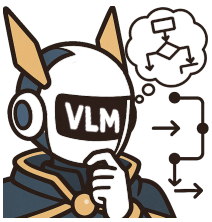


CoSPlan: Corrective Sequential Planning via Scene Graph Incremental Updates

Abstract

*Large-scale Vision-Language Models (VLMs) exhibit impressive complex reasoning capabilities but remain largely unexplored in 'visual sequential planning', i.e., executing multi-step actions towards a goal. Additionally, practical sequential planning often involves non-optimal (erroneous) steps, challenging VLMs to detect and correct such steps. We propose **Co**rrective **S**equential **Plan**ning Benchmark (**CoSPlan**) to evaluate VLMs in error-prone, vision-based sequential planning tasks across 4 domains: maze navigation, block re-arrangement, image reconstruction, and object re-organization.* CoSPlan *assesses two key abilities: **Error Detection** (identifying non-optimal action) and **Step Completion** (correcting and completing action sequences to reach the goal). Despite using state-of-the-art reasoning techniques such as Chain-of-Thought and Scene Graphs, VLMs (*e.g. *Intern-VLM and Qwen2) struggle on* CoSPlan*, failing to leverage contextual cues to reach goals. Addressing this, we propose a novel training-free method, **S**cene **G**raph **I**ncremental updates (**SGI**), which introduces intermediate reasoning steps between the initial and goal states. SGI helps VLMs reason about sequences, yielding an avg. performance gain of $\simeq$5.2%. In addition to enhancing reliability in corrective sequential planning, SGI generalizes to traditional planning tasks such as PlanBench and VQA. Code and dataset will be made public.*

1. Introduction

Vision-Language Models (VLMs) [1, 6] demonstrate strong zero-shot generalization across diverse tasks, increasingly integrating into complex workflows [15, 37]. This raises a key question: *"How well can VLMs handle practical decision-making?" e.g.* robotics, autonomous navigation *etc.* A particularly challenging scenario is **Sequential Planning** [23, 24, 38], where models must execute a series of actions to reach a goal. Realistically, as the number of instructions increases, errors become more likely (Fig. 1). Hence, detecting errors and course-correcting towards the goal is essential for assessing VLMs robustness against errors.

Figure 1. **Corrective Sequential Planning**: Given the initial and final states, with already performed actions w/ some errors (initial context), model identifies errors in the provided context, and picks the optimal action steps to reach the final goal, correcting the error.

Existing work on sequential planning has largely focused on the text domain [26, 37], with limited exploration in the vision domain [32]. Moreover, these works assume ideal conditions, with perfect instructions [30], limiting their applicability in the physical world. However, handling of errors in such sequential planning tasks is vastly unexplored.

Motivated by this, we introduce **CoSPlan** (**Co**rrective **S**equence **Plan**ning), a benchmark designed to study VLMs' planning capabilities in erroneous scenarios. CoSPlan focuses on *2D spatial vision tasks* guided by *text-based instructions*, requiring models to plan a sequence of actions toward a goal *(temporal)*, while *detecting and correcting an erroneous* action. CoSPlan introduces error to simulate realistic challenges and evaluates VLMs on two key abilities: **Error Detection** (identify the error in the initial context (already performed actions)) and **Step Completion** (reach the final goal while correcting the error). CoSPlan includes four diverse tasks: 1) **Maze-E**: Navigation in a 2D maze 2) **Blocks-World-E**: Re-arranging colored blocks 3) **Shuffle-E**: Reconstructing shuffled image tiles 4) **Robo-VQA-E**: Re-organizing real-world objects.

We evaluate five leading VLMs on our benchmark, namely *GPT-4o* [1], *CoG-VLM* [41], *InternVLM-26B* [7], *Janus-pro-7B* [6], and *Qwen2 VL-8B* [40]. On average, the zero-shot performance of these models is close to random guessing. Hence we opt for two popular techniques, known for complex visual reasoning: (i) Chain-of-Thought

(CoT) [42], which guides VLM through intermediate reasoning steps, and (ii) Scene Graph [5], which provides structured representations of objects by modeling their attributes and relationships. This setup enables us to assess our benchmark's complexity and validate its effectiveness.

While Scene Graph structured representations perform well in error-free sequence planning tasks, our evaluation shows that they struggle in error-prone ones. Specifically, they fail to internalize the sequence of steps and miss contextual cues needed to reach the goal. This is likely because solving corrective sequential planning in a single step, from the initial state to the final goal, is inherently difficult. VLMs lack representations of intermediate states and struggle with tracking the evolution of scenes across multiple actions. Addressing this, we propose **SGI** (**S**cene **G**raph **I**ncremental updates), a novel training-free method that refines Scene Graphs step-by-step for each action, generating intermediate states. SGI significantly enhances VLM's capability to i) handle long instruction sequences, ii) track evolving scenes, iii) detect and correct errors to reach the final goal, making corrective sequence planning more robust.

In summary, we make the following contributions: i) **CoSPlan** (**Co**rrective **S**equence **Plan**ning) is the first to reveal limitations of VLMs in handling error-prone sequence planning, with temporal sequences of actions in vision + language domain. CoSPlan includes four diverse planning tasks to test the abilities of *Error Detection* and *Step Completion* to reach a desired goal. ii) We benchmark VLMs like GPT-4o, exposing their vulnerabilities in handling errors, sequence planning in the vision domain, and a lack of context understanding, among other insights. iii) We propose **SGI**, a **S**cene **G**raph **I**ncremental update technique, that refines structured representations step-by-step for every action, enhancing robustness in CoSPlan and traditional datasets like VQA [39], and Plancbench [37].

2. Related work

Reasoning in Foundation Models Enhancing Reasoning without fine-tuning [8] is of great interest in LLM / VLMs. Wei et al. introduced the Chain-of-Thought (COT) via a series of intermediate reasoning steps to guide LLMs toward the final answer. CoT has been shown to greatly improve performance in reasoning tasks like mathematical problems [34], but in visual reasoning, it does not account for spatial relationships. [28] explores planning using GPT. Chen et al. proposed using structured representations (Scene Graphs) of objects, scenes, and relationships, to improve VLM's complex reasoning abilities on tasks like VQA [11], visual grounding [5], image generation [19], spatial reasoning [21], *etc.* However, SG faces challenges in erroneous sequential planning and multi-step reasoning. Our **S**cene **G**raph **I**ncremental updates (**SGI**) approach relies on a pair of images, and an instruction set to update the scene graphs, unlike [27], which uses ground-truth 3D scene graphs.

Table 1. **Previous Reasoning Benchmarks**: Existing works are mostly text only. For modality: 'V' is vision, 'T' is Text. Temporal means change in scene, and synthetic means source of tasks.

