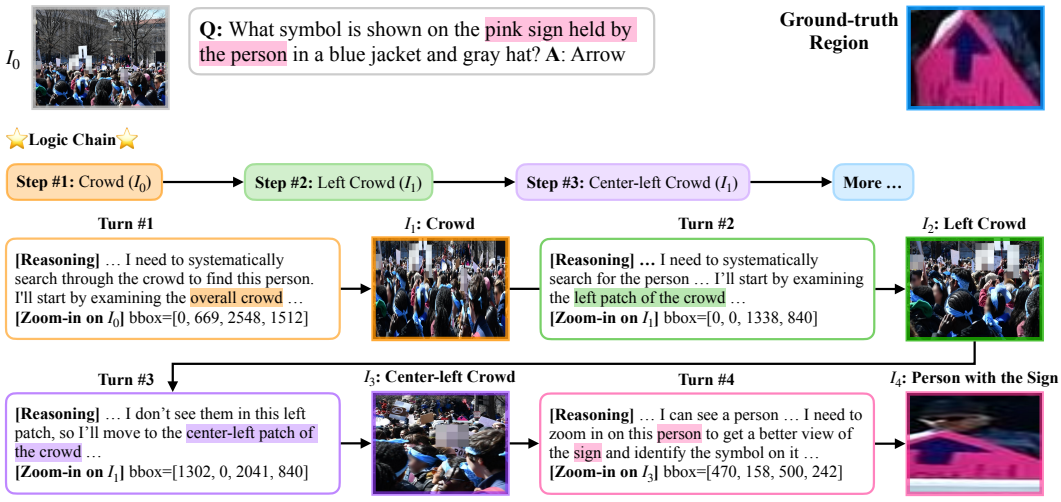


Figure 9: **An example of an exhaustive search logic chain and its corresponding tool-use reasoning trace.** To locate the pink sign held by a person in the crowd, the logic chain begins by instructing the trace generator (see Section 3.2) to identify the crowd, then systematically search it from left to right. Since the trace generator successfully detects the target sign and the person in the crop returned by the third ZOOM-IN call, it stops scanning the rest of the crowd and instead zooms in on the identified region to extract the final answer.

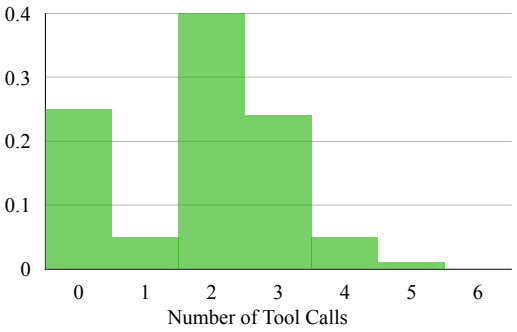


Figure 10: **Normalized distribution of ZOOM-IN call counts in the SFT dataset generated by LOGT.** The number of tool calls per trace in the dataset ranges from 0 (direct answering without tool use) to 6.

