SPATIAL TOOL FORESTS FOR STRUCTURED SPATIAL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Existing tool-augmented spatial reasoning methods rely on predefined tools whose limited coverage restricts complex reasoning, while directly expanding the tool library increases selection ambiguity. We propose Spatial Tool Forest (STF), a hierarchical framework that organizes spatial tools into an operation-specific forest and uses constrained Monte Carlo Tree Search (MCTS) for tool-chain planning. Given an image sequence and a natural-language query, the framework analyzes the query to extract its spatial requirements and searches legal trunks from compatible tool-forest branches for a reliable, task-adaptive tool chain. When no existing candidate trunk fully covers the required spatial operations, the framework retains the candidate trunk that best matches the current task as a scaffold, and a constrained LLM-based module generates only the missing tool under explicit interface, spatial-semantic, and evidence constraints. The resulting chain is executed to produce traceable evidence and the final answer. We evaluate STF on MINDCUBE-1k, OMNI3D-BENCH, and SPAR-Bench, observing consistent improvements across datasets and task categories. On MINDCUBE-1k, it improves overall accuracy over pySpatial by 6.38 percentage points and increases accuracy in the challenging around category from 51.60% to 70.40%. These results demonstrate improvements in spatial reasoning accuracy, stability, and adaptability through hierarchical tool organization, constrained planning, and targeted tool generation.

1 INTRODUCTION

Multimodal large language models (MLLMs) have made remarkable progress in image understanding, visual question answering, video understanding, and complex reasoning. However, these capabilities do not naturally translate into reliable 3D spatial reasoning. Controlled evaluations reveal persistent weaknesses in relative positions, depth, reference-frame changes, and multi-view spatial inference (Liu et al., 2023; Kamath et al., 2023; Cheng et al., 2024; Wang et al., 2026).

To alleviate this problem, prior work has introduced visual programming and explicit 3D spatial tools (Gupta & Kembhavi, 2023; Surís et al., 2023; Ma et al., 2024). These approaches enable a model to reason over geometric scenes by invoking tools for 3D reconstruction, camera-pose recovery, camera movement, and novel-view synthesis. In particular, pySpatial transforms otherwise implicit multi-view spatial reasoning into executable operations over reconstructed scenes (Luo et al., 2026). Figure 1 contrasts flat tool expansion with the structured tool-forest design adopted in STF.

However, explicit 3D representations and executable spatial tools do not automatically solve complex spatial reasoning problems. A model must still translate spatial conditions expressed in natural language into suitable tool-call paths. Complex tasks such as rotation and around require not only basic geometric operations but also task-adapted operations such as reference-frame updating, viewpoint-displacement interpretation, motion-chain simulation, and local relation judgment. Fixed tool sets provide incomplete coverage of these fine-grained operations, whereas directly expanding the tool library enlarges the candidate space and makes it easier for LLMs to select superficially relevant but unsuitable tool chains.

Recent work has attempted to improve tool coverage through dynamic tool construction, for example by asking LLMs to generate tools for the current problem or abstract reusable tools from prior experience (Marsili et al., 2025). Although these methods improve expressiveness, they introduce new sources of instability: generated tools may have inconsistent interfaces, fail during execution,

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

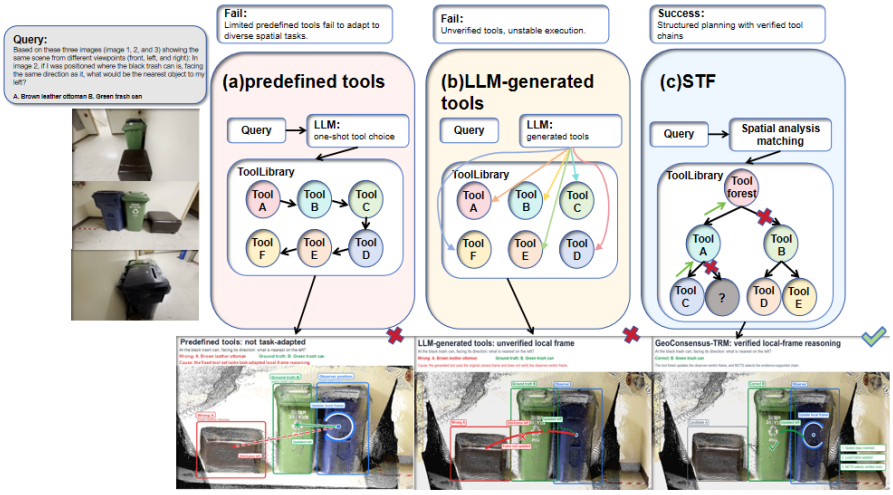


Figure 1: Motivation for STF. Flat tool expansion improves spatial-operation coverage but enlarges the candidate space and increases selection ambiguity. Our structured tool forest restricts planning to compatible branches and legal trunks for more reliable, evidence-grounded spatial reasoning.

or produce answers without sufficient supporting evidence. Multiple rounds of generation and verification also increase inference time and cost. The key challenge is therefore to organize and select reliable tool chains for each problem, rather than simply enlarge the tool library.

For this purpose, we propose STF, an MCTS-based framework for hierarchical tool organization and selection. It organizes spatial tools into a structured forest in which each branch represents a class of operations, such as reference-frame updating, view-displacement interpretation, motion-chain simulation, or local relation judgment. Given an image sequence and a natural-language query, the framework first extracts the required spatial operations and reference-frame constraints, and then performs constrained MCTS planning over legal trunks from compatible branches to select a reliable, task-adaptive tool chain, drawing on UCT-style exploration and structured search (Kocsis & Szepesvári, 2006; Yao et al., 2023). When no existing candidate trunk fully covers the required spatial operations, the framework retains the candidate trunk that best matches the current task as a scaffold, and a constrained LLM-based tool-generation module generates only the missing tool under explicit interface, spatial-semantic, and evidence constraints. This design balances tool coverage, selection robustness, and inference cost. Figure 2 presents the complete processing workflow.