Benchmark	Modality	Temporal	Synthetic	Error
ALFWorld [33] (2020)	T	✓	✓	
PlanBench [37] (2022)	T	✓	✓	
WebArena [49] (2023)	T	✓	✓	
SpatialEval [39] (2024)	V+T		✓	
CoSPlan (*Ours*)	V + T	✓	✓	✓

Sequential Planning Valmeekam et al. proposed variations of Sequence Planning, including completing steps based on a partial context. Most sequential planning datasets [2, 22, 30, 31, 44] assume ideal instructions, which may not hold outside the lab environment. Many rely on human / video-based supervision [4, 10, 46], limiting scalability. Most benchmarks focus on textual planning [3, 26, 43, 48], with limited exploration in the vision domain [9, 32, 36] (Tab. 1). SpatialEval [39] only evaluates static images. Recent works have also examined VLMs for planning [14, 16, 20, 36, 45]. CoSPlan is the first benchmark to evaluate VLMs on sequential planning under *vision-language, and temporal domain*, under erroneous instructions.

3. CoSPlan Benchmark

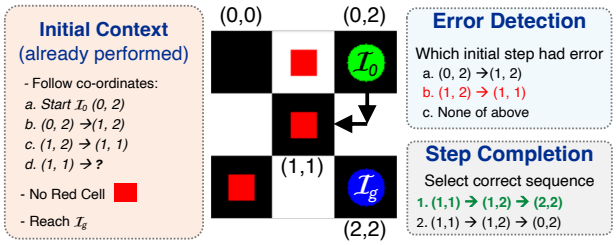
Corrective **S**equence **Plan**ning (**CoSPlan**) mimics general decision-making by evaluating the model's ability to navigate a complex challenge of detecting and correcting non-optimal (error) steps in a sequential planning task. In this setup, • Model $\mathcal{M}$ progresses from an initial state $\mathcal{I}_0$ to a goal state $\mathcal{I}_g$ through a sequence of N actions: $(\mathcal{A}_1, \mathcal{A}_2, ...\mathcal{A}_N)$. • We introduce an intentional non-optimal (error) action $\mathcal{A}_\mathcal{E}$ within the *initial context (already performed k actions)* $(\mathcal{A}_1, \mathcal{A}_2, ..\mathcal{A}_\mathcal{E}..\mathcal{A}_{k<N})$. • The model must *detect* this erroneous action $\mathcal{A}_\mathcal{E}$ and course-correct to *complete* the remaining actions steps $(\mathcal{A}_{k+1}, \mathcal{A}_{k+2}, ...\mathcal{A}_N)$ towards the final goal. Mathematically, it can be shown as

$$\textbf{CoSPlan} \begin{array}{l} \mathcal{M}(\mathcal{A}_{1,..\mathcal{E},..k}; \mathcal{I}_0; \mathcal{I}_g) \\ \text{Initial State: } \mathcal{I}_0 \\ \text{Goal: } \mathcal{I}_g \\ \text{Performed actions: } \mathcal{A}_{1,..\mathcal{E},..k} \end{array} \rightarrow \begin{cases} \textbf{Error Detection} \\ \text{Identify} \mathcal{A}_\mathcal{E}. \\ \textbf{Step Complete} \\ \mathcal{A}_{k+1,k+2,..N} \end{cases} \quad (1)$$

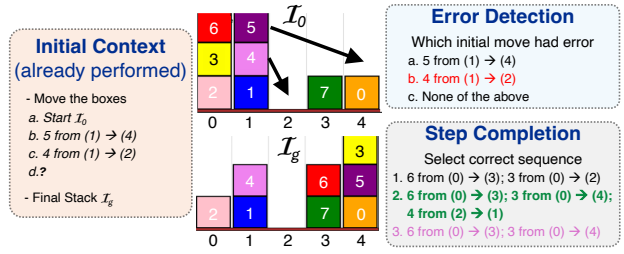
This setup is used to solve diverse scenarios, such as *reconstructing* a correct image from shuffled image tiles, *rearranging & re-organizing* the objects / blocks into a coherent order (obeying physics), and *navigating* through a maze. Success relies on addressing and resolving non-optimal (errors) steps encountered along the way (Fig. 2). Sec. 3.1 describes each dataset with proposed planning tasks.

Table 2. **CoSPlan Dataset Details:** 'Initial Context Length' is the avg. number of actions already performed, and 'Remaining Step Length' are avg. additional steps required to reach the goal after the initial context. The 'Source' is where images (or text) are taken from.

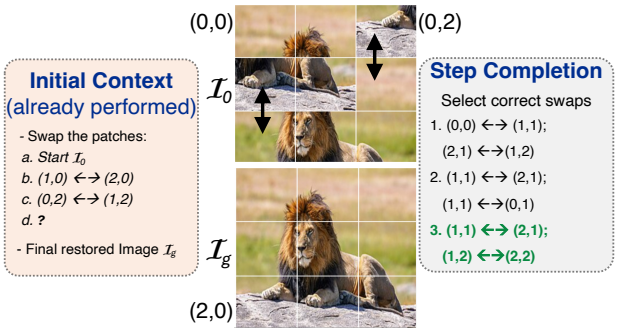
Dataset	Task	Type	# Test Samples (Images & Text)	Initial Context Length (Avg.)	Remaining Step Length (Avg.)	Source
Maze-E	Navigation	Path Planning	5000	2.0	4.6	Synthetic
Blocks-World-E	Re-arrangement	Blocks	5000	2.0	3.8	Synthetic
Shuffle-E	Re-construction	Puzzle	1000	3.7	7.1	ImageNet (2009)
Robo-VQA-E	Re-organization	Real-world	350	5.5	4.1	ROM (2024)



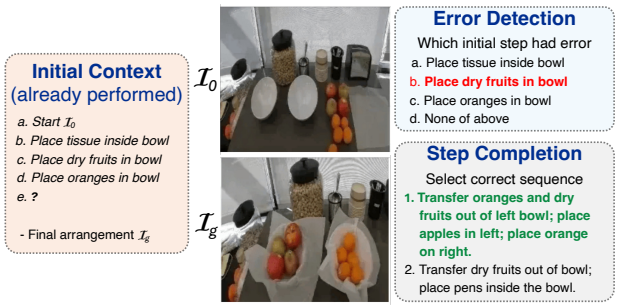
(a) **Maze-E (Navigating)**: → denotes movement.



(b) **Blocks-World-E (Re-arrangement)**: X from $(a) \rightarrow (b)$ indicates move box # X from column (a) to column (b)



(c) **Shuffle-E (Re-construction)**: ↔ indicates patch swap



(d) **Robo-VQA (Re-organization)**: Real World

Figure 2. **Overview of CoSPlan Benchmark**: Given initial ($\mathcal{I}_s$) and final state ($\mathcal{I}_e$) and initial set of instructions (orange), the model needs to perform two tasks: *Step completion*, choosing right set of future paths (green) to complete the task, and *Error detection* the sub-optimal / erroneous action in past actions (initial context). Shown coordinates (row, column) are 0-indexed. Initial steps visualized as black arrows, and safeguard against cheating (Section 3.2) highlighted in pink for Blocks-World-E. Shuffle-E sub-optimal errors are computationally infeasible, hence ignored (Sec. 3.1). Rest all errors (red) violate the rules of the environment or unnecessary (sub-optimal).