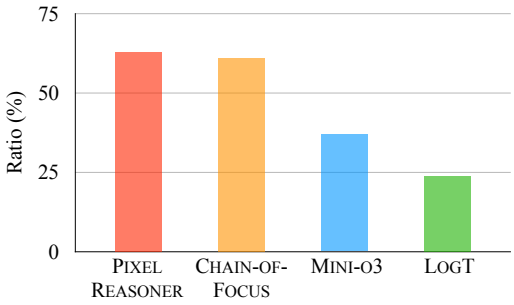
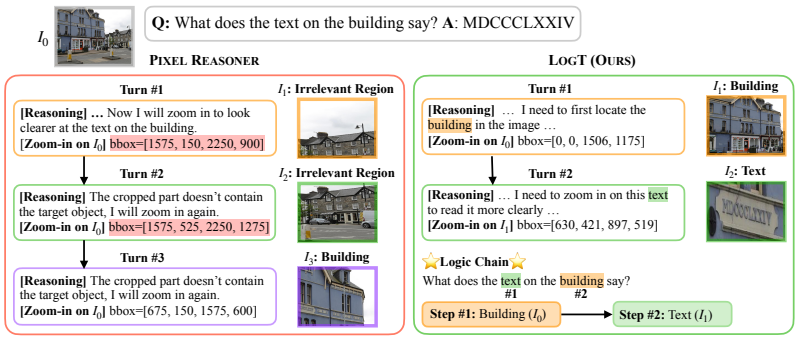
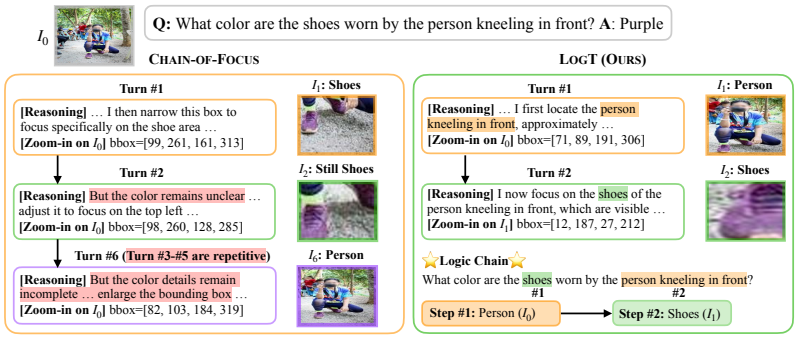


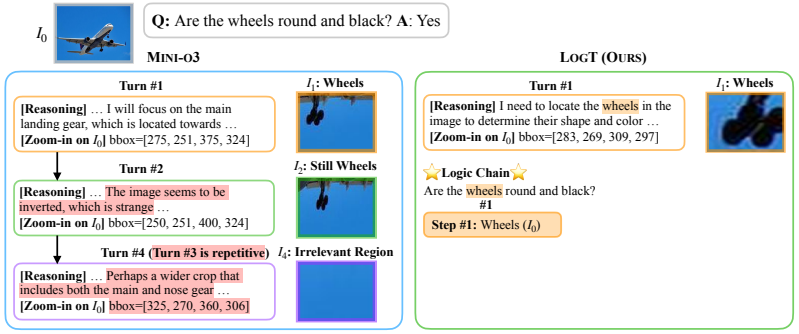
Figure 11: **Ratio of easy questions.** We compute the ratio of questions in existing and our SFT datasets for which Qwen2.5-VL-7B achieves an average@8 $=$ 1 without needing tools (easy questions). Our dataset synthesized by LOGT contains only 24%, significantly lower than existing datasets.



(a) **PIXEL REASONER vs. LOGT.** The example trace in PIXEL REASONER SFT data randomly introduces errors when synthesizing multi-turn tool calls, and the erroneous bounding boxes are highlighted in red. In contrast, ours follows a coherent logic chain: first locating the building, and then locating the text.



(b) **CHAIN-OF-FOCUS vs. LOGT.** The example trace in CHAIN-OF-FOCUS SFT data repeatedly zooms in on the shoes over six turns, despite correctly identifying the target region in the first turn. This redundant reasoning, highlighted in red, reflects tool call inefficiency. In contrast, LOGT demonstrates a coherent reasoning chain: first locating the person, then progressively narrowing down to the shoes.



(c) **MINI-O3 vs. LOGT.** The example trace in MINI-O3 SFT data correctly finds the wheels in the first turn but mistakenly claims that the crop returned by ZOOM-IN is inverted, leading to two redundant ZOOM-IN calls. Ultimately, it zooms in on an entirely irrelevant region. The redundant reasoning and the incorrect ZOOM-IN call are highlighted in red. In contrast, LOGT makes a single ZOOM-IN call to locate the wheels and then directly reaches the correct answer.

Figure 12: **Qualitative case study of SFT data synthesized by PIXEL REASONER, CHAIN-OF-FOCUS, MINI-O3 and LOGT.**

Table 2: **Performance comparison between our models and additional baselines.** We extend the main results in Table 1 by including additional open-source models (InternVL3.5-8B, LLaVA-OneVision-7B), modular-agent frameworks (VISUAL SKETCHPAD, PYVISION), and visual-search-based methods (SEAL, DYFO). The green rows refer to our models. Both LOGT-SFT and LOGT-RL significantly outperform these baselines across all seven test sets, further demonstrating the effectiveness of our approach.

