ByteDance | Seed

Revisiting the Necessity of Lengthy Chain-of-Thought in Vision-centric Reasoning Generalization

Yifan Du[1,2,*], **Kun Zhou**[1,*], **Yingqian Min**[1], **Yue Ling**[2], **Wayne Xin Zhao**[1,†], **Youbin Wu**[2,†]

[1]Renmin University of China, [2]ByteDance Seed

[*]Equal Contribution, [†]Corresponding authors

Abstract

We study how different Chain-of-Thought (CoT) designs affect the acquisition of the generalizable visual reasoning ability in vision-language models (VLMs). While CoT data, especially long or visual CoT such as "think with image", has been widely used to supervise intermediate reasoning, it remains unclear why specific CoT designs help and which ones truly support generalizable reasoning. To systematically evaluate this, we focus on a controlled maze-solving benchmark where reasoning rules are fully visual, difficulty can be tuned by grid size, and all the intermediate steps can be automatically generated. Using Qwen2.5-VL-7B under a standard SFT-then-RL pipeline, we compare three representative CoT formats: Language CoT, Grounding CoT (with spatial coordinate trajectories), and Visual CoT (with image manipulations). Our experiments reveal that visual and longer CoT mainly accelerate convergence but do not lift the final performance ceiling; concise CoT containing only essential grounding steps outperforms longer traces; and, strikingly, CoT retaining only the minimal grounding results generalizes best across different maze sizes. We further validate these insights on other vision-centric tasks. These findings highlight a "short is long" effect and provide practical guidance for constructing more generalizable SFT datasets for visual reasoning.

Date: December 1, 2025
Correspondence: Yifan Du at yifandu1999@gmail.com
Project Page: https://github.com/RUCAIBox/Revisiting-Visual-CoT

1 Introduction

Visual reasoning is emerging as a key capability for vision-language models (VLMs) [21, 23, 24, 43], enabling them to support broader real-world, vision-centric applications[1] rather than merely recognizing objects or generating captions [14, 35, 36]. To make such models truly reliable and versatile, it is essential to endow them with *generalizable reasoning capabilities*. Instead of overfitting to a specific benchmark, generalizable reasoning aims to acquire abstract reasoning skills and reusable patterns that transfer across tasks, domains, and prompts, *e.g.*, learning puzzle-solving rules on small mazes and simple sudoku games that can be transferred to larger and harder ones [38, 52].

[1]**Vision-centric reasoning** refers to tasks where the core reasoning process must occur *within the image*. The model must extract, track, and manipulate spatial or structural cues directly from the image.

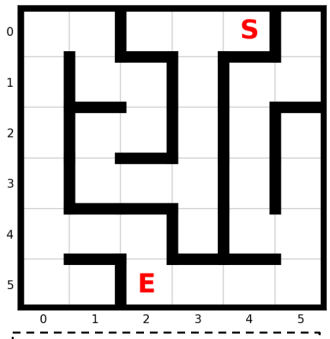
To learn such generalizable visual reasoning capabilities, recent work widely employs Chain-of-Thought (CoT) data for supervised fine-tuning, to guide both the training and inference of VLMs [7, 45]. CoT reasoning encourages the model to generate explicit intermediate steps instead of jumping directly to the answer, decomposing complex problems into a sequence of simpler sub-goals [49]. By increasing the length of these CoT traces, with more detailed explanations, multi-step deliberation, and self-reflection, the VLM can reason more thoroughly and refine its predictions, often yielding more accurate and higher-quality results [8, 42]. Moreover, supervised fine-tuning VLMs on long CoT–formatted data has been shown to further amplify these benefits, leading to significant performance improvements across diverse visual reasoning benchmarks [19, 46, 55]. Among these methods, a special form of CoT expressed directly in the visual space, exemplified by o3's "think with image" [28], has proven particularly effective: this *visual CoT* allows the model to manipulate the image itself (*e.g.*, cropping regions or drawing marks to highlight evidence), making its reasoning process more aligned with how humans naturally think with images.

Despite these promising results, it remains unclear why these strategies help and which forms of CoT actually contribute to generalizable visual reasoning. In particular, different CoT paradigms externalize intermediate reasoning in fundamentally different ways (linguistic, spatial, and visual). It is unknown whether and how specific CoT designs contribute to learning generalizable visual reasoning ability. To close this gap, we conduct a rigorous and controlled comparison of three representative CoT formats: (1) *Language CoT*, which follows the conventional LLM paradigm and expresses the entire reasoning process in natural language; (2) *Grounding CoT* [31, 41], where the model directly outputs a sequence of spatial coordinates so that the reasoning chain is implicitly encoded in the coordinate trajectory; and (3) *Visual CoT* [28, 58], which manipulates the image via predefined operations (*e.g.*, cropping or marking) and feeds the updated image back into the model to enable interleaved image–text reasoning. These formats reflect three different ways of externalizing intermediate reasoning (*e.g.*, linguistic, spatial, and visual), allowing us to isolate how each affects learning and generalization.

To systematically and fairly compare these CoT strategies, we require a controlled task environment free from data contamination. Therefore, we select the maze, a classic vision-centric visual reasoning task, as our evaluation setting. The maze offers a clean and interpretable testbed: its underlying reasoning rule is fully expressed by the visual input, while the difficulty can be smoothly controlled by adjusting the grid size (*e.g.*, from 4×4 to 6×6), and current VLMs still perform poorly on it (*e.g.*, Qwen2.5-VL-7B [1] achieves below 10% success on 4×4 mazes), allowing us to focus on studying their visual reasoning ability rather than saturating performance. In addition, maze data are easy to synthesize, and both final solutions and intermediate steps (paths, coordinates, or annotated images) can be automatically generated, making it convenient to generate and filter large-scale CoT data. Built on this task, we design experiments that systematically test the widely used SFT-then-RL generalization paradigm under different CoT formats, by constructing CoT-specific cold-start datasets, performing Supervised Fine-Tuning (SFT) on Qwen2.5-VL-7B [1] to obtain policy models for each format, and then further improving them with Reinforcement Learning (RL). After applying RL, our experiments reveal three key findings:

- **Visual and longer CoT accelerates but does not lift the ceiling.** Incorporating visual CoT and increasing CoT length can speed up convergence during training, yet the final performance plateau remains similar to (or only marginally better than) shorter-CoT baselines.
- **Grounded short CoT surpasses verbose ones.** Short CoT containing essential grounding information achieves higher and more stable performance than longer, step-by-step CoT, suggesting that excessive intermediate explanation may not be useful for generalization.
- **CoT with the least grounded results generalizes the best.** CoT formats that retain only the least amount of grounding results (*e.g.*, sparse coordinate paths or final trajectories) achieve the strongest generalization across maze sizes.

In summary, our results show the "short is long" effect, suggesting that concise but well-grounded supervision better encourages reusable reasoning patterns. We also validate the effectiveness of short CoT on other visual reasoning tasks (*e.g.*, visual search and visual puzzle). It offers a practical guideline for constructing more generalizable SFT datasets for visual reasoning.



Instruction:
Given an image of a n*n maze: ……
Your task is to compute a valid path from S to E without crossing any walls. Output: A list of coordinate tuples representing the path from S to E, in the format: [(row1, col1), (row2, col2), ...]. The path must start with the coordinates of S and end with the coordinates of E. The final path MUST BE put in \boxed{}.