What's an Error? We loosely define "error" as a plausible but suboptimal action that deviates from the optimal path to the goal, potentially resulting in longer sequences. Error can also be a purely wrong action that makes it impossible to reach the goal without correction, *e.g.* referencing non-existent objects, violating task/physics constraints *etc.*

3.1. Benchmark Datasets

We introduce four sequential planning datasets, each featuring diverse tasks with intentional sub-optimal errors (except 'Shuffle-E') posing unique challenges in corrective sequence planning. The datasets are structured as multiple-choice questions that test: i) **Error Detection**: Identifying the non-optimal erroneous action from initial context (already performed actions), or selecting "none of the above". ii) **Step Completion**: Selecting the correct answer among 5 options that would correct the mistake and lead to the final goal. The use of synthetic datasets [25, 39] has been shown to test reasoning vulnerabilities in VLMs. A overview is provided in Tab. 2 & Fig. 2, respectively.

Maze-E (Fig. 2a): The *goal* is to solve a maze while navigating from the start cell to the goal cell. *Inputs* is a maze layout with starting position (green, $\mathcal{I}_0$), a destination position (blue $\mathcal{I}_g$), and a sequence of initial moves with error like moving into a red dead-end cell, detours, diagonal move, moving out of maze *etc.* requiring backtraction. Dataset is constructed via randomly sampled grid of size $\in [3\times3, 8\times8]$, and up to 5 obstacles,susing OpenCV [17][1].

Blocks-World-E (Fig. 2b): The *goal* is to stack blocks in a specific (target) configuration. *Inputs* is the initial block arrangement ($\mathcal{I}_0$) and the final arrangement ($\mathcal{I}_g$), initial sequence for stacking with an erroneous (sub-optimal) step

[1]Black & white pattern helps distinguish cells/navigate.

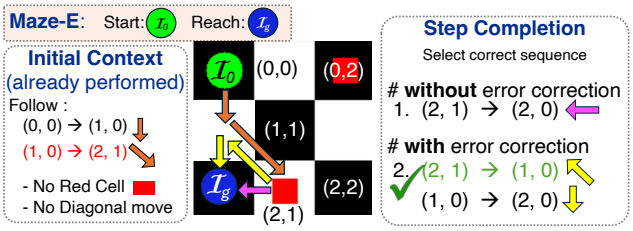


Figure 3. **Error Correction** Initial context (orange arrows) with error move $\mathcal{A}_{\mathcal{E}}$ (diagonal $(1,0) \rightarrow (2,1)$) to cell. Step completion to I_g w/ error correction (yellow) is correct while w/o (pink) is not.

to mislead the model. Suboptimal errors involve inefficient stacking, placing blocks in impossible positions (*e.g.* air), moving inner blocks (not on top), *etc.* The dataset is generated using OpenCV by rendering start and end configurations with 3–8 boxes randomly placed across columns.

Shuffle-E (Fig. 2c): The *goal* is to restore a shuffled scene to its original order, via swapping image patches (tiles). *Input* is a starting shuffled image ($\mathcal{I}_0$), the final restored image ($\mathcal{I}_g$), and the initial sequence of image patch swaps. *Step completion* is correct sequence of swaps to generate the restored image. *Error Detection* is skipped here, as one initial erroneous swap will cascade into subsequent incorrect swaps, breaking the one-error assumption applied to other datasets (limitations Sec. 5). ImageNet [12] sampled uniformly from each class totaling 1000 images were used.

Robo-VQA-E (Fig. 2d): The *goal* is to manipulate / reorganize the objects via a robot, collected in the real-word scenarios. The dataset (restructured ROM [31]) consists of 350 image pairs, curated by us (humans). *Inputs* include starting ($\mathcal{I}_0$) image, final image ($\mathcal{I}_g$), and a sequence of initial actions for object placements. A suboptimal error might involve unnecessarily picking of objects, arbitrarily placements, manipulate an out-of-scene object, invalid action (*e.g.* open a already open door), *etc.* *Goal* is the remaining sequence of object placements to get final organization.

3.2. Safeguard against Cheating

Revealing the final image ($\mathcal{I}_g$) to the model can help cheat the planning task by simply picking the option that best describes $\mathcal{I}_g$, without looking at intial context or identifying the error. To prevent this leakage, we include an incorrect option that is identical to the correct one but **omits** the error correction step. For example, in Fig. 2b 'Step Completion', the green (2) and pink options (3) both match $\mathcal{I}_g$, but only the green option (2) has the error correction step of moving block 4 from 2nd column to 1st column. Similarly, in Fig. 3, the 'Step Completion' that reverts the error (#2 yellow arrow) is the solution compared to the other option (#1 pink arrow) that reaches the target but doesnt correct the error.

Chain of Thought on Maze-E

"Identify": *Describe problem constraints*

e.g. Start from (0,0), reach (2,1), and avoid obstacles at (1,1)

"Context": *Describe initial context (actions already taken)*

e.g. Move from (0,0) to (2,0), then move from (2,0) to (1,1)

"Verify": *Take actions to reach goal & verify all constraints*

e.g. Stated from (0,0), path avoided (1,1), path reached (2,1), actions taken only involve vertical and horizontal paths

Figure 4. **CoT for Maze-E**: Detailed description in Sec. 3.5.1

Scene Graph on Robo-VQA-E

 # node 1
{"name": "bowl_1", "state": "contains dry fruits with tissue", "location": "table, left"} # Edges = location

 # node 2
{"name": "bowl_2", "state": "contains dry fruits with tissue", "location": "table, right"} # Edges = location

 # node 3
{"name": "apples", "state": "", "location": "on table, near bowls"} # Edges = location

Figure 5. **Scene Graph for Robo-VQA-E** SG generated via GPT-4o, with objects as *nodes*, location as *edge*, state as *attributes*.

3.3. Sequence Completion Design

`CoSPlan` design choice for error correction within step completion mimics general scenarios where agents must detect and recover from errors in ongoing sequences, while completing the task. Alternatives like separating tasks into (i) explicit error correction and (ii) continuation from a valid state assume the error-free steps for reaching the goal, which may not reflect practical decision-making. Instead, our 'correct' option may begin from the erroneous state but proposes a recovery sequence that leads to the goal, without additional errors. Similarly, 'incorrect' options may perpetuate the error or introduce new ones.