	Data		V^*	VisualProbe			HR-Bench		MME-RW-Lite	Avg.
	SFT	RL		Easy	Medium	Hard	4K	8K		
VISUAL SKETCHPAD	-	-	80.30	-	-	-	-	-	-	-
PYVISION	-	-	75.90	-	-	-	-	-	-	-
Open-sourced Models w/o Tools										
InternVL3.5-8B	-	-	60.73	29.79	10.07	13.21	57.38	49.00	41.12	37.33
LLaVA-OneVision-7B	-	-	72.25	41.84	16.42	14.15	64.62	58.00	44.71	44.57
Open-sourced Models with Visual Search Tools										
SEAL	-	-	75.39	-	-	-	-	-	-	-
DYFO	-	-	81.15	-	-	-	-	-	-	-
LOGT-SFT	7.1k	0	**87.43**	**60.99**	**41.04**	**39.62**	**72.00**	**70.12**	**45.96**	**59.59**
LOGT-RL	7.1k	6.4k	**92.15**	**70.92**	**52.24**	**51.89**	**76.12**	**74.62**	**53.00**	**67.28**

B Additional Experiment Setups and Results

B.1 Additional Evaluation Details

During evaluation, we adopt a maximum image resolution budget of $8192 \times 28 \times 28$ pixels for models in the Qwen2.5-VL-7B family, where each 28×28 patch represents a visual token. Images exceeding this budget are downsampled. For comparison, MINI-O3 uses a smaller default budget of approximately $2551 \times 28 \times 28$ pixels. We observe that MINI-O3 consistently performs better under our higher budget. Therefore, we adopt our budget for all Qwen2.5-VL-7B-based evaluations.

B.2 Additional RL Implementation Details

We provide additional details on our RL implementation that are not covered in Section 4.2. During RL, to reduce GPU idle time and improve training efficiency, we cap the number of ZOOM-IN calls at 6. Our RL training data comes from MINI-O3, and we filter out trivial cases by removing any question where Qwen2.5-VL-7B achieves an average@8 of 100%. To monitor training dynamics during RL, we use a validation set composed of samples from V^*, VisualProbe, and HR-Bench.

B.3 Performance Comparison Between LOGT and Additional Baselines

To further demonstrate the effectiveness of LOGT, we evaluate our approach against a broader set of baselines. Specifically, we include two more open-source models of comparable scale (7B–8B): InternVL3.5-8B [16] and LLaVA-OneVision-7B [49]. We also consider modular-agent frameworks that enable proprietary models to use external tools, such as VISUAL SKETCHPAD [6] and PYVISION [17]. In addition, we evaluate two visual-search-based methods that rely on external visual experts: SEAL [3] and DYFO [50]. The performance of these baselines, along with LOGT-SFT and LOGT-RL, is shown in Table 2. Results indicate that both LOGT-SFT and LOGT-RL significantly outperform the additional baselines, consistent with the findings in Table 1.

B.4 SFT Results on an Additional Base Model: InternVL3.5-8B

To evaluate whether our data engine generalizes to other base models, we fine-tune InternVL3.5-8B using 7.1k SFT samples generated by LOGT. As shown in Table 3, the fine-tuned model substantially outperforms the base model, achieving a 14-point overall improvement.

B.5 Additional Human Evaluation Details

We present the instructions given to human annotators in Figure 13. For each question, the evaluation interface randomly assigns MINI-O3-RL and LOGT-RL to either Model A or Model B. In addition to

Table 3: **Performance of fine-tuning InternVL3.5-8B with LOGT SFT data.** We fine-tune InternVL3.5-8B using 7.1k samples synthesized by LOGT. The green rows refer to our models. The resulting model outperforms the base model across all test sets, improving overall performance by 14.00 points.

