T-SPATIAL: LEARNING UTILITY AWARE TOOL-USE FOR SPATIAL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

External tools are increasingly used to improve the performance of vision-language models (VLMs) on spatial reasoning tasks. However, executing a tool call successfully does not guarantee better performance. Inappropriate tool calls may even harm reasoning. Existing credit-assignment methods typically assign a single final-answer reward to an entire multi-turn trajectory, providing limited guidance on whether an individual call is beneficial, harmful, or redundant. This coarse-grained credit assignment makes it difficult to further improve the utility of tool use in long, multi-turn interactions. To address this challenge, we propose the Tool-Augmented Spatial Reasoning Agent (T-Spatial), trained with Difficulty-Aware Trajectory Synthesis (DATS) for supervised fine-tuning (SFT) and Tool Utility-based Policy Optimization (TUPO) for reinforcement learning (RL). DATS evaluates the target VLM on each training question and assigns a difficulty label based on its performance. According to these labels, DATS constructs a tool-assisted or a direct-answer trajectory for each question and teaches the target model to adaptively use spatial tools according to the difficulty of the question. TUPO combines sample-, trajectory-, branch-, and node-level tool-utility signals that are mainly computed from differences in answer scores across branches or before and after tool calls, allowing TUPO to reuse the existing answer evaluator without an auxiliary reward model. These complementary signals reduce the reward sparsity and make tool utility explicit at multiple granularity. Experiments on multiple spatial reasoning benchmarks show that T-Spatial improves spatial reasoning accuracy and tool-use utility.

1 INTRODUCTION

Specialized spatial tools can complement the limited spatial evidence in the internal representation of a vision-language model (VLM). Prior studies have shown that tools such as 3D reconstruction and object detection can improve the VLM performance on spatial reasoning tasks(Wu et al., 2026; Dai et al., 2026; Zhang et al., 2026). However, executing a tool call successfully does not guarantee better reasoning. Incorrect arguments or misinterpretation of the returned evidence may make a call unhelpful or even harmful. Too many unnecessary calls can also introduce redundant evidence and reduce answer accuracy(Guo et al., 2026). The key challenge is therefore to train the model to call tools when needed and increase the utility of tool use.

Constructing high-quality supervised fine-tuning (SFT) data requires considering both final-answer correctness and whether tool assistance is appropriate. A trajectory with a correct final answer may still contain calls that contribute little to the solution, allowing redundant tool-use behavior to enter the training data (Wang et al., 2026b; Xiang et al., 2026). Existing approaches often use a stronger model to generate tool-use trajectories for a weaker target model. However, the stronger model may either call tools by default or rely mainly on its own reasoning ability. The former introduces redundant tool calls, while the latter produces trajectories whose success may be difficult for the target model to reproduce.

In reinforcement learning (RL), a final-answer reward reflects the overall outcome of a multi-turn trajectory but provides limited information about the contribution of each tool call. Figure 1 shows the difficulty of distinguishing beneficial and harmful tool calls using only the final-answer reward. Penalizing the number of calls can encourage efficiency, but it does not distinguish necessary attempts

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.

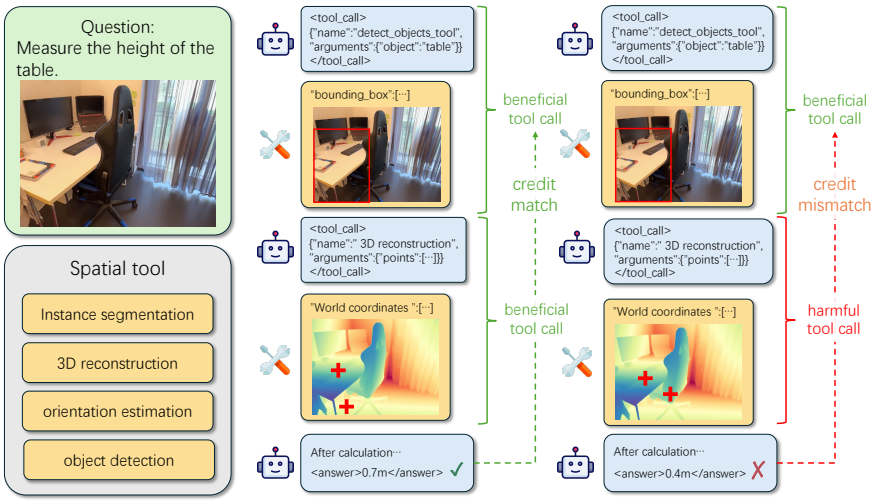

Figure 1: An example in which coarse-grained credit assignment cannot distinguish beneficial from harmful tool calls.

from redundant ones. This coarse-grained credit assignment makes it difficult to further improve the utility of tool use in long, multi-turn interactions. Recent studies have explored necessity-aware rewards, tool-output assessment, answer probing, and tree-based credit assignment as complementary ways to improve tool-use supervision (Hou et al., 2026; Feng et al., 2026; Ji et al., 2026; Wu et al., 2025b). These observations motivate a joint treatment of training-data construction and multi-level credit assignment.

To address these challenges, we introduce T-Spatial, trained through an SFT-then-RL pipeline that coordinates Difficulty-Aware Trajectory Synthesis (DATS) with Tool Utility-based Policy Optimization (TUPO). DATS assigns model-relative difficulty labels to training questions based on the target VLM's performance. According to these labels, DATS constructs a tool-assisted or a direct-answer trajectory for each question and teaches the target model to adaptively use spatial tools according to the difficulty of the question, reducing unnecessary tool calls.

During RL, T-Spatial applies TUPO, which extends Group Relative Policy Optimization (GRPO) (Shao et al., 2024) with tool-utility signals at the sample, trajectory, branch, and node levels. These signals respectively allocate rollout budgets, scale final-answer rewards, aggregate downstream advantages, and provide tool-use rewards based on answer changes after tool use. This multi-level design turns a sparse final-answer reward into fine-grained credit assignment, helping the model favor beneficial tool use while avoiding harmful or redundant behavior. TUPO constructs these signals mainly from final-answer evaluation and tool-use statistics. This allows it to reuse the existing answer evaluator without an auxiliary learned reward model.