Language CoT:
<think>From my starting position, I can only move West. After that, I have two intermediate options: West or South… The Westward path seems to head towards a deadend, which is away from my goal in the bottom. Therefore, ……</think>
\boxed{[(0,4), (0,3), (1,3), (2,3), (3,3), (3,2), (3,1), (2,1), …, (4,0), (4,1), (4,2), (5,2)]}

Grounding-CoT:
<think>From S at [452, 59], the only available move is West to [359, 59] …… </think>
\boxed{[([(0,4), (0,3), (1,3), (2,3), (3,3), (3,2), (3,1), (2,1), (2,2), (1,2), (1,1), (0,1), (0,0), (1,0), (2,0), (3,0), (4,0), (4,1), (4,2), (5,2)]}

Grounding-CoT-least:
\boxed{[([(0,4), (0,3), (1,3), (2,3), (3,3), (3,2), (3,1), (2,1), (2,2), (1,2), (1,1), (0,1), (0,0), (1,0), (2,0), (3,0), (4,0), (4,1), (4,2), (5,2)]}

Visual CoT:
<think>From S at [452, 59], the only available move is West to [359, 59] …… From [81, 523], the only path is East to [174, 523].
Let's to visualize this try: ①
My path covered a large portion of the maze, but terminated in a dead end at [174, 523]. I need to backtrack and reconsider a critical decision. ……I will re-trace my steps from S [452, 59] all the way to the junction at [81, 430].
Let's visualize this: ②
The successful path has been found and visualized. </think>
\boxed{[(0,4), (0,3), (1,3), (1,1), (0,1), (0,0), …(1,0), (2,0), (3,0), (4,0), (4,1), (4,2), (5,2)]}

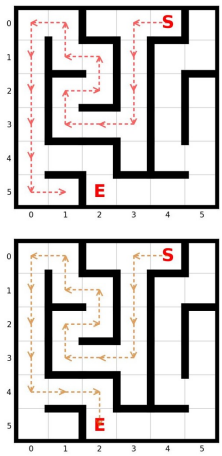


Figure 1 Illustration of four types of CoT reasoning strategies. We show examples of solving a 6×6 maze navigation task.

2 Preliminary

We first formalize the Chain-of-Thought (CoT) reasoning process of a vision-language model (VLM). Given a textual query Q and an image I as input, the VLM π_θ generates a textual answer A. Its reasoning is decomposed into a sequence of intermediate steps (or thoughts) $R_T = s_1, s_2, \ldots, s_T$, where each s_t denotes the reasoning state at time step t. The CoT strategy specifies how these intermediate reasoning states are represented. In this paper, we mainly study the following three types of CoT strategies in visual reasoning tasks.

2.1 Language CoT

Language CoT follows the classical LLM paradigm, in which the reasoning process is expressed purely through textual tokens. Formally, the reasoning trajectory is $R_T^{\text{lang}} = r_1^{(l)}, r_2^{(l)}, \ldots, r_T^{(l)}, \quad r_t^{(l)} \in \mathcal{V}_{\text{text}}$, where $\mathcal{V}_{\text{text}}$ is the language vocabulary. At each reasoning step t, the model generates a reasoning step $r_t^{(l)} \sim \pi_\theta\left(\cdot|Q, I, R_{t-1}^{\text{lang}}\right)$ conditioned on the initial query Q, image I, and all the preceding reasoning steps $R_{t-1}^{\text{lang}} = r_1^{(l)}, r_2^{(l)}, \ldots, r_{t-1}^{(l)}$. This formulation corresponds to explicit reasoning in the language space, without any manipulation on visual information. It has been widely used to enhance multimodal reasoning tasks that rely on stepwise logical deduction, such as visual math reasoning [7, 42] and physical reasoning [3].

2.2 Grounding CoT

Grounding CoT [31, 41] expresses the reasoning process by explicitly associating linguistic references with their corresponding visual evidence in the image[2]. While bounding boxes are a popular grounding form in existing work [10, 30], we explore more flexible alternatives, including points, lines, and regions, which adapt to the reasoning context. Formally, each grounded element is represented by a spatial descriptor $g_k = (G_k, C_k)$, where $G_k \in \{\text{point, line, region}\}$ denotes the grounding type, and $C_k \subset [0,1]^2$ represents its spatial coordinates in the image:

$$C_k = \begin{cases} (x, y), & G_k = \text{point} \\ \{(x_1, y_1), (x_2, y_2)\}, & G_k = \text{line} \\ \{(x_i, y_i)\}_{i=1}^{n}, & G_k = \text{region} \end{cases} \tag{1}$$

The trajectory using grounding CoT is denoted as $R_T^{\text{grd}} = \left(r_1^{(g)}, g_1\right), \left(r_2^{(g)}, g_2\right), \ldots, \left(r_T^{(g)}, g_T\right), \quad r_t^{(g)} \in \mathcal{V}_{\text{text}}$. At each reasoning step t, the model generates a reasoning step as $\left(r_t^{(g)}, g_t\right) \sim \pi_\theta\left(\cdot|Q, I, R_{t-1}^{\text{grd}}\right)$. By incorporating explicit yet flexible grounding information, the model learns to connect each reasoning step with the precise spatial evidence it refers to. This strategy enhances reasoning on vision-centric tasks such as counting [29], spatial relation inference [50], and path tracing [11], where visual references may be points, lines, or regions.

2.3 Visual CoT

Visual CoT extends grounding CoT by enabling active visual manipulation during reasoning, with the help of specific tools or functions [55, 58]. Instead of merely referencing visual elements, the model can dynamically operate on grounded visual evidence, such as highlighting a region, tracing a path, or cropping a subimage, and feed the modified visual context back into subsequent reasoning steps. Formally, at each reasoning step t, the model selects a grounding target g_t, and then an operation function $\phi_t(\cdot)$ acts on the current image I_t conditioned on g_t:

$$I_{t+1} = \phi_t(I_t, g_t), \tag{2}$$

where ϕ_t may perform operations such as point marking, line drawing, and region cropping. The reasoning trajectory interleaves textual reasoning and visual updates: $R_T^{\text{vis}} = \left(r_1^{(v)}, g_1, I_1\right), \left(r_2^{(v)}, g_2, I_2\right), \ldots, \left(r_T^{(v)}, g_T, I_T\right)$. At each reasoning step t, the model generates an intermediate reasoning result $\left(r_t^{(v)}, g_t, I_t\right) \sim \pi_\theta\left(\cdot|Q, I, R_{t-1}^{\text{vis}}\right)$. This formulation enables iterative, interleaved image–text reasoning, where the model reasons about visual content, performs grounded visual actions, and refines its understanding through updated visual contexts.

3 Experimental Setup

In this section, we design our experiment to study *how CoT strategies affect the learning of generalizable reasoning ability*. To investigate it, we adopt the *maze navigation task* as our evaluation testbed. This choice offers several advantages: (1) **Minimal interference from pretrained priors**: existing VLMs perform poorly on maze-related tasks, reducing confounding effects of strong pretrained capabilities; (2) **Pure visual reasoning**: maze solving mainly requires spatial reasoning and does not rely on external domain knowledge; and (3) **Controlled difficulty**: task complexity can be precisely tuned by adjusting maze size or path length. In this section, we introduce the experimental setup, including dataset construction, training strategy, and implementation details.

3.1 Maze Data Construction

The maze is represented as an $N \times N$ grid $\mathcal{M} = \{(i, j) \mid i, j \in \{1, \ldots, N\}\}$, where all grids are reachable. Walls are defined between adjacent cells, rather than occupying cells themselves. Let $\mathcal{W} = \left\{w_{(i,j)\to(i',j')} \mid (i, j), (i', j') \in \mathcal{M}\right\}$

[2]We name it as grounding CoT because it strictly localizes and grounds language concepts with the visual information at each step using a special format. While commonly-used language CoTs may include grounding steps, these are often sporadic and poorly structured.

denote the set of walls, where $w_{(i,j)\to(i',j')} = 1$ indicates a wall between cells (i, j) and (i', j'), and 0 otherwise. Given a start cell $s = (i_s, j_s)$ and an end cell $e = (i_e, j_e)$, the model can produce a stepwise reasoning trajectory $R_T = s_1, s_2, \ldots, s_T$ culminating in the correct path sequence $P = [(i_s, j_s), (i_2, j_2), \ldots, (i_e, j_e)]$, satisfying $w_{(i_k,j_k)\to(i_{k+1},j_{k+1})} = 0$. We visualize $\mathcal{M}$ as an image I, accompanied by an instruction Q, and the model is required to generate the reasoning process and the final path. Since current VLMs do not perform well in maze navigation and cannot even generate a reasonable thought process for maze navigation tasks [50], we follow existing SFT-then-RL paradigm and first synthesize three types of data for supervised fine-tuning.