	Data		V^*	VisualProbe			HR-Bench		MME-RW-Lite	Avg.
	SFT	RL		Easy	Medium	Hard	4K	8K		
InternVL3.5-8B	-	-	60.73	29.79	10.07	13.21	57.38	49.00	41.12	37.33
+LOGT-SFT	7.1k	0	**75.92**	**56.74**	**27.24**	**28.30**	**66.12**	**63.38**	**41.64**	**51.33**
Δ vs. Base Model	-	-	**+15.19**	**+26.95**	**+17.17**	**+15.09**	**+8.74**	**+14.38**	**+0.52**	**+14.00**

Instructions for Human Evaluation

Which model response do you prefer?
Evaluation Criteria:
1. Answer Correctness & No Hallucination: Does the model arrive at the correct answer? It is NOT preferred that the answer is obtained by hallucination.
2. Coherent Logic: Does the model's reasoning make sense and match how you would solve the problem?
3. Tool Call Efficiency: Does the model use the minimum necessary number of tool calls (fewer iterations) to answer the question?

[Question]
...
[Ground Truth]
...
[Model A's Response]
...
[Model B's Response]
...

Your Preference:
Please select one:
Model A; Tie; Model B

Figure 13: **Instructions for Human Evaluation.**

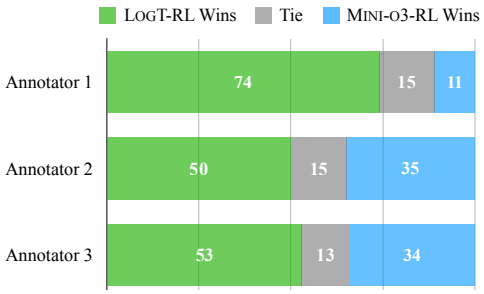


Figure 14: **Per-annotator breakdown of human evaluation.** All three annotators preferred responses generated by LOGT-RL over those from MINI-O3-RL.

Table 4: **Ablation on question difficulty.** Training the VLM on the harder half of our SFT dataset leads to significantly improved performance on challenging tasks like those in VisualProbe-Hard.

Split	V^*	VisualProbe		
		Easy	Medium	Hard
Easier	85.34	**59.57**	**35.07**	35.85
Harder	**87.43**	58.87	34.70	**40.57**

the overall human evaluation results reported in Section 4.5, the per-annotator breakdown in Figure 14 shows that all annotators prefer responses generated by LOGT-RL over those from the prior SOTA, MINI-O3-RL. This underscores that our data generation framework enables VLMs to reason more effectively using the ZOOM-IN tool.

Table 5: **Ablation on logic chain-driven reasoning trace synthesis.** Incorporating logical chains into the data generation process of LOGT improves overall SFT performance and yields substantial additional gains after subsequent RL.

Training Recipe	Logic Chain	V^*	VisualProbe			HR-Bench		MME-RW-Lite	Avg.
			Easy	Medium	Hard	4K	8K		
SFT	✗	87.43	59.57	38.06	39.62	**72.25**	69.50	44.09	58.65
	✓	87.43	**60.99**	**41.04**	39.62	72.00	**70.12**	**45.96**	**59.59**
SFT+RL	✗	90.05	66.67	46.27	46.23	74.75	**75.12**	51.80	64.41
	✓	**92.15**	**70.92**	**52.24**	**51.89**	**76.12**	74.62	**53.00**	**67.28**

Table 6: **Ablation on models used for data synthesis.** We replace the models used by LOGT in Section 3 with much smaller variants and re-synthesize 4k SFT samples. After SFT, the average performance across V^*, VisualProbe, and HR-Bench remains nearly unchanged, indicating that our method is robust to the choice of synthesis models.

Model Usage	Avg.
Original	59.55
Qwen3-VL-8B, Qwen2.5-VL-32B & GPT-OSS-20B	59.43

Table 7: **Error rates across different stages.** We randomly sampled 100 samples from each stage and manually evaluated their error rates.

Stage	**Error Rate (%)**
QA Generation & Filtering	12
Logic Chain Generation	2
Reasoning Trace Generation	7

Table 8: **Performance on V^*-Rotated.** We construct V^*-Rotated by rotating images in V^* by angles randomly sampled from 30° to 330°. LOGT-RL uses a rotation tool at inference time that was not seen during training, improving over the base model by 21.99 points.