We evaluate STF on MINDCUBE-1k, OMNI3D-BENCH, and SPAR-Bench. On MINDCUBE-1k, the method improves overall accuracy over the original pySpatial by 6.38 percentage points and produces its largest gain in the around category, from 51.60% to 70.40%. Consistent advantages on OMNI3D-BENCH and SPAR-Bench further indicate that hierarchical tool organization and MCTS-based selection improve spatial reasoning accuracy, stability, and task adaptability.

Our contributions can be summarized as follows:

- We present STF, a framework that organizes spatial tools into a structured forest, analyzes each query and identifies compatible tool-forest branches, and allocates the planning budget among reliable legal trunks through constrained MCTS planning.
- We construct operation-specific branches for reference-frame updates, viewpoint displacement, motion chains, object relations, and visibility, together with controlled local completion and evidence-based experience writeback. This design expands coverage without exposing the controller to an unconstrained flat tool space.
- We systematically evaluate the framework on MINDCUBE-1k, OMNI3D-BENCH, and SPAR-Bench. STF improves pySpatial by 6.38 percentage points on MINDCUBE-1k, with the largest gain in the around category, and remains competitive across benchmarks.

2 Related Work

2.1 MLLMs for Spatial Reasoning

Recent MLLMs have shown strong performance across image understanding, visual question answering, video understanding, and complex reasoning. However, controlled spatial benchmarks reveal persistent errors in left–right relations, object orientation, depth, viewpoint changes, and multi-view geometry (Liu et al., 2023; Kamath et al., 2023; Ma et al., 2023). SpatialVLM (Chen et al., 2024a) and SpatialRGPT (Cheng et al., 2024) improve grounded single-view spatial reasoning through scalable spatial supervision, while SpatialPIN (Ma et al., 2024) injects explicit 3D priors at inference time. MINDCUBE (Wang et al., 2026) and SPAR-Bench (Zhang et al., 2025a) further expose limitations in limited-view mental modeling, reference-frame transformation, and multi-view inference. Unlike methods that enhance representations through additional data or pre-training, our work structures the tool space to plan reliable chains for tool-augmented reasoning.

2.2 3D Reconstruction and Explicit Spatial Tools

Explicit 3D representations provide an important geometric foundation for complex spatial reasoning. Classical Structure-from-Motion pipelines require staged correspondence estimation and iterative optimization (Schönberger & Frahm, 2016). More recent feed-forward systems—including DUSt3R (Wang et al., 2024), MASt3R (Leroy et al., 2024), CUT3R (Wang et al., 2025b), and VGGT (Wang et al., 2025a)—directly predict camera parameters, depth, point maps, or other pixel-aligned 3D representations from sparse images. These advances make reconstruction, camera-pose recovery, viewpoint movement, and novel-view synthesis practical components of tool-augmented reasoning. Rather than proposing a new reconstruction model, our work studies how to organize and select chains built from these spatial tools.

2.3 Tool-Augmented Visual Programming and Spatial Reasoning

Tool-augmented reasoning decomposes complex tasks into calls to external vision models, expert modules, and symbolic operations. Visual ChatGPT (Wu et al., 2023), MM-REACT (Yang et al., 2023), and HuggingGPT (Shen et al., 2023) use LLMs to coordinate multimodal tools. VisProg (Gupta & Kembhavi, 2023) and ViperGPT (Surís et al., 2023) generate executable programs that invoke visual modules, improving interpretability and compositionality. VADAR (Marsili et al., 2025) introduces a dynamically generated API for single-image 3D reasoning, whereas pySpatial (Luo et al., 2026) composes reconstruction, camera-motion, and novel-view tools for multi-view reasoning. These systems demonstrate the value of explicit tools but also expose a central challenge: fixed APIs have limited coverage, while flat expansion or unconstrained generation increases selection ambiguity and execution risk. STF instead combines a structured tool forest, forest-constrained planning over compatible branches, and constrained missing-tool generation.

3 Method

In this section, we introduce STF, a hierarchical tool organization and selection framework based on MCTS that improves tool-augmented spatial reasoning on complex tasks. Rather than allowing an LLM to combine all available tools freely, STF organizes them into a structured spatial tool forest, analyzes each query to extract its spatial requirements, identifies compatible branches, and performs Forest-MCTS planning over their legal candidate trunks. When no existing candidate fully covers the required spatial operations, the framework retains the candidate trunk that best matches the current task as a scaffold and constrains the LLM to generate only the unavailable tool needed to complete it. As illustrated in Figure 2, the completed trunk is instantiated and executed, and evidence-supported trajectories are written to the experience library to guide subsequent trunk planning.

3.1 Problem Formulation

We consider spatial reasoning from an image sequence. Given $\mathcal{I} = \{I_i\}_{i=1}^{N}$, where $N \geq 1$, a natural-language query q, and an answer space $\mathcal{A}$, the goal is to predict the correct answer $r^\star \in \mathcal{A}$. When multiple views are available, the task requires the model to jointly reason over object locations,

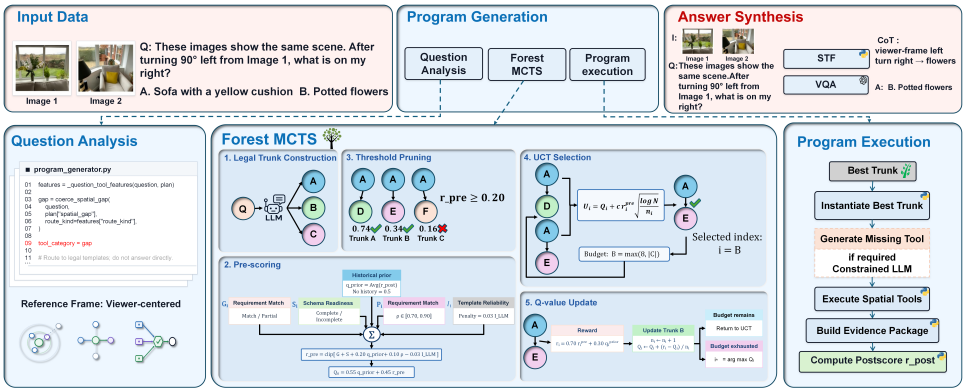


Figure 2: Overall framework of STF. The system analyzes each multimodal query to extract its spatial requirements and identify compatible tool-forest branches, over whose legal trunks Forest-MCTS selects a reliable tool chain for execution. If no candidate trunk fully covers the required operations, the candidate trunk that best matches the current task is completed with a constrained LLM-generated tool before execution. The final answer is derived from traceable execution evidence.

camera movements, and local geometric cues across viewpoints; in the single-view setting, the same framework operates on the available geometric and visual evidence.

To obtain an actionable geometric basis, we first utilize a feed-forward 3D reconstruction module to convert the limited 2D observations into an explicit scene representation, denoted as

$$\mathcal{G} = (\mathbf{K}, \{\mathbf{T}_i\}_{i=1}^{N}, \{D_i\}_{i=1}^{N}, \mathcal{P}).$$

Here, $\mathbf{K}$ denotes the camera intrinsics, $\mathbf{T}_i \in SE(3)$ denotes the camera pose of the i-th view, D_i denotes the depth estimate, and $\mathcal{P}$ denotes the scene-level point cloud. This representation enables the model to perform spatial operations such as viewpoint movement, reference-frame transformation, local relation analysis, and occlusion verification in a unified geometric space.

On this basis, we represent a reasoning process as an executable tool invocation path $\tau = (t_1, t_2, \ldots, t_L)$, where each t_l is a spatial tool or a step instantiated from spatial tools. After executing τ, the system obtains intermediate results o_τ, an evidence set e_τ, and a candidate answer r_τ. The core problem is therefore to select, from an expanded tool library, a structurally valid, task-adaptive, and evidence-supported path conditioned on $(\mathcal{I}, \mathcal{G}, q)$.

3.2 Spatial Tool Forest

Existing approaches typically provide spatial tools to LLMs through tool APIs. This approach is straightforward when the number of tools is small, but the candidate space grows rapidly as the library expands to cover more complex tasks. The model may then select a seemingly relevant tool that is unsuitable for the current problem. To address this issue, we organize spatial tools into a hierarchical tool forest. Specifically, we define a tool forest

$\mathcal{F} = \{B_m\}_{m=1}^{M}$, where each branch B_m represents a class of spatial operations, such as reference-frame updates, viewpoint displacement, motion chains, local relations, or occlusion verification. The canonical branches are public_spatial_tree, viewer_frame_tree, egocentric_frame_tree, motion_transform_tree, object_relation_tree, visibility_occlusion_tree, and close_pair_tree.

Before trunk retrieval, the query analyzer extracts and normalizes spatial-requirement signals. View displacement, pose chains, and distance deltas serve as compatibility cues for motion_transform_tree; reference-frame and camera-turn queries serve as compatibility cues for viewer_frame_tree; observer-centric queries serve as compatibility cues for egocentric_frame_tree; local, multi-view, and yes/no relations serve as compatibility cues for object_relation_tree; visibility queries serve as compatibility cues for visibility_occlusion_tree; close top-2 ambiguity serves as a compatibility cue for close_pair_tree; and ordinary relation queries are compatible with public_spatial_tree. Multiple compatible branches may be retained when the query requires cross-branch spatial operations. Each

branch contains candidate trunks $\tau \in B_m$. A trunk specifies a tool chain's invocation order, dependencies, and applicable problem type, rather than a flat tool list.

For each trunk, we record (i) its applicable spatial-analysis type, (ii) its required inputs and preconditions, (iii) its tool-chain structure, (iv) its expected execution evidence, and (v) its call cost and historical performance. The same underlying tool may therefore appear in multiple trunks while serving different high-level patterns through distinct contexts, parameter constraints, and evidence requirements.

During reasoning, the system first extracts the spatial operations required by the current problem and matches them against the capabilities of the tool-forest branches, rather than conducting an unconstrained search over the entire tool library. All compatible branches remain available to the constrained search. This transforms the question of "whether a tool exists" into "which types of spatial analysis best match the current problem," thereby narrowing the search space and making tool selection more relevant.

3.3 Tool Chain Planning Based on MCTS

After query analysis and branch-compatibility matching, STF first retrieves the legal candidate trunks from the compatible branches. Each trunk represents a structured tool-chain plan that specifies the order of spatial operations, tool dependencies, input conditions, and expected execution evidence. Forest-MCTS pre-scores, prunes, and searches these candidates according to the spatial operations required by the query, prioritizing trunks that can fully cover the current task.

For each legal trunk τ, the system computes a pre-execution score

$$r_{\mathrm{pre}}(\tau) = s_{\mathrm{gap}}(\tau) + s_{\mathrm{schema}}(\tau) + 0.20 q_{\mathrm{prior}}(\tau) + 0.10 s_{\mathrm{reliability}}(\tau) - 0.03 \mathbb{I}_{\mathrm{LLM}}(\tau),$$

where s_{gap} measures compatibility with the extracted spatial requirement, s_{schema} evaluates whether the required inputs and parameters are available, q_{prior} summarizes the historical post-execution performance of the trunk, and $s_{\mathrm{reliability}}$ denotes its template reliability. The indicator $\mathbb{I}_{\mathrm{LLM}}$ introduces a small cost when completing the trunk requires LLM-based tool generation. Trunks below the pre-score threshold are pruned; if none survive, the highest-scoring trunks provide a fallback.

Forest-MCTS then applies an upper-confidence rule (Kocsis & Szepesvári, 2006) to allocate the search budget among the retained trunks:

$$\mathrm{UCT}(\tau) = q(\tau) + c\, r_{\mathrm{pre}}(\tau) \sqrt{\frac{\log N}{n(\tau)}},$$

where $q(\tau)$ is the current value estimate, $n(\tau)$ is the visit count of the trunk, N is the total number of visits, and c controls exploration. As planning precedes execution, each planning iteration uses a lightweight pre-execution reward combining the structural pre-execution score with historical execution experience:

$$r(\tau) = 0.70 r_{\mathrm{pre}}(\tau) + 0.30 q_{\mathrm{prior}}(\tau).$$

The pre-execution reward is used to update the value estimate of the selected trunk through the incremental update

$$n(\tau) \leftarrow n(\tau) + 1, \qquad q(\tau) \leftarrow q(\tau) + \frac{r(\tau) - q(\tau)}{n(\tau)}.$$

After the search budget is exhausted, trunks are ranked by updated search values and pre-execution scores; the highest-ranked are retained for program instantiation.

Restricting planning to legal trunks from compatible branches avoids unconstrained selection over the complete tool library while preserving alternative tool-chain structures for the required spatial operation. Appendix B provides implementation-oriented pseudocode for this procedure.

3.4 Constrained LLM-Based Tool Generation and Program Execution

When none of the existing candidate trunks can fully support all spatial operations required by the query, the system does not ask the LLM to regenerate the entire tool chain. Instead, it retains the

candidate trunk that best matches the current task as a scaffold and identifies the necessary operation that cannot be implemented by the available tools. A constrained LLM-based tool-generation module then receives the selected trunk, its existing tool sequence, the uncovered spatial operation, the surrounding input–output interfaces, the outputs of available public tools, and the expected evidence requirements.

Under these constraints, the LLM generates only an adapted tool for the uncovered operation. The generated tool must match the surrounding input and output signatures, may implement only the uncovered spatial operation, and cannot replace existing reconstruction, detection, or ranking tools. It must also expose verifiable evidence that allows the system to assess whether the generated tool improves the reasoning result. Inserting the tool at the appropriate position completes a structurally valid, interface-consistent trunk adapted to the task.

The completed trunk is subsequently instantiated as an executable program $\hat{\tau}$ and run on $(\mathcal{I}, \mathcal{G}, q)$. Execution produces intermediate geometric results, verifiable evidence, an execution trace, and a candidate answer. The resulting trajectory is assigned a post-execution score r_{post} according to execution success, geometric consistency, evidence sufficiency, and candidate discrimination. Trajectories whose scores exceed the writeback threshold are stored in the experience library, and their accumulated statistics update q_{prior} for subsequent Forest-MCTS searches. Thus, pre-execution rewards update trunk values during the current search, while evidence-supported execution results influence future searches through experience writeback.

4 Experiments

4.1 Experimental Settings

Benchmarks. We evaluate STF on three spatial reasoning benchmarks: MINDCUBE-1k (Wang et al., 2026), OMNI3D-BENCH (Marsili et al., 2025), and SPAR-Bench (Zhang et al., 2025a). MINDCUBE-1k contains 1,050 multi-view spatial questions covering rotation, around, and among camera motions, and evaluates spatial mental modeling and reference-frame reasoning under limited observations. OMNI3D-BENCH focuses on single-view 3D spatial reasoning and includes numerical, yes/no, and multiple-choice tasks, allowing us to assess generalization beyond multi-view settings. SPAR-Bench evaluates spatial reasoning across low-, medium-, and high-difficulty subsets, providing a complementary measure of robustness under varying levels of task complexity.

Baselines. We compare STF with four categories of existing baselines: (1) open-weight multi-image MLLMs, including LLaVA-OneVision (Li et al., 2025), Qwen2.5-VL (Bai et al., 2025), DeepSeek-VL2 (Wu et al., 2024), InternVL (Chen et al., 2024b), LLaVA-Video (Zhang et al., 2025b), and mPLUG-Owl3 (Ye et al., 2025); (2) proprietary MLLMs, including GPT-4o (OpenAI, 2024), GPT-4.1 (OpenAI, 2025), and Claude (Anthropic, 2025b); (3) specialized spatial models, including VLM-3R (Fan et al., 2026), Spatial-MLLM (Wu et al., 2025), and RoboBrain (Ji et al., 2025); and (4) visual-programming systems, including ViperGPT (Surís et al., 2023), VisProg (Gupta & Kembhavi, 2023), VADAR (Marsili et al., 2025), and pySpatial (Luo et al., 2026). These baselines cover general-purpose, specialized spatial, and tool-based reasoning.

Implementation Details. By default, we build STF on top of the spatial tools used in pySpatial (Luo et al., 2026). We use GPT-4o (OpenAI, 2024) as the LLM controller for query analysis, tool-branch compatibility matching, MCTS-based trunk planning, constrained missing-tool generation, and final answer generation. For tasks requiring explicit 3D geometry, we use VGGT (Wang et al., 2025a) to obtain camera parameters, depth estimates, and scene-level point clouds. Whereas pySpatial directly generates calls to predefined tools, STF organizes tools into a forest, analyzes the query, and applies MCTS to legal candidates from compatible branches. All methods use the same splits and protocols.

4.2 Quantitative Results

Results on MINDCUBE-1k. Table 1 compares STF with baseline models, spatial mental models, and visual programming approaches on the multi-view MINDCUBE-1k benchmark. Overall, STF achieves the best performance at 57.14% accuracy, 6.38 percentage points above the original pySpatial. The improvement is particularly clear on the around category, where STF increases the accuracy from

Table 1: Results on MINDCUBE-1k. Best and second-best values are bold and underlined, respectively.

Method	Reference	Overall	Rotation	Among	Around
Baseline Models					
Qwen2.5-VL-3B-Instruct	(Bai et al., 2025)	37.81	34.00	36.00	45.20
GPT-4o	(OpenAI, 2024)	42.29	35.00	43.00	46.40
Spatial Mental Models					
Chain-of-Thought	–	40.48	32.00	36.00	<u>58.00</u>
View Interpolation	(Wang et al., 2026)	37.81	35.50	36.50	42.80
Cognitive Map	–	41.43	<u>37.00</u>	41.67	44.40
Visual Programming Approaches					
ViperGPT	(Surís et al., 2023)	36.95	20.50	41.00	40.40
VADAR	(Marsili et al., 2025)	40.76	33.50	40.67	46.80
VADAR w/ Recon.	–	35.62	31.00	36.83	36.40
pySpatial	(Luo et al., 2026)	<u>50.76</u>	35.00	<u>55.67</u>	51.60
STF (Ours)	–	**57.14**	**40.50**	**57.17**	**70.40**

Table 2: Results on OMNI3D-BENCH (MCQ: multiple-choice questions). Best and second-best values are bold and underlined, respectively.

Method	Reference	Total	Num. (ct)	Num. (other)	Y/N	MCQ
Baseline Models						
GPT-4o	(OpenAI, 2024)	**42.90**	**28.10**	**35.50**	**66.70**	57.20
Claude-3.5-Sonnet	(Anthropic, 2024)	32.20	22.40	20.60	62.20	50.60
Llama-3.2	(Meta, 2024)	25.60	24.30	19.30	47.50	27.40
Gemini-1.5-Pro	(Gemini Team, 2024)	32.00	25.20	28.10	46.20	37.60
SpaceMantis	(Chen et al., 2024a; Jiang et al., 2024)	30.30	20.00	21.70	50.60	48.20
Visual Programming Approaches						
ViperGPT	(Surís et al., 2023)	26.70	20.00	15.40	56.00	42.40
VisProg	(Gupta & Kembhavi, 2023)	13.50	2.90	0.90	54.70	25.90
VADAR	(Marsili et al., 2025)	40.40	21.70	**35.50**	56.00	57.60
pySpatial	(Luo et al., 2026)	39.30	25.70	28.80	62.70	<u>62.80</u>
STF (Ours)	–	<u>42.80</u>	<u>27.10</u>	<u>32.80</u>	<u>64.00</u>	**68.60**

51.60% to 70.40%. STF shows a substantial improvement over pySpatial in this category, which requires reasoning about local object configurations and viewpoint-dependent spatial relations. The contribution of individual components is examined further in Section 4.3.

Results on OMNI3D-BENCH. We evaluate STF on OMNI3D-BENCH, a single-view benchmark, to examine whether its tool-organization and selection framework generalizes beyond multi-view settings. Table 2 reports the results following the evaluation protocol of VADAR (Marsili et al., 2025): mean relative accuracy (MRA) is used for the numeric (other) subtask, and standard accuracy is used for the remaining categories. STF achieves the best performance among visual programming approaches, obtaining a total score of 42.8, with gains of 3.5 points over the original pySpatial and 2.4 points over VADAR. The improvement is most evident on multiple-choice questions, where STF improves pySpatial from 62.8 to 68.6. Notably, its overall performance is also competitive with GPT-4o, indicating that structured tool-forest planning can improve tool-based spatial reasoning even in single-view settings where explicit multi-view geometric cues are not available.

Results on SPAR-Bench. Table 3 tests STF's generalization across SPAR-Bench's low-, medium-, and high-difficulty subsets. STF achieves the best average performance, reaching 38.40%, which improves over the original pySpatial by 2.30 percentage points. Compared with pySpatial, our method obtains consistent gains across all difficulty levels, including improvements from 24.75% to 26.50% on low, from 55.33% to 56.67% on medium, and from 39.78% to 42.89% on high. These

Table 3: Results on SPAR-Bench. Best and second-best values are bold and underlined, respectively.

Method	Reference	Avg.	Low	Medium	High
Baselines					
Random	–	32.74	**31.19**	38.25	32.29
API Models					
GPT-4o	(OpenAI, 2024)	<u>36.39</u>	<u>29.25</u>	24.93	**45.11**
Claude-3.7-Sonnet	(Anthropic, 2025a)	21.77	25.43	7.33	23.33
Open-source Models (< 8B)					
Qwen2.5-VL-7B	(Bai et al., 2025)	33.07	28.75	22.97	40.27
InternVL2-8B	(Chen et al., 2024b)	33.02	26.83	36.49	37.47
LLaVA-OV-7B	(Li et al., 2025)	31.20	21.79	26.13	40.14
Visual Programming Approaches					
pySpatial (Original)	(Luo et al., 2026)	36.10	24.75	<u>55.33</u>	39.78
STF (Ours)	–	**38.40**	26.50	**56.67**	<u>42.89</u>