3.4. Evaluation

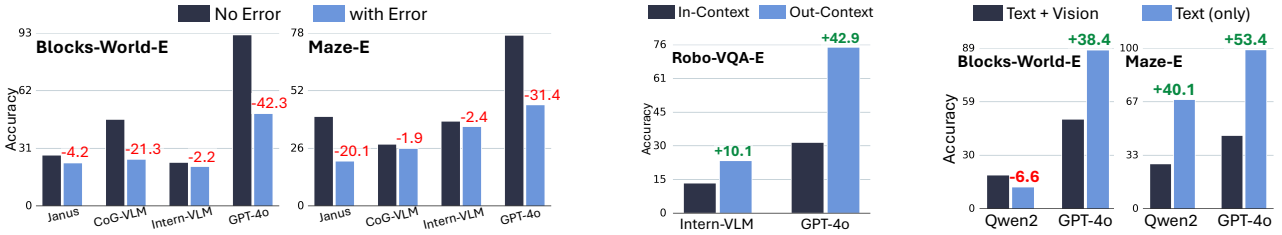
`CoSPlan` is evaluated using multiple-choice question (MCQ) framework (shown in Fig. 2), with Top-1(%) accuracy as the evaluation metric. We *independently* evaluate **Step Completion:** the model chooses *one correct option among 5* options (random accuracy $\frac{1}{5}$), and **Error Detection:** MCQ setup presents *initial context actions as choices*, with an additional option of 'none of the above' denoting no error present (random accuracy $\mathbb{E}[\frac{1}{\text{Initial context length}+1}]$).

3.5. Models & Techniques

We follow the OpenVLM leaderboard (Huggingface) for selecting Vision-Language Models (VLMs) for our corrective sequential planning tasks. We incorporate both closed

Table 3. **CoSPlan benchmark:** Evaluation described in Section 3.4. Higher number (↑) implies better performance. 'V' is Vanilla VLM (no CoT and SG modification), CoT is Chain-of-Thought, and SG is Scene Graphs (both initial and final states are input). † indicates GPT-4o vanilla wasnt evaluated because of its inferior performance compared CoT and SG in other VLMs, and its paid (budget constraint).

VLM	Step Completion (% ↑)												Error Detection (% ↑)								
	Robo-VQA-E			Shuffle-E			Maze-E			Blocks-World-E			Robo-VQA-E			Maze-E			Blocks-World-E		
	V	CoT	SG	V	CoT	SG	V	CoT	SG	V	CoT	SG	V	CoT	SG	V	CoT	SG	V	CoT	SG
Random	20			20			20			20			25.4			26.1			26.1		
Qwen2 VL-8B	17.1	17.6	18.9	24.1	24.9	25.1	26.5	27.9	28.3	18.1	18.6	18.8	9.2	9.1	9.6	20.5	20.8	20.7	32.3	30.6	35.2
CoG-VLM	13.1	12.5	21.5	23.1	27.1	23.7	25.1	25.9	26.5	25.5	25.2	26.7	32.1	33.4	35.3	6.4	8.4	13.3	41.3	43.1	44.5
Janus-pro-7B	14.1	14.7	21.3	23.2	23.1	23.5	20.4	20.2	21.7	24.2	23.1	25.1	17.5	18.1	26.1	20.5	19.1	21.0	29.3	31.0	27.6
Intern-VLM	22.1	23.5	25.1	20.1	23.2	23.4	21.6	35.8	41.2	18.3	21.2	18.9	24.3	25.2	26.1	32.8	33.1	33.4	36.5	37.9	37.3
GPT-4o †	-	48.2	52.2	-	27.6	30.1	-	45.6	46.1	-	49.7	54.3	-	45.3	44.2	-	40.3	35.3	-	35.1	42.1



(a) **Effect of error in context** (b) **Out vs In-context errors** (c) **Multi-modal vs Text-only**

Figure 6. *(a)* VLMs excel in error-free settings, highlighting the complexity of error-prone ones. *(b)* Errors from within context (scene) are harder than random ones (out-context). *(c)* VLMs struggle on visual reasoning; however perform exceptionally well on text-only domain.

(**GPT-4o** [1]) and open-source (*e.g.* **CoG-VLM** [41], **InternVLM-26B** [7], **Qwen2 VL-8B** [40], **Janus-pro-7B** [6]). Since GPT-4o is not open-source and requires payment per use, we have used it judiciously for selected experiments. More details about each model in *Supplementary*. We next describe how Chain-of-Thought (CoT) and Scene Graphs (SG) are used to further improve these VLMs.

3.5.1. Baseline Reasoning

Chain-of-Thought (CoT [42]) We adapt CoT for our CoSPlan datasets by i) *Identify:* Providing models with a detailed description of the problem and constraints; ii) *Context:* Step-by-step description of each action in the initial context; iii) *Verify:* Ask model to plan a path to reach goal while verifying it follows all constraints. An example CoT for the Maze-E is shown in Fig. 4. This approach is model-agnostic, and the same set of instructions is provided to all models. Detailed examples in *supplementary*.

Scene Graphs (SG [42]) In our work, we use the Scene Graph (SG) in addition to CoT as added context to aid VLM reasoning abilities. Given an initial state ($\mathcal{I}_0$) and a goal state ($\mathcal{I}_g$), we 'QUERY' (prompt) VLM to *"Construct scene graphs for the initial and goal states, capturing key objects, attributes, spatial relationships, and target configurations"*. The VLMs then generate state graph consisting of three key components: *a) Nodes*: Objects present in the scene, *b) Edges*: Relationships (*e.g.* spatial position) between objects *c) Attributes*: Object properties and interactions. An example SG for the Robo-VQA-E is shown in Fig. 5 (detailed examples in *supplementary*).

Comparison Different VLM have different structural representations, hence SGs are very model-specific. We have standardized attributes (*e.g.* nodes for objects, edges for relations) via unified prompts, rejecting invalid formats. To ensure fairness across models, identical SG schemas and prompts were enforced across models, with strict JSON validation for outputs. Cross-model comparisons thus focus on task performance under consistent structures, despite inherent differences, *e.g.* GPT-4o vs. Qwen2 VL-8B verbosity.

3.6. Results & Analysis

Vanilla vs CoT vs SG: Table 3 compares VLMs on CoSPlan benchmark, via Vanilla method (raw image-text input) and via enhanced reasoning (CoT and Scene Graph (SG)). CoT improves performance on vanilla models, and SG provides additional gains (with few exceptions), underscoring the value of structured representations for corrective sequence planning. GPT-4o makes relatively informed reasoning decisions, while Janus-pro-7B, CoG-VLM, and Qwen2 VL-8B perform near or below random chance, indicating the difficulty of the task. Task difficulty follows: Shuffle-E > Maze-E > Robo-VQA-E >Blocks-World-E. Accuracy less than random can partially be explained by overwhelmingly picking certain options [47] (Janus selects 'option A' 94% times) or options without error correction (cheating, Sec. 3.2). Exploring VLM's pattern for answering MCQ is *left as future work*. *Error Detection* provably is substantially harder than *Step Completion*, likely because step completion can leak answers while error detection requires a deeper understanding of context and task. Higher

