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

Supplementary Material

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.