Our contributions are threefold:

- Difficulty-Aware Trajectory Synthesis (DATS), which assigns difficulty labels based on the target VLM's performance and constructs training data with difficulty-adaptive tool-use, reducing unnecessary tool calls at the data construction stage.
- Tool Utility-based Policy Optimization (TUPO), an RL method that improves credit assignment with coarse-to-fine tool-utility signals at the sample, trajectory, branch, and node levels without an auxiliary learned reward model.
- Extensive experiments with T-Spatial-4B and T-Spatial-8B, trained from Qwen3-VL-4B and Qwen3-VL-8B (Bai et al., 2025), demonstrate improvements in spatial reasoning accuracy and tool-use utility across diverse spatial reasoning benchmarks.

2 Related Work

2.1 Spatial agents

Spatial agents augment reasoning with explicit evidence acquisition. SpatialAgent combines specialized spatial perception tools with Plan-Execute and ReAct inference (Wu et al., 2026); pySpatial generates executable programs for reconstruction, camera recovery, and novel-view reasoning (Luo et al., 2026). MSSR explicitly pursues a minimal sufficient set of spatial evidence through iterative acquisition and pruning (Guo et al., 2026). S-Agent develops hierarchical spatial experts and temporal memory, and transfers generated trajectories to a compact model through SFT (Dai et al., 2026).

Several spatial agents also learn their interaction policies. Think3D-RL trains active 3D exploration using final-answer rewards (Zhang et al., 2026). Skill-3D uses scene-aware skills and SFT followed by GRPO, including correctness, format, and tool-efficiency feedback (Li et al., 2026b). The house-scale SpatialReasoner of Zheng et al. (2025) uses SFT and exploration-oriented RL rewards to discourage redundant operations. SpatialCLI combines cold-start SFT, live-tool RL, and subsequent internalization of tool-assisted reasoning (Zhou et al., 2026). These works establish both spatial tool augmentation and efficient interaction learning. T-Spatial pairs a data synthesis pipeline for trajectories with or without tools with multi-level RL signals, coupling sample-level rollout-budget allocation to descendant returns and answer probes at intermediate decisions. It does not require a cross-question skill library or a separate tool-internalization stage.

2.2 Visual agents

Visual programming systems such as ViperGPT compose perception modules through executable code (Surís et al., 2023). More recent approaches optimize the interaction policy: OpenThinkIMG provides a visual tool-use framework with SFT initialization and reinforcement learning (Su et al., 2025), whereas DeepEyes learns active visual perception through RL without a cold-start SFT dataset (Zheng et al., 2026). Beyond answer correctness, CodeV evaluates the relevance of intermediate tool outputs with a VLM judge, aggregates these scores into a trajectory reward, and broadcasts its advantage across the trajectory (Hou et al., 2026). OpenVisTool instead verifies the usefulness of tool observations during offline trajectory selection (Xiang et al., 2026). Beacon combines direct and code-assisted SFT examples with group-dependent necessity rewards and hint-assisted exploration (Wang et al., 2026b). Thus, mixed-mode supervision and selective tool-use are not new in isolation.

TACO introduces differential answer probes and outcome-gated advantage routing for visual tool branches (Feng et al., 2026). Tree-GRPO (Ji et al., 2026) and PORTool (Wu et al., 2025b) establish tree-based credit assignment for general tool-using agents. Our distinction is the joint use of sample-conditioned exploration, spatial interaction trees, and evaluator-based node signals, rather than the introduction of answer probes or tree credit assignment alone.

2.3 Spatially specialized models

A parallel line develops spatial competence through data, representations, and model post-training. SpatialVLM constructs large-scale spatial question–answer supervision (Chen et al., 2024), while SpatialRGPT combines region-level training with a depth-aware visual interface (Cheng et al., 2024). Spatial-MLLM fuses semantic and geometric encoders and uses SFT and GRPO to strengthen visual-based spatial reasoning (Wu et al., 2025a). The explicit-representation SpatialReasoner of Ma et al. (2025b) links 3D perception, computation, and reasoning through a shared representation. Integrating a geometry encoder or depth input into a VLM is distinct from learning when to issue external tool requests.

Training-based approaches further develop internal spatial reasoning. SpaceR uses map imagination with verifiable-reward RL (Ouyang et al., 2025), and MindCube studies map-then-reason training with subsequent RL (Wang et al., 2026a). SpatialLadder progresses from localization to spatial understanding and reasoning (Li et al., 2026c); Visual Spatial Tuning similarly combines perception-oriented SFT with spatial RL (Yang et al., 2026a). Cambrian-S (Yang et al., 2026b) and SenseNova-SI (Cai et al., 2026) study spatial capabilities through large-scale data and training.

These approaches improve the model's spatial representations and intrinsic competence. T-Spatial addresses the complementary question of how a policy should acquire and use external spatial evidence when its existing competence is insufficient.

3 METHOD

We first describe how T-Spatial calls spatial tools and incorporates the returned evidence into reasoning. We then introduce Difficulty-Aware Trajectory Synthesis (DATS) for constructing training data with difficulty-adaptive tool-use, followed by Tool Utility-based Policy Optimization (TUPO) for utility-aware credit assignment to spatial tool calls.

3.1 OVERVIEW

Let x denote a spatial reasoning problem and $\mathcal{U} = \{T_1, \ldots, T_K\}$ the set of available spatial tools, which provide capabilities such as 3D reconstruction, object detection, and orientation estimation. At interaction turn t, T-Spatial maintains an interaction state s_t and an external memory M_t. The state s_t contains the original problem x, the system prompt, and the remaining tool-call budget. Separately, M_t records and organizes the spatial evidence returned by previous tool calls, with $M_0 = \varnothing$.