Language CoT Synthesis. Language CoT in the maze navigation task describes the path through directions like "north", "south", "west", and "east". We employ language CoT because maze navigation inherently involves sequential spatial reasoning that can be naturally expressed as a series of linguistic direction tokens, allowing the model to align spatial movement with stepwise textual reasoning. To synthesize reasoning trajectories with language CoT, we first utilize a rule-based function to convert the path $P = [(i_s, j_s), (i_2, j_2), \ldots, (i_e, j_e)]$ to a list of directions describing every step going from s to e. Then we prompt Gemini-2.5-Pro [5] to synthesize the reasoning trajectories $R_T^{\text{lang}} = r_1^{(l)}, r_2^{(l)}, \ldots, r_T^{(l)}$. We name these data as **L-CoT**. The detailed prompt is shown in Appendix, and an example of language CoT is shown in Figure 1.

Grounding CoT Synthesis. Grounding CoT in the maze navigation task refers to every stepped grid with its central coordinate. To synthesize reasonable trajectories with grounding CoT, we first utilize a rule-based function to convert each grid (i_k, j_k) in the path $P = [(i_s, j_s), (i_2, j_2), \ldots, (i_e, j_e)]$ to its absolute coordinate in the image $[x_k, y_k]$. To enhance reasoning depth, we additionally introduce reflection patterns by synthesizing several incorrect paths (e.g., hitting a wall or entering a dead end) along with corresponding correction reasoning. Then we prompt Gemini-2.5-Pro [5] to synthesize the reasoning trajectories $R_T^{\text{grd}} = \left(r_1^{(g)}, g_1\right), \left(r_2^{(g)}, g_2\right), \ldots, \left(r_T^{(g)}, g_T\right)$, where $g_k = (\text{point}, [x_k, y_k])$. The detailed prompt is shown in Appendix. We name this dataset as **G-CoT**. It is worth noting that in the maze navigation task, the target output is a path from the start s to the end e. It is equivalent to a grounding CoT, as it represents the sequence of visited grid positions without additional textual explanation or absolute coordinates. We therefore refer to this naturally aligned variant as **G-CoT-least**, indicating that the reasoning process is inherently embedded in the path sequence rather than being explicitly expressed. An example of the synthesized grounding CoT is shown in Figure 1.

Visual CoT Synthesis. Visual CoT in the maze navigation task utilizes coordinates to refer to the stepped grid, and employ a predefined function to draw a line connecting these coordinates on the image. Compared to language CoT, visual CoT further enhances spatial reasoning by enabling explicit visual interaction, allowing the model to externalize its intermediate thoughts on the image and reason over the updated visual context. To synthesize reasoning trajectories with visual CoT, we first utilize a rule-based function to convert each grid (i_k, j_k) in the path $P = [(i_s, j_s), (i_2, j_2), \ldots, (i_e, j_e)]$ to its absolute coordinate in the image $[x_k, y_k]$. Then we define the operation function $\phi_t(\cdot)$ as a line-drawing operation on the image to reflect the drawing process to generate a sequence of intermediate images $\{I_1, I_2, \ldots, I_T\}$, where each I_t visualizes the partial path from the starting point s to the current step t. Finally, we prompt Gemini-2.5-Pro [5] to synthesize the multimodal reasoning trajectories $R_T^{\text{vis}} = \left(r_1^{(v)}, g_1, I_1\right), \left(r_2^{(v)}, g_2, I_2\right), \ldots, \left(r_T^{(v)}, g_T, I_T\right)$ based on these interleaved image-text pairs. We name these data as **V-CoT**. The detailed prompt is shown in Appendix, and an example of visual CoT is shown in Figure 1.

3.2 Training Strategy

To investigate how different CoT strategies affect model performance, we first perform supervised fine-tuning (SFT) using the three types of synthesized CoT data introduced in Section 3.1, obtaining three policy models with distinct reasoning styles. We then further train these models using reinforcement learning (RL) under the RLVR framework.

Supervised Fine-Tuning. We format the CoT data for SFT by enclosing the synthesized reasoning process within `<think></think>` tags and the final answer within `\boxed{}` tags. Since the answers in G-CoT-least inherently contain the reasoning process, we do not separate them. For each CoT type, we synthesize 8K reasoning trajectories. In the visual CoT setting, the reasoning process is represented as interleaved image-text data, and the cross-entropy loss is computed only over textual tokens.

Reinforcement Learning. Although the fine-tuned VLM acquires the corresponding reasoning strategy, it still struggles to generalize to both in-domain and out-of-domain test data. To further enhance its reasoning robustness, we synthesize an additional 20K maze samples and apply reinforcement learning. The training dynamics reveal how different CoT strategies influence the optimization process. We employ Group Relative Policy Optimization (GRPO) [32] to update the policy model. The reward function is defined as:

$$r = \alpha \cdot r_{\text{acc}} + (1 - \alpha) \cdot r_{\text{format}} \tag{3}$$

The r_{acc} is calculated based on whether the predicted path connects the start and end points without crossing walls, where we implement a rule-based function based on the maze structure. The r_{format} is used to encourage the model to follow a specific format: enclosing the reasoning process in `<think></think>` and wrapping the final path using the symbol `\boxed{}`. We set α to 0.1 in our experiments.

3.3 Implementation Details

We adopt Qwen2.5-VL-7B [1] as the base model. During the SFT stage, the model is fine-tuned on the synthesized CoT data using the LLaMAFactory [57] framework. SFT is conducted for three epochs with a learning rate of 1×10^{-5}, a warm-up ratio of 0.1, and a batch size of 64. In the RL stage, we further optimize the model on the original maze dataset using the verl [33] framework. The rollout batch size is set to 128, the mini-batch size to 32, and the number of rollouts to 8. RL training is continued until the model's performance converges. Prior works in visual reinforcement learning [7, 20, 25] typically train models for only a few hundred—or even just tens of—steps, which often leaves the model under-trained and makes its true performance ceiling unclear. To address this limitation, we train for up to 1000 RL steps, ensuring that the model fully converges and enabling a fair and reliable comparison across different CoT strategies. Throughout all training stages, we freeze the vision encoder and only update the parameters of LLM.

4 Experimental Analysis

4.1 Experimental Framework

To systematically investigate the mechanisms, benefits, and limitations of different CoT strategies in VLMs, our analysis is structured around the following research questions:

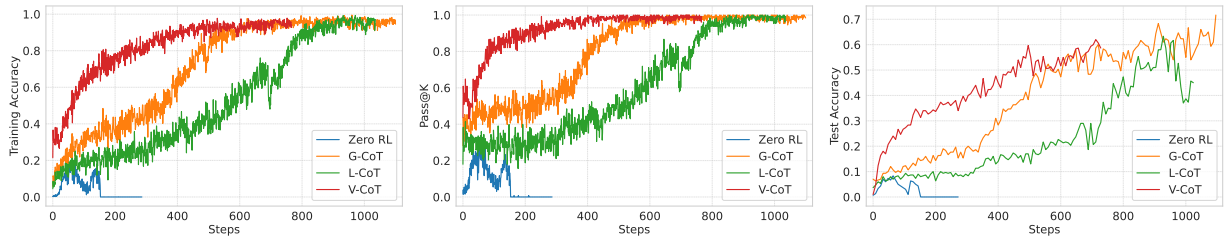
- *Q1: What benefits do these CoT strategies bring to VLMs in an unseen task?*
- *Q2: What fundamental capability enables CoT to work in vision-centric tasks?*
- *Q3: How is the generalizability of these CoT strategies?*

We first conduct experiments on the controlled maze navigation task and answer these questions in Section 4.2. We then extend our analysis to more realistic vision-centric tasks in Section 4.3 to verify whether the observed mechanisms and benefits generalize beyond the maze setting.