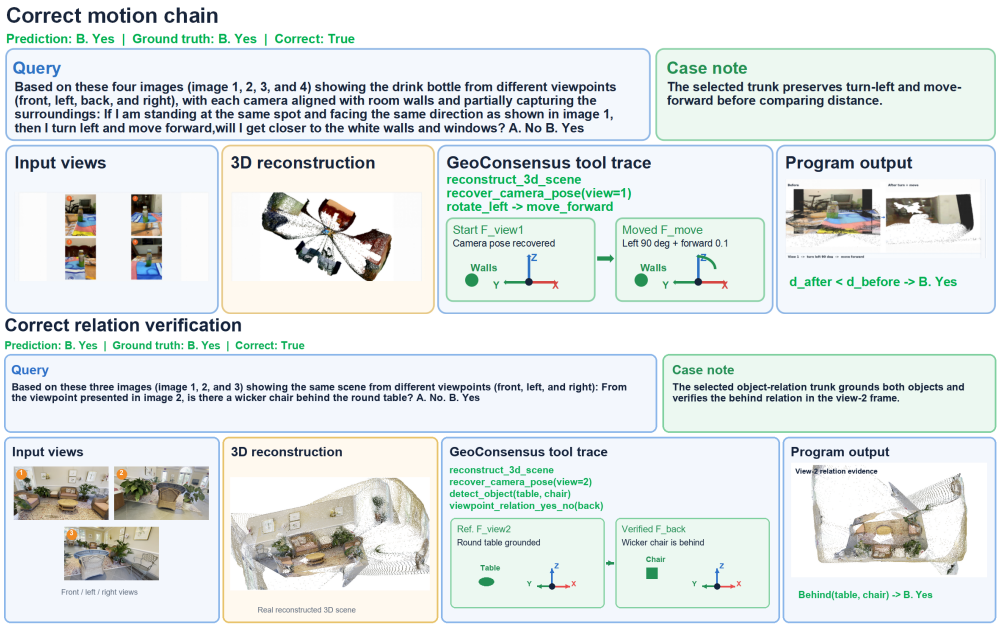


Figure 3: Successful MINDCUBE-1k executions. The motion-chain case preserves operation order and produces verifiable distance evidence; the relation case grounds both objects and verifies the behind relation in view 2. Each example shows input views, the reconstructed scene, the tool trace, and program output. Best viewed when zoomed in.

results indicate that the proposed tool-forest organization and MCTS-based selection can improve the stability of tool-based spatial reasoning across different levels of task complexity.

Qualitative Analysis Figure 3 presents two complementary successful executions. The motion-chain case preserves the required turn-left and move-forward sequence before comparing distance. The relation-verification case grounds the round table and wicker chair, constructs the view-2 reference frame, and verifies the queried behind relation. Additional successful around-relation and rotation examples are provided in Appendix A.3.

4.3 Ablation Study and Discussion

Backbone Generalization. Figure 4 compares GPT-4o, GPT-4.1, and Qwen3-VL-8B-Instruct with their STF-enhanced counterparts. STF improves overall accuracy for all three controllers by 12.38–20.57 percentage points and yields particularly large gains on the around category (20.80–

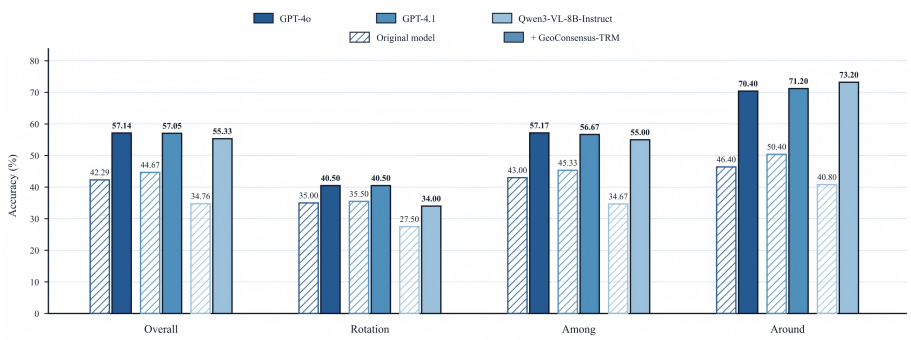


Figure 4: Backbone generalization on MINDCUBE-1k: overall and category accuracy for GPT-4o, GPT-4.1, and Qwen3-VL-8B-Instruct, with and without STF.

Table 4: Ablation results on MINDCUBE-1k using Full STF as the shared reference. Bold values indicate column maxima within each panel, including ties.

Method	Overall	Rotation	Among	Around
(a) Component ablation				
Extended Spatial Tools	54.29	34.00	56.33	65.60
MCTS-based Trunk Planning	56.48	39.50	56.67	69.60
Constrained LLM Tool Completion	57.05	**40.50**	56.67	**71.20**
Full STF	**57.14**	**40.50**	**57.17**	70.40
(b) Tool organization controls				
Predefined Tools	54.95	40.00	55.83	64.80
Tool Trees	55.05	32.50	56.50	69.60
Unconstrained LLM-Tool Forest	54.00	32.50	55.00	68.80
Full STF	**57.14**	**40.50**	**57.17**	**70.40**

32.40 points). The strongest enhanced controller is category-dependent: GPT-4o achieves the highest overall accuracy and the highest accuracy in the among category, whereas Qwen3-VL-8B-Instruct achieves the highest accuracy in the around category. This result shows that the framework generalizes across proprietary and open-weight controllers while remaining sensitive to their underlying spatial reasoning profiles.

Component Ablation. Table 4(a) quantifies the contribution of planning and constrained completion over extended spatial tools. Full STF achieves the highest overall and among accuracy and ties for the highest rotation accuracy, while constrained completion gives the strongest around result. This pattern suggests that broader tool coverage benefits from compatibility-based search. The around result also highlights the need for task-appropriate chains in local reference-frame reasoning. Appendix A.2 reports detailed planner controls.

Tool Organization Controls. Table 4(b) exposes a category trade-off: tool trees improve around accuracy but reduce rotation accuracy relative to predefined tools, with little overall change. Unconstrained generation also performs worse overall. Thus, a larger or merely grouped candidate set is insufficient. The pattern motivates compatibility-constrained selection rather than flat expansion. Full STF leads in all categories, although the controls do not isolate the effects of organization, planning, completion, and evidence-aware selection.

5 Conclusion

In this work, we present STF, a hierarchical tool-forest framework combining constrained MCTS planning with controlled missing-tool completion. Across MINDCUBE-1k, OMNI3D-BENCH, and SPAR-Bench, STF consistently improves over pySpatial, including a 6.38 percentage-point gain on MINDCUBE-1k and a substantial improvement on around questions. These results support structured tool organization and constrained planning for tool-based spatial reasoning.

AI USE STATEMENT

AI tools were used to assist with programming, improve the clarity and fluency of the manuscript, and support literature search and background research. They were not used to generate the core research ideas or to conduct experiments in a fully autonomous manner. The authors take full responsibility for the implementation, experimental results, references, and claims presented in this paper.

REPRODUCIBILITY STATEMENT

The main paper specifies the task, tool forest, MCTS planning, controlled completion, and evidence writeback, and reports backbone, component, and tool-organization comparisons. Appendix A separates full-MINDCUBE and MINDCUBE-1k results and details the planning controls. Appendices B–D document runtime defaults, the Forest-MCTS implementation, evidence writeback, the public spatial-tool interface, and the controlled generation prompt. Experiments use the public MINDCUBE, OMNI3D-BENCH, and SPAR-Bench protocols; comparisons within each table use the same split and metric.

REFERENCES

Anthropic. Claude 3.5 sonnet. Anthropic technical announcement, 2024. URL https://www.anthropic.com/news/claude-3-5-sonnet.

Anthropic. Claude 3.7 sonnet and claude code. Anthropic technical announcement, 2025a. URL https://www.anthropic.com/news/claude-3-7-sonnet.

Anthropic. Claude 4 system card. Technical report, 2025b. URL https://www-cdn.anthropic.com/6be99a52cb68eb70eb9572b4cafad13df32ed995.pdf.

Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, et al. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025. URL https://arxiv.org/abs/2502.13923.

Boyuan Chen, Zhuo Xu, Sean Kirmani, Brian Ichter, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14455–14465, 2024a. URL https://openaccess.thecvf.com/content/CVPR2024/html/Chen_SpatialVLM_Endowing_Vision-Language_Models_with_Spatial_Reasoning_Capabilities_CVPR_2024_paper.html.

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*, pp. 24185–24198, 2024b. URL https://openaccess.thecvf.com/content/CVPR2024/html/Chen_InternVL_Scaling_up_Vision_Foundation_Models_and_Aligning_for_Generic_CVPR_2024_paper.html.

An-Chieh Cheng, Hongxu Yin, Yang Fu, Qiushan Guo, Ruihan Yang, Jan Kautz, Xiaolong Wang, and Sifei Liu. Spatialrgpt: Grounded spatial reasoning in vision-language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/f38cb4cf9a5eaa92b3cfa481832719c6-Abstract-Conference.html.

Zhiwen Fan, Jian Zhang, Renjie Li, Junge Zhang, Runjin Chen, Hezhen Hu, Kevin Wang, Peihao Wang, Huaizhi Qu, Shijie Zhou, Dilin Wang, Zhicheng Yan, Hongyu Xu, Justin Theiss, Tianlong Chen, Jiachen Li, Zhengzhong Tu, Zhangyang Wang, and Rakesh Ranjan. Vlm-3r: Vision-language models augmented with instruction-aligned 3d reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 31054–31065, 2026. URL https://openaccess.thecvf.com/content/CVPR2026/html/Fan_VLM-3R_Vision-Language_Models_Augmented_with_Instruction-Aligned_3D_Reconstruction_CVPR_2026_paper.html.

Gemini Team. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024. URL https://arxiv.org/abs/2403.05530.

Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14953–14962, 2023. URL https://openaccess.thecvf.com/content/CVPR2023/html/Gupta_Visual_Programming_Compositional_Visual_Reasoning_Without_Training_CVPR_2023_paper.html.

Yuheng Ji, Huajie Tan, Jiayu Shi, Xiaoshuai Hao, Yuan Zhang, Hengyuan Zhang, Pengwei Wang, Mengdi Zhao, Yao Mu, Pengju An, et al. Robobrain: A unified brain model for robotic manipulation from abstract to concrete. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1724–1734, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Ji_RoboBrain_A_Unified_Brain_Model_for_Robotic_Manipulation_from_Abstract_CVPR_2025_paper.html.

Dongfu Jiang, Xuan He, Huaye Zeng, Cong Wei, Max Ku, Qian Liu, and Wenhu Chen. Mantis: Interleaved multi-image instruction tuning. *Transactions on Machine Learning Research*, 2024. URL https://openreview.net/forum?id=skLtdUVaJa.

Amita Kamath, Jack Hessel, and Kai-Wei Chang. What's "up" with vision-language models? investigating their struggle with spatial reasoning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 9161–9175, 2023. doi: 10.18653/v1/2023.emnlp-main.568. URL https://aclanthology.org/2023.emnlp-main.568/.

Levente Kocsis and Csaba Szepesvári. Bandit based monte-carlo planning. In *Machine Learning: ECML 2006*, pp. 282–293, 2006. doi: 10.1007/11871842_29. URL https://doi.org/10.1007/11871842_29.

Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *European Conference on Computer Vision*, 2024. URL https://eccv.ecva.net/virtual/2024/poster/523.

Bo Li, Yuanhan Zhang, Dong Guo, Renrui Zhang, Feng Li, Hao Zhang, Kaichen Zhang, Peiyuan Zhang, Yanwei Li, Ziwei Liu, and Chunyuan Li. Llava-onevision: Easy visual task transfer. *Transactions on Machine Learning Research*, 2025. URL https://openreview.net/forum?id=zKv8qULV6n.

Fangyu Liu, Guy Emerson, and Nigel Collier. Visual spatial reasoning. *Transactions of the Association for Computational Linguistics*, 11:635–651, 2023. doi: 10.1162/tacl_a_00566. URL https://aclanthology.org/2023.tacl-1.37/.

Zhanpeng Luo, Ce Zhang, Silong Yong, Cunxi Dai, Qianwei Wang, Haoxi Ran, Guanya Shi, Katia Sycara, and Yaqi Xie. pyspatial: Generating 3d visual programs for zero-shot spatial reasoning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=yv15C8ql24.