For concise notation, π_θ represents the process of VLM generation followed by rule-based parsing. At interaction turn t, the policy generates an answer or a tool call based on the current interaction state s_t and memory M_t:

$$a_t \sim \pi_\theta(\cdot \mid s_t, M_t), \qquad a_t \in \{\mathrm{Answer}, \mathrm{ToolCall}\}\,. \tag{1}$$

An Answer action contains the final answer, while a $\mathrm{ToolCall}$ action contains a selected tool in $\mathcal{U}$ and its arguments. Let $\mathrm{Execute}$ denote tool execution and let Update denote the joint update of the interaction state and external memory. When the policy produces a valid tool call, the transition is

$$(s_{t+1}, M_{t+1}) = \mathrm{Update}\left(s_t, M_t, \mathrm{Execute}(a_t; x, \mathcal{U})\right), \quad \text{if } a_t = \mathrm{ToolCall}\,. \tag{2}$$

The returned tool result is incorporated into M_{t+1}, while s_{t+1} records the updated tool-call budget for the next turn. The process continues until the agent returns a final answer or exhausts its tool-call budget.

3.2 TRAINING DATA CONSTRUCTION

We propose Difficulty-Aware Trajectory Synthesis (DATS) as the training-data construction pipeline for T-Spatial. DATS builds a training question pool, assigns difficulty labels, constructs framework-compliant SFT trajectories, and prepares labeled questions for RL.

We construct the initial pool from two open-source spatial reasoning datasets. Questions are sampled according to their original task categories to preserve distribution of different spatial reasoning skills. Let π_{base} denote the base model and π_{syn} denote the data synthesis model equipped with the tool-interaction framework in Section 3.1. For each question x, we first obtain a tool-free reasoning trajectory from π_{base} and retain it if it is correct. Otherwise, π_{syn} generates a new tool-assisted trajectory for the same question.

Let $z_x^{\mathrm{direct}} \in \{0, 1\}$ indicate whether the direct answer from π_{base} is correct. Let $z_x^{\mathrm{tool}} \in \{0, 1\}$ indicate whether π_{syn} produces at least one correct tool-assisted solution. DATS assigns three difficulty labels:

$$\ell_x = \begin{cases} \textsc{Easy-Direct}, & z_x^{\mathrm{direct}} = 1, \\ \textsc{Easy-Tool}, & z_x^{\mathrm{direct}} = 0 \ \wedge \ z_x^{\mathrm{tool}} = 1, \\ \textsc{Hard}, & z_x^{\mathrm{direct}} = 0 \ \wedge \ z_x^{\mathrm{tool}} = 0. \end{cases} \tag{3}$$

For SFT, DATS retains correct tool-free trajectories for EASY-DIRECT questions and correct tool-assisted trajectories for EASY-TOOL questions. All retained trajectories are converted to the

framework-compliant interaction format. HARD questions are excluded because no correct trajectory is obtained during synthesis. DATS further removes trajectories that violate the interaction format, exceed the length limit, or contain invalid or repeated tool calls.

RL examples follow the same sampling and difficulty-labeling process but do not include reference trajectories. Each example retains only the original input, ground-truth answer, and difficulty label.

3.3 TOOL UTILITY-BASED POLICY OPTIMIZATION

Figure 2 provides an overview of TUPO. TUPO organizes tool-utility signals at four levels: the DATS difficulty label provides sample-level guidance for reward scaling and rollout allocation; the trajectory-level signal combines final-answer correctness with tool-use cost; the branch-level signal aggregates descendant leaf advantages; and the node-level signal measures answer correctness changes before and after tool use. These four levels form a coarse-to-fine credit-assignment process, rather than being combined as independent reward terms.

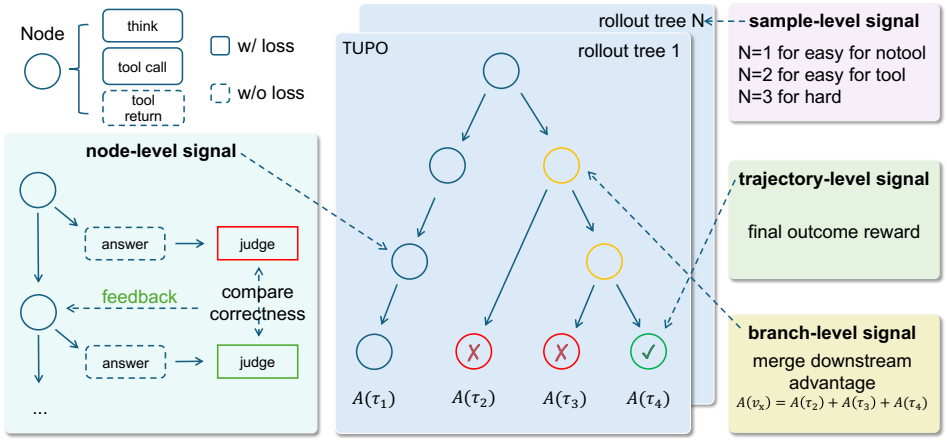

Figure 2: Overview of Tool Utility-based Policy Optimization.

3.3.1 TREE ROLLOUTS

For each question, we first sample a *trunk trajectory* and randomly select several non-terminal states along it. At each selected state, we retain the preceding interaction history as a shared prefix, sample an alternative policy response, and independently roll out the policy until termination. The shared prefix and new continuation form a *branch trajectory*. A node v represents one interaction turn, including the model's reasoning and either a tool call with its returned result or a final answer. Both trunk and branch trajectories are complete root-to-leaf paths, denoted by τ. Let $\mathcal{T}_v$ denote the set of trajectories that contain node v.

3.3.2 SAMPLE- AND TRAJECTORY-LEVEL SIGNALS

The sample-level signal comes from the DATS difficulty label, which records whether a question can be solved directly, solved with tools, or is not solved during data synthesis. This signal is used to adjust the reward for unnecessary tool use and allocate the rollout budget. The trajectory-level signal scales the final-answer correctness reward using a tool-use cost $c_{\text{tool}}(\tau) = \min(0.05,\, 0.01 N_{\text{call}}(\tau))$, where $N_{\text{call}}(\tau)$ is the number of tool calls in trajectory τ. Let $r_{\text{acc}}(\tau) \in [0, 1]$ denote the correctness reward assigned by the answer evaluator. The reward of the trajectory is defined as

$$R_{\text{traj}}(x, \tau) = \begin{cases} [0.25 - c_{\text{tool}}(\tau)]\, r_{\text{acc}}(\tau), & \ell_x = \text{EASY-DIRECT} \ \wedge \ N_{\text{call}}(\tau) > 0, \\ [1 - c_{\text{tool}}(\tau)]\, r_{\text{acc}}(\tau), & \text{otherwise.} \end{cases} \tag{4}$$

The sample-level signal also controls rollout allocation. We sample one tree for each EASY-DIRECT question, two trees for each EASY-TOOL question, and two or three trees for each HARD question to match the total rollout count of standard GRPO. Each tree contains four leaf trajectories.