4.2 Experimental Results

4.2.1 Visual CoT Improves Efficiency, Not Efficacy

To answer *Q1*, we first compare language CoT, grounding CoT, and visual CoT under the same training setting. Specifically, we perform SFT on L-CoT, G-CoT, and V-CoT respectively to obtain three policy models, and then apply RL on the maze data. Models are trained on mazes of sized 4×4 to 6×6 and evaluated on unseen 7×7 mazes. The RL training dynamics in Figure 2 lead to three observations:

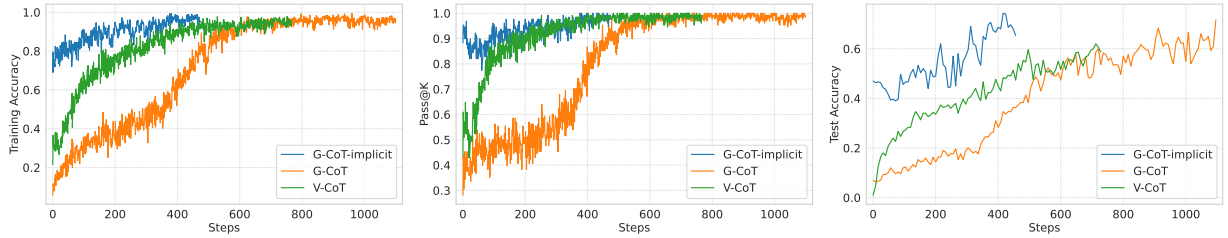


(a) Training Accuracy on Mazes Sized 4×4 to 6×6.

(b) Pass@8 Training Accuracy on Mazes Sized 4×4 to 6×6.

(c) Test Accuracy on Mazes Sized 7×7.

Figure 2 Training dynamics of zero RL, Grounding CoT (G-CoT), Languge CoT (L-CoT), and Visual CoT (V-CoT). The model is cold-started on mazes sized 4×4 to 6×6.



(a) Training Accuracy on Mazes Sized 4×4 to 6×6.

(b) Pass@8 Training Accuracy on Mazes Sized 4×4 to 6×6.

(c) Test Accuracy on Mazes Sized 7×7.

Figure 3 Training Dynamics of least Grounding CoT (G-CoT-least), Grounding CoT (G-CoT), and Visual CoT (V-CoT). The model is cold-started on mazes sized 4×4 to 6×6.

• **Necessity of cold-start**. An SFT initialization is essential for stable and progressive RL training: models trained from scratch tend to collapse, whereas all SFT-initialized models steadily improve and eventually reach 100% accuracy on the training mazes. This suggests that SFT provides a reasonably shaped policy space and mitigates the exploration and reward-sparsity issues that arise when starting RL from random behavior.

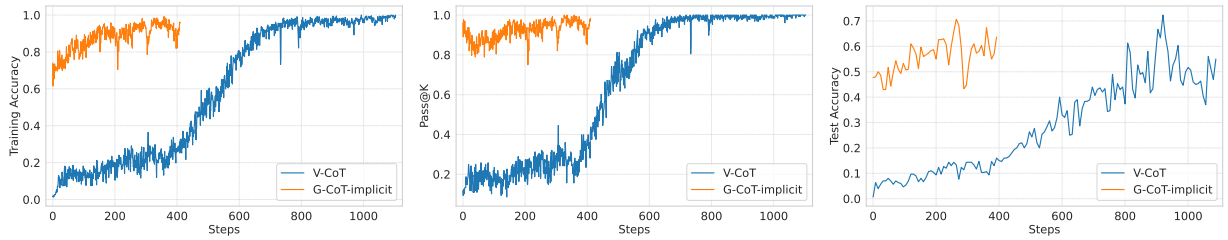
• **Training efficiency**. Among the three CoT strategies, visual CoT yields the fastest RL convergence, requiring roughly half the training steps of language CoT. A plausible explanation is that visual CoT offers more structured, spatially grounded supervision, which aligns better with the underlying maze geometry and thus provides a more informative optimization signal.

• **Performance upper bound**. Despite its faster convergence and higher initial performance, visual CoT does not surpass grounding CoT or language CoT in final accuracy. This indicates that visual CoT mainly accelerates optimization rather than expanding the model's ultimate reasoning capacity: once RL has distilled an efficient internal policy for maze navigation, additional visual manipulation offers diminishing returns on asymptotic performance.

Takeaway Findings
Visual CoT does not offer a higher performance upper bound compared to language or grounding CoT, but only accelerates training.

4.2.2 Short CoT Surpasses Longer Ones

Section 4.2.1 shows that grounding CoT attains performance comparable to visual CoT, suggesting that the model's reasoning is largely mediated by its grounding capability. Here, we further ask whether this grounding-based reasoning depends on the explicit coordinate system used during pre-training. To probe this,

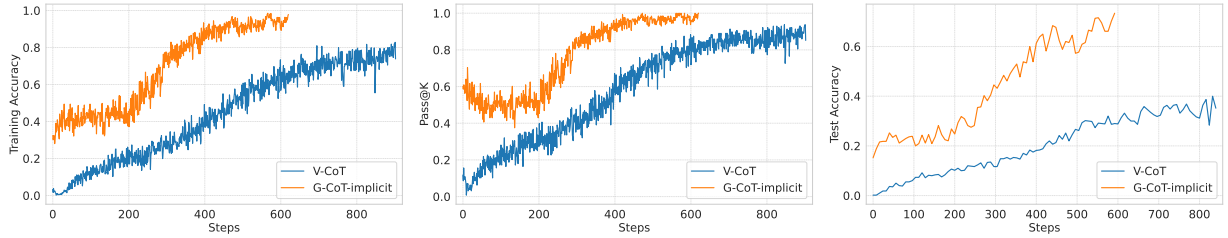


(a) Training Accuracy on Mazes Sized 6×6.

(b) Pass@8 Training Accuracy on Mazes Sized 6×6.

(c) Test Accuracy on Mazes Sized 7×7.

Figure 4 Training Dynamics of least Grounding CoT (G-CoT-least) and Visual CoT (V-CoT). The model is cold-started on mazes sized 6×6 to validate the single-scale generalization.



(a) Training Accuracy on Mazes Sized 7×7 to 9×9.

(b) Pass@8 Training Accuracy on Mazes Sized 7×7 to 9×9.

(c) Test Accuracy on Mazes Sized 10×10.

Figure 5 Training Dynamics of least Grounding CoT (G-CoT-least) and Visual CoT (V-CoT). The model is cold-started on mazes sized 4×4 to 6×6 to validate the cross-scale generalization.

we fine-tune Qwen2.5-VL-7B on both G-CoT and G-CoT-least datasets and then apply RL under the same training and evaluation setup as in Section 4.2.1. As shown in Figure 3, the model trained with the shorter *least* grounding CoT not only starts from a higher performance level but also converges faster than the one trained with explicit grounding CoT, even surpassing the efficiency of visual CoT. At the same time, it still reaches 100% accuracy after RL, despite never seeing explicit coordinates. These results suggest that once the model's grounding ability is properly aligned with the visual environment, it can internalize and operate over its own latent spatial representations without relying on externally specified coordinate systems. Least grounding may thus serve as a more flexible and compact inductive bias, avoiding overfitting to a particular coordinate frame while still supporting robust, generalizable visual reasoning.