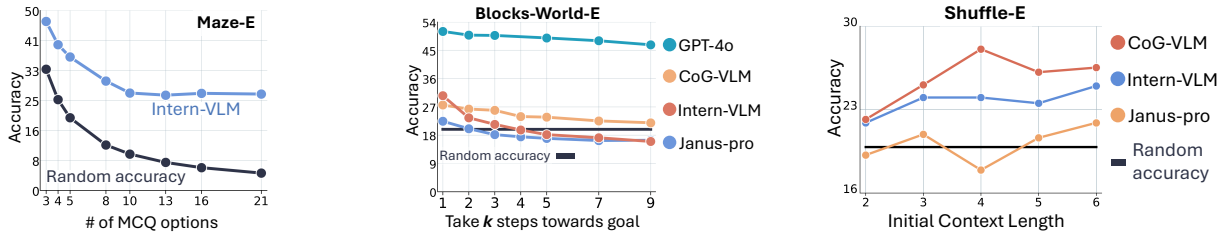


Figure 7. All tasks (a,b,c) shown for Step completion, with (a) using Scene Graph (SG), and (b,c) using Chain-of-Though (CoT). **a) Effect of # MCQ options** As the number of options ↑, Intern-VLM accuracy starts to drop. **b) Information from MCQ options** With 9 remaining steps, VLMs take next K steps toward the goal. Constant accuracy indicates VLMs ignore additional context from MCQ. **c) Length of Initial Context** VLM accuracy shows a positive correlation with context length, *i.e.* as # of already performed steps ↑, accuracy ↑.

performance on Blocks-World-E error detection might be due to data leakage, since variants of Blocks-World are widely used in pretraining these VLMs.

Complexity of Errors: Figure 6a shows VLMs perform relatively well in error-free settings (GPT-4o near-perfect accuracy) but struggle when errors are introduced, revealing VLMs' dependency on ideal, error-free settings. These highlight the need for challenging, error-aware benchmarks like `CoSPlan` to highlight the vulnerable gap between training and practical error-prone scenarios (CoG-VLM and Janus-pro-7B predict randomly under errors).

Effect of Context: `CoSPlan` includes a random mix of two types of errors: i) *In-Context* Suboptimal erroneous step that involves objects present in the scene, ii) *Out-Context* Error uses random objects not in scene. Figure 6b shows lower performance on In-Context errors, suggesting VLMs struggle more when erroneous suboptimal actions involve plausible objects from within the scene while they can handle the out-of-context errors with relative ease.

Necessity of Vision modality: Figure 6c shows that transforming multi-modal tasks (vision + text) into text-only formats significantly boosts model reasoning, validating recent works [18, 29]. Qwen2's near-random prediction in Blocks-World-E reveals its limitations for `CoSPlan`, while other tasks like Robo-VQA-E and Shuffle-E cannot be faithfully represented as text-only without visual aid. This exposes the VLM's vulnerability in visual reasoning, and the need for a sequence planning benchmark in the visual domain.

MCQ Options: Fig. 7(a) shows that increasing the number of options (the default is 5 for step completion), exposes the randomness in picking options, as the accuracy drops with the number of options (first reported by [13]). This reveals the added complexity of MCQ in `CoSPlan`. Alternative to MCQ (plan generation) is *left as future work*.

Ignoring additional context: We input a constant context of length 2 (1 initial step and 1 error), and evaluate step completion, where the models need to take 9 steps to reach the goal $\mathcal{I}_g$ (inclusive of 1 error correction). Fig. 7(b) evaluates step completion where only the next k steps towards $\mathcal{I}_g$ are available, *e.g.* $k = 2$ means MCQ options will show only the next 2 steps (won't reach I_g). $k = 9$ would have options where I_g is reached. This helps us to measure if models use additional available context ($\propto k$) in MCQ options to reach goal $\mathcal{I}_g$. All models (including GPT-4o) **maintain stable accuracy regardless of k**, indicating additional information via MCQ options (# of the remaining steps or 'k') does not influence the reasoning to reach the goal. This can be explained by models' preference towards certain options (*e.g.* Janus selects 'option A' 94% times) and models strong dependency on initial context (only two steps provided here) to make reasonable predictions.

Initial Context importance Fig. 7(c) shows that as the length of the initial context goes up, model accuracy goes up, signifying the importance of already performed actions in sequence planning. We hypothesize that models may not understand context (Fig. 7(b)), but seeing more steps likely help weed out erroneous step. We use this as the motivation for our novel **SGI** (**S**cene **G**raph **I**ncremental update) technique to maximize information gain from this context.

4. Scene Graph Incremental update (SGI)

Tab. 3 shows that Scene Graphs (and CoT) enhance VLM's performance on sequential planning tasks. However, with only initial and final states, the model is forced to internally interpolate intermediate steps. This is because SG and CoT both attempt to encapsulate the entire task transformation within a single-step graph, making it challenging to coherently capture the intermediate states (and transitions) across sequential actions. This places a heavy burden on the model's ability to simulate long action sequences, something VLMs struggle with. Addressing this, we propose a dynamic approach that adaptively represents evolving scenes, incrementally updating the Scene Graph as actions unfold. Simulating each action generates *explicit representations* of intermediate states, breaking down sequences into smaller transitions. Explicit intermediate states bridge the gap between the initial and final states, improving VLMs corrective sequence planning and error detection.

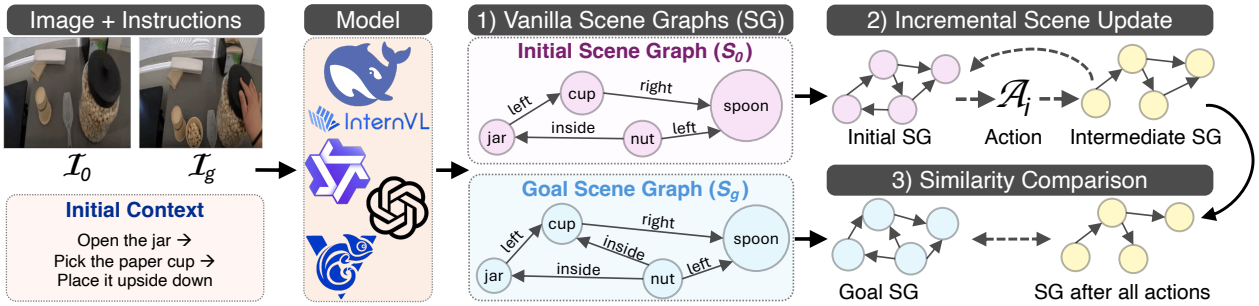


Figure 8. **SGI** 1) Initial and Goal Scene Graphs (SG) are generated. 2) Incremental Scene Update sequentially modifies SG for each action A_i 3) Similarity Comparison matches the resultant SG with Goal graph for searching for the best-aligned sequence.

Algorithm 1 SGI for Step Completion (Sec. 4)

Input: Initial state $\mathcal{I}_0$, Goal state $\mathcal{I}_g$, Initial Context actions $\mathcal{A}_1, \mathcal{A}_2, ..\mathcal{A}_\mathcal{E}..\mathcal{A}_{k<N}$

Objective: Pick the best option m' from MCQ for Step Completion $\mathcal{A}_{k+1}, \mathcal{A}_{k+2}...\mathcal{A}_N$

Require: VLM $\mathcal{M}$, Step Completion MCQ m options.

1) Vanilla Scene Graph, (ref Section 3.5.1)

$S_0 \leftarrow \text{QUERY}\left[\mathcal{M}(\mathcal{I}_0)\right]$ // Obtain initial Scene Graph

$S_g \leftarrow \text{QUERY}\left[\mathcal{M}(\mathcal{I}_g)\right]$ // Obtain goal Scene Graph

2) Incremental Scene Update ($S_0 \rightarrow S_c \rightarrow S_m$)

$S_c \leftarrow S_0$

for $\mathcal{A}_i$ in $[\mathcal{A}_1, \mathcal{A}_2, ..\mathcal{A}_\mathcal{E}..\mathcal{A}_{k<N}]$ **do**

  $S_c \leftarrow \text{SIMULATE}\left[\mathcal{M}(S_c, \mathcal{A}_i)\right]$

  // $\mathcal{M}$ simulates i-th action $\mathcal{A}_i$ to incrementally update intermediate context Scene Graph S_c

end for

for each option $m \in MCQ$ **do**

  $\mathcal{A}^m_{k+1}, \mathcal{A}^m_{k+2}...\mathcal{A}^m_N \leftarrow m$ // actions from option m

  $S_m \leftarrow S_c$ // make a copy for option m

  for $\mathcal{A}^m_i$ in $[\mathcal{A}^m_{k+1}, \mathcal{A}^m_{k+2}...\mathcal{A}^m_N]$ **do**

    $S_m \leftarrow \text{SIMULATE}\left[\mathcal{M}(S_m, \mathcal{A}^m_i)\right]$

    // Simulate test actions to reach goal for the m^{th} option

  end for

end for

3) Similarity Comparison

$m' \leftarrow \arg\max_{m \in MCQ} \text{SIMILARITY}\left[\mathcal{M}(S_m, S_g)\right]$

// Ask VLM to determine the similarity between all MCQs derived Scene Graphs and goal Scene Graph S_g

Output: m'

4.1. Algorithm

An overview (Fig. 8) and pseudo code for step completion is shown in Algorithm 1, with detailed steps below. More description and SGI for error detection in *supplementary*.

1) Vanilla Scene Graphs (SG): We 'QUERY' VLMs (feed the states to the model to generate Scene Graphs, described in Section 3.5.1) to generate the Scene Graphs for the initial state $\mathcal{I}_0$ as S_0 and the final goal $\mathcal{I}_g$ as S_g. We have already evaluated the performance of these vanilla Scene Graphs on the baselines in Tab. 3 as $SG = [S_0, S_g]$

2) Incremental Scene Update: Starting from the initial SG (S_0), we feed a textual description of each action ($\mathcal{A}_{1,..\mathcal{E},..k}$) to VLM and ask it 'SIMULATE' the action on the SG producing intermediate SG (S_c). The 'SIMULATE' prompt to VLM: *"Simulate the given action sequence from the initial state, incrementally updating the scene graph."* modifies nodes, attributes, and edges of SG. We then use intital context SG S_C to 'SIMULATE' each MCQ option independently producing for final scene graph S_m for the 'm-th' option encompassing $\mathcal{A}_{k+1,k+2..,..N}$. Note, SG S_C after all the intial context steps is same for all MCQ options.

3) Similarity Comparison: After simulating each action, *VLM is asked to compare* 'SIMILARITY' between resultant SG S_m and goal SG S_g. Prompt used for this: ***"Compare** the resulting scene graph with the goal scene graph to identify incorrect relationships, misplaced objects, or unmet constraints and score them between 0-100."* compares mismatches in SGs. The option with best similarity score between the option S_m and S_g is chosen as prediction.

4.2. Difference between SGI vs SG & CoT

Chain-of-Thought (CoT) represents the most basic form, where VLMs break complex tasks into a sequence of step-by-step reasoning steps. Scene Graph (SG) builds on CoT via a structured representation of the scene, enabling more coherent tracking and reasoning. Both CoT and SG focus on reasoning within a single scene and interpolating decisions from that. Our **S**cene **G**raph **I**ncremental update (**SGI**) extends this framework by adding a *temporal, 3D* component, where scene graphs not only to represent the current scene but also to derive next-time-frame scene graphs. Effectively, SGI interpolates CoT and SG reasoning across sequential scenes, allowing VLMs to reason through evolving scenes rather than interpolating scene-level decisions. In terms of reasoning hierarchy, CoT $\subseteq$ SG $\subseteq$ SGI.

4.3. Results

`CoSPlan` Comparison Table 4 shows that SGI significantly outperforms the vanilla Scene Graph (SG) approach

Table 4. **Scene Graph Incremental update (SGI)**: SGI improvement relative to vanilla SG, same naming convention as Tab. 3.

Method	Step Completion (% ↑)								Error Detection (% ↑)					
	Robo-VQA-E		Shuffle-E		Maze-E		Blocks-World-E		Robo-VQA-E		Maze-E		Blocks-World-E	
	SG	SGI	SG	SGI	SG	SGI	SG	SGI	SG	SGI	SG	SGI	SG	SGI
Intern-VLM	25.1	32.1 (+7.0)	23.4	25.2 (+1.8)	41.2	43.2 (+2.0)	18.9	29.2 (+10.3)	26.1	31.5 (+5.4)	33.4	34.8 (+1.4)	37.3	42.9 (+5.6)
GPT-4o	52.2	56.4 (+4.2)	30.1	37.0 (+6.9)	46.1	56.1 (+10.0)	54.3	55.3 (+1.0)	44.2	57.4 (+13.2)	35.3	41.1 (+5.8)	42.1	50.7 (+8.6)

Table 5. **SGI on VQA dataset [39]** Format same as Tab. 3.

Method	Step Completion (% ↑)								
	Spatial-Map			Maze-Nav			Spatial-Grid		
	CoT	SG	SGI	CoT	SG	SGI	CoT	SG	SGI
CoG -VLM	25.1	36.7	35.8	32.3	32.4	31.2	30.1	34.3	38.2
Janus -pro-7B	42.4	47.4	47.8	20.8	27.3	29.3	34.4	35.8	36.3
Intern -VLM	36.3	41.3	44.3	28.6	40.5	42.1	33.3	33.8	35.1

Table 6. **SGI on Planbench [37]** PlanBench score (Plan completion)

Method	Variant	Task 8 Score (↑)
Qwen2 VL-8B	Vanilla	13.8
	CoT	14.1
	SG	13.9
(our)	SGI	14.7

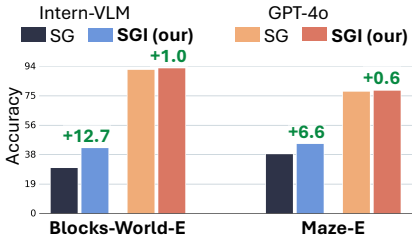


Figure 9. **Error Free Step Completion**

across all benchmark tasks/datasets. For *Step Completion*, SGI achieves a 1.8%–10.3% improvement for Intern-VLM and 1%–10% for GPT-4o. For *Error Detection*, SGI improves performance by 1.4%–5.6% for Intern-VLM and 5.8%–13.2% for GPT-4o. **Gemini-2.5-pro** [35] observes 67% for CoT, 70% with SG and 71.5% for SGI on Blocks-World-E for *Step Completion*. Compute-wise, SGI makes 1 VLM call/step, *i.e.* length of initial context + # of MCQ options × Avg. # of steps per option. The added compute is justified by up to a 13% boost in error detection, with similarly consistent improvements across all tasks, proving the effectiveness of incorporating intermediate scene representations in robust sequential planning.

External Dataset Unlike Sequence Planning, which involves a final state and a sequence of intermediate transitions, Visual Question Answering (VQA) [39] lacks *temporal structure*. VQA typically presents a static scene accompanied by MCQs. In our formulation, we treat this figure as both the initial and the final state, and ask VLM to iteratively simulate all MCQ options (checking for feasibility). In contrast to Scene Graph or CoT, SGI independently and iteratively **evaluates each MCQ option. This extra emphasis on options can enables more informed decision-making**, as highlighted in the superior performance of SGI in Tab. 5. Description of each task in *Supplementary*. We also evaluate our SGI algorithm on text-only PlanBench [37], plan completion with *blockworld* (task 8). We use Qwen2 VL-8B, with SGI applied to textual SG, yielding the best PlanBench score (Tab. 6). Note, other tasks of PlanBench are plan generation and not plan completion (outside the scope of the work).

Error-Free Scenario: Figure 9 shows SGI not only outperforms SG on error-prone corrective sequence planning but also in the error-free ideal scenario, making our appraoch generic for all types of sequence planning tasks, further validating the robustness of our approach in enhancing VLMs for structured sequential reasoning and decision-making.

5. Limitation

`CoSPlan` reveals near-random predictions of VLMs in error-prone sequential planning with *just one error*. Expanding our analysis to multiple error cases requires automation currently limited by VLMs (*e.g.* GPT-4o) not being able to handle even one error. Additionally, `CoSPlan` adds a temporal dimension to sequence planning, but extending it to videos requires VLM's processing multiple videos (start and goal states), *left as future work*.

6. Conclusion

In this work, we introduce `CoSPlan`, a benchmark designed to evaluate the decision-making capabilities of VLMs in error-prone, sequential planning tasks that simulate practical scenarios. `CoSPlan` challenges VLMs to solve *2D spatial vision* tasks with *text-based instructions*, requiring *temporal* reasoning over previously executed actions. Our empirical analysis reveals key limitations in current VLMs: i) they often make random predictions, ignoring contextual understanding, ii) struggle with in-context errors, and iii) exhibit a bias toward text-based reasoning over multimodal decision-making. Notably, even advanced models like GPT-4o fail to leverage contextual cues effectively to reach goals. To address these challenges, we propose **SGI** (**S**cene **G**raph **I**ncremental update), a technique that refines scene graphs step-by-step with each action, generating intermediate steps. SGI substantially boosts performance across error-prone, error-free, and VQA settings compared to vanilla scene graphs, highlighting its robustness and ability to enhance corrective sequence planning in VLMs.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 1, 5

[2] Masataro Asai. Photo-realistic blocksworld dataset, 2018. 2

[3] Masataro Asai, Hiroshi Kajino, Alex Fukunaga, and Christian Muise. Classical planning in deep latent space, 2022. 2

[4] Smail Ait Bouhsain, Rachid Alami, and Thierry Simeon. Learning to predict action feasibility for task and motion planning in 3d environments. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3736–3742. IEEE, 2023. 2

[5] Ming Chen, Yuan Li, et al. Measuring and improving chain-of-thought reasoning in vision-language models. *arXiv preprint arXiv:2309.04461*, 2023. 2

[6] Xiaokang Chen, Zhiyu Wu, Xingchao Liu, Zizheng Pan, Wen Liu, Zhenda Xie, Xingkai Yu, and Chong Ruan. Janus-pro: Unified multimodal understanding and generation with data and model scaling. *arXiv preprint arXiv:2501.17811*, 2025. 1, 5

[7] Zhe Chen, Jiannan Wu, Wenhai Wang, Weijie Su, Guo Chen, Sen Xing, Muyan Zhong, Qinglong Zhang, Xizhou Zhu, Lewei Lu, et al. Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 24185–24198, 2024. 1, 5

[8] Anjie Cheng et al. Spatialrgpt: Grounded spatial reasoning in vision language model. *OpenReview*, 2024. 2

[9] 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. 2

[10] Breanne Crockett, Carl L Mueller, and Bradley Hayes. Human demonstrations enable efficient solutions to sequential manifold planning problems. In *Proceedings of the 2025 ACM/IEEE International Conference on Human-Robot Interaction*, pages 800–809, 2025. 2

[11] Vinay Damodaran, Sharanya Chakravarthy, Akshay Kumar, Anjana Umapathy, Teruko Mitamura, Yuta Nakashima, Noa Garcia, and Chenhui Chu. Understanding the role of scene graphs in visual question answering, 2021. 2

[12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009. 3, 4

[13] Shresth Grover, Vibhav Vineet, and Yogesh S Rawat. Navigating hallucinations for reasoning of unintentional activities, 2024. 6

[14] Shibo Hao, Yi Gu, Haodi Ma, Joshua Jiahua Hong, Zhen Wang, Daisy Zhe Wang, and Zhiting Hu. Reasoning with language model is planning with world model. *arXiv preprint arXiv:2305.14992*, 2023. 2

[15] Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. Large language models cannot self-correct reasoning yet. In *The Twelfth International Conference on Learning Representations*, 2024. 1

[16] Xu Huang, Weiwen Liu, Xiaolong Chen, Xingmei Wang, Hao Wang, Defu Lian, Yasheng Wang, Ruiming Tang, and Enhong Chen. Understanding the planning of llm agents: A survey. *arXiv preprint arXiv:2402.02716*, 2024. 2

[17] Itseez. Open source computer vision library. `https://github.com/itseez/opencv`, 2015. 3

[18] Anubhooti Jain, Mayank Vatsa, and Richa Singh. Words over pixels? rethinking vision in multimodal large language models. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI-25*, pages 10481–10489. International Joint Conferences on Artificial Intelligence Organization, 2025. Survey Track. 6

[19] Justin Johnson, Agrim Gupta, and Li Fei-Fei. Image generation from scene graphs, 2018. 2

[20] Subbarao Kambhampati, Karthik Valmeekam, Lin Guan, Mudit Verma, Kaya Stechly, Siddhant Bhambri, Lucas Paul Saldyt, and Anil B Murthy. Position: Llms can't plan, but can help planning in llm-modulo frameworks. In *Forty-first International Conference on Machine Learning*, 2024. 2

[21] Xinghang Li, Di Guo, Huaping Liu, and Fuchun Sun. Embodied semantic scene graph generation. In *5th Annual Conference on Robot Learning*, 2021. 2

[22] Kartik Nagpal and Negar Mehr. Optimal robotic assembly sequence planning: A sequential decision-making approach, 2025. 2