3.3.3 BRANCH- AND NODE-LEVEL SIGNALS

The branch-level signal evaluates whether an intermediate decision can lead to a successful final outcome, while the node-level signal measures the change in answer quality observed across a tool-use transition.

Each TUPO rollout tree contains one trunk trajectory and three branch trajectories. We flatten all trees for the same sample x into complete root-to-leaf trajectories and treat them as a single GRPO group $\mathcal{G}_x$. The leaf-level advantage is obtained by normalizing the trajectory rewards within this group:

$$A_{\text{leaf}}(\tau_i) = \frac{R_{\text{traj}}(x, \tau_i) - \text{mean}_{\tau_j \in \mathcal{G}_x} R_{\text{traj}}(x, \tau_j)}{\text{std}_{\tau_j \in \mathcal{G}_x} R_{\text{traj}}(x, \tau_j)}. \tag{5}$$

For each node v, TUPO aggregates the normalized advantages of its descendant trajectories:

$$A_{\text{branch}}(v) = \sum_{\tau_i \in \mathcal{T}_v} A_{\text{leaf}}(\tau_i). \tag{6}$$

To obtain the node-level signal, we append `<answer>` to the context at the start of each interaction turn, after receiving the tool results from the previous turn, and greedily decode a probe answer without further tool calls. The existing answer evaluator assigns each probe answer a correctness score $r_{\text{acc}}(s_{i,t}) \in [0, 1]$. For tool-use turn t, the current-turn probe provides the before-tool score, while the next-turn probe provides the after-tool score. Each probe is therefore reused across two adjacent turns. An increase in correctness indicates a positive effect of tool-use, while a decrease indicates a negative effect. This provides fine-grained tool-utility signals without an auxiliary learned reward model. The probe reward is defined as

$$\Delta R_{\text{probe}}(i, t) = r_{\text{acc}}(s_{i,t+1}) - r_{\text{acc}}(s_{i,t}). \tag{7}$$

3.3.4 TUPO OPTIMIZATION

At the sample and trajectory levels, the DATS difficulty label and tool-use cost scale the final-answer correctness reward of each complete trajectory. GRPO normalization converts these rewards into leaf advantages, which are aggregated over descendant trajectories to obtain the branch-level advantage. The local probe reward estimates the change in answer correctness after tool use. Let $R_{\text{format}}(i, t) \in [0, 1]$ denote a rule-based reward for following the required response format. TUPO combines these terms to construct the advantage of node $v_{i,t}$:

$$A_{\text{node}}(i, t) = 0.6 A_{\text{branch}}(v_{i,t}) + 0.3 \Delta R_{\text{probe}}(i, t) + 0.1 R_{\text{format}}(i, t). \tag{8}$$

The same node-level advantage is assigned to all trainable policy tokens in $v_{i,t}$. Let $\rho_{i,t,k}(\theta)$ denote the probability ratio between the current policy π_θ and the rollout policy π_{old} for the k-th token of this node, and let ϵ_{clip} denote the clipping threshold. TUPO updates the policy by minimizing the following clipped GRPO loss:

$$\mathcal{L}_{\text{TUPO}}(\theta) = -\mathbb{E}_{i,t,k} \left[\min \left(\rho_{i,t,k}(\theta) A_{\text{node}}(i, t), \text{clip}(\rho_{i,t,k}(\theta), 1 - \epsilon_{\text{clip}}, 1 + \epsilon_{\text{clip}}) A_{\text{node}}(i, t) \right) \right]. \tag{9}$$

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Training settings. We train T-Spatial in two stages using SFT followed by TUPO. Both stages update all LLM parameters, while the vision encoder and multimodal aligner remain frozen.

SFT uses AdamW with a learning rate of 5×10^{-5} for two epochs, an effective batch size of 16, cosine decay, 5% warm-up, and a maximum sequence length of 65,536 tokens. The loss is applied only to policy-generated tokens, while system prompts, visual inputs, and tool outputs are masked. TUPO starts from the SFT checkpoint with a learning rate of 5×10^{-6} and a rollout temperature of 1.0. Each rollout tree contains four leaf trajectories, with question-dependent budgets as described in Section 3.3.2. Answer probes use greedy decoding. Both stages use BF16. Both models are trained on eight NVIDIA H800 GPUs, taking approximately 4.1 days for T-Spatial-8B and 2.7 days for T-Spatial-4B.

Table 1: Overall performance. Avg. is the arithmetic mean over all six benchmarks. Bold numbers indicate the best results among models with fewer than 10B parameters.

Method	VSI-Bench	MMSI-Bench	3DSR Bench	View Spatial	CV-Bench	MindCube	Avg.
Base models							
Gemini 3 Pro (Google, 2025)	52.5	45.2	68.9	50.4	85.9	70.9	62.3
GPT-5.2 (OpenAI, 2025)	54.5	41.3	60.2	47.3	84.2	60.4	58.0
Qwen3.5-9B (Qwen Team, 2026)	54.3	38.7	56.8	48.2	85.8	60.1	57.3
Qwen3-VL-4B-Instruct (Bai et al., 2025)	53.0	28.1	60.6	36.7	84.3	25.2	48.0
Qwen3-VL-4B-Thinking (Bai et al., 2025)	52.1	27.7	61.9	44.6	83.9	37.0	51.2
Qwen3-VL-8B-Instruct (Bai et al., 2025)	55.3	31.1	53.9	42.2	85.1	29.4	49.5
Qwen3-VL-8B-Thinking (Bai et al., 2025)	54.3	28.6	54.5	47.3	85.0	43.2	52.2
Specialized spatial models							
SpatialLadder-3B (Li et al., 2026c)	44.9	27.4	42.8	39.9	73.7	43.5	45.4
VST-7B (Yang et al., 2026a)	**61.2**	34.8	61.0	50.6	**86.5**	46.6	56.8
SpatialCorpus-8B (Wu et al., 2026)	54.3	32.6	62.4	45.1	81.8	54.8	55.2
Tool-augmented models							
Think3D-4B (Zhang et al., 2026)	45.3	27.0	59.5	43.5	80.7	46.0	50.3
Skill3D-8B (Li et al., 2026b)	57.6	43.0	56.7	41.3	84.2	50.5	55.6
SpatialAgent-8B (Wu et al., 2026)	54.0	34.1	63.3	45.5	84.0	56.5	56.2
S-Agent-8B† (Dai et al., 2026)	–	41.6	–	46.8	–	–	–
Our models							
T-Spatial-4B	55.3	42.9	62.1	48.6	84.5	64.7	59.7
Δ_{4B} vs. base	+3.2	+15.2	+0.2	+4.0	+0.6	+27.7	+8.5
T-Spatial-8B	57.8	**46.2**	**65.2**	**51.4**	85.6	**68.6**	**62.5**
Δ_{8B} vs. base	+3.5	+17.6	+10.7	+4.1	+0.6	+25.4	+10.3

† Results are taken directly from the original paper. Models with undisclosed parameter counts are excluded from the sub-10B best-result comparison.

Training data. We use a 22k-question pool sampled from VSI-590K (Yang et al., 2026b) and SpatialLadder-26K (Li et al., 2026c), as described in Section 3.2. Of these questions, 20k are used to construct SFT data with Qwen3.5-Plus (Qwen Team, 2026) as the synthesis model, while a disjoint set of 2k questions is used for RL.

Benchmarks. We use six benchmarks: VSI-Bench (Yang et al., 2025), MMSI-Bench (Yang et al., 2026c), 3DSRBench (Ma et al., 2025a), ViewSpatial (Li et al., 2026a), CV-Bench (Tong et al., 2024), and MindCube (Wang et al., 2026a). These benchmarks cover single-image, multi-image, and video reasoning.

Tool interaction and evaluation. We provide four spatial perception tools: Depth Anything 3 (Lin et al., 2026) for depth estimation and geometric reconstruction, SAM 3 (Carion et al., 2026) for promptable object segmentation, Grounding DINO (Liu et al., 2024) for open-vocabulary object detection and localization, and Orient Anything (Wang et al., 2025b) for object orientation estimation. The model invokes these tools by generating `<tool_call>` blocks that specify the selected tool and its arguments. The framework executes each request and returns the resulting spatial evidence as an observation. The model then continues reasoning until it produces an `<answer>`. We use VLMEvalKit (Duan et al., 2024) to extract final answers and compute accuracy.

4.2 Overall performance

Table 1 compares T-Spatial with general-purpose VLMs, specialized spatial models, and tool-augmented methods across six spatial reasoning benchmarks. T-Spatial consistently improves its corresponding backbone and outperforms previous models of comparable scale on average. T-Spatial-8B achieves the strongest overall performance while remaining competitive with strong general-purpose VLMs, demonstrating the effectiveness of spatial tool-use training.

4.3 Tool invocation rates by question type

We analyze how T-Spatial-8B selects tools across different question types. The tool invocation rate is the percentage of questions that invoke a given tool at least once.

Figure 3 shows clear task-dependent patterns. Depth estimation is mainly used for scale and distance reasoning, orientation estimation for viewpoint questions, and camera information for multi-view and motion reasoning. These results show that T-Spatial adapts its tool selection to the spatial requirements of each question.

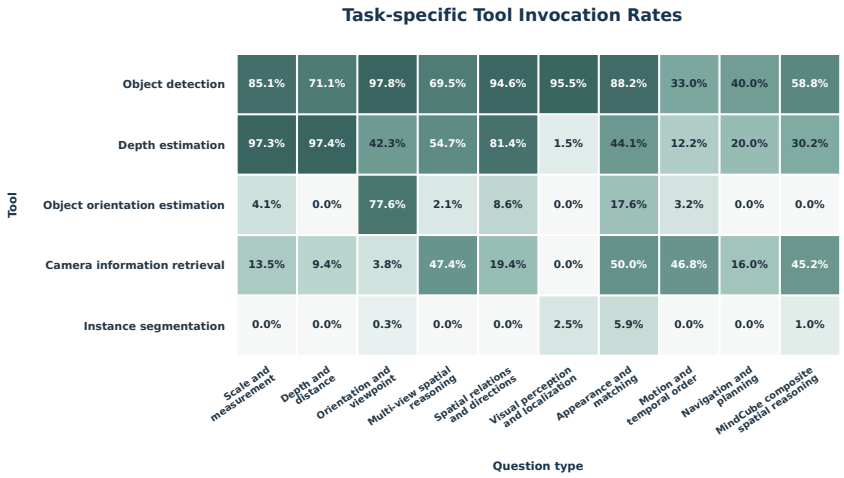

Figure 3: Tool invocation rates of T-Spatial-8B across spatial reasoning question types.

4.4 Tool utility analysis

We evaluate tool utility through paired tool-free and tool-available inference. Specifically, we run each evaluated model twice on every evaluation example under the same decoding settings: once without access to tools and once with the tool interface enabled. Following MED (Ma et al., 2026), Call Gain is the percentage of examples for which the tool-free prediction is incorrect, the tool-available prediction is correct, and the model actually invokes a tool. Call Harm measures the reverse transition from correct to incorrect when a tool is invoked. For examples in which the tool interface is available but no tool is called, Schema Gain and Schema Harm measure the corresponding incorrect-to-correct and correct-to-incorrect transitions, respectively. These schema terms capture changes caused by exposing the model to the tool schema and associated prompt rather than by returned tool evidence. We define net tool utility as the overall accuracy change after tools become available:

$$U_{\text{tool}} = \text{Acc}_{\text{w}} - \text{Acc}_{\text{wo}} = G_{\text{call}} + G_{\text{schema}} - H_{\text{call}} - H_{\text{schema}}. \tag{10}$$

As shown in Table 2, both T-Spatial variants achieve the largest difference between tool gain and tool harm. Their Call Gain consistently exceeds Call Harm, with actual tool calls accounting for most of the improvement. These results show that T-Spatial improves both spatial reasoning performance and the utility of tool use.