Takeaway Findings
The VLM can conduct implicit reasoning after its grounding ability is aligned with the visual environment, achieving full accuracy and faster convergence.

4.2.3 Least CoT Achieves Better Generalization

Our previous experiments show that the model can perform implicit reasoning through its grounding ability. We now ask whether this capability can generalize to mazes of different sizes, and evaluate two types of generalization:

• **Single-scale generalization:** We examine whether a model trained on mazes of a single size can generalize to slightly larger mazes. Concretely, we perform both SFT and RL on 6×6 mazes and evaluate the resulting model on unseen 7×7 mazes.

• **Cross-scale generalization:** We further test whether the model can extrapolate to completely unseen maze sizes after RL, when exposed to diverse scales during SFT. Specifically, we conduct SFT on mazes sized 4×4

Table 1 Evaluation results on other vision reasoning tasks, where the best-performed results are marked as bond.

Model	V^* Bench			HR-Bench 4K			Frozenlake	Jigsaw
	Attr	Spatial	Overall	FSP	FCP	Overall		
Qwen2.5-VL-7B	67.83	78.95	72.25	88.00	57.00	72.50	20.00	0.00
+ V-CoT RL	86.09	78.95	83.25	87.00	57.00	72.00	-	-
+ G-CoT-least RL	**87.83**	**82.89**	**85.86**	**90.75**	**57.50**	**74.12**	**90.33**	**75.60**

to 6×6, perform RL on mazes sized 7×7 to 9×9, and then evaluate on unseen 10×10 mazes.

As shown in Figure 4 and Figure 5, the G-CoT-least policy model generalizes robustly in both settings, maintaining high success rates on unseen maze sizes, whereas Visual CoT saturates after about 800 RL steps and remains inferior to G-CoT-least on both training and test mazes. This suggests that grounding-based implicit reasoning encourages the model to internalize scale-invariant, local navigation rules (*e.g.*, following corridors, backtracking from dead-ends), while visual CoT is more prone to overfitting to specific visual layouts or operation patterns. By relying on compact grounded signals rather than explicit visual edits, G-CoT-least provides a more fundamental and transferable spatial representation, supporting consistent generalization to larger and more complex environments.

Takeaway Findings
Properly aligned grounding ability allows the model to generalize its spatial reasoning effectively to new visual environments.

4.3 Experimental Results on Other Tasks

The previous experiments on the maze dataset demonstrate that the model can conduct implicit reasoning via its grounding ability without the need to generate coordinates explicitly. In this part, we extend our analysis to more realistic vision-centric tasks to verify whether the observed mechanisms and benefits generalize beyond the maze setting. Specifically, we expand the experiments to visual games and real-world VQA tasks.

Visual Games. We evaluate the effectiveness of G-CoT-least on two classic visual reasoning games: FrozenLake and Jigsaw. In FrozenLake, the model must find a path from the start point to the gift while avoiding all holes. The standard environment introduces stochasticity by allowing the agent to slip with a certain probability, but for stability and reproducibility, we set the slipping probability to 0. Since multiple optimal paths may exist, we adopt the environment implementation from Towers et al. [40] to reliably determine whether the agent reaches the target cell. All training and evaluation are conducted on 4×4 grids. In FrozenLake, G-CoT-least corresponds to a series of actions from the start point to the gift (*e.g.*, ["Left", "Up", "Right", "Down"]). In jigsaw, the model must assemble nine puzzle pieces into a coherent image arranged in a 3×3 grid. We use the dataset provided by Wu et al. [51] and follow their 3×3 configuration for both training and evaluation. G-CoT-least in jigsaw task corresponds to the tile indices in raster-scan order (*e.g.*, [7, 5, 1, 6, 8, 3, 9, 2, 4]). Consistent with our maze experiments, we employ a two-stage SFT-then-RL training pipeline. The results in Table 1 demonstrate that G-CoT-least significantly improves model performance on both tasks, with the gains being especially pronounced on Jigsaw, where accuracy rises from 0% to over 70%, demonstrating a substantial enhancement in visual reasoning capability.

Real-world VQA. We follow the experimental setup of DeepEyes [58], performing zero-shot RL on the V^* training set [39]. Unlike visual game environments, a model's grounding ability is largely established during pre-training on massive collections of natural images, so additional SFT is unnecessary. We evaluate the resulting model on both the V^* [39] test set and HR-Bench [44]. In these tasks, V-CoT refers to cropping the region of interest and appending it to the model's context [28, 55, 58]. In contrast, G-CoT-least directly answers the visual questions without introducing explicit visual CoT steps. We compare the performance of V-CoT and G-CoT-least, with results summarized in Table 1. Across all benchmarks and sub-tasks,

the G-CoT-least variants achieve the best performance, demonstrating that the model can perform visual reasoning implicitly, without relying on explicit visual CoT.

5 Related Work

Vision-centric Reasoning. Existing visual reasoning tasks can be broadly categorized into two types. The first involves tasks where reasoning can be entirely performed in the linguistic space once the visual content is understood, such as geometric problem solving [2, 54], physical reasoning [3], or chart understanding [26, 48]. These are language-dominant reasoning tasks, where vision primarily serves as contextual grounding for symbolic inference. The second category comprises tasks that require active visual exploration and manipulation of the image to extract and reason over spatial clues, such as maze navigation [16], puzzle solving [52], and jigsaw [47, 51]. In these vision-centric reasoning tasks, the reasoning process is grounded in perceptual understanding rather than linguistic abstraction. This work focuses on the latter category. We use maze navigation as a controlled testbed for studying vision-centric reasoning and extend our analysis to a broader set of visual games and real-world vqa tasks.

Chain-of-thought Reasoning. Chain-of-Thought (CoT) reasoning was first introduced in [49] as an emergent capability of LLM [9, 53, 56], enabling them to decompose complex problems into intermediate reasoning steps. This approach has been shown to significantly enhance performance across a range of challenging reasoning tasks [4, 17]. Recent studies on long CoT [13] further demonstrates that scaling the length of the reasoning trace enhances model performance, particularly in domains requiring multi-step inference, such as math reasoning [15] and code generation [18]. Building upon CoT for LLMs, researchers have proposed more expressive CoT formulations for VLMs, including grounding-based CoT [31, 41] and visual CoT [28, 34, 58]. These variants enable models to reference specific objects or even manipulate visual content during the reasoning process. In this work, we conduct a systematic comparison of these CoT variants under a controlled experimental setting to understand their underlying mechanisms.

Reinforcement Learning in Vision Language Models. Building on the success of reinforcement learning (RL) in significantly enhancing the reasoning abilities of large language models [6, 27, 36], recent efforts have begun extending RL to VLMs [35, 37]. Prior work explores both zero-RL and cold-start RL regimes, showing substantial improvements in multimodal mathematical and STEM reasoning [7, 42, 45]. For vision-centric tasks, existing studies show that RL can enhance spatial reasoning [30, 50], object recognition [22], and puzzle-solving abilities [12, 47]. Interestingly, these works also observe that, unlike language-dominant tasks, the CoT trajectories induced by RL in vision-centric settings tend to be short. Our work further reveals the underlying mechanism: in vision-centric environments, RL primarily strengthens the model's established grounding capability. Once grounding is sufficiently reinforced, the model can perform effective reasoning with very brief CoT traces.

6 Conclusion

In this work, we systematically investigated how different Chain-of-Thought (CoT) formats affect the acquisition of *generalizable* visual reasoning in vision-language models. Using a controlled maze-solving benchmark and an SFT-then-RL training paradigm on Qwen2.5-VL-7B, we compared language CoT, grounding CoT, and visual CoT. Our experiments show that visual and longer CoT traces can accelerate convergence but do not substantially raise the final performance ceiling; short CoT often surpasses longer ones; and, most notably, CoT formats containing only the least grounded information (*e.g.*, sparse coordinate paths) generalize best across different maze sizes and related visual reasoning tasks.