[23] Siddharth Nayak, Adelmo Morrison Orozco, Marina Ten Have, Vittal Thirumalai, Jackson Zhang, Darren Chen, Aditya Kapoor, Eric Robinson, Karthik Gopalakrishnan, Brian Ichter, James Harrison, Anuj Mahajan, and Hamsa Balakrishnan. MAP-THOR: Benchmarking long-horizon multi-agent planning frameworks in partially observable environments. In *Multi-modal Foundation Model meets Embodied AI Workshop @ ICML2024*, 2024. 1

[24] Swarna Kamal Paul. Sequential planning in large partially observable environments guided by llms, 2023. 1

[25] Atin Pothiraj, Elias Stengel-Eskin, Jaemin Cho, and Mohit Bansal. Capture: Evaluating spatial reasoning in vision language models via occluded object counting. *arXiv preprint arXiv:2504.15485*, 2025. 3

[26] Santhosh Kumar Ramakrishnan, Erik Wijmans, Philipp Kraehenbuehl, and Vladlen Koltun. Does spatial cognition emerge in frontier models? *arXiv preprint arXiv:2410.06468*, 2024. 1, 2

[27] Krishan Rana, Jesse Haviland, Sourav Garg, Jad Abou-Chakra, Ian Reid, and Niko Suenderhauf. Sayplan: Grounding large language models using 3d scene graphs for scalable task planning. In *7th Annual Conference on Robot Learning*, 2023. 2

[28] Nicholas Rossetti, Massimiliano Tummolo, Alfonso Emilio Gerevini, Luca Putelli, Ivan Serina, Mattia Chiari, and Matteo Olivato. Learning general policies for planning through gpt models. *Proceedings of the International Conference on*

Automated Planning and Scheduling, 34(1):500–508, 2024. 2

[29] Luca M Schulze Buschoff, Elif Akata, Matthias Bethge, and Eric Schulz. Visual cognition in multimodal large language models. *Nature Machine Intelligence*, 7(1):96–106, 2025. 6

[30] F. Sener, D. Chatterjee, D. Shelepov, K. He, D. Singhania, R. Wang, and A. Yao. Assembly101: A large-scale multi-view video dataset for understanding procedural activities. *CVPR 2022*, 2022. 1, 2

[31] Pierre Sermanet, Tianli Ding, Jeffrey Zhao, Fei Xia, Debidatta Dwibedi, Keerthana Gopalakrishnan, Christine Chan, Gabriel Dulac-Arnold, Sharath Maddineni, Nikhil J Joshi, et al. Robovqa: Multimodal long-horizon reasoning for robotics. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 645–652. IEEE, 2024. 2, 3, 4

[32] Parshin Shojaee, Iman Mirzadeh, Keivan Alizadeh, Maxwell Horton, Samy Bengio, and Mehrdad Farajtabar. The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity, 2025. 1, 2

[33] Mohit Shridhar, Xingdi Yuan, Marc-Alexandre Côté, Yonatan Bisk, Adam Trischler, and Matthew Hausknecht. Alfworld: Aligning text and embodied environments for interactive learning. *arXiv preprint arXiv:2010.03768*, 2020. 2

[34] Zayne Sprague, Fangcong Yin, Juan Diego Rodriguez, Dongwei Jiang, Manya Wadhwa, Prasann Singhal, Xinyu Zhao, Xi Ye, Kyle Mahowald, and Greg Durrett. To cot or not to cot? chain-of-thought helps mainly on math and symbolic reasoning. *arXiv preprint arXiv:2409.12183*, 2024. 2

[35] Gemini Team, Rohan Anil, Sebastian Borgeaud, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, Katie Millican, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023. 8

[36] Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. On the planning abilities of large language models - a critical investigation. In *Advances in Neural Information Processing Systems*, pages 75993–76005. Curran Associates, Inc., 2023. 2

[37] Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. On the planning abilities of large language models-a critical investigation. *Advances in Neural Information Processing Systems*, 36:75993–76005, 2023. 1, 2, 8

[38] Chaojie Wang, Yanchen Deng, Zhiyi Lyu, Liang Zeng, Jujie He, Shuicheng Yan, and Bo An. Q*: Improving multi-step reasoning for llms with deliberative planning, 2024. 1

[39] Jiayu Wang, Yifei Ming, Zhenmei Shi, Vibhav Vineet, Xin Wang, Yixuan Li, and Neel Joshi. Is a picture worth a thousand words? delving into spatial reasoning for vision language models, 2024. 2, 3, 8

[40] 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. 1, 5

[41] Weihan Wang, Qingsong Lv, Wenmeng Yu, Wenyi Hong, Ji Qi, Yan Wang, Junhui Ji, Zhuoyi Yang, Lei Zhao, Xixuan Song, et al. Cogvlm: Visual expert for pretrained language models. *arXiv preprint arXiv:2311.03079*, 2023. 1, 5

[42] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837, 2022. 2, 5

[43] Ruixuan Xiao, Wentao Ma, Ke Wang, Yuchuan Wu, Junbo Zhao, Haobo Wang, Fei Huang, and Yongbin Li. FlowBench: Revisiting and benchmarking workflow-guided planning for LLM-based agents. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 10883–10900, Miami, Florida, USA, 2024. Association for Computational Linguistics. 2

[44] Haoran Zhang, Hangyu Guo, Shuyue Guo, Meng Cao, Wenhao Huang, Jiaheng Liu, and Ge Zhang. Ing-vp: Mllms cannot play easy vision-based games yet, 2024. 2

[45] Jiatao Zhang, Lanling Tang, Yufan Song, Qiwei Meng, Haofu Qian, Jun Shao, Wei Song, Shiqiang Zhu, and Jason Gu. Fltrnn: Faithful long-horizon task planning for robotics with large language models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6680–6686. IEEE, 2024. 2

[46] He Zhao, Isma Hadji, Nikita Dvornik, Konstantinos G. Derpanis, Richard P. Wildes, and Allan D. Jepson. P3iv: Probabilistic procedure planning from instructional videos with weak supervision, 2022. 2

[47] Chujie Zheng, Hao Zhou, Fandong Meng, Jie Zhou, and Minlie Huang. Large language models are not robust multiple choice selectors. *arXiv preprint arXiv:2309.03882*, 2023. 5

[48] Huaixiu Steven Zheng, Swaroop Mishra, Hugh Zhang, Xinyun Chen, Minmin Chen, Azade Nova, Le Hou, Heng-Tze Cheng, Quoc V. Le, Ed H. Chi, and Denny Zhou. Natural plan: Benchmarking llms on natural language planning, 2024. 2

[49] Shuyan Zhou, Frank F. Xu, Hao Zhu, Xuhui Zhou, Robert Lo, Abishek Sridhar, Xianyi Cheng, Tianyue Ou, Yonatan Bisk, Daniel Fried, Uri Alon, and Graham Neubig. Webarena: A realistic web environment for building autonomous agents, 2024. 2