SPATIALWEAVE: CO-EVOLVING TOOLS, WORKFLOWS, AND SKILLS FOR SPATIAL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Tool-augmented vision-language models can perceive complex spatial environments, but they struggle to calculate the exact answers that questions require. Existing methods either recalculate everything from scratch for each question and repeat the same mistakes, or they use fixed steps that fail on new problems. We introduce SPATIALWEAVE, which improves spatial reasoning without modifying the underlying language model. Instead, it automatically optimizes the external tools, workflows, and usage rules the model relies on. During evolution, a temporary procedure scaffold guides the model to explore various tool combinations. Successful tool sequences are saved as reusable workflows, and common mistakes are corrected by adding specific skills. After evolution, this scaffold is completely removed. Across seven spatial reasoning benchmarks, SPATIALWEAVE consistently improves performance on three different models. Notably, it raises Qwen3.8-27B to 80.73%, beating the base GPT-6 Astra model at 80.28%. It also improves Gemma-4-31B to 73.58%, outperforming the specialized SpatialClaw agent on the same model. Furthermore, removing the scaffold during inference makes the process much more efficient. It reduces token generation by up to 34.9% and tool calls by up to 46.0% without losing accuracy. Ultimately, SPATIALWEAVE shows that automatically building a lightweight set of external tools and rules is a practical way to achieve strong spatial reasoning while keeping inference fast and simple.

1 INTRODUCTION

Spatial reasoning requires converting visual evidence into the quantity a question asks for, in the entities, reference frame, and units it specifies. When a camera slides right while turning left, answering "*did it move left or right?*" requires ordering the images, fixing a ground frame, and separating translation from rotation (Figure 1a). Vision–language models still lose this alignment (Fu et al., 2024; Yang et al., 2025; Jia et al., 2026), and tool-augmented agents that supply detections, depth, reconstructions, and measurements (Marsili et al., 2025; Zhang et al., 2026b; Chen et al., 2026a; Luo et al., 2026) often fail at the conversion itself: they read a depth offset in the wrong frame, mistake heading change for displacement, or answer with a diagnostic value instead of the requested quantity (Figure 1b–c).

Current methods structure this conversion in one of two ways, each with a cost. Agents that plan every question from scratch (Yao et al., 2023; Surís et al., 2023; Cho et al., 2026) rebuild the same geometric computation each time and can repeat the same binding errors, while fixed procedures written into the prompt carry redundant steps into every interaction and can transfer poorly to question types they were not written for, a limitation also noted by Tian et al. (2026a). Experience, skill, and workflow memory methods (Wang et al., 2025; Zhao et al., 2024; Wu et al., 2026; Yang et al., 2026b; Li et al., 2026a) reuse past trajectories but mostly evolve a single kind of component. Yet a computation and its usage are coupled: a new executor needs binding rules, and one interpretation error can call for both a program check and a recovery rule. Revising either side alone leaves executors that agents misuse, or guidance that re-describes a computation step by step in every prompt.

In this work, we introduce SPATIALWEAVE, which keeps the solver frozen and co-evolves its external *harness* $\mathcal{H} = (\mathcal{T}, \mathcal{W}, \mathcal{S})$: **tools**, executable **workflows** that compile recurring tool chains into callable programs, and **skills** that specify how to bind and read them. The solver keeps task-level

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.

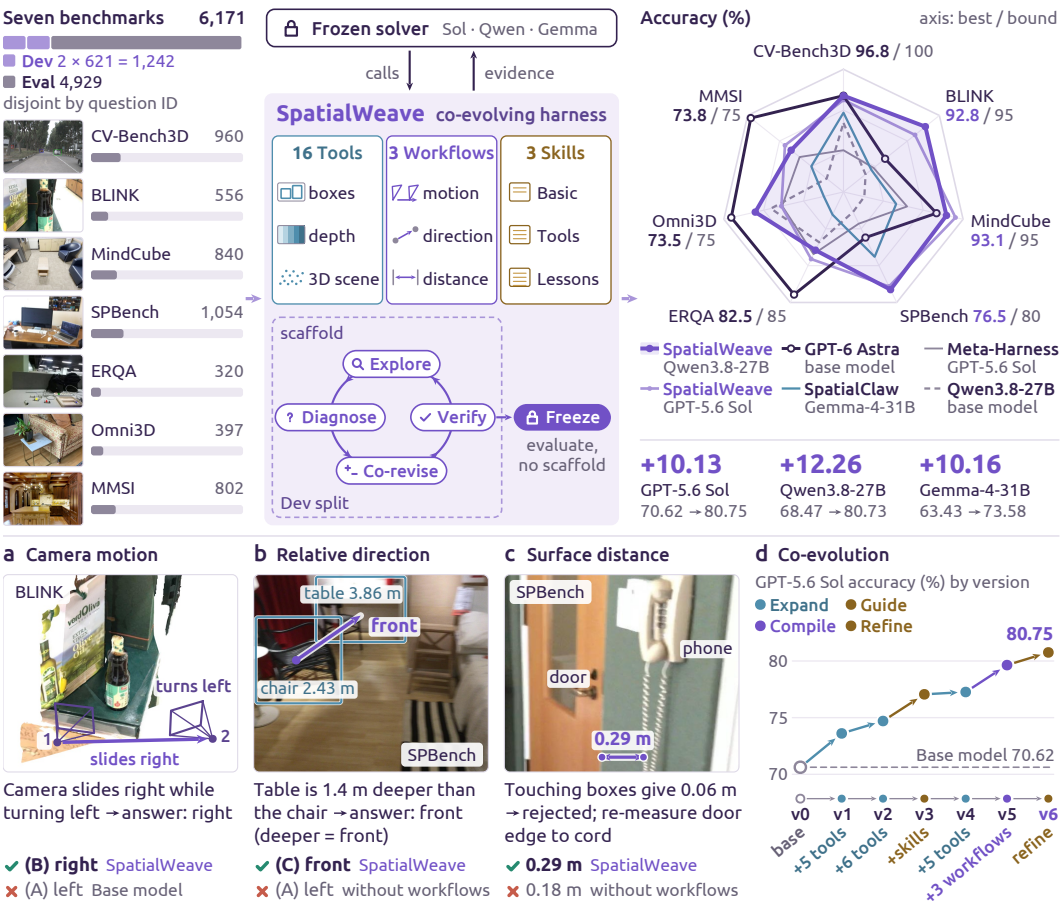

Figure 1: **SPATIALWEAVE at a glance.** *Top:* data split, co-evolved harness, and accuracy (each radar axis spans its own range, labeled best value / bound). *Bottom:* (a–c) recorded GPT-5.6 Sol executions, SPATIALWEAVE (✓) vs. a failing comparison (×); (d) Sol accuracy of versions v0–v6.

planning; a workflow runs on cached evidence without additional model calls, adding computation without adding interactions. During evolution only, a *procedure scaffold* enforces broad tool composition to elicit diverse call chains, and trajectory analysis drives four operators (Expand, Guide, Compile, Refine) that revise all three components together. The scaffold is removed by design once evolution ends, leaving 16 tools, three workflows, and three compact skill packages for leaner inference.

On seven benchmarks, the harness improves all **21 benchmark–backbone combinations** of three backbones (GPT-5.6 Sol, Qwen3.8-27B, Gemma-4-31B) and adds more than 10 points overall to each; with it, Qwen3.8-27B (**80.73**) scores slightly above GPT-6 Astra (base model; 80.28), Gemma-4-31B surpasses SpatialClaw on the same checkpoint, and the harness also lifts GPT-6 Astra by 5.07 points overall. The evolution trace (Figure 1d) locates the gains: five more specialized tools add 0.20 points, whereas compiling recurring chains into three workflows adds 2.37 points, and workflows and interpretation skills repair different benchmarks. Removing the scaffold at inference cuts Sol's tokens by **27.58%** and tool calls by 46.03% without losing accuracy. In summary, our contributions are threefold:

- **Method:** SPATIALWEAVE, a spatial harness for a frozen solver that couples tools, callable executable workflows, and skills that bind and interpret their returns (Section 3).
- **Co-evolution:** four operators that revise all three components together, with an evolution-only procedure scaffold whose elicited chains are compiled into workflows (Section 3.3).

- **Findings:** gains reach every benchmark of the three backbones; beyond a core perception toolset, they come from compilation and guidance rather than more tools; and they persist at lower cost once the scaffold is removed (Section 4).

2 Related Work

Spatial reasoning benchmarks and models. Benchmarks probe depth and relative position (Tong et al., 2024; Fu et al., 2024), mental modeling from limited views (Wang et al., 2026b), multi-image and embodied reasoning (Yang et al., 2026a; Gemini Robotics Team, 2025), spatial memory from video (Yang et al., 2025), and broad spatial taxonomies (Jia et al., 2026). Training on synthesized 3D spatial data (Chen et al., 2024) or progressive curricula (Li et al., 2026b) improves a given model; SpatialWeave instead keeps the solver frozen and improves what surrounds it.

Tool-augmented and programmatic visual reasoning. Language models learn to call tools (Schick et al., 2023) or interleave reasoning with actions (Yao et al., 2023), and VisProg and ViperGPT compose vision modules through generated programs (Gupta & Kembhavi, 2023; Surís et al., 2023), and TroVE and LATM induce reusable tools (Wang et al., 2024c; Cai et al., 2024); geometric models such as Depth Anything and DUSt3R (Yang et al., 2024; Wang et al., 2024b) make depth and reconstruction available as tools. Spatial agents add programs over a dynamic API (VADAR; Marsili et al., 2025), 3D visual programs (Luo et al., 2026), rendered views (Zhang et al., 2026b), reinforcement-learned tool use (Chen et al., 2026a), geometric constraints (Chen et al., 2026b), 3D memory (He et al., 2026), scene graphs (Du et al., 2026), tools as hints (Tian et al., 2026b), collaborating agents (Zhang et al., 2026c; Hwang et al., 2026), agentic harnesses (Ruan et al., 2026), and a persistent Python interface (SpatialClaw; Cho et al., 2026). Most assemble the computation anew per question or follow hand-designed procedures; SpatialWeave reuses this evidence layer and learns which computations to compile and how to read them.

Self-evolving agents, skills, and workflows. Agents improve without weight updates by reflecting on failures (Shinn et al., 2023), distilling insights (Zhao et al., 2024), storing workflows and procedural memory (Wang et al., 2025; Fang et al., 2026), or building, discovering, and distilling skills (Wang et al., 2024a; Zheng et al., 2025; Ni et al., 2026; Yang et al., 2026b; Li et al., 2026a). Prompt and context optimization (Agrawal et al., 2026; Zhang et al., 2026a), workflow and agent-design search (Zhang et al., 2025; Hu et al., 2025), and harness optimization (Wang et al., 2026a; Lee et al., 2026) adapt what surrounds a fixed model. For spatial reasoning, TVP abstracts recurring programs into tools (Wu et al., 2026), and NeSy-Spatial evolves tool-use and geometry skills (Tian et al., 2026a). SpatialWeave differs in three respects: it co-evolves tools, workflows, and skills as one harness, so each new executor ships with its binding guidance; its scaffold serves only exploration and is removed before inference; and the solver stays frozen, so all improvement resides in the harness.

3 SpatialWeave

SpatialWeave turns spatial reasoning experience into reusable computation and the guidance to apply it. It operates at two timescales. At inference (Figure 2a), a frozen solver answers each question with a fixed harness, selecting actions, calling tools or workflows, reading skills, and updating its workspace until it answers; during evolution (Figure 2c), scaffolded trajectories on the development split drive coordinated revisions of tools, workflows, and skills.