Table 2: Gain–harm decomposition under paired tool-free and tool-available evaluation. All values are percentages.

Method	Acc_{wo}	Acc_{w}	Call G.	Schema G.	Call H.	Schema H.	$\text{Acc}_{\text{w}} - \text{Acc}_{\text{wo}}$
Think3D-4B	48.8	50.3	7.6	8.0	7.0	7.1	1.5
Skill3D-8B	50.3	55.6	15.8	11.4	12.3	9.6	5.3
SpatialAgent-8B	49.5	56.2	13.4	9.1	8.1	7.7	6.7
T-Spatial-4B	49.3	59.7	17.6	10.3	8.6	8.9	10.4
T-Spatial-8B	50.8	62.5	18.1	9.1	8.2	7.3	11.7

4.5 Ablation study

We compare T-Spatial with SFT, Standard GRPO, and variants that remove one signal level at a time. All RL variants use the same SFT initialization, training data, tool interface, and rollout

budget. Standard GRPO uses only the final-answer reward, while each *w/o* variant removes the corresponding signal from TUPO.

Table 3: Ablation results on the 8B backbone. Avg. is the arithmetic mean over the six benchmarks.

Variant	VSI-Bench	MMSI-Bench	3DSR Bench	View Spatial	CV-Bench	MindCube	Avg.
SFT	51.7	41.9	59.1	49.6	83.8	59.4	57.6
Standard GRPO	55.1	42.6	60.3	50.4	85.3	64.8	59.8
w/o sample-level signal	56.3	**47.6**	63.1	50.8	85.1	67.4	61.7
w/o branch-level signal	55.4	45.3	61.3	49.2	84.1	66.5	60.3
w/o node-level signal	55.9	41.4	60.5	**51.7**	**85.6**	65.7	60.1
T-Spatial-8B	**57.8**	46.2	**65.2**	51.4	**85.6**	**68.6**	**62.5**

The complete T-Spatial model achieves the highest average performance. Removing either the branch-level or node-level signal lowers the overall score, confirming the value of downstream outcome information and local tool-utility estimation. The sample-level signal has a smaller and more task-dependent effect, while combining all signal levels gives the strongest overall result.

4.6 Robustness to geometry-backend replacement

We test backend robustness by replacing Depth Anything 3 with VGGT (Wang et al., 2025a) at inference time without policy retraining. The evaluation data, tool interface, and remaining tools are kept unchanged. Although the replacement produces task-specific score changes, the average performance remains stable for both model sizes. This shows that the learned tool-use policy can transfer across geometry backends, while the usefulness of the returned spatial evidence remains task-dependent.

Table 4: Performance after inference-time geometry-backend replacement.

Checkpoint	Backend	VSI	MMSI	3DSR	View	CV	Mind	Avg.	Δ Avg.
T-Spatial-4B	Depth Anything 3	**55.3**	**42.9**	62.1	48.6	**84.5**	**64.7**	**59.7**	–
T-Spatial-4B	VGGT	51.4	41.6	**64.3**	**50.8**	83.5	64.1	59.3	-0.4
T-Spatial-8B	Depth Anything 3	**57.8**	**46.2**	65.2	51.4	**85.6**	68.6	**62.5**	–
T-Spatial-8B	VGGT	52.8	45.0	**67.2**	**53.8**	84.4	**69.6**	62.1	-0.3

5 Conclusion

We introduced the Tool-Augmented Spatial Reasoning Agent (T-Spatial), trained through an SFT-then-RL pipeline that coordinates Difficulty-Aware Trajectory Synthesis (DATS) with Tool Utility-based Policy Optimization (TUPO). DATS constructs difficulty-adaptive SFT trajectories to reduce unnecessary tool use in the training data, while TUPO provides finer-grained credit assignment through sample-, trajectory-, branch-, and node-level signals without an auxiliary learned reward model. Experiments across six spatial reasoning benchmarks show that T-Spatial outperforms its corresponding base-model baselines and achieves strong overall performance. Further analysis demonstrates task-dependent tool selection, a larger positive difference between tool gain and tool harm, and stable performance when the geometry backend is replaced. These results highlight the importance of optimizing tool utility with fine-grained, multi-level signals rather than relying only on final-answer rewards.

AI USE STATEMENT

AI models were used to synthesize training data, as detailed in Section 3.2. AI-assisted tools, including Codex, were also used for code development and debugging, as well as for English translation and language polishing of the manuscript. The generated data were subject to rule-based format and correctness checks, while all other AI-assisted outputs were reviewed by the authors.

REFERENCES

Shuai Bai, Yuxuan Cai, Ruizhe Chen, Keqin Chen, Xionghui Chen, Zesen Cheng, Lianghao Deng, Wei Ding, Chang Gao, Chunjiang Ge, Wenbin Ge, Zhifang Guo, Qidong Huang, Jie Huang, Fei Huang, Binyuan Hui, Shutong Jiang, Zhaohai Li, Mingsheng Li, Mei Li, Kaixin Li, Zicheng Lin, Junyang Lin, Xuejing Liu, Jiawei Liu, Chenglong Liu, Yang Liu, Dayiheng Liu, Shixuan Liu, Dunjie Lu, Ruilin Luo, Chenxu Lv, Rui Men, Lingchen Meng, Xuancheng Ren, Xingzhang Ren, Sibo Song, Yuchong Sun, Jun Tang, Jianhong Tu, Jianqiang Wan, Peng Wang, Pengfei Wang, Qiuyue Wang, Yuxuan Wang, Tianbao Xie, Yiheng Xu, Haiyang Xu, Jin Xu, Zhibo Yang, Mingkun Yang, Jianxin Yang, An Yang, Bowen Yu, Fei Zhang, Hang Zhang, Xi Zhang, Bo Zheng, Humen Zhong, Jingren Zhou, Fan Zhou, Jing Zhou, Yuanzhi Zhu, and Ke Zhu. Qwen3-VL technical report. *arXiv preprint arXiv:2511.21631*, 2025.