These findings reveal a "short is long" effect: concise but well-grounded supervision appears more effective for learning reusable visual reasoning patterns than verbose, heavily supervised traces. Looking forward, we plan to extend this analysis to richer task families beyond mazes and VLMs.

Acknowledgement

We sincerely thank Renrui Zhang, Xiaoying Zhang, Jialong Wu, Xinchen Zhang, and other colleagues at ByteDance Seed for their support of this project.

This project is conducted solely for academic research purposes. The findings are independent of ByteDance's commercial products and are not incorporated into any ByteDance products or commercial applications.

References

[1] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv*, 2025.

[2] Jiaqi Chen, Jianheng Tang, Jinghui Qin, Xiaodan Liang, Lingbo Liu, Eric Xing, and Liang Lin. Geoqa: A geometric question answering benchmark towards multimodal numerical reasoning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 513–523, 2021.

[3] Wei Chow, Jiageng Mao, Boyi Li, Daniel Seita, Vitor Guizilini, and Yue Wang. Physbench: Benchmarking and enhancing vision-language models for physical world understanding. *arXiv preprint arXiv:2501.16411*, 2025.

[4] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *CoRR*, abs/2110.14168, 2021.

[5] Google DeepMind. Gemini 2.5 pro: Best for coding and highly complex tasks. *https://deepmind.google/models/gemini/pro/*, 2025.

[6] DeepSeek-AI. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv*, 2025.

[7] Yihe Deng, Hritik Bansal, Fan Yin, Nanyun Peng, Wei Wang, and Kai-Wei Chang. Openvlthinker: An early exploration to complex vision-language reasoning via iterative self-improvement. *arXiv preprint arXiv:2503.17352*, 2025.

[8] Yifan Du, Zikang Liu, Yifan Li, Wayne Xin Zhao, Yuqi Huo, Bingning Wang, Weipeng Chen, Zheng Liu, Zhongyuan Wang, and Ji-Rong Wen. Virgo: A preliminary exploration on reproducing o1-like mllm. *arXiv preprint arXiv:2501.01904*, 2025.

[9] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, Arun Rao, Aston Zhang, Aurélien Rodriguez, Austen Gregerson, Ava Spataru, Baptiste Rozière, Bethany Biron, Binh Tang, Bobbie Chern, Charlotte Caucheteux, Chaya Nayak, Chloe Bi, Chris Marra, Chris McConnell, Christian Keller, Christophe Touret, Chunyang Wu, Corinne Wong, Cristian Canton Ferrer, Cyrus Nikolaidis, Damien Allonsius, Daniel Song, Danielle Pintz, Danny Livshits, David Esiobu, Dhruv Choudhary, Dhruv Mahajan, Diego Garcia-Olano, Diego Perino, Dieuwke Hupkes, Egor Lakomkin, Ehab AlBadawy, Elina Lobanova, Emily Dinan, Eric Michael Smith, Filip Radenovic, Frank Zhang, Gabriel Synnaeve, Gabrielle Lee, Georgia Lewis Anderson, Graeme Nail, Grégoire Mialon, Guan Pang, Guillem Cucurell, Hailey Nguyen, Hannah Korevaar, Hu Xu, Hugo Touvron, Iliyan Zarov, Imanol Arrieta Ibarra, Isabel M. Kloumann, Ishan Misra, Ivan Evtimov, Jade Copet, Jaewon Lee, Jan Geffert, Jana Vranes, Jason Park, Jay Mahadeokar, Jeet Shah, Jelmer van der Linde, Jennifer Billock, Jenny Hong, Jenya Lee, Jeremy Fu, Jianfeng Chi, Jianyu Huang, Jiawen Liu, Jie Wang, Jiecao Yu, Joanna Bitton, Joe Spisak, Jongsoo Park, Joseph Rocca, Joshua Johnstun, Joshua Saxe, Junteng Jia, Kalyan Vasuden Alwala, Kartikeya Upasani, Kate Plawiak, Ke Li, Kenneth Heafield, Kevin Stone, and et al. The Llama 3 herd of models. *CoRR*, abs/2407.21783, 2024.

[10] Yue Fan, Xuehai He, Diji Yang, Kaizhi Zheng, Ching-Chen Kuo, Yuting Zheng, Sravana Jyothi Narayanaraju, Xinze Guan, and Xin Eric Wang. Grit: Teaching mllms to think with images. *arXiv preprint arXiv:2505.15879*, 2025.

[11] Sicheng Feng, Song Wang, Shuyi Ouyang, Lingdong Kong, Zikai Song, Jianke Zhu, Huan Wang, and Xinchao Wang. Can mllms guide me home? a benchmark study on fine-grained visual reasoning from transit maps. *arXiv preprint arXiv:2505.18675*, 2025.

[12] Yichen Feng, Zhangchen Xu, Fengqing Jiang, Yuetai Li, Bhaskar Ramasubramanian, Luyao Niu, Bill Yuchen Lin, and Radha Poovendran. Visualsphinx: Large-scale synthetic vision logic puzzles for rl. *arXiv preprint arXiv:2505.23977*, 2025.

[13] Etash Guha, Ryan Marten, Sedrick Keh, Negin Raoof, Georgios Smyrnis, Hritik Bansal, Marianna Nezhurina, Jean Mercat, Trung Vu, Zayne Sprague, et al. Openthoughts: Data recipes for reasoning models. *arXiv preprint arXiv:2506.04178*, 2025.

[14] Jarvis Guo, Tuney Zheng, Yuelin Bai, Bo Li, Yubo Wang, King Zhu, Yizhi Li, Graham Neubig, Wenhu Chen, and Xiang Yue. Mammoth-vl: Eliciting multimodal reasoning with instruction tuning at scale. *arXiv preprint arXiv:2412.05237*, 2024.

[15] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *NeurIPS Datasets and Benchmarks*, 2021.

[16] Michael Igorevich Ivanitskiy, Rusheb Shah, Alex F Spies, Tilman Räuker, Dan Valentine, Can Rager, Lucia Quirke, Chris Mathwin, Guillaume Corlouer, Cecilia Diniz Behn, et al. A configurable library for generating and manipulating maze datasets. *arXiv preprint arXiv:2309.10498*, 2023.

[17] Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. Livecodebench: Holistic and contamination free evaluation of large language models for code. *arXiv preprint arXiv:2403.07974*, 2024.

[18] Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. *CoRR*, abs/2403.07974, 2024.

[19] 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.

[20] Sicong Leng, Jing Wang, Jiaxi Li, Hao Zhang, Zhiqiang Hu, Boqiang Zhang, Yuming Jiang, Hang Zhang, Xin Li, Lidong Bing, et al. Mmr1: Enhancing multimodal reasoning with variance-aware sampling and open resources. *arXiv preprint arXiv:2509.21268*, 2025.

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

[22] Ming Li, Jike Zhong, Shitian Zhao, Yuxiang Lai, Haoquan Zhang, Wang Bill Zhu, and Kaipeng Zhang. To think or not to think: A study of thinking in rule-based visual reinforcement fine-tuning. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*.

[23] Yifan Li, Yifan Du, Kun Zhou, Jinpeng Wang, Wayne Xin Zhao, and Ji-Rong Wen. Evaluating object hallucination in large vision-language models. *arXiv preprint arXiv:2305.10355*, 2023.

[24] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023.

[25] Xiangyan Liu, Jinjie Ni, Zijian Wu, Chao Du, Longxu Dou, Haonan Wang, Tianyu Pang, and Michael Qizhe Shieh. Noisyrollout: Reinforcing visual reasoning with data augmentation. *arXiv preprint arXiv:2504.13055*, 2025.