3.1 Harness and Inference

A harness version r is a triple of tools, workflows, and skills, and its action space exposes every workflow as a callable tool:

$$\mathcal{H}_r = (\mathcal{T}_r, \mathcal{W}_r, \mathcal{S}_r), \qquad \mathcal{A}_r = \mathcal{T}_r \cup \{\mathrm{Expose}(w) : w \in \mathcal{W}_r\}. \tag{1}$$

Expose registers a workflow through the same Model Context Protocol (MCP) interface as a tool. The deployed library exposes 19 interfaces: three compiled workflows and 16 tools for inspection,

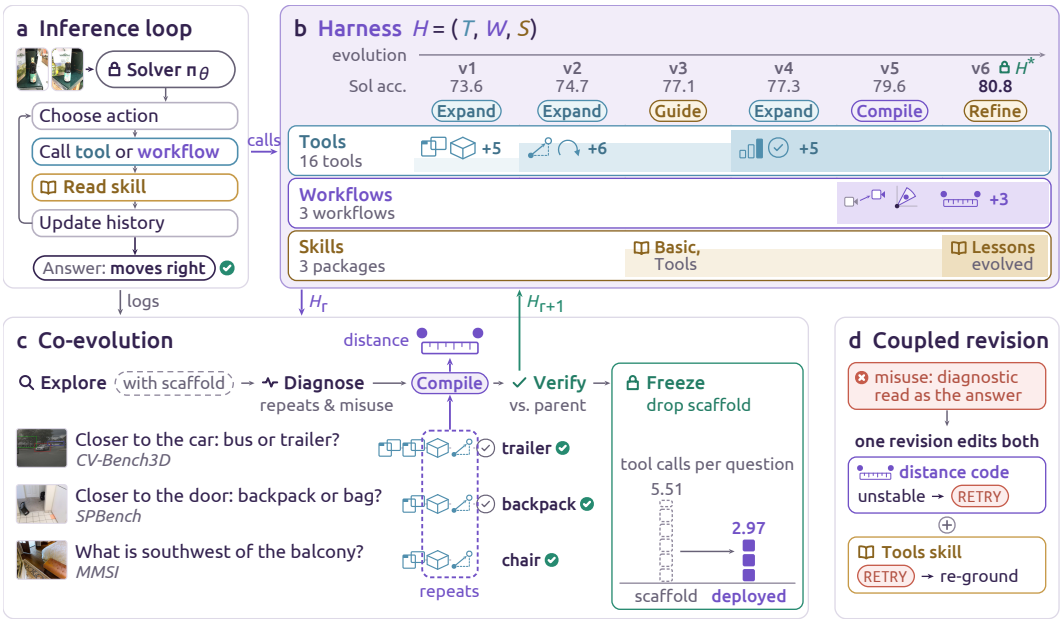

Figure 2: **The harness gets leaner as it gets stronger.** (a) A frozen solver calls tools and workflows and reads skills from the harness, built over v1–v6 (b). (c) During evolution, a procedure scaffold drives broad tool use; a recurring chain is compiled into a workflow that joins the harness, each round turns $\mathcal{H}_r$ into a verified $\mathcal{H}_{r+1}$, and the harness is frozen without the scaffold, cutting spatial-tool calls from 5.51 to 2.97 per question (GPT-5.6 Sol) at similar accuracy. (d) One revision fixes a misuse by editing the distance workflow's code and the Tools skill together.

detection (Liu et al., 2024), segmentation (Ravi et al., 2025), depth (Piccinelli et al., 2026), reconstruction (Wang et al., 2026c), geometric queries, counting, and diagnostics. Given the visual observations and question x and the interaction history h_k, the solver selects the next action

$$a_k \sim \pi_\theta\big(\cdot \mid x,\, h_k,\, \mathcal{S}_r,\, \mathrm{Schema}(\mathcal{A}_r)\big), \tag{2}$$

where θ and the runtime stay fixed and Schema holds the model-visible argument definitions. Skills act through context; a selected tool or workflow runs its programmed computation on a per-question cache of evidence, and the solver answers or acquires more evidence from the returned readouts (details in Appendix A.1).

3.2 Executable Workflows and Skills

A workflow fixes a recurring computation and leaves its question-dependent bindings to the solver:

$$w = (\Sigma_w, P_w, G_w, \Omega_w), \qquad o_w = P_w(b_w; \mathcal{T}_r, G_w) \in \Omega_w, \tag{3}$$

with signature Σ_w (parameters, image order, observer roles, units), program P_w, local checks G_w, return convention Ω_w, and bindings b_w supplied by the solver. Workflows run on cached evidence without additional model calls. The camera-motion workflow (`camera_motion`) builds a ground frame from reconstructed poses and separates translation bearing from yaw and pitch; the pair-measurement workflow (`measure_pair`) selects the quantity the question asks for (center, closest-surface, or ground distance, or depth) in the requested unit; the relation-readout workflow (`relation_readout`) binds observer, facing, and target and maps the bearing to a directional sector. Each returns ok, a retry status that asks the solver to re-ground, or not applicable, with readouts and diagnostics.

Skills bind these computations to the question. *Basic* sets reasoning and answer-format conventions, *Tools* specifies interface use and local recovery (including the action for each return status), and *Lessons* holds narrow interpretation rules, e.g., objects that merely look alike do not establish that two views share a scene. The solver keeps the task-level plan: for a relocated-view question, it

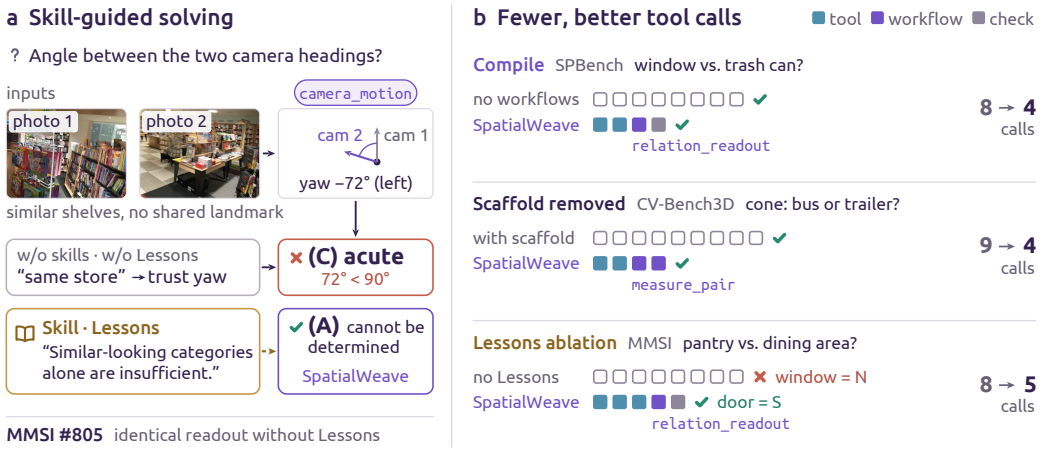

Figure 3: **Skills and workflows change how the solver uses tools.** (a) With the same camera-motion readout, runs without Lessons answer *acute*; SPATIALWEAVE applies a Lessons rule, finds no shared landmark, and answers *cannot be determined*. (b) One square per call (hollow: the named ablation or the scaffolded variant; filled: SPATIALWEAVE): compiled workflows and removing the scaffold halve the calls at the same answer (top two); without Lessons, eight calls end in a wrong answer vs. five for a correct one (bottom). Each case is one recorded run.

identifies observer, facing object, and target, acquires their evidence, and binds them to the relation workflow, which performs the directional computation. Recovery rules preserve semantics; surface-gap re-grounding keeps both object identities and the closest-surface quantity. Skills thus bind and interpret but do not prescribe evidence procedures, the opposite of the development scaffold below (cards in Appendix A).

3.3 HARNESS CO-EVOLUTION

The object of improvement is $\mathcal{H}_r$; model parameters and runtime stay fixed. Each round runs Explore, Diagnose, Co-revise, Verify, and Freeze (Figure 2c).

Explore and Diagnose. During evolution only, the skill context is augmented with a *procedure scaffold* $\mathcal{P}_r$ of task-level evidence procedures that enforce broad tool composition to elicit diverse call chains. Trajectories on the development questions fill a buffer that analysis sorts into three finding sets:

$$\mathcal{S}_r^{\text{dev}} = \mathcal{S}_r \cup \mathcal{P}_r, \qquad \mathcal{B}_r = \{\tau_i\}_{i \in \mathcal{D}_{\text{dev}}}, \qquad \text{Diagnose}(\mathcal{B}_r) = \big(\mathcal{F}_r^{\text{gap}}, \mathcal{F}_r^{\text{rep}}, \mathcal{F}_r^{\text{use}}\big), \tag{4}$$

where each τ_i follows Eq. (2) with $\mathcal{S}_r^{\text{dev}}$ in place of $\mathcal{S}_r$, and the findings are missing evidence or diagnostics, recurring call chains, and misused returns (wrong bindings, frames, or quantities).

Co-revise. A revision $\delta_r = (\Delta\mathcal{T}_r, \Delta\mathcal{W}_r, \Delta\mathcal{S}_r)$ applies four operators:

$$\begin{aligned} &\text{EXPAND}: \ \Delta\mathcal{T}_r \leftarrow \mathcal{F}_r^{\text{gap}}, \qquad \text{GUIDE}: \ \Delta\mathcal{S}_r \leftarrow \mathcal{F}_r^{\text{use,inv}}, \\ &\text{COMPILE}: \ \Delta\mathcal{W}_r \leftarrow \{w_c : c \in \mathcal{F}_r^{\text{rep}}, \ n_r(c) \geq n_{\min} \wedge \text{Rebind}(c)\}, \\ &\text{REFINE}: \ \Delta\mathcal{S}_r \leftarrow \mathcal{F}_r^{\text{use,int}}, \quad \mathcal{P}_{r+1} = \mathcal{P}_r \setminus \text{Subsumed}(\mathcal{W}_{r+1}). \end{aligned} \tag{5}$$

Guide turns invocation findings $\mathcal{F}_r^{\text{use,inv}} \subseteq \mathcal{F}_r^{\text{use}}$ into guidance and Refine interpretation findings $\mathcal{F}_r^{\text{use,int}}$ into rules. Compile turns a tool-call chain c that occurs in at least $n_{\min}$ trajectories into a workflow when $\text{Rebind}(c)$ holds, i.e., its arguments can be re-instantiated from question entities, image order, and reference frames. The scaffold serves exploration only: by design, it is removed at Freeze, and the compiled workflows carry the computation it spelled out. Revisions are coordinated: a new workflow ships with binding guidance, and a repeatedly misused return can change both executor and skill. For example, when agents answered with the diagnostic distance of a retry (re-ground) return, the executor revision cleared its answer fields, and a Tools rule directed re-grounding (Figure 1c).

Verify and Freeze. A candidate $\widetilde{\mathcal{H}}_r = \mathrm{Apply}(\mathcal{H}_r, \delta_r)$ passes interface checks and artifact replay and is compared with its parent on the same development questions (accuracy change ΔJ, repairs, regressions, cost):

$$\mathrm{Select}(\widetilde{\mathcal{H}}_r) \iff \mathrm{Check}(\widetilde{\mathcal{H}}_r) \wedge \big[\Delta J > \epsilon_J \vee (|\Delta J| \le \epsilon_J \wedge \Delta\mathrm{cost} < 0)\big],$$
$$\mathcal{H}_{r+1} = \begin{cases} \widetilde{\mathcal{H}}_r & \text{if selected,} \\ \mathcal{H}_r & \text{otherwise,} \end{cases} \qquad \mathcal{H}^\star = \mathrm{Freeze}(\mathcal{H}_R), \quad \mathcal{P} \notin \mathcal{H}^\star. \tag{6}$$

The second branch keeps a candidate that matches accuracy at lower cost. Code, descriptions, skills, and configuration are frozen together. Coding agents generate every candidate from the mined findings, and researchers only accept or reject candidates; Section 4.3 traces the resulting versions and Section 4.4 describes what the scaffold contributes.

4 Experiments

We test whether SpatialWeave transfers across backbones, how evolution builds capability and what each component contributes, and what the procedure scaffold contributes and costs.

4.1 Experimental Setup

Benchmarks and scoring. We use the 3D subset of CV-Bench (Tong et al., 2024), five spatial subtypes of BLINK (Fu et al., 2024), MindCube (Wang et al., 2026b), SPBench (Li et al., 2026b), ERQA (Gemini Robotics Team, 2025), Omni3D-Bench (Marsili et al., 2025), and MMSI-Bench (Yang et al., 2026a), covering depth and relative position, multi-view correspondence, mental manipulation, metric measurement, and embodied multi-image reasoning. The 6,171 selected questions form two development partitions of 621 questions each and 4,929 evaluation questions, disjoint by question ID (sizes in Table 1). The split is by question, not scene (overlap statistics in Appendix B). Each question scores $s_i \in [0, 1]$ under a per-item rule fixed before prediction (Appendix B). Multiple-choice questions (all of CV-Bench3D, BLINK, MindCube, ERQA, and MMSI; 361 SPBench and 126 Omni3D-Bench questions) score 1 for the correct option letter. Numerical answers receive mean relative accuracy, the fraction of ten relative-error tolerances $0.50, 0.45, \ldots, 0.05$ that the answer meets (693 SPBench and 215 Omni3D-Bench questions); 56 integer Omni3D-Bench answers must match exactly. Only the final-answer position is parsed; unparsable, missing, or failed answers score 0 and stay in the denominator. Benchmark and overall scores are micro averages, $100 \sum_i s_i/n$; differences use unrounded scores, and ties share emphasis.