Chenyang Ma, Kai Lu, Ta-Ying Cheng, Niki Trigoni, and Andrew Markham. Spatialpin: Enhancing spatial reasoning capabilities of vision-language models through prompting and interacting 3d priors. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/7f2257d2b291b8d7e712c70b67e09412-Abstract-Conference.html.

Xiaojian Ma, Silong Yong, Zilong Zheng, Qing Li, Yitao Liang, Song-Chun Zhu, and Siyuan Huang. Sqa3d: Situated question answering in 3d scenes. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=IDJx97BC38.

Damiano Marsili, Rohun Agrawal, Yisong Yue, and Georgia Gkioxari. Visual agentic ai for spatial reasoning with a dynamic api. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19446–19455, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Marsili_Visual_Agentic_AI_for_Spatial_Reasoning_with_a_Dynamic_API_CVPR_2025_paper.html.

Meta. Llama 3.2: Revolutionizing edge ai and vision with open, customizable models. Meta AI technical announcement, 2024. URL https://ai.meta.com/blog/llama-3-2-connect-2024-vision-edge-mobile-devices/.

OpenAI. Hello gpt-4o. OpenAI technical announcement, 2024. URL https://openai.com/index/hello-gpt-4o/.

OpenAI. Introducing gpt-4.1 in the api. OpenAI technical announcement, 2025. URL https://openai.com/index/gpt-4-1/.

Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4104–4113, 2016. URL https://openaccess.thecvf.com/content_cvpr_2016/html/Schonberger_Structure-From-Motion_Revisited_CVPR_2016_paper.html.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. Hugginggpt: Solving ai tasks with chatgpt and its friends in hugging face. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/77c33e6a367922d003ff102ffb92b658-Abstract-Conference.html.

Dídac Surís, Sachit Menon, and Carl Vondrick. Vipergpt: Visual inference via python execution for reasoning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 11888–11898, 2023. URL https://openaccess.thecvf.com/content/ICCV2023/html/Suris_ViperGPT_Visual_Inference_via_Python_Execution_for_Reasoning_ICCV_2023_paper.html.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5294–5306, 2025a. URL https://openaccess.thecvf.com/content/CVPR2025/html/Wang_VGGT_Visual_Geometry_Grounded_Transformer_CVPR_2025_paper.html.

Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10510–10522, 2025b. URL https://openaccess.thecvf.com/content/CVPR2025/html/Wang_Continuous_3D_Perception_Model_with_Persistent_State_CVPR_2025_paper.html.

Qineng Wang, Baiqiao Yin, Pingyue Zhang, Jianshu Zhang, Kangrui Wang, Zihan Wang, Jieyu Zhang, Keshigeyan Chandrasegaran, Han Liu, Ranjay Krishna, Saining Xie, Jiajun Wu, Li Fei-Fei, and Manling Li. Spatial mental modeling from limited views. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/858de961874816f894285ec02e50c7fe-Abstract-Conference.html.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jérôme Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20697–20709, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Wang_DUSt3R_Geometric_3D_Vision_Made_Easy_CVPR_2024_paper.html.

Chenfei Wu, Shengming Yin, Weizhen Qi, Xiaodong Wang, Zecheng Tang, and Nan Duan. Visual chatgpt: Talking, drawing and editing with visual foundation models. *arXiv preprint arXiv:2303.04671*, 2023. URL https://arxiv.org/abs/2303.04671.

Diankun Wu, Fangfu Liu, Yi-Hsin Hung, and Yueqi Duan. Spatial-mllm: Boosting mllm capabilities in visual-based spatial intelligence. In *Advances in Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=RnXS7aK4rK.

Zhiyu Wu, Xiaokang Chen, Zizheng Pan, Xingchao Liu, Wen Liu, Damai Dai, Huazuo Gao, Yiyang Ma, Chengyue Wu, Bingxuan Wang, et al. Deepseek-vl2: Mixture-of-experts vision-language models for advanced multimodal understanding. *arXiv preprint arXiv:2412.10302*, 2024. URL https://arxiv.org/abs/2412.10302.

Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Ehsan Azarnasab, Faisal Ahmed, Zicheng Liu, Ce Liu, Michael Zeng, and Lijuan Wang. Mm-react: Prompting chatgpt for multimodal reasoning and action. *arXiv preprint arXiv:2303.11381*, 2023. URL https://arxiv.org/abs/2303.11381.

Shunyu Yao, Dian Yu, Jeffrey Zhao, Izhak Shafran, Thomas L. Griffiths, Yuan Cao, and Karthik Narasimhan. Tree of thoughts: Deliberate problem solving with large language models. In *Advances in Neural Information Processing Systems*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/271db9922b8d1f4dd7aaef84ed5ac703-Abstract.html.

Jiabo Ye, Haiyang Xu, Haowei Liu, Anwen Hu, Ming Yan, Qi Qian, Ji Zhang, Fei Huang, and Jingren Zhou. mplug-owl3: Towards long image-sequence understanding in multi-modal large language models. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=pr37sbuhVa.

Jiahui Zhang, Yurui Chen, Yueming Xu, Ze Huang, Jilin Mei, Chunhui Chen, Yanpeng Zhou, Yu-Jie Yuan, Xinyue Cai, Guowei Huang, Xingyue Quan, Hang Xu, and Li Zhang. From flatland to space: Teaching vision-language models to perceive and reason in 3d. In *Advances in Neural Information Processing Systems*, volume 38, 2025a. doi: 10.52202/085713-5180. URL https://papers.neurips.cc/paper_files/paper/2025/hash/e3a0db7c0a191854c176af1d20cdec80-Abstract-Datasets_and_Benchmarks_Track.html. Datasets and Benchmarks Track.

Yuanhan Zhang, Jinming Wu, Wei Li, Bo Li, Zejun Ma, Ziwei Liu, and Chunyuan Li. Llava-video: Video instruction tuning with synthetic data. *Transactions on Machine Learning Research*, 2025b. URL https://openreview.net/forum?id=EElFGvt39K.