[26] Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. Chartqa: A benchmark for question answering about charts with visual and logical reasoning. *arXiv preprint arXiv:2203.10244*, 2022.

[27] OpenAI. Learning to reason with LLMs, 2024. URL https://openai.com/index/learning-to-reason-with-llms/.

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

[29] Jonathan Roberts, Mohammad Reza Taesiri, Ansh Sharma, Akash Gupta, Samuel Roberts, Ioana Croitoru, Simion-Vlad Bogolin, Jialu Tang, Florian Langer, Vyas Raina, et al. Zerobench: An impossible visual benchmark for contemporary large multimodal models. *arXiv preprint arXiv:2502.09696*, 2025.

[30] Gabriel Sarch, Snigdha Saha, Naitik Khandelwal, Ayush Jain, Michael J Tarr, Aviral Kumar, and Katerina Fragkiadaki. Grounded reinforcement learning for visual reasoning. *arXiv preprint arXiv:2505.23678*, 2025.

[31] 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. *Advances in Neural Information Processing Systems*, 37:8612–8642, 2024.

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

[33] 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.

[34] Alex Su, Haozhe Wang, Weiming Ren, Fangzhen Lin, and Wenhu Chen. Pixel reasoner: Incentivizing pixel-space reasoning with curiosity-driven reinforcement learning. *arXiv preprint arXiv:2505.15966*, 2025.

[35] ByteDance Seed Team. Seed1.5-vl technical report. *arXiv preprint arXiv:2505.07062*, 2025.

[36] Kimi Team, Angang Du, Bofei Gao, Bowei Xing, Changjiu Jiang, Cheng Chen, Cheng Li, Chenjun Xiao, Chenzhuang Du, Chonghua Liao, et al. Kimi k1. 5: Scaling reinforcement learning with llms. *arXiv preprint arXiv:2501.12599*, 2025.

[37] Kwai Keye Team, Biao Yang, Bin Wen, Changyi Liu, Chenglong Chu, Chengru Song, Chongling Rao, Chuan Yi, Da Li, Dunju Zang, et al. Kwai keye-vl technical report. *arXiv preprint arXiv:2507.01949*, 2025.

[38] Jingqi Tong, Jixin Tang, Hangcheng Li, Yurong Mou, Ming Zhang, Jun Zhao, Yanbo Wen, Fan Song, Jiahao Zhan, Yuyang Lu, et al. Code2logic: Game-code-driven data synthesis for enhancing vlms general reasoning. *arXiv preprint arXiv:2505.13886*, 2025.

[39] Shengbang Tong, Zhuang Liu, Yuexiang Zhai, Yi Ma, Yann LeCun, and Saining Xie. Eyes wide shut? exploring the visual shortcomings of multimodal llms. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9568–9578, 2024.

[40] Mark Towers, Ariel Kwiatkowski, Jordan Terry, John U Balis, Gianluca De Cola, Tristan Deleu, Manuel Goulão, Andreas Kallinteris, Markus Krimmel, Arjun KG, et al. Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*, 2024.

[41] Haochen Wang, Xiangtai Li, Zilong Huang, Anran Wang, Jiacong Wang, Tao Zhang, Jiani Zheng, Sule Bai, Zijian Kang, Jiashi Feng, et al. Traceable evidence enhanced visual grounded reasoning: Evaluation and methodology. *arXiv preprint arXiv:2507.07999*, 2025.

[42] Haozhe Wang, Chao Qu, Zuming Huang, Wei Chu, Fangzhen Lin, and Wenhu Chen. Vl-rethinker: Incentivizing self-reflection of vision-language models with reinforcement learning. *arXiv preprint arXiv:2504.08837*, 2025.

[43] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, et al. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*, 2024.

[44] 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*, volume 39, pages 7907–7915, 2025.

[45] Xiyao Wang, Zhengyuan Yang, Chao Feng, Hongjin Lu, Linjie Li, Chung-Ching Lin, Kevin Lin, Furong Huang, and Lijuan Wang. Sota with less: Mcts-guided sample selection for data-efficient visual reasoning self-improvement. *arXiv preprint arXiv:2504.07934*, 2025.

[46] Ye Wang, Qianglong Chen, Zejun Li, Siyuan Wang, Shijie Guo, Zhirui Zhang, and Zhongyu Wei. Simple o3: Towards interleaved vision-language reasoning. *arXiv preprint arXiv:2508.12109*, 2025.

[47] Zifu Wang, Junyi Zhu, Bo Tang, Zhiyu Li, Feiyu Xiong, Jiaqian Yu, and Matthew B Blaschko. Jigsaw-r1: A study of rule-based visual reinforcement learning with jigsaw puzzles. *arXiv preprint arXiv:2505.23590*, 2025.

[48] Zirui Wang, Mengzhou Xia, Luxi He, Howard Chen, Yitao Liu, Richard Zhu, Kaiqu Liang, Xindi Wu, Haotian Liu, Sadhika Malladi, Alexis Chevalier, Sanjeev Arora, and Danqi Chen. Charxiv: Charting gaps in realistic chart understanding in multimodal llms. *arXiv preprint arXiv:2406.18521*, 2024.

[49] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed H. Chi, Quoc Le, and Denny Zhou. Chain of thought prompting elicits reasoning in large language models. *CoRR*, abs/2201.11903, 2022. URL https://arxiv.org/abs/2201.11903.

[50] Junfei Wu, Jian Guan, Kaituo Feng, Qiang Liu, Shu Wu, Liang Wang, Wei Wu, and Tieniu Tan. Reinforcing spatial reasoning in vision-language models with interwoven thinking and visual drawing. *arXiv preprint arXiv:2506.09965*, 2025.

[51] Penghao Wu, Yushan Zhang, Haiwen Diao, Bo Li, Lewei Lu, and Ziwei Liu. Visual jigsaw post-training improves mllms. *arXiv preprint arXiv:2509.25190*, 2025.

[52] Weiye Xu, Jiahao Wang, Weiyun Wang, Zhe Chen, Wengang Zhou, Aijun Yang, Lewei Lu, Houqiang Li, Xiaohua Wang, Xizhou Zhu, et al. Visulogic: A benchmark for evaluating visual reasoning in multi-modal large language models. *arXiv preprint arXiv:2504.15279*, 2025.

[53] An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, et al. Qwen2.5 technical report. *arXiv:2412.15115*, 2024.

[54] Renrui Zhang, Dongzhi Jiang, Yichi Zhang, Haokun Lin, Ziyu Guo, Pengshuo Qiu, Aojun Zhou, Pan Lu, Kai-Wei Chang, Yu Qiao, et al. Mathverse: Does your multi-modal llm truly see the diagrams in visual math problems? In *European Conference on Computer Vision*, pages 169–186. Springer, 2024.

[55] 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.

[56] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2), 2023.

[57] Yaowei Zheng, Richong Zhang, Junhao Zhang, Yanhan Ye, Zheyan Luo, Zhangchi Feng, and Yongqiang Ma. Llamafactory: Unified efficient fine-tuning of 100+ language models. *arXiv preprint arXiv:2403.13372*, 2024.

[58] 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.

Appendix

Data Synthesis

In Section 3, we construct the maze dataset and synthesize three types of CoT data: language CoT, grounding CoT, and visual CoT. In this section, we introduce how we synthesize these CoT data.

Language CoT Synthesis. To synthesize reasoning trajectories with language CoT, we first utilize a rule-based function to convert the path $P = [(i_s, j_s), (i_2, j_2), \ldots, (i_e, j_e)]$ to a list of directions describing every step going from s to e. Then we prompt Gemini-2.5-Pro [5] to synthesize the reasoning trajectories. The detailed prompt is shown in Figure 6.