Model	V^*-Rotated
Qwen2.5-VL-7B	46.07
LOGT-RL (with Rotation Tool)	**68.06**
Δ vs. Base Model	**+21.99**

B.6 Zero-Shot Generalization to Unseen Tools

To evaluate whether our method can generalize to tools unseen during training, we construct a new benchmark, V^*-Rotated, by rotating images in V^* by angles uniformly sampled from 30° to 330°. We evaluate the base model and LOGT-RL on this benchmark. At inference time, LOGT-RL is given access to a rotation tool that was not seen during training. As shown in Table 8, LOGT-RL learns to use this tool in a zero-shot setting despite receiving no rotation-related supervision, outperforming the base model by 21.99 points. These results suggest that LOGT can generalize to novel tasks and previously unseen tools.

B.7 Qualitative Case Study of Model Responses Generated By Baselines and LOGT-RL

To illustrate how SFT data synthesized by LOGT helps VLMs reason logically and use ZOOM-IN accurately, we present a qualitative case study comparing model responses to a question from the V^* benchmark. These comparisons are shown in Figure 15. We refer to Qwen2.5-VL-7B-RL as the model obtained by directly applying our RL training procedure from Section 4.2 to the base model Qwen2.5-VL-7B. For all other baselines, we use the officially released checkpoints.

- **Qwen2.5-VL-7B-RL** answers the question correctly but hallucinates non-existent white text on the tissue box.



Figure 15: **Qualitative case study comparing model responses to a V* question between baseline models and LOGT-RL.** We show the generated reasoning traces, including ZOOM-IN calls, for both baseline models and LOGT-RL. While Qwen2.5-VL-7B-RL, PIXEL REASONER-RL, and CHAIN-OF-FOCUS-RL produce the correct final answer, their responses contain hallucinations or self-contradictions. The remaining baselines make erroneous ZOOM-IN calls and ultimately produce incorrect answers. In contrast, LOGT-RL follows a coherent and consistent reasoning chain: it first identifies the front cabin of the truck, then locates the tissue box, and finally answers the question correctly without hallucinations.

- **PIXEL REASONER-RL** also produces the correct answer; however, its ZOOM-IN call targets an irrelevant region, and its final answer contradicts its own statement that "There is no tissue box."
- **DEEPEYES-RL** focuses on white sheets of paper and incorrectly concludes that the tissue box is white.
- **CHAIN-OF-FOCUS-RL** zooms in on an irrelevant region and hallucinates, though it ultimately answers correctly.
- **MINI-O3-RL** selects regions that include the tissue box, but still arrives at an incorrect answer.
- **LOGT-RL** first identifies the likely location of the tissue box (the truck's front cabin), then zooms in on the tissue box itself, ultimately producing the correct answer without any hallucinations.

Among all the models, LOGT-RL is the only one that reaches the correct answer and shows no hallucinations. This example further demonstrates that our data engine produces high-quality SFT data that effectively trains VLMs to reason about images in a logical and faithful manner.

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: Our abstract and introduction in Section 1 clearly reflect our paper's contributions and scope.

   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: We discuss the limitations of our work in Section 5.3.

   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: This is not a theoretical paper.

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: We present our method in Section 3, and detail the experimental setup in Section 4.1, Section 4.2, Section B.1 and Section B.2.

   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: We provide the code and artifacts required to reproduce our work at https://github.com/Dynamite321/LogT.

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: We provide details of the experimental settings in Section 4.1, Section 4.2, Section B.1 and Section B.2.

   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: We do not report error bars due to limited computational resources. As described in Section 4.1, we use greedy decoding to reduce the effect of randomness.

   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: Details of the computational resources used for data synthesis are provided in Section 4.6 and Section A.5.

   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: We adhere strictly to 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: [N/A]

    Justification: This work does not pose any direct societal impacts.

    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: This work does not pose any risk for misuse.

    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: In this paper, we credit them properly.

    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: We provide documentation for the newly introduced assets at https://github.com/Dynamite321/LogT.

    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: [Yes]

    Justification: We provide the full instructions given to participants in Section B.5.

    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: The paper does not involve research with human subjects. The human evaluation described in Section 4.5 and Section B.5 does not require IRB approval.

    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: [N/A]

Justification: The core method development in this paper does not involve LLMs as any important, original, or non-standard components.

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.