Zhongang Cai, Ruisi Wang, Chenyang Gu, Fanyi Pu, Junxiang Xu, Yubo Wang, Wanqi Yin, Zhitao Yang, Chen Wei, Tongxi Zhou, Qingping Sun, Hui En Pang, Jiaqi Li, Oscar Qian, Zhiqian Lin, Xuanke Shi, Kewang Deng, Xiaoyang Han, Zukai Chen, Xiangyu Fan, Hanming Deng, Lewei Lu, Liang Pan, Bo Li, Ziwei Liu, Quan Wang, Dahua Lin, and Lei Yang. Scaling spatial intelligence with multimodal foundation models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7879–7890, 2026.

Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris Coll-Vinent, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, Jie Lei, Tengyu Ma, Baishan Guo, Arpit Kalla, Markus Marks, Joseph Greer, Meng Wang, Peize Sun, Roman Rädle, Triantafyllos Afouras, Effrosyni Mavroudi, Katherine Xu, Tsung-Han Wu, Yu Zhou, Liliane Momeni, Rishi Hazra, Shuangrui Ding, Sagar Vaze, Francois Porcher, Feng Li, Siyuan Li, Aishwarya Kamath, Ho Kei Cheng, Piotr Dollar, Nikhila Ravi, Kate Saenko, Pengchuan Zhang, and Christoph Feichtenhofer. SAM 3: Segment anything with concepts. In *International Conference on Learning Representations*, 2026.

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, 2024.

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.

Yalun Dai, Hao Li, Shulin Tian, Runmao Yao, Yuhao Dong, Fangzhou Hong, Zhaoxi Chen, Fangfu Liu, Baoliang Tian, Dingwen Zhang, Tao Wang, Kim-Hui Yap, and Ziwei Liu. S-Agent: Spatial Tool-Use Elicits Reasoning for Spatial Intelligence. *arXiv preprint arXiv:2606.20515*, 2026.

Haodong Duan, Junming Yang, Yuxuan Qiao, Xinyu Fang, Lin Chen, Yuan Liu, Xiaoyi Dong, Yuhang Zang, Pan Zhang, Jiaqi Wang, Dahua Lin, and Kai Chen. VLMEvalKit: An open-source toolkit for evaluating large multi-modality models. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pp. 11198–11201, 2024. doi: 10.1145/3664647.3685520.

Mingkuan Feng, Jinyang Wu, Hao Gu, Fangrui Lv, Ruihan Jin, Chuyuan Zhang, Zhengqi Wen, and Jianhua Tao. TACO: Tool-augmented credit optimization for agentic tool use. *arXiv preprint arXiv:2606.30251*, 2026.

Google. Introducing Gemini 3, November 2025. URL `https://blog.google/products-and-platforms/products/gemini/gemini-3-collection/`. Accessed: 2026-09-26.

Yejie Guo, Yunzhong Hou, Wufei Ma, Meng Tang, and Ming-Hsuan Yang. Pursuing minimal sufficiency in spatial reasoning. In *International Conference on Learning Representations*, 2026.

Xinhai Hou, Shaoyuan Xu, Manan Biyani, Moyan Li, Jia Liu, Todd C. Hollon, and Bryan Wang. CodeV: Code with images for faithful visual reasoning via tool-aware policy optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21500–21510, 2026.

Yuxiang Ji, Ziyu Ma, Yong Wang, Guanhua Chen, Xiangxiang Chu, and Liaoni Wu. Tree search for LLM agent reinforcement learning. In *International Conference on Learning Representations*, 2026.

Dingming Li, Hongxing Li, Zixuan Wang, Yuchen Yan, Hang Zhang, Siqi Chen, Guiyang Hou, Shengpei Jiang, Wenqi Zhang, Yongliang Shen, Weiming Lu, and Yueting Zhuang. ViewSpatial-Bench: Evaluating multi-perspective spatial localization in vision-language models. In *European Conference on Computer Vision*, pp. 95–111, 2026a.

Haoyuan Li, Zhengdong Hu, Jun Wang, Hehe Fan, and Yi Yang. Skill-3D: Evolving Scene-Aware Skills for Agentic 3D Spatial Reasoning. *arXiv preprint arXiv:2606.07436*, 2026b.

Hongxing Li, Dingming Li, Zixuan Wang, Yuchen Yan, Hang Wu, Wenqi Zhang, Yongliang Shen, Weiming Lu, Jun Xiao, and Yueting Zhuang. SpatialLadder: Progressive training for spatial reasoning in vision-language models. In *International Conference on Learning Representations*, 2026c.

Haotong Lin, Sili Chen, Jun Hao Liew, Donny Y. Chen, Zhenyu Li, Yang Zhao, Sida Peng, Hengkai Guo, Xiaowei Zhou, Guang Shi, Jiashi Feng, and Bingyi Kang. Depth Anything 3: Recovering the visual space from any views. In *International Conference on Learning Representations*, 2026.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection. In *European Conference on Computer Vision*, pp. 38–55, 2024.

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 *International Conference on Learning Representations*, 2026.

Wufei Ma, Haoyu Chen, Guofeng Zhang, Yu-Cheng Chou, Jieneng Chen, Celso M. de Melo, and Alan Yuille. 3DSRBench: A comprehensive 3D spatial reasoning benchmark. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 6924–6934, 2025a.

Wufei Ma, Yu-Cheng Chou, Qihao Liu, Xingrui Wang, Celso de Melo, Jianwen Xie, and Alan L. Yuille. SpatialReasoner: Towards explicit and generalizable 3D spatial reasoning. In *Advances in Neural Information Processing Systems*, volume 38, 2025b.

Yan Ma, Weiyu Zhang, Tianle Li, Linge Du, Xuyang Shen, and Pengfei Liu. What does vision tool-use reinforcement learning really learn? disentangling tool-induced and intrinsic effects for crop-and-zoom. In *International Conference on Machine Learning*, 2026.

OpenAI. Introducing GPT-5.2, December 2025. URL `https://openai.com/index/introducing-gpt-5-2/`. Accessed: 2026-09-26.

Kun Ouyang, Yuanxin Liu, Haoning Wu, Yi Liu, Hao Zhou, Jie Zhou, Fandong Meng, and Xu Sun. SpaceR: Reinforcing MLLMs in Video Spatial Reasoning. *arXiv preprint arXiv:2504.01805*, 2025.

Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL `https://qwen.ai/blog?id=qwen3.5`. Accessed: 2026-09-26.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. DeepSeekMath: Pushing the Limits of Mathematical Reasoning in Open Language Models. *arXiv preprint arXiv:2402.03300*, 2024.

Zhaochen Su, Linjie Li, Mingyang Song, Yunzhuo Hao, Zhengyuan Yang, Jun Zhang, Guanjie Chen, Jiawei Gu, Juntao Li, Xiaoye Qu, and Yu Cheng. OpenThinkIMG: Learning to Think with Images via Visual Tool Reinforcement Learning. *arXiv preprint arXiv:2505.08617*, 2025.

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.

Shengbang Tong, Ellis Brown, Penghao Wu, Sanghyun Woo, Manoj Middepogu, Sai Charitha Akula, Jihan Yang, Shusheng Yang, Adithya Iyer, Xichen Pan, Austin Wang, Rob Fergus, Yann LeCun, and Saining Xie. Cambrian-1: A fully open, vision-centric exploration of multimodal LLMs. In *Advances in Neural Information Processing Systems*, volume 37, 2024.

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.

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 *International Conference on Learning Representations*, 2026a.

Qixun Wang, Yang Shi, Letian Cheng, Zhuoran Zhang, Yan He, Yuqi Tang, Qi Zhang, Xinlei Yu, Ruizhe Chen, Tianrun Xu, Yuanxing Zhang, Pengfei Wan, Haotian Wang, and Xianghua Ying. Beacon: Knowing When and How to Perform Agentic Visual Reasoning. *arXiv preprint arXiv:2607.28595*, 2026b.

Zehan Wang, Ziang Zhang, Tianyu Pang, Chao Du, Hengshuang Zhao, and Zhou Zhao. Orient anything: Learning robust object orientation estimation from rendering 3d models. In *Proceedings of the 42nd International Conference on Machine Learning*, pp. 65556–65574, 2025b.

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*, volume 38, 2025a.

Feijie Wu, Weiwu Zhu, Yuxiang Zhang, Soumya Chatterjee, Jiarong Zhu, Fan Mo, Rong Luo, and Jing Gao. PORTool: Importance-Aware Policy Optimization with Rewarded Tree for Multi-Tool-Integrated Reasoning. *arXiv preprint arXiv:2510.26020*, 2025b.

Haoning Wu, Xiao Huang, Yaohui Chen, Ya Zhang, Yanfeng Wang, and Weidi Xie. SpatialScore: Towards Comprehensive Evaluation for Spatial Intelligence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 31029–31041, 2026.

Changhao Xiang, Shilin Zhang, Zheng Ma, Kanzhi Cheng, Ruize Ma, Yi Feng, Jianbing Zhang, Zhi Wang, Zhen Wu, Xinyu Dai, and Lewei Lu. OpenVisTool: An Open Recipe for Synthesizing Instructive Visual Tool-Use Trajectories. *arXiv preprint arXiv:2608.08557*, 2026.

Jihan Yang, Shusheng Yang, Anjali W. Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in space: How multimodal large language models see, remember, and recall spaces. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10632–10643, 2025.

Rui Yang, Ziyu Zhu, Yanwei Li, Jingjia Huang, Shen Yan, Siyuan Zhou, Zhe Liu, Xiangtai Li, Shuangye Li, Wenqian Wang, Yi Lin, and Hengshuang Zhao. Visual Spatial Tuning. In *European Conference on Computer Vision*, pp. 192–211, 2026a.

Shusheng Yang, Jihan Yang, Pinzhi Huang, Ellis Brown, Zihao Yang, Yue Yu, Shengbang Tong, Zihan Zheng, Yifan Xu, Muhan Wang, Rob Fergus, Yann LeCun, Fei-Fei Li, and Saining Xie. Cambrian-S: Towards spatial supersensing in video. In *International Conference on Learning Representations*, 2026b.

Sihan Yang, Runsen Xu, Yiman Xie, Sizhe Yang, Mo Li, Jingli Lin, Chenming Zhu, Xiaochen Chen, Haodong Duan, Xiangyu Yue, Dahua Lin, Tai Wang, and Jiangmiao Pang. MMSI-Bench: A benchmark for multi-image spatial intelligence. In *International Conference on Learning Representations*, 2026c.

Zaibin Zhang, Yuhan Wu, Lianjie Jia, Yifan Wang, Zhongbo Zhang, Yijiang Li, Binghao Ran, Fuxi Zhang, Zhuohan Sun, Yizhuang Peng, Zhenfei Yin, Lijun Wang, and Huchuan Lu. Think3D: Thinking with Space for Spatial Reasoning. *arXiv preprint arXiv:2601.13029*, 2026.

Hongpei Zheng, Shijie Li, Yanran Li, and Hujun Yin. SpatialReasoner: Active Perception for Large-Scale 3D Scene Understanding. *arXiv preprint arXiv:2512.03284*, 2025.

Ziwei Zheng, Minghao Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, Chao Shen, and Xing Yu. DeepEyes: Incentivizing "thinking with images" via reinforcement learning. In *International Conference on Learning Representations*, 2026.

Yang Zhou, Zixuan Huang, Sunzhu Li, Zhuo Yang, Chen Zhang, Shunian Chen, Caijun Yan, Jianyao Xu, Shunyu Liu, Weijie Fu, Peiliang Li, Xiaozhi Chen, and Yuxiang Cai. SpatialCLI: Learning to Reason With Spatial Tools, Then Without Them. *arXiv preprint arXiv:2607.27703*, 2026.