Grounding CoT Synthesis. We first utilize a rule-based function to convert each grid (i_k, j_k) in the path $P = [(i_s, j_s), (i_2, j_2), \ldots, (i_e, j_e)]$ to its absolute coordinate in the image $[x_k, y_k]$. Then we prompt Gemini-2.5-Pro [5] to synthesize the reasoning trajectories. The detailed prompt is shown in Figure 7.

Visual CoT Synthesis. To obtain coherent reflective patterns in visual CoT, we define a line-drawing function on the image and plot each point $[x_k, y_k]$ on the image. Then we prompt Gemini-2.5-Pro [5] to synthesize the reasoning trajectories. The prompt is shown in Figure 8.

You are a professional maze explorer and data annotator tasked with solving a maze step-by-step while narrating your process in English.
You will be provided with three inputs:
1. **Maze Structure**: The cooredinates of start (S) and end (E).
2. **DFS Paths**: A list of failed DFS paths (with dead ends and backtrack points) and one successful path.
3. **Images**: A list of maze diagrams with the following visual cues:
- 'S' marking the **start point**.
- 'E' marking the **end point**.
- The dashed line indicating the successful path from S to E.
Your Task:
Use all these information to simulate the process of an intelligent person walking through a maze: observe the maze, analyze possible actions, and narrate your reasoning step-by-step.
Note that while the successful path and its visualization on the image give you very helpful prior information, you must NOT reference or rely on them directly in your explanation. Instead, use them to silently guide your reasoning simulation. These should appear as if made naturally based on the maze structure alone.
You should only use the direction: ["North", "South", "East", "West"] to describe your movements and not use the coordinates of the path.
Guidelines:
1. **Reason Step-by-Step**:
- First provide a general description of the maze, the start point, and the end point, as prompted by start and end in the Maze Structure.
- You should not analyze the information of every grid cell. Instead, you only need to think carefully when you are at a fork. For most of the cases, you should directly outline the path.
2. **Tone and Detail**:
- Use a thoughtful, methodical tone. Be verbose and thorough in reflecting and planning.
- You are training another AI to generalize this kind of reasoning, so aim for maximum clarity and insight.
Now use the following information to generate the CoT:
Maze Structure:
{meta_info}
DFS Paths:
Failed path:
{dfs_path}
Note: These failed DFS paths are only provided to help you select representative exploration steps to simulate. You must NOT describe the DFS paths as if you knew the outcome in advance. Always simulate decisions as if you are reasoning them out in real-time based on the maze structure. You must use the DFS paths and not create new paths.
Successful path:
{success_path}

Figure 6 Prompt for language CoT synthesis.

You are a professional maze explorer and data annotator tasked with solving a maze step-by-step while narrating your process in English.
You will be provided with three inputs:
1. **Maze Structure**: The cooredinates of S and E. A description of which direction is blocked by the wall of each cell.
2. **DFS Paths**: A list of failed paths obtained by DFS, with the dead end and backtrack point. The final successful path does not have a dead end or a backtrack point.
3. **Images**: A list of maze diagrams with the following visual cues:
- 'S' marking the **start point**.
- 'E' marking the **end point**.
- The first image is the original maze structure, and the following images display the paths obtained by DFS. The dashed lines are the explored path.
Your Task:
It is highly inefficient to obtain the correct path through a maze using DFS. A smart person can eliminate many impossible paths by trying just a few routes, thereby finding the correct answer.
Your task is to use all these information to simulate the process of an intelligent person walking through a maze: first try the first DFS path, then reflect and plan for the next DFS path; try the second DFS path, then reflect and plan for the next DFS path, etc.
Note that while the DFS paths and their visualization on the image give you very helpful prior information, you must NOT reference or rely on them directly in your explanation. Instead, use them to silently guide your reasoning simulation. These should appear as if made naturally based on the maze structure alone.
Guidelines:
1. **Reason Step-by-Step**:
- First provide a general description of the maze, the start point, and the end point, as prompted by start and end in the Maze Structure.
- You should not analyze the information of every grid cell. Instead, you only need to think carefully when you are at a fork or encountering a dead end. For most of the cases, you should directly outline the path.
2. **Reflect the Wrong Exploration**:
- In the DFS paths, all the paths are incorrect and leads to a dead end. Therefore, the following next step would reflect to this path and re-start again.
- In the failed path, you should also pretend to make this mistake and pretend to explain why this move is correct, even if you will reflect this in the next step.
- In your output of this step, you must use a certain tone for the move, since you shouldn't know it will lead to a dead end now.
3. **Tone and Detail**:
- Use a thoughtful, methodical tone. Be verbose and thorough in reflecting and planning.
- You are training another AI to generalize this kind of reasoning, so aim for maximum clarity and insight.
Now use the following information to generate the CoT:
Maze Structure:
{meta_info}
DFS Paths:
Failed path:
{dfs_path}
Note: These failed DFS paths are only provided to help you select representative exploration steps to simulate. You must NOT refer to them directly or imply foreknowledge of failure in your narration. You must use the DFS paths and not create new paths.
Successful path:
{success_path}

Figure 7 Prompt for grounding CoT synthesis.

You are a professional maze explorer and annotator. Given three inputs:
1. **Maze Structure**: The cooredinates of start (S) and end (E).
2. **DFS Paths**: A list of failed DFS paths (with dead ends and backtrack points) and one successful path.
3. **Images**: A list of maze diagrams with the following visual cues:
- 'S' marking the **start point**.
- 'E' marking the **end point**.
- The first image is the original maze structure, and the following images display the paths obtained by DFS. The dashed lines are the explored path.
Your Task:
It is highly inefficient to obtain the correct path through a maze using DFS. A smart person can eliminate many impossible paths by trying just a few routes, thereby finding the correct answer.
Your task is to use **ALL** these information to simulate the process of an intelligent person walking through a maze: first try the first DFS path, then reflect and plan for the next DFS path; try the second DFS path, then reflect and plan for the next DFS path, etc.
Note that while the DFS paths and their visualization on the image give you very helpful prior information, you must NOT reference or rely on them directly in your explanation. Instead, use them to silently guide your reasoning simulation. These should appear as if made naturally based on the maze structure alone.
Guidelines:
1. **Reason Step-by-Step**:
- Start with a general description of the maze, start/end positions. Also mentions that you will use **image_draw_points_tool** function to help myself visualize the path.
- You should not analyze the information of every grid cell. For most of the cases, you should make your thought concise and directly outline the path. However, when you are at a fork or encountering a dead end, you must plan and reflection carefully.
2. **Call Function to Visualize the Path**:
- Call a **image_draw_points_tool** function when you finish a try. The format is: Let's use the image_draw_points_tool to visualize this try: <path>[[x1,y1], [x2,y2], [x3,y3], ...]</path>
- The drawn image from the last step will be given before the next step for visual reference. Remember to check this image of the previous path for the analysis of the current path.
3. **Reflect the Wrong Exploration**:
- In the DFS paths, all the paths are incorrect and leads to a dead end. Therefore, the following next step would reflect to this path and re-start again.
- In the failed path, you should also pretend to make this mistake and pretend to explain why this move is correct, even if you will reflect this in the next step.
- You should NOT mention the outcome of an explored path before visualization, you must first call **image_draw_points_tool** function to visualize the path, and then reflect on the path.
4. **Tone and Detail**:
- Use a thoughtful, methodical tone. Be verbose and thorough in reflecting and planning.
- You are training another AI to generalize this kind of reasoning, so aim for maximum clarity and insight.
Now use the following information to generate the CoT:
Maze Structure:
{meta_info}
DFS Paths:
Failed path:
{dfs_path}
Note: These failed DFS paths are only provided to help you select representative exploration steps to simulate. You must NOT describe the DFS paths as if you knew the outcome in advance. Always simulate decisions as if you are reasoning them out in real-time based on the maze structure. You must use the DFS paths and not create new paths.
Successful path:
{success_path}

Figure 8 Prompt for visual CoT synthesis.