Configurations and references. SpatialWeave exposes 16 tools and three workflows with the Basic, Tools, and Lessons skills. SpatialWeave + *procedure guidance* adds the development scaffold $\mathcal{P}$, as it stood at the end of evolution, back at inference while keeping the complete executable library, so the two differ only in task-level textual procedures. *Tools + workflows* keeps every callable interface, schema, and description but removes all skills. Backbones are GPT-5.6 Sol (medium reasoning effort), Qwen3.8-27B (xhigh thinking), and Gemma-4-31B-it (FP8). A *base model (no tools)* row reports the backbone without external tools or skill documents. The library is evolved with Sol alone and then transferred: Qwen uses it unchanged, and Gemma uses the same core tools and skills with backbone-specific interface adaptations (Appendix B). We compare with GPT-6 Astra (base model), Meta-Harness (Lee et al., 2026), pySpatial (Luo et al., 2026), and SpatialClaw (Cho et al., 2026), which runs on our Gemma checkpoint; protocols are in Appendix B.

4.2 Main Results: Transfer across Backbones

On the three backbones, SpatialWeave improves every benchmark: all **21 benchmark–backbone comparisons** are positive, and each backbone gains more than **10 points** overall (Table 1). The gains are largest where visual evidence must be mapped to structured relations or metric quantities. SpatialWeave raises GPT-5.6 Sol to **80.75** (+10.13), well above Meta-Harness (72.13) and pySpatial (68.94). The same library raises Qwen3.8-27B from 68.47 to **80.73** (+12.26), slightly above GPT-6 Astra (base model) at 80.28, a gain that comes entirely from the external harness. Both Sol and Qwen exceed Astra overall, while Astra leads on ERQA, Omni3D-Bench, and MMSI. The harness also lifts GPT-6 Astra by 5.07 points overall (grey rows; 80.28 → **85.35**).

Table 1: **Spatial reasoning accuracy on seven benchmarks** (4,929 questions, micro accuracy, %). Bold marks the best result within each backbone group; SPATIALWEAVE rows give the gain over their base model. Grey rows are a reference with GPT-6 Astra.

Method	CV-Bench3D	BLINK	MindCube	SPBench	ERQA	Omni3D	MMSI	Overall
Questions	960	556	840	1,054	320	397	802	4,929
GPT-5.6 Sol · medium								
GPT-5.6 Sol[a]	90.10	84.71	71.67	58.98	70.00	62.82	55.86	70.62
Meta-Harness[b]	90.10	84.71	80.95	58.61	69.69	62.82	55.99	72.13
pySpatial[c]	92.92	81.29	83.69	53.46	61.25	57.30	45.39	68.94
SpatialWeave	**96.25**	**91.19**	**93.10**	**75.65**	**71.25**	**62.97**	**61.35**	**80.75** (+10.13)
Qwen3.8-27B · xhigh								
Qwen3.8-27B	93.44	83.45	70.24	55.42	65.94	65.16	46.13	68.47
SpatialWeave	**96.77**	**92.81**	**90.83**	**76.51**	**68.44**	**68.44**	**59.10**	**80.73** (+12.26)
Gemma-4-31B-it-FP8								
Gemma-4-31B	91.77	77.88	64.88	53.83	58.13	53.88	37.41	63.43
SpatialClaw	94.69	84.35	78.21	**67.59**	57.19	54.11	**51.75**	72.23
SpatialWeave	**94.79**	**90.65**	**84.17**	66.68	**58.75**	**54.94**	49.50	**73.58** (+10.16)
Reference · GPT-6 Astra, medium								
GPT-6 Astra (base model)	96.77	86.51	88.33	62.38	82.50	73.45	73.82	80.28
+ SPATIALWEAVE	97.40	92.09	94.64	77.46	81.56	72.70	74.69	85.35 (+5.07)

Group rows give the backbone and its reasoning setting. Unmarked backbone rows are base models without tools. Gemma systems share one FP8 checkpoint; interaction budgets are system-specific. [a]Shared answer normalization. [b]Conditional visual review; mixed provider metadata. [c]Reevaluated after fixes and task adaptation.

Table 2: **Step-wise capability construction with GPT-5.6 Sol** (4,929 questions). Each version adds one revision to the previous library; v0–v6 match Figure 1d.

Ver.	Configuration	Operator	Executable library	Skill guidance	Overall
v0	GPT-5.6 Sol (base model)	–	None	None	70.62
v1	Perception tools	Expand	5 tools	None	73.61
v2	Extended tools	Expand	11 tools	None	74.70
v3	Generic skills	Guide	11 tools	Generic	77.06
v4	Specialized tools	Expand	16 tools	Generic	77.26
v5	Executable workflows	Compile	16 tools + 3 workflows	Generic	79.64
v6	SpatialWeave	Refine	16 tools + 3 workflows	Evolved	**80.75**
	+ procedure guidance		16 tools + 3 workflows	Evolved + $\mathcal{P}$	<u>80.46</u>

Perception tools: detection, segmentation, depth, reconstruction, and image inspection. Generic: initial Basic and Tools skills; evolved: evolved Basic, Tools, and Lessons skills; $\mathcal{P}$: procedure scaffold.

The comparison with SpatialClaw holds the backbone fixed. On the same Gemma checkpoint, SPATIALWEAVE raises the score from 63.43 to **73.58** (+10.16), **1.35 points** above SpatialClaw (72.23) and leads SpatialClaw on five of seven benchmarks. Section 4.4 compares their structure.

4.3 HOW THE HARNESS EVOLVES

Step-wise construction. We evaluate each cumulative harness in the order its components were introduced during evolution (v0–v6 in Figure 1d) on the same 4,929 questions (Table 2). Five perception tools lift Sol from 70.62 to 73.61, and six more reach 74.70. Two controlled increments then dominate. Generic Basic and Tools skills (Guide, v3) add +2.36 without changing the executable library. With skills fixed, five further specialized computation and diagnostic tools add only +0.20 (v4), whereas compiling recurring chains into three workflows (Compile, v5) adds +2.37

Table 3: **Component analysis with GPT-5.6 Sol.** (a) Removals from SPATIALWEAVE + procedure guidance (four benchmarks, 2,770 questions); (b) skills on all seven benchmarks.

(a) Configuration	BLINK	MindCube	SPBench	ERQA	Overall
SpatialWeave + procedure guidance (reference)	**93.88**	89.76	<u>76.20</u>	69.69	<u>83.11</u>
without Basic skill	<u>92.63</u>	<u>90.60</u>	75.87	69.38	82.95
without Tools skill	92.09	89.88	**76.37**	**71.88**	83.10
without Lessons skill	**93.88**	80.71	76.02	65.31	79.79
without workflows	88.13	88.81	75.63	70.63	81.56
Tools + workflows (no skills)	<u>92.63</u>	82.74	69.40	68.44	78.00
SpatialWeave (without procedure guidance)	91.19	**93.10**	75.65	<u>71.25</u>	**83.55**

(b) Configuration	CV-Bench3D	BLINK	MindCube	SPBench	ERQA	Omni3D	MMSI	Overall
Tools + workflows (no skills)	96.15	**92.63**	82.74	69.40	68.44	**64.69**	**61.72**	77.81
SpatialWeave	**96.25**	91.19	**93.10**	**75.65**	**71.25**	62.97	61.35	**80.75**

All rows keep the 16 tools. Best per column in bold, second underlined in (a). The ERQA reference in (a) is the paired component run (69.69; seven-benchmark run: 72.50).

to 79.64, including $+9.17$ on SPBench and $+2.52$ on BLINK; compilation barely changes output length (Appendix E). SPATIALWEAVE (Refine, v6), with evolved Basic, Tools, and Lessons skills and no scaffold, reaches 80.75, 7.15 points above the perception-only library. Beyond the core perception tools, more interfaces plateau; the further gains come from how computations are composed and guided.

Workflows compute, Lessons interpret. The component ablations in Table 3 confirm this split. Panel (a) removes one component at a time from SPATIALWEAVE + procedure guidance on BLINK, MindCube, SPBench, and ERQA (2,770 questions), changing only the named skill package or the workflow registrations; its paired ERQA control scores 69.69 (72.50 in the seven-benchmark run of Sections 4.3 and 4.4). Removing the three workflows lowers the score from 83.11 to 81.56 (-1.55), with the largest loss on BLINK (-5.76), where camera motion and relocated-view relations need a fixed computation. Removing Lessons lowers it to 79.79 (-3.32), including -9.05 on MindCube, where the computation is simple but its reading is not, and removing all skills costs the most (-5.11, to 78.00). Because the two losses fall on different benchmarks, workflows and Lessons are complementary, not redundant. By contrast, removing the Basic or Tools skill alone changes the aggregate by only -0.16 and -0.01, and removing the procedure scaffold gives the best score, 83.55 ($+0.44$): guidance content matters, not package count. Panel (b) isolates skills on all seven benchmarks. With the tools, workflows, and interface descriptions fixed, the three skill packages add 2.94 points over Tools + workflows (77.81 $\rightarrow$ 80.75), led by MindCube ($+10.36$) and SPBench ($+6.25$); the unguided configuration is higher on BLINK, Omni3D-Bench, and MMSI.

Figure 3 shows both mechanisms on recorded runs. In (a), on an MMSI question whose two photos share no identifiable landmark, the libraries without skills or without Lessons trust a stable camera-motion readout (yaw -72°) and answer *acute*, whereas SPATIALWEAVE applies the Lessons rule that similar-looking categories alone are insufficient and correctly answers that the angle cannot be determined; in (b), a compiled workflow halves the atomic chain on SPBench #SI_105 (eight calls without workflows, four with) for the same correct answer. On MMSI #69, the library without Lessons anchors directions on the wrong landmark and answers incorrectly after eight calls, while SPATIALWEAVE answers correctly after five.

4.4 THE PROCEDURE SCAFFOLD AND EFFICIENCY

The scaffold explores what Compile consolidates. We measure the scaffold on the evaluation trajectories of the same frozen library with and without $\mathcal{P}$; since the scaffolded runs already include the three workflows, this is an analogue of evolution-time Explore, not a replay of it. On the same 4,929 questions, Sol with the scaffold produces 803 distinct call chains (consecutive repeats merged) vs. 575 without it, and 168 vs. 145 distinct tool transitions; Qwen, which calls tools densely either way, produces 1,452 vs. 1,346 chains. The gap comes mostly from higher tool coverage and longer

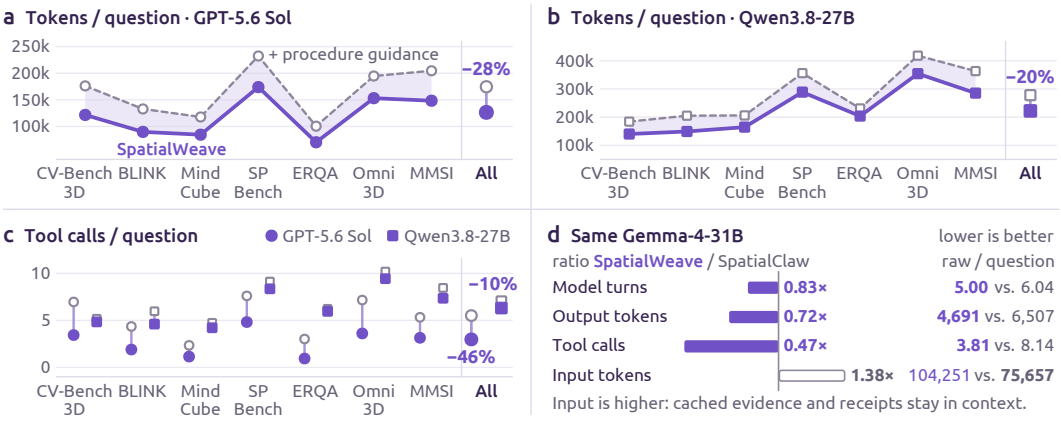

Figure 4: **Structural efficiency.** (a–c) Tokens and spatial-tool calls per question for SPATIALWEAVE with and without the procedure scaffold at inference. (d) SPATIALWEAVE vs. SpatialClaw on the same Gemma-4-31B checkpoint as the per-question ratio (lower is better): fewer turns, output tokens, and tool calls, but $1.38\times$ as many input tokens.

chains, not richer compositions at a fixed chain length. The chains the scaffold elicits are those that Compile consolidates: on CV-Bench3D, Sol trajectories with a reconstruction-then-geometry chain (3D reconstruction followed by a geometric query) number 177 with the scaffold and 1 without it, while pair-measurement workflow calls rise from 6 with the scaffold to 120 without it (Qwen: 403 to 12 trajectories, 193 to 632 calls), at an unchanged CV-Bench3D score (Sol 96.04 vs. 96.25).

Removing it at inference cuts cost. Removing the scaffold preserves accuracy (Sol 80.46 vs. 80.75; Qwen 80.93 vs. 80.73), and without it every structural cost falls (Figure 4a–c): Sol uses 34.93% fewer output tokens, 27.58% fewer total tokens, 21.42% fewer model calls, and 46.03% fewer tool calls (Qwen: 18.07%, 20.13%, 13.63%, and 10.35%). Workflow calls hold steady or rise while atomic calls fall (Appendix D): the compiled workflows carry the computation the scaffold spelled out step by step, as on CV-Bench3D #1047 (Figure 3b), where nine calls become four for the same answer.

Structure and reasoning profiles. Against SpatialClaw on the same Gemma checkpoint (Figure 4d), SPATIALWEAVE scores higher with fewer model turns, output tokens, and tool calls, but reads $1.38\times$ as many input tokens, since every turn carries the skill context, cached evidence, and earlier tool receipts; it trades a longer context for fewer turns and calls, not for fewer total tokens. Qwen and Sol reach nearly the same accuracy through different profiles: Qwen at xhigh thinking calls tools on 98% of questions, Sol at medium effort on 72% (Appendix D). Relative to the base models, these gains cost more inference (Appendix E).

5 CONCLUSION

We introduced SPATIALWEAVE, which keeps the solver frozen and co-evolves its harness of tools, executable workflows, and skills. During evolution, a procedure scaffold drives broad tool composition and recurring chains are compiled into workflows; the scaffold is removed by design once evolution ends. Evolved with GPT-5.6 Sol alone, the harness improves all 21 benchmark–backbone pairs across three backbones, by more than 10 points overall for each, and also lifts GPT-6 Astra by 5.07 points. Beyond a core perception toolset, its gains come from compilation and guidance rather than more tools, and the deployed harness makes up to 46% fewer tool calls than its scaffolded counterpart without losing accuracy. Spatial reasoning can thus improve through an external, inspectable harness that transfers across models. The harness still inherits the errors of the perception and reconstruction models it wraps, and our benchmarks use static images; stronger models can be swapped in behind the same interfaces, and extending co-evolution to video and embodied settings is a natural next step.

AI USE STATEMENT

We used large language models for writing, literature search, research execution, and drafting. For writing, we used Claude to polish the language of the abstract, introduction, method, and experiments, to unify terminology, captions, and citation formatting, and to draft some paragraphs and the implementation details in the appendix; the authors then substantially revised every passage. For literature search, the model helped locate related work on spatial reasoning benchmarks, tool-augmented agents, and skill and workflow evolution; the authors verified the source and content of every cited reference. For research execution, two uses should be distinguished. First, SpatialWeave studies LLM-based spatial reasoning agents, so the solver models in our experiments (GPT-5.6 Sol, Qwen3.8-27B, Gemma-4-31B, and GPT-6 Astra) are part of the system under study. Second, harness co-evolution combines automated trajectory mining with agent-generated revisions: all candidate tool, workflow, and skill revisions were generated and implemented by Codex, and the authors only decided which candidates to retain after verification. Experiment scripts, data analysis, and plotting code were also written with model assistance and were reviewed and tested by the authors. The core ideas, method design, and conclusions of this paper were conceived and determined by the authors, and every reported number was checked against the original run records. The authors take full responsibility for the content of this paper.

REPRODUCIBILITY STATEMENT

Section 3 specifies the harness and the co-evolution procedure, and Section 4 the backbones, reasoning settings, data split, and scoring. The appendices give the remaining details: workflow and skill cards (Appendix A), per-benchmark scoring rules (Appendix B), full usage tables (Appendix E), skill package excerpts (Appendix G), solver prompts and evolution details (Appendix H), and the complete tool and workflow interfaces (Appendix I). All seven benchmarks are publicly available. GPT-5.6 Sol and GPT-6 Astra are accessed through commercial APIs, so exact outputs may vary with provider updates. We plan to publicly release the implementation, including the frozen harness, to support reproduction and further research.

REFERENCES

Lakshya A. Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J. Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. In *The Fourteenth International Conference on Learning Representations*. OpenReview.net, 2026. URL https://openreview.net/forum?id=RQm2KQTM5r.

Tianle Cai, Xuezhi Wang, Tengyu Ma, Xinyun Chen, and Denny Zhou. Large language models as tool makers. In *The Twelfth International Conference on Learning Representations*. OpenReview.net, 2024. URL https://arxiv.org/abs/2305.17126.

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. URL https://openaccess.thecvf.com/content/CVPR2024/html/Chen_SpatialVLM_Endowing_Vision-Language_Models_with_Spatial_Reasoning_Capabilities_CVPR_2024_paper.html.

Siyi Chen, Mikaela Angelina Uy, Chan Hee Song, Faisal Ladhak, Adithyavairavan Murali, Qing Qu, Stan Birchfield, Valts Blukis, and Jonathan Tremblay. SpaceTools: Tool-augmented spatial reasoning via double interactive RL. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 37109–37120, 2026a. URL https://openaccess.thecvf.com/content/CVPR2026/html/Chen_SpaceTools_Tool-Augmented_Spatial_Reasoning_via_Double_Interactive_RL_CVPR_2026_paper.html.

Zeren Chen, Xiaoya Lu, Zhijie Zheng, Pengrui Li, Lehan He, Yijin Zhou, Jing Shao, Bohan Zhuang, and Lu Sheng. Geometrically-constrained agent for spatial reasoning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 38689–38699,

2026b. URL https://openaccess.thecvf.com/content/CVPR2026/html/Chen_Geometrically-Constrained_Agent_for_Spatial_Reasoning_CVPR_2026_paper.html.

Seokju Cho, Ryo Hachiuma, Abhishek Badki, Hang Su, Byung-Kwan Lee, Chan Hee Song, Sifei Liu, Subhashree Radhakrishnan, Seungryong Kim, Yu-Chiang Frank Wang, and Min-Hung Chen. SpatialClaw: Rethinking action interface for agentic spatial reasoning, 2026. URL https://arxiv.org/abs/2606.13673.

Junqing Du, Fernando Ropero, Erkin Turkoz, Yanfeng Zhang, and Lu Liu. GraFT: A training-free framework for spatial reasoning in multimodal large language models via 3D scene graphs, 2026. URL https://arxiv.org/abs/2609.03892.

Runnan Fang, Yuan Liang, Xiaobin Wang, Jialong Wu, Shuofei Qiao, Pengjun Xie, Fei Huang, Huajun Chen, and Ningyu Zhang. Memp: Exploring agent procedural memory. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 17490–17502, San Diego, California, United States, July 2026. Association for Computational Linguistics. doi: 10.18653/v1/2026.findings-acl.866. URL https://aclanthology.org/2026.findings-acl.866/.

Xingyu Fu, Yushi Hu, Bangzheng Li, Yu Feng, Haoyu Wang, Xudong Lin, Dan Roth, Noah A. Smith, Wei-Chiu Ma, and Ranjay Krishna. BLINK: Multimodal large language models can see but not perceive. In *Computer Vision – ECCV 2024*, pp. 148–166. Springer Nature Switzerland, 2024. doi: 10.1007/978-3-031-73337-6_9. URL https://arxiv.org/abs/2404.12390.

Gemini Robotics Team. Gemini Robotics: Bringing AI into the physical world. Technical report, Google DeepMind, 2025. URL https://storage.googleapis.com/deepmind-media/gemini-robotics/gemini_robotics_report.pdf.

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. doi: 10.1109/CVPR52729.2023.01436. URL https://openaccess.thecvf.com/content/CVPR2023/html/Gupta_Visual_Programming_Compositional_Visual_Reasoning_Without_Training_CVPR_2023_paper.html.

Jixuan He, Xueting Li, Chieh Hubert Lin, and Ming-Hsuan Yang. Reasmory: 3D reconstruction as explicit memory for VLMs spatial reasoning, 2026. URL https://arxiv.org/abs/2606.00963.

Shengran Hu, Cong Lu, and Jeff Clune. Automated design of agentic systems. In *The Thirteenth International Conference on Learning Representations*. OpenReview.net, 2025. URL https://arxiv.org/abs/2408.08435.

Chan Yeong Hwang, Miso Choi, Sunghyun On, Jinkyu Kim, and Jungbeom Lee. SpatiO: Adaptive test-time orchestration of vision-language agents for spatial reasoning, 2026. URL https://arxiv.org/abs/2604.21190.

Mengdi Jia, Zekun Qi, Shaochen Zhang, Wenyao Zhang, Xinqiang Yu, Jiawei He, He Wang, and Li Yi. OmniSpatial: Towards comprehensive spatial reasoning benchmark for vision language models. In *The Fourteenth International Conference on Learning Representations*. OpenReview.net, 2026. URL https://openreview.net/forum?id=6nZKT2rL0H.

Yoonho Lee, Roshen Nair, Qizheng Zhang, Kangwook Lee, Omar Khattab, and Chelsea Finn. Meta-Harness: End-to-end optimization of model harnesses, 2026. URL https://arxiv.org/abs/2603.28052.

Haoyuan Li, Zhengdong Hu, Jun Wang, Hehe Fan, and Yi Yang. Skill-3D: Evolving scene-aware skills for agentic 3D spatial reasoning, 2026a. URL https://arxiv.org/abs/2606.07436.

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 *The Fourteenth International Conference on Learning Representations*. OpenReview.net, 2026b. URL https://openreview.net/forum?id=KtrFXlvgrK.

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 *Computer Vision – ECCV 2024*, pp. 38–55. Springer Nature Switzerland, 2024. doi: 10.1007/978-3-031-72970-6_3. URL https://arxiv.org/abs/2303.05499.

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*. OpenReview.net, 2026. URL https://openreview.net/forum?id=yv15C8ql24.

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. doi: 10.1109/CVPR52734.2025.01811. URL https://openaccess.thecvf.com/content/CVPR2025/html/Marsili_Visual_Agentic_AI_for_Spatial_Reasoning_with_a_Dynamic_API_CVPR_2025_paper.html.

Jingwei Ni, Yihao Liu, Xinpeng Liu, Yutao Sun, Mengyu Zhou, Pengyu Cheng, Dexin Wang, Erchao Zhao, Xiaoxi Jiang, and Guanjun Jiang. Trace2Skill: Distill trajectory-local lessons into transferable agent skills, 2026. URL https://arxiv.org/abs/2603.25158.

Luigi Piccinelli, Christos Sakaridis, Yung-Hsu Yang, Mattia Segu, Siyuan Li, Wim Abbeloos, and Luc Van Gool. UniDepthV2: Universal monocular metric depth estimation made simpler. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(3):2354–2367, 2026. doi: 10.1109/TPAMI.2025.3628473. URL https://doi.org/10.1109/TPAMI.2025.3628473.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment anything in images and videos. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=Ha6RTeWMd0.

Shouwei Ruan, Bin Wang, Zhenyu Wu, Qihui Zhu, Yuxiang Zhang, Jingzhi Li, Yubin Wang, and Xingxing Wei. AlloSpatial: Agentic harness framework for spatial reasoning in foundation models, 2026. URL https://arxiv.org/abs/2606.08952.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems*, volume 36, pp. 68539–68551. Curran Associates, Inc., 2023. URL https://arxiv.org/abs/2302.04761.

Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 36, pp. 8634–8652. Curran Associates, Inc., 2023. URL https://arxiv.org/abs/2303.11366.

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. 11854–11864, 2023. doi: 10.1109/ICCV51070.2023.01092. URL https://doi.org/10.1109/ICCV51070.2023.01092.

Shi-Yu Tian, Zhuo-Xia Wang, Xuan-Yi Zhu, Zhi Zhou, Xinwei Yang, Kun-Yang Yu, Ming Yang, Yang Chen, and Yu-Feng Li. Self-evolving neuro-symbolic skills for tool-augmented spatial reasoning, 2026a. URL https://arxiv.org/abs/2608.07955.

Shi-Yu Tian, Zhi Zhou, Kun-Yang Yu, Ming Yang, Yang Chen, Ziqiao Shang, Lan-Zhe Guo, and Yu-Feng Li. LAST: Leveraging tools as hints to enhance spatial reasoning for multimodal large language models, 2026b. URL https://arxiv.org/abs/2604.09712.

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, pp. 87310–87356. Curran Associates, Inc., 2024. doi: 10.52202/079017-2771. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/9ee3a664ccfeabc0da16ac6f1f1cfe59-Abstract-Conference.html.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models. *Transactions on Machine Learning Research*, 2024a. URL https://arxiv.org/abs/2305.16291.

Peidong Wang, Zhiming Ma, Ying Chang, Xufang Luo, Yiqun Zhang, Zihan Wang, Xiaocui Yang, Shi Feng, Yuqing Yang, and Dongsheng Li. Self-evolving embodied agents via skill-harness evolution, 2026a. URL https://arxiv.org/abs/2608.11350.

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*. OpenReview.net, 2026b. URL https://openreview.net/forum?id=0FhrtdKLtD.

Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 20697–20709, 2024b. URL https://arxiv.org/abs/2312.14132.

Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Permutation-equivariant visual geometry learning. In *The Fourteenth International Conference on Learning Representations*. OpenReview.net, 2026c. URL https://openreview.net/forum?id=DTQIjngDta.

Zhiruo Wang, Graham Neubig, and Daniel Fried. TroVE: Inducing verifiable and efficient toolboxes for solving programmatic tasks. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 51177–51191. PMLR, 2024c. URL https://proceedings.mlr.press/v235/wang24az.html.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 63897–63911. PMLR, 2025. URL https://proceedings.mlr.press/v267/wang25bx.html.

Shengguang Wu, Xiaohan Wang, Yuhui Zhang, Hao Zhu, and Serena Yeung-Levy. Transductive visual programming: Evolving tool libraries from experience for spatial reasoning. In *The Fourteenth International Conference on Learning Representations*. OpenReview.net, 2026. URL https://openreview.net/forum?id=XCW1l9qcxy.

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. doi: 10.1109/CVPR52734.2025.00994. URL https://openaccess.thecvf.com/content/CVPR2025/html/Yang_Thinking_in_Space_How_Multimodal_Large_Language_Models_See_Remember_CVPR_2025_paper.html.

Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth Anything: Unleashing the power of large-scale unlabeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10371–10381, 2024. URL https://arxiv.org/abs/2401.10891.

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 *The Fourteenth International Conference on Learning Representations*. OpenReview.net, 2026a. URL https://openreview.net/forum?id=gHRoX4vXm3.

Yifan Yang, Ziyang Gong, Weiquan Huang, Qihao Yang, Ziwei Zhou, Zisu Huang, Yan Li, Xuemei Gao, Qi Dai, Bei Liu, Kai Qiu, Yuqing Yang, Dongdong Chen, Xue Yang, and Chong Luo. SkillOpt: Executive strategy for self-evolving agent skills, 2026b. URL https://arxiv.org/abs/2605.23904.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *The Eleventh International Conference on Learning Representations*. OpenReview.net, 2023. URL https://openreview.net/forum?id=WE_vluYUL-X.

Jiayi Zhang, Jinyu Xiang, Zhaoyang Yu, Fengwei Teng, Xiong-Hui Chen, Jiaqi Chen, Mingchen Zhuge, Xin Cheng, Sirui Hong, Jinlin Wang, Bingnan Zheng, Bang Liu, Yuyu Luo, and Chenglin Wu. AFlow: Automating agentic workflow generation. In *The Thirteenth International Conference on Learning Representations*. OpenReview.net, 2025. URL https://openreview.net/forum?id=z5uVAKwmjf.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Y. Zou, and Kunle Olukotun. Agentic context engineering: Evolving contexts for self-improving language models. In *The Fourteenth International Conference on Learning Representations*. OpenReview.net, 2026a. URL https://openreview.net/forum?id=eC4ygDs02R.

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, 2026b. URL https://arxiv.org/abs/2601.13029.

Zhehao Zhang, Ryan A. Rossi, Tong Yu, Franck Dernoncourt, Ruiyi Zhang, Jiuxiang Gu, Sungchul Kim, Xiang Chen, Zichao Wang, and Nedim Lipka. VipAct: Visual-perception enhancement via specialized VLM agent collaboration and tool-use. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(43):36536–36546, 2026c. doi: 10.1609/aaai.v40i43.40976. URL https://ojs.aaai.org/index.php/AAAI/article/view/40976.

Andrew Zhao, Daniel Huang, Quentin Xu, Matthieu Lin, Yong-Jin Liu, and Gao Huang. ExpeL: LLM agents are experiential learners. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(17):19632–19642, 2024. doi: 10.1609/aaai.v38i17.29936. URL https://ojs.aaai.org/index.php/AAAI/article/view/29936.

Boyuan Zheng, Michael Y. Fatemi, Xiaolong Jin, Zora Zhiruo Wang, Apurva Gandhi, Yueqi Song, Yu Gu, Jayanth Srinivasa, Gaowen Liu, Graham Neubig, and Yu Su. SkillWeaver: Web agents can self-improve by discovering and honing skills, 2025. URL https://arxiv.org/abs/2504.07079.

A WORKFLOWS AND SKILL CARDS

The deployed harness $\mathcal{H}^\star$ contains three executable workflows and three skill packages (Basic, Tools, Lessons). Figure 5 shows one card of each kind: the `camera_motion` workflow, with its trigger, computation, and return statuses, and the Lessons rule `directed-relation-tuple`, with its trigger, interpretation rule, and fallback. Appendix A.1 describes all three workflows.

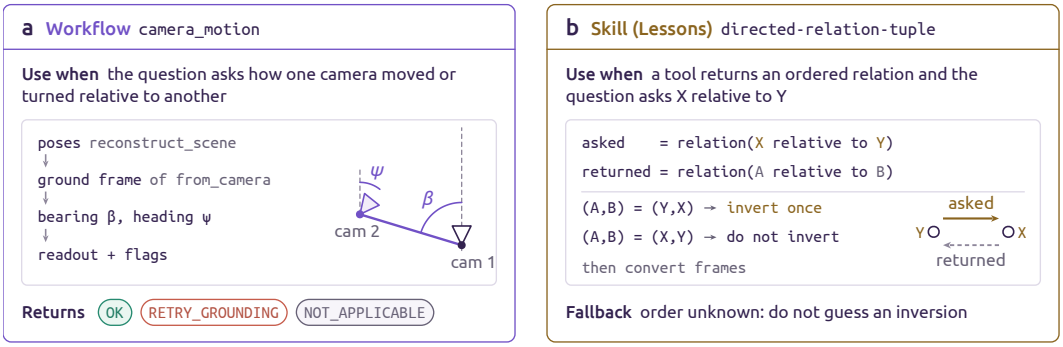

Figure 5: **Example workflow and skill cards of the deployed harness.** (a) The `camera_motion` workflow: trigger, program over cached poses, and return statuses Ω_w (Eq. (3)). (b) A Lessons rule: trigger, interpretation of an ordered relation, and fallback.

A.1 WORKFLOW DETAILS

Execution model. Each question has a workspace Z_k of images, cached depth and reconstructions, artifacts, and tool receipts that later calls reuse. A workflow call executes its program on the bindings and the workspace, $(o_w, Z_{k+1}) = \mathrm{Exec}(P_w; b_w, Z_k, \mathcal{T}_r, G_w)$ with $o_w \in \Omega_w$. Along an executed path, each operation consumes the bindings and preceding results, $z_\ell = f_\ell(\beta_\ell(b_w, Z_k, z_{<\ell}))$: it reads cached evidence, invokes a geometric implementation, transforms coordinates or units, or produces a task-relevant readout, and conditional checks in G_w add diagnostics or terminate an inapplicable computation. The three workflows are Python functions over the workspace; they make no language-model or perception-model calls, while evidence acquisition and semantic binding (object anchors, image order, observer roles) stay with the solver.

Evolution records. Development trajectories record, per step, the observation, history, action, arguments, returns, artifacts, predictions, and development scores, $\tau = ((x, h_k, a_k, o_k))_{k=1}^{K}$; a chain $c = (t_1, \ldots, t_m)$ has support $n_r(c) = |\{\tau \in \mathcal{B}_r : c \sqsubseteq \tau\}|$ (Eq. (5)). Verification records $m_r = (\Delta J, \mathrm{repair}, \mathrm{regress}, \mathrm{exec}, \mathrm{cost})$ for each candidate; selection can retain, revise, reject, or merge candidates, and merged candidates are evaluated jointly.

Camera motion. `camera_motion` reads reconstructed camera-to-world poses, checks image order and camera identities, and constructs a reference ground frame. It projects displacement and orientation into this frame to report translation bearing, yaw, pitch, and qualitative directions. Separating displacement from orientation is essential when the camera moves in one direction while turning in another. The workflow handles missing or degenerate poses explicitly; near-pure rotation suppresses the translation label while preserving rotational readouts. Small baselines and inconsistent up axes are returned as diagnostic flags.

Pair measurement. `measure_pair` connects a geometric measurement to the quantity requested in the question. It takes grounded anchors, question text, and optional scale information, invokes the distance implementation once, and applies ordered wording rules to select center distance, closest-surface distance, ground distance, or depth. It then converts the selected quantity to the requested unit. Its single-image branch can estimate ground-related quantities from cached point geometry. Unavailable closest-surface evidence or overlapping near-contact regions trigger re-grounding, while scale information accompanies the numerical readout. The resulting return exposes both the selected quantity and the evidence supporting it.

Relation readout. `relation_readout` makes the frame of a spatial relation explicit. In multiple views, it uses observer, facing, and target bindings to invoke the relative-position implementation, then maps the resulting bearing to a directional sector. In a single image, it combines horizontal and cached-depth relations before applying the sector mapping. Reference-frame, overlap, and local-stability checks accompany the result. When a full point map is unavailable, the single-image branch uses an approximate horizontal quantity derived from image displacement and depth.

Readouts and recovery. A workflow returns status, task-relevant readouts, diagnostics, and evidence sources as an MCP text record. `OK` indicates that the implemented computation produced a readout; `RETRY_GROUNDING` requests revised evidence bindings; `NOT_APPLICABLE` identifies an unmet prerequisite or unavailable computation. Retry records can carry diagnostic numbers that are not answers. Skills connect these conditions to the agent's next action: inspect the evidence, refine an anchor, use a supported alternative, or proceed to interpretation.

B Evaluation Details

Scoring rules. Table 4 lists the per-question rules of every benchmark. Each evaluation item carries a scoring contract, fixed before any prediction from the benchmark and the answer type of its source record. *Exact option match*: the key is one letter in A–H, and the prediction scores 1 if the parsed letter equals it and 0 otherwise; Omni3D-Bench yes/no questions are posed as two options. *MRA*: for ground truth y and parsed number $\hat{y}$, $e = |\hat{y} - y|/y$ and

$$s = \tfrac{1}{10} \textstyle\sum_{\theta \in \{0.50, 0.55, \ldots, 0.95\}} \mathbf{1}[e \leq 1 - \theta] \,,$$

so an answer within 5% of the truth scores 1, one within 50% scores at least 0.1, and one off by more than 50% scores 0. SPBench uses the inclusive comparison ($\leq$) and Omni3D-Bench the strict one ($<$), each on the binary64 thresholds of its reference scorer; SPBench counting questions are numerical and are scored by MRA, whereas Omni3D-Bench integer answers (52 of 56 are counts) must match exactly after numerical parsing (`3` and `3.0` agree).

Answer parsing and failures. Only a deliberate final-answer position is read: the whole response, a final `<answer>` tag, a trailing `\boxed{}`, or the last non-empty line, optionally prefixed by "Final answer:" or "Answer:"; for options, a bare, parenthesized, or dotted letter is accepted there. Letters and numbers inside the reasoning are never used, ambiguous endings (e.g., "A or B") are unparsable, and numerical payloads must be one finite literal without units, ranges, or fractions. The Gemma base-model run also accepts an explicit option label followed by its text on the final line (e.g., "(B) left"); normalization of the Sol base model is described below. An unparsable answer, an empty response, a provider refusal, or an unresolved model failure scores 0 and remains in the denominator.

Aggregation. Every benchmark score and the overall score are micro averages over questions, $100 \sum_i s_i/n$, with numerical partial credit retained; the overall score is therefore the question-weighted mean of the seven benchmark scores. We do not report the category-macro variants of BLINK (mean over its five subtypes) and SPBench (mean of its single-view and multi-view category means), which weight questions unequally.

Data overlap. The development and evaluation partitions are disjoint by question ID, not by scene: 340 SPBench evaluation questions share development image paths and 685 share scene identifiers, and 119 Omni3D-Bench evaluation questions share images and scenes.

Reference systems. Meta-Harness uses a frozen spatial adaptation with conditional visual review and no external tools. pySpatial is reevaluated after documented implementation fixes and task adaptation. SpatialClaw uses the same Gemma-4-31B-it-FP8 checkpoint as our Gemma runs. Our Gemma runs use the same core tool and skill inventory as Sol and Qwen, with backbone-specific interface and activation refinements. All three enable thinking with a 32,768-token limit per main generation, with interaction budgets following each system's protocol. Each condition keeps one scored outcome per question; missing answers and model failures count in the denominator.

Table 4: **Per-question scoring rules** on the 4,929 evaluation questions. MRA: mean relative accuracy over ten relative-error tolerances $0.50, 0.45, \ldots, 0.05$.

Benchmark	Questions	Answer type	Per-question score
CV-Bench3D	960	options	exact option match
BLINK (five subtypes)	556	options	exact option match
MindCube	840	options	exact option match
ERQA	320	options	exact option match
MMSI-Bench	802	options	exact option match
SPBench	361	options (relative direction and distance)	exact option match
	693	number (size, distance, count)	MRA, $e \leq$ tolerance
Omni3D-Bench	126	options (yes/no)	exact option match
	215	real number	MRA, $e <$ tolerance
	56	integer	exact numerical match

Prediction reuse and normalization. Completed frozen predictions are reused when their configuration and question IDs match; infrastructure recovery is recorded separately. Shared answer normalization recovers one previously unparsed correct Sol base-model answer on CV-Bench3D (865/960). Requests to the Sol service specify GPT-5.6 Sol, while some returned receipts report `gpt-6-sol`. Some evaluation questions appeared in earlier baseline experiments; they were excluded from the candidate-mining partitions. The construction and efficiency studies use the retained seven-benchmark run of SPATIALWEAVE + procedure guidance (ERQA 72.50); the component panel uses its designated ERQA control run (69.69), and the other three component-panel benchmarks reuse the same predictions.

C ADDITIONAL RESULTS

These figures complement Section 4 with per-cell views of Tables 1 and 3. Figure 6 overlays each backbone's benchmark profile with and without SPATIALWEAVE: the profile expands most along MindCube and SPBench and least along ERQA and Omni3D-Bench. Figure 7 gives the gain of every benchmark–backbone cell; the smallest gains are on ERQA ($+0.62$ to $+2.50$) and Omni3D-Bench ($+0.15$ to $+3.27$), benchmarks on which GPT-6 Astra (base model) remains ahead of all three backbones with SPATIALWEAVE. Differences use unrounded scores and can differ by 0.01 from the rounded entries of Table 1. Figure 8 shows panel (a) of Table 3 as changes from the reference. Beyond the aggregate effects discussed in Section 4.3, it shows per-benchmark tradeoffs: removing the Tools skill raises ERQA by 2.19 while lowering BLINK by 1.80, removing workflows raises ERQA by 0.94, and removing the procedure scaffold lowers BLINK by 2.70 while raising MindCube by 3.33.

Table 5, the fourth panel of Figure 6, and the last row of Figure 7 apply the same harness (19 interfaces, Basic, Tools, and Lessons, no procedure guidance) unchanged to GPT-6 Astra at medium effort on all 4,929 questions. Accuracy rises from 80.28 to 85.35 (+5.07), with gains concentrated on SPBench (+15.08), MindCube (+6.31), and BLINK (+5.58), while ERQA and Omni3D-Bench, where the base model is already strong, stay within one point. Astra calls a spatial tool on 34.25% of questions overall (about 61% on SPBench and Omni3D-Bench, 11–13% on CV-Bench3D and BLINK), invoking the harness selectively rather than by default.

D INFERENCE BEHAVIOR

Figure 9 breaks down the spatial-tool calls of Section 4.4 by tool group. Without the scaffold, workflow calls make up a larger share of a smaller call budget: 8.0% of Sol's 2.97 calls per question vs. 4.0% of 5.51 with SPATIALWEAVE + procedure guidance, and 5.5% of Qwen's 6.31 vs. 3.1% of 7.04. For Sol, the scaffold mainly adds inspection calls (33.5% vs. 19.0% of calls), while Qwen's mix changes little across the two configurations. Figure 10 places the Sol and Qwen profiles of Section 4.4 side by side on common per-unit scales. Complete usage counts are in Appendix E; token counts are not compared across model vocabularies.

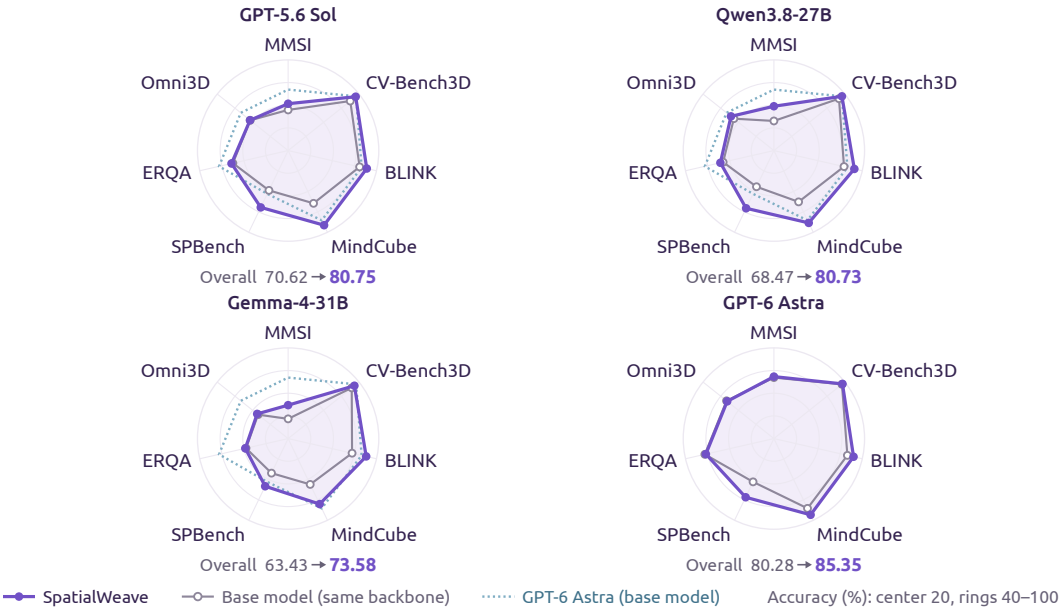

Figure 6: **Per-backbone benchmark profiles** (micro accuracy, %; radial axis starts at 20%). Solid purple: SPATIALWEAVE; thin grey: the same backbone as the base model; dotted blue: GPT-6 Astra (base model). The fourth panel applies the same harness to GPT-6 Astra.

	CV-Bench3D	BLINK	MindCube	SPBench	ERQA	Omni3D	MMSI	Overall
GPT-5.6 Sol	+6.15	+6.47	+21.43	+16.67	+1.25	+0.15	+5.49	**+10.13**
Qwen3.8-27B	+3.33	+9.35	+20.60	+21.09	+2.50	+3.27	+12.97	**+12.26**
Gemma-4-31B	+3.02	+12.77	+19.29	+12.85	+0.62	+1.06	+12.09	**+10.16**
GPT-6 Astra	+0.62	+5.58	+6.31	+15.08	−0.94	−0.76	+0.87	**+5.07**

SpatialWeave minus same-backbone base model (points, micro accuracy)

Figure 7: **Gain of SPATIALWEAVE over the same backbone as the base model** for every benchmark–backbone cell (points, 4,929 questions); the right column is the overall gain, and the last row is GPT-6 Astra with the same harness.

GPT-5.6 Sol 4 benchmarks · 2,770 questions	BLINK ref 93.88	MindCube ref 89.76	SPBench ref 76.20	ERQA ref 69.69	**4-bench overall** ref 83.11
Without Basic skill	−1.26	+0.83	−0.32	−0.31	−0.16
Without Tools skill	−1.80	+0.12	+0.17	+2.19	−0.01
Without Lessons skill	0.00	−9.05	−0.18	−4.38	**−3.32**
Without workflows	−5.76	−0.95	−0.57	+0.94	−1.55
Tools + workflows (no skills)	−1.26	−7.02	−6.79	−1.25	**−5.11**
SpatialWeave (without procedure guidance)	−2.70	+3.33	−0.54	+1.56	**+0.44**

Change vs. SpatialWeave + procedure guidance (points) lower higher

Figure 8: **Component ablations as changes from the reference** (*ref*: SPATIALWEAVE + procedure guidance; GPT-5.6 Sol, 2,770 questions on four benchmarks). The right column is the four-benchmark aggregate. ERQA changes use the paired control run (69.69); against the seven-benchmark run (72.50), the ERQA change of SPATIALWEAVE would be -1.25 instead of $+1.56$.

Table 5: **GPT-6 Astra with SPATIALWEAVE**: accuracy (micro, %; 4,929 questions) and inference behavior (per-question means over all answered questions). Differences (Δ, percentage points) are computed from unrounded scores. Tool use: questions with at least one spatial-tool call.

		Accuracy (%)			Inference behavior (per question)			
Benchmark	Questions	Base model	+ SPATIALWEAVE	Δ	Model calls	Tool calls	Tool use (%)	Tokens
CV-Bench3D	960	96.77	97.40	+0.63	4.39	0.39	11.46	86,188
BLINK	556	86.51	92.09	+5.58	4.17	0.23	13.49	80,339
MindCube	840	88.33	94.64	+6.31	5.61	0.61	28.33	109,626
SPBench	1,054	62.38	77.46	+15.08	8.46	2.40	60.82	203,609
ERQA	320	82.50	81.56	−0.94	4.89	0.39	28.75	94,485
Omni3D-Bench	397	73.45	72.70	−0.76	7.25	1.73	60.71	164,374
MMSI	802	73.82	74.69	+0.87	5.87	0.92	36.28	132,593
Overall	4,929	80.28	**85.35**	+5.07	5.95	1.04	34.25	129,018

Tokens count uncached input, output (including reasoning), and cache-read input over the completed trajectory; of the overall 129,018 tokens per question, 118,017 are cache reads, 10,695 uncached input, and 306 output. Token counts are not comparable across backbones.

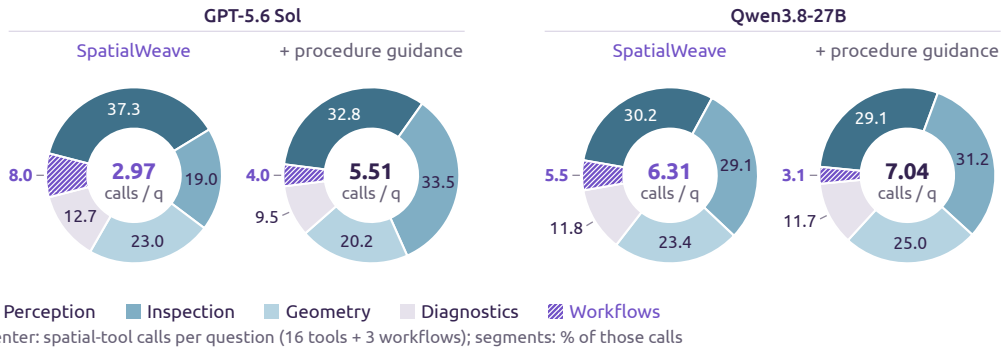

Figure 9: **Spatial-tool call mix with and without the procedure scaffold** (16 tools + 3 workflows, 4,929 questions). Ring centers give calls per question; segments are shares of that configuration's own calls.

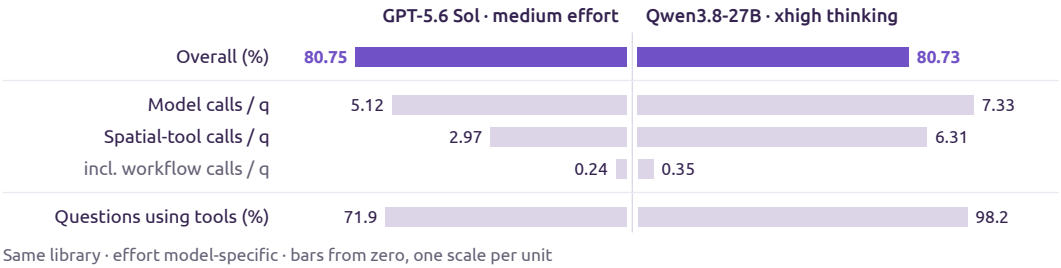

Figure 10: **Two routes to the same accuracy** with the same SPATIALWEAVE library (Sol 80.75, Qwen 80.73). Spatial-tool calls include the workflow calls shown below them.

E COMPLETE USAGE TABLES

Table 6 gives scores and inference usage for the base models, SPATIALWEAVE, and SPATIALWEAVE + procedure guidance on Sol and Qwen (Section 4.4); Table 7 gives the same quantities, with input and output tokens separated, for every version of Table 2 and every configuration of Table 3a. Tokens sum input and output over all language-model turns in the accepted trajectory; development runs, superseded predictions, and failed or recovery attempts are excluded, cached input counts once

per turn, and reasoning tokens are included in output. Model calls count language-model turns. Input tokens grow with every construction step, because each turn re-reads the skill context and the accumulated tool receipts. Workflow compilation (v4 → v5) leaves output and model calls nearly unchanged (807 vs. 820 output tokens and 4.37 vs. 4.32 model calls per question) while adding 2.37 points, but raises input from 88,885 to 105,812 tokens per question; Refine (v6) raises input further to 125,447. Only removing the procedure scaffold lowers input at matched accuracy.

Table 6: **Performance and inference usage** on all seven benchmarks (4,929 questions per configuration). Tokens sum input and output over the retained trajectory; output tokens include reasoning.

Configuration	Overall score	Tokens / question	Output tokens / question	Model calls / question	Tool calls / question	Questions using tools (%)
GPT-5.6 Sol · medium						
GPT-5.6 Sol (base model)	70.62	2,690	311	1.00	0.00	0.00
SpatialWeave + procedure guidance	80.46	174,597	1,543	6.51	5.51	95.78
SpatialWeave	80.75	126,451	1,004	5.12	2.97	71.88
change vs. + procedure guidance		−27.58%	−34.93%	−21.42%	−46.03%	
Qwen3.8-27B · xhigh						
Qwen3.8-27B (base model)	68.47	8,379	6,381	1.00	0.00	0.00
SpatialWeave + procedure guidance	80.93	277,983	12,300	8.48	7.04	99.55
SpatialWeave	80.73	222,026	10,078	7.33	6.31	98.15
change vs. + procedure guidance		−20.13%	−18.07%	−13.63%	−10.35%	

Group rows give the reasoning setting (Sol: reasoning effort; Qwen: thinking level). Percentages compare SPATIALWEAVE with SPATIALWEAVE + procedure guidance on the same backbone; token counts are not comparable across backbones. Tool-use rate: questions with at least one spatial-tool call.

Table 7: **Inference usage across construction stages and component ablations** (GPT-5.6 Sol, medium effort; per question). (a) Versions of Table 2 on all seven benchmarks (4,929 questions); (b) configurations of Table 3a on BLINK, MindCube, SPBench, and ERQA (2,770 questions).

Configuration	Overall	Input tokens	Output tokens	Model calls	Tool calls	Workflow calls	Tool use (%)
(a) Construction stages							
v0 GPT-5.6 Sol (base model)	70.62	2,378	311	1.00	0.00	0.00	0.00
v1 Perception tools	73.61	15,974	357	1.54	0.69	0.00	37.72
v2 Extended tools	74.70	27,644	385	1.94	1.33	0.00	44.67
v3 Generic skills	77.06	47,991	632	3.31	1.82	0.00	60.54
v4 Specialized tools	77.26	88,885	820	4.32	3.36	0.00	74.64
v5 Executable workflows	79.64	105,812	807	4.37	3.32	0.49	74.25
v6 SpatialWeave	80.75	125,447	1,004	5.12	2.97	0.24	71.88
+ procedure guidance	80.46	173,053	1,543	6.51	5.51	0.22	95.78
(b) Component ablations							
SpatialWeave + procedure guidance (reference)	83.11	161,926	1,379	6.18	4.77	0.21	92.71
without Basic skill	82.95	160,982	1,281	6.18	4.62	0.20	93.43
without Tools skill	83.10	147,036	1,123	5.83	4.54	0.15	92.09
without Lessons skill	79.79	159,470	1,249	6.22	4.55	0.18	97.98
without workflows	81.56	135,148	1,330	5.94	4.51	0.00	92.78
Tools + workflows (no skills)	78.00	65,237	469	2.73	2.43	0.37	55.70
SpatialWeave (without procedure guidance)	83.55	117,035	913	4.77	2.67	0.22	63.50

Tool calls include workflow calls. Tool use: questions with at least one spatial-tool call. Input includes cached input, counted once per turn.

F Complete Worked Examples

We give two complete, successful SpatialWeave trajectories of GPT-5.6 Sol (medium effort) from the evaluation runs of Table 1: a figure of the inputs and key artifacts, then a trace with one numbered step per model turn. Grey italics summarize the model's text in that turn (or quote Sol's reasoning-summary title where it wrote none); call, return, and skill lines give the tool (blue) or workflow (purple) and key arguments, the deciding receipt fields, and the skill files opened (gold) with one quoted rule, copied from the recorded trajectories ("…" marks omissions).

F.1 Case 1: Camera Translation versus Camera Rotation (BLINK)

Question. BLINK Multi-view Reasoning #110: "The images are frames from a video. The video is shooting a static scene. The camera is either moving clockwise (left) or counter-clockwise (right) around the object. … Is the camera moving left or right when shooting the video? (A) left (B) right." Ground truth: (B). The camera translates right while turning left by about 42° to keep the bottle in view, so the turn points to the wrong option.

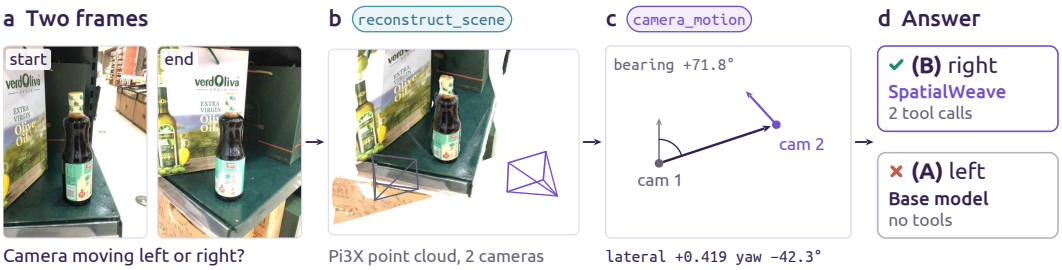

Figure 11: **Case 1, BLINK Multi-view Reasoning #110.** (a) Input frames. (b) Pi3X reconstruction with the two camera poses (grey: start, purple: end). (c) `camera_motion` readout in camera 1's ground frame: camera 2 lies at bearing +71.8° and is turned −42.3° (left). (d) SpatialWeave answers from the translation (correct); the base model answers (A).

1 *Loads the required spatial guidance before comparing the two frames.*
read `spatial-basic/SKILL.md, spatial-lessons/SKILL.md, spatial-tools/SKILL.md`
skill `spatial-tools`: *"Optional executor 'camera_motion': use this when n_images >= 2; prerequisite_tools ['reconstruct_scene'] …"*

2 *A two-view camera-motion question: reconstruction first, then the camera-motion readout.*
skill `references/reconstruction.md`: *"Before calling it, name at least one shared static landmark …"*

3 call **`reconstruct_scene`**`(image_path=[2 frames], rotation_reference_camera=1, ...)`
return `success: true; points_count: 585538; "... producing 2 camera viewpoints"`

4 *The poses put the ending camera to the first camera's right; map this to the question's convention.*
call **`camera_motion`**`(image_path=[...], from_camera=1, to_camera=2)`
return `branch: OK; "...bearing +71.8 deg -> forward-right"; "yaw -42.3 deg (left)"`
return `motion_words: {"translation_cardinal4": "right", ..., "turn": "left", ...}`

5 *Clarifying camera rotation direction … Confirming counterclockwise camera motion*
answer **B**

Outcome. SpatialWeave answers **(B) right** (correct) with 5 model calls, 2 spatial calls, and 108.7k tokens (input + cached input + output). The Tools skill routes the question to `camera_motion` once a reconstruction exists, and the workflow returns translation and turn as separate fields (`translation_cardinal4` is `right`, `turn` is `left`); the question asks about the camera's path, so the translation decides. No Lessons reference is opened. The **base model (no tools)** answers **(A) left** (incorrect) in one call without a written rationale (base-model run).

F.2 Case 2: Closest-Point Distance with Guarded Re-Grounding (SPBench)

Question. SPBench MV #297, eight views of one room: "Measuring from the closest point of each object, what is the distance between the door and the telephone (in meters)?" Ground truth: 0.3; scored by mean relative accuracy (MRA). The door and a wall telephone with a hanging cord sit side by side (frame 7). The box-based closest-surface distance (0.06 m) is far too small; the harness flags it, and the model re-grounds on the facing edges.

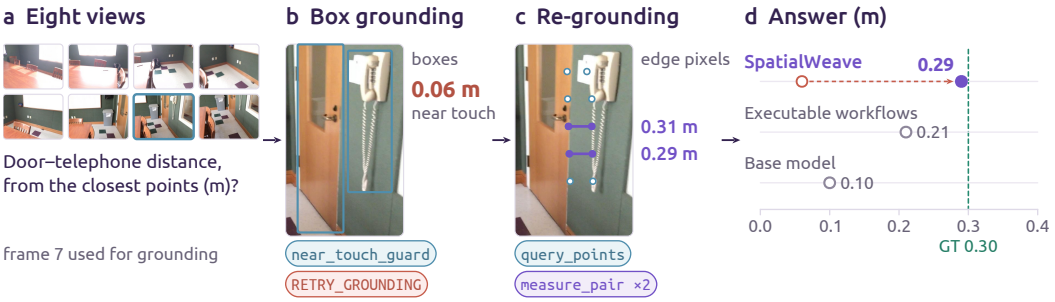

Figure 12: **Case 2, SPBench MV #297.** (a) Eight input views; both objects are grounded in frame 7. (b) On the boxes: closest point 0.06 m vs. center 0.93 m; `near_touch_guard` returns `RETRY_GROUNDING`. (c) Re-grounding on the door's facing edge and the cord: 0.31 m and 0.29 m. (d) Final answers: SpatialWeave 0.29 (MRA 1.0; open red circle: rejected box reading), v5 harness 0.21 (MRA 0.5), base model 0.10 (MRA 0.0).

1 *Loads the skill guidance, then plans to localize both objects and measure their closest points.*
read `spatial-{basic,lessons,tools}/SKILL.md`
skill `spatial-tools`: *"On RETRY_GROUNDING, diagnostic numbers are not answer values. Use … one role-preserving re-grounding pass …"*

2 *Prioritizing references for localization and reconstruction*
skill `geometry-query.md`: *"State the semantic roles … whether distance means … closest surface …"*

3 *One rigid room; the door and the wall telephone are clearest in frame 7, so ground them there.*
call `reconstruct_scene(8 images, rotation_reference_camera=7, ...); estimate_depth, inspect_image(frame 7)`
call `detect_objects(frame 7, text_prompt="wooden door . wall telephone")`
return `"wooden door" 0.82 [604,1,771,687]; "wall telephone" 0.68 [787,24,947,540]`

4 *The boxes do not overlap; measure the closest-point distance and apply the guard.*
call `measure_pair, near_touch_guard(point_a, point_b = both detection boxes, ...)`
return `measure_pair: value: 0.06; currency_rule: R2_closest; near_touch: True`
return `guard: branch: RETRY_GROUNDING; value: None; answer_value: None`

5 *The box minimum fails the guard; re-ground on the door's facing edge and the telephone cord.*
call `query_points(points=[[7,750,100], ..., [7,844,500]] (10 pixels))`
return `"Pairwise 3D distances ...: ... P3-P8 0.44, ... P4-P9 0.44, ..."`

6 *Assessing point distances and scaling metrics*
call `measure_pair(point_a=[7,755,300,"door facing edge"], point_b=[7,840,300,"telephone cord"], ...)`
call `measure_pair(point_a=[7,758,400,...], point_b=[7,842,400,...], ...)`
return `"closest-point ...: 3D 0.31 m"; "...: 3D 0.29 m"; near_touch: False`

7 *The two facing-edge separations are 0.31 m and 0.29 m; the closest-point minimum is 0.29 m.*
call `answer_check(answer_kind="numeric", proposed_answer="0.29", unit="m")`
return `branch: OK; label: consistent; {"agreements": 1, "contradictions": 0}`

8 answer `"Final answer: \boxed{0.29}"`

Outcome. SPATIALWEAVE answers **0.29** (ground truth 0.3; MRA 1.0) with 8 model calls, 10 spatial calls, and 326.3k tokens. The `measure_pair` workflow selects the closest-point quantity from the question wording, but on the detection boxes it reads only 0.06 m against a 0.93 m center distance. The guard withholds that value (`answer_value` is `None`) and names the fallback. The model then makes one re-grounding pass that keeps both object identities and the closest-surface quantity, as the Tools skill prescribes, and reports the smaller of the two facing-edge separations. For comparison, the **base model (no tools)** answers **0.1** (MRA 0.0). The **v5 harness** (three executable workflows, generic Basic and Tools skills; Table 2) calls `measure_pair` on tight detection boxes and accepts the box-based closest-point reading **0.21** (MRA 0.5), without running the guard.

G SKILL PACKAGES

Each skill package is a directory with an entry point `SKILL.md`, whose front matter gives the package name and a one-sentence description, optional `references/` files that the entry point routes to, and a small interface metadata file (`agents/openai.yaml`: display name, short description, default prompt). Table 8 summarizes the three deployed packages. The frozen package keeps the retained skill files byte-identical to the evaluated bundle (hashes are recorded in the snapshot manifest).

Table 8: **Skill packages of the deployed harness.** Word counts are measured on the frozen files (entry point / all references).

Package	Role	Words	References
Basic `spatial-basic`	Benchmark-agnostic reasoning checklist: entities and requested quantity, distinct reference frames, decisive evidence, exact answer format; points to `answer_check`.	151 / –	none
Tools `spatial-tools`	Tool selection, parameters, and reading of returned evidence; a trigger and prerequisites for each v4/v5 interface; binding and recovery rules, including the action for each return status.	584 / 630	`object-localization`, `region-segmentation`, `depth-estimation`, `artifact-evidence`, `reconstruction`, `geometry-query`
Lessons `spatial-lessons`	Narrow interpretation rules of three kinds: terminal contracts, evidence gates, and modifiers. Each applies only when its trigger matches exactly.	259 / 1,751	`four-view-direct-left-right`, `cross-image-connector`, `current-camera-depth-label`, `cross-image-shared-scale`, `drawn-trajectory-evidence`, `directed-relation-tuple`, `canonical-named-extent`, `counting-unit`

Basic. The entry point is the only file. It lists five steps (identify the entities and the requested quantity; keep camera, observer, object, and temporal frames distinct; identify the decisive evidence; check the exact question and format; give one final answer). It forbids inferring a dataset name or hidden label and claiming inspections that were not made.

Tools. The entry point routes by uncertainty: *"Choose a tool to resolve a named uncertainty, not merely to increase the number of calls"*, and *"Use only the one or two references relevant to the decisive uncertainty."* For each v4/v5 interface, it lists the trigger and prerequisites copied from the interface description, followed by five usage rules. The references are short checklists, one per evidence type. Two rules govern recovery:

Tools rule on retry returns (`spatial-tools/SKILL.md`)

```
- On RETRY_GROUNDING, diagnostic numbers are not answer values. Use the indicated atomic
  fallback or one role-preserving re-grounding pass; do not loop until a preferred value
  appears.
- `answer_check` compares the proposed answer with available tool evidence;
  NOT_APPLICABLE is absence of a check, not support for the answer. Tools not exposed in
  this session are unavailable.
```

Tools re-grounding rule (`spatial-tools/references/geometry-query.md`)

```
For a decisive sign, include a known-side or facing anchor when the pipeline
specifies one. One same-identity re-grounding is allowed when an anchor is noisy
or a reported range crosses an answer boundary. Do not change semantic roles,
image order, frame, or measurement definition during recovery.
```

Lessons. The entry point defines the three rule roles and lists the eight published lessons, each with its visible trigger. It ends: *"Do not generalize a lesson to nearby wording. If no trigger matches, use the other selected guidance or solve normally rather than inventing a lesson."* Every reference has the same six sections: *Use this when*, *Do not use this when*, *Contract*, *Required evidence*, *Workflow interaction*, and *Fallback*. The gate applied in Figure 3a and the argument-order rule shown in Figure 5b read:

Lessons evidence gate, Contract (`spatial-lessons/references/cross-image-connector.md`)

```
Check for at least one answer-relevant connector: the same identifiable entity
or landmark in both images, visible scene overlap, an ordered continuous
sequence whose motion constrains the asked variable, or an explicit transform
given in the question. Similar-looking categories alone are insufficient.

If no connector exists, choose the supplied indeterminate option. A tool's
reconstruction output is not itself evidence that a connector existed. If a
connector exists, the gate passes and reasoning continues with the other
selected guidance or the supplied evidence.
```

Lessons modifier, Contract (`spatial-lessons/references/directed-relation-tuple.md`)

```
Write `asked = relation(X relative to Y)` and
`returned = relation(A relative to B)`. If `(A,B)=(Y,X)`, invert the directed
relation exactly once; if `(A,B)=(X,Y)`, do not invert. Convert frames only
after argument order is correct.
```

H Solver Prompts and Evolution Details

H.1 Solver Prompts and Skill Loading

Every run uses the same short system prompt and a task prompt built from a fixed template (Prompt 1). The template varies only in the selected skills, the image list, the question, the answer-format lines, and, in the + *procedure guidance* configuration, two added scaffold lines. The runtime exposes its built-in file-read action together with the 19 interfaces. It places the selected packages under `.agents/skills/` in the per-question workspace and does not inject skill bodies into the prompt. The task prompt names the entry points. The solver opens each `SKILL.md` with the read action, and each entry point routes to its references ("Read … when …"), so the solver opens a reference only when its trigger matches the visible question. Skill text therefore enters the context on demand. In the *Tools + workflows* configuration, no skills are selected, and the skill lines are absent. The + *procedure guidance* configuration also selects the scaffold package `spatial-pipelines`; its two lines are shown at the end of the template.

Prompt 1: System prompt and task prompt template (`case.json`); angle brackets mark per-question fields.

```
--- system prompt ---
You are a spatial reasoning agent working inside a session workspace. Follow the task
  message exactly, use the available tools and skill files as instructed, and finish with
  the requested final answer format.
--- task prompt ---
Solve the spatial reasoning question using only the provided question, choices, and images.
Spatial MCP tools and the `inspect_image` MCP helper are available.
When a named, answer-relevant visual uncertainty matches a tool capability, prefer one
  targeted MCP tool or helper call over a confident guess. Tools are not required when the
  supplied evidence already settles the question.
For comparisons across multiple images, `inspect_image` can create a contact sheet; for a
  small or ambiguous region, it can crop the relevant area.
Tool results may return workspace-relative artifacts under `artifacts/` and attach
  decision-relevant images inline. Inspect any returned artifact that could affect the
  answer before drawing the final conclusion.
Do not invent tool calls or results. Do not use ad-hoc shell or Python image processing
  when its visual output is not returned for inspection.
The selected skills $spatial-basic, $spatial-lessons, $spatial-tools are part of this
  arm's execution contract. You MUST open and follow every selected SKILL.md entry point
  before solving.
The entry points are `.agents/skills/spatial-basic/SKILL.md`,
  `.agents/skills/spatial-lessons/SKILL.md`, `.agents/skills/spatial-tools/SKILL.md`.
Follow each entry point's own routing instructions. Open and execute only the linked
  references it identifies as relevant to the visible question.
The attached images are available at these session-local paths, in order:
- inputs/0001.jpg
- <...>
Use these paths when an image tool asks for image_path.

Question:
<question text and options>

--- choice questions ---
Return the final answer as exactly one option letter shown in the question.
--- numeric questions ---
<benchmark answer-format sentence, e.g. "... end with exactly one line `Final answer:
  \boxed{N}` ...">
Return the final answer as one finite numeric literal only, without units or other text.
Decimals and scientific notation are allowed. Use the unit requested by the question, and
  do not round or convert units unless the question explicitly asks you to.
--- added with the procedure scaffold (+ procedure guidance) ---
Do not skip a matching published pipeline merely because the answer seems evident.
When a published pipeline matches, you MUST execute every tool step that it marks as
  required and inspect the resulting evidence.
```

H.2 Evolution Implementation

Table 9 maps the operators of Section 3.3 to their implementation. All mining, compilation, and replay modules are deterministic Python. The one exception is the optional translation of a trajectory into a glue program, which uses a low-effort model call and falls back to parametric templates. Mining reads only development trajectories, and run prefixes of held-out evaluations are excluded in code. Coding agents generated every candidate revision (tool code, workflow programs, and skill text) from the mined reports; the researchers wrote no revision content and acted only as an acceptance gate, keeping or rejecting candidates by the replay checks, paired results, and the selection rule of Section 3.3.

Procedure scaffold. The scaffold $\mathcal{P}$ is a fourth skill package, `spatial-pipelines`: one routing entry point and ten family references (e.g., relocated observer, current camera, measurement, camera self-motion, counting identity), about 4.4k words in total. Unlike the deployed skills, it prescribes evidence procedures: *"Once entered, every required step is mandatory"*. For example, its relocated-observer reference requires `reconstruct_scene` followed by `relative_position` with `targets=[B, C]`, a check that B is straight ahead, a sector-stability test, one role-preserving re-grounding, and a sector mapping. During development, these mandatory steps produced the recurring chains that mining found (Section 4.4). The

Table 9: **Operators and their implementation.** Thresholds are those in the chain configuration.

Operator	Implementation
Explore	Scaffolded solver runs on the development splits. Each run is stored as an episode: tool calls with arguments, readouts, and SHA-256 hashes of artifacts; skill files read; assistant messages; prediction; official score.
Diagnose	Success and failure trajectories of the same question are aligned (longest common subsequence over tool and argument provenance), and the first divergence is classified as a missing decisive call, a different input source, equivalent readouts with different answers, or different readouts (D1–D4; D0 for non-tool failures). Anchor audits and tool-hygiene counterfactuals, run on stored point maps, find missing evidence or diagnostics ($\mathcal{F}_r^{\mathrm{gap}}$) and misuse ($\mathcal{F}_r^{\mathrm{use}}$).
Compile (mining)	Successful trajectories are translated into short programs in a restricted glue language that recompute the answer from parsed readouts; a program is kept only if it reproduces the trajectory's own prediction. Object names, the answer unit, and rounding become question-bound slots, bound by order of mention in the question; this is the REBIND test. Programs are canonicalized and clustered, and support $n_r(c)$ counts distinct questions ($n_{\min} = 30$; 15 for families under 60 questions). A counterfactual on stored readouts counts repairs $\hat{R}$ and regressions $\hat{G}$, and a cluster is kept if $\hat{G} \le \hat{R}/2$ and $\hat{R} - \hat{G} \ge \max(3, 0.05\, n_f)$ (recorded in $\mathcal{F}_r^{\mathrm{rep}}$).
Compile (build), Expand	A candidate comes from a mined glue cluster, a diagnostic patch, or a chain already reproduced on support trajectories. It is written as a spec (trigger, slot bindings, glue, record schema, applicability conditions, statistics) and rendered as a Python executor over the workspace that returns the shared record; it runs no perception model and no language model.
Guide / Refine	A skill overlay copies the parent skills and adds a Tools index entry (trigger, prerequisites, inputs, return fields) plus an optional-executor line in the matching scaffold reference; Lessons are added or edited as separate references. Every changed file is recorded with before/after hashes.
Verify	Check: the overlay server must register every interface with its declared schema; the frozen package is additionally validated offline, with network access blocked, for the registered interface set and schema parity. Replay: the executor is rerun on the stored, hash-verified artifacts of development episodes. It must reproduce 100% of its support trajectories, and the replay tallies repairs and regressions on the remaining episodes. Paired evaluation: the parent (A), a repeat of the parent (A′), the candidate (B), and a twin (C) with the same text but a description-only server run on the same development questions; C separates the effect of the text from that of the computation.
Freeze	Executor modules and the shared runtime are frozen with a manifest of hashes; the evaluated package is snapshotted with its server, schemas, and skills; the scaffold package is removed.

`relation_readout` spec lists steps 4–8 of this procedure as the steps it absorbs, and the workflow was attached to the reference as an optional executor. At Freeze, the whole package is removed: the deployed harness keeps Basic, Tools, and Lessons, and the same 19 interfaces (Section 4.4).

I TOOL AND WORKFLOW LIBRARY

The deployed harness registers 19 interfaces on one MCP server: 16 tools and the three workflows of Section 3.2. The solver sees each interface as a name, an argument schema, and a natural-language description, and it calls tools and workflows the same way. Tables 10 and 11 list all 19. The eight interfaces added in stages v4 and v5 share one return record: a status `branch` (`OK`, `RETRY_GROUNDING`, or `NOT_APPLICABLE`), a `value`, a `label`, a `decision_stable` flag, `evidence` lines, a `not_applicable_reason`, and interface-specific extras. Their descriptions are generated from the executor contract and state the record fields, a *Use this when* trigger with prerequisite tools, and the exact conditions for each non-`OK` status. Some descriptions also include an example call executed on a stored development workspace (Prompt 2). The 11 earlier tools return the server's own readout text, structured results, and artifact paths.

Prompt 2 excerpts two model-visible descriptions verbatim: a v1 perception tool and part of the `measure_pair` workflow. The full `measure_pair` description also lists its ordered quantity rules (R0–R5), all record fields, and every `NOT_APPLICABLE` reason.

Table 10: **The 19 interfaces of the deployed harness**, part 1: the 11 tools of stages v1 and v2. Inputs list the main arguments; arguments in brackets are optional. *Cat.*: perception (perc.), inspection (insp.), geometry (geom.), diagnostics (diag.), workflow (wf.). *Stage*: the cumulative harness that introduced the interface (v1: perception tools; v2: geometric queries and counting; v4: specialized computation and diagnostic tools; v5: workflows; Section 4.3). *status* denotes the shared return record described in the text. Anchors are `[i,x,y]` pixels, `[i,x1,y1,x2,y2]` boxes (optionally labeled), or `camera:N`; "recon. units" are arbitrary reconstruction units.

Interface	Cat.	Stage	Inputs	Returns	Purpose
`detect_objects`	perc.	v1	`image_path`, `text_prompt` (noun phrases joined by " . "); [thresholds]	boxes, labels, confidences; near-miss boxes just below threshold; visualization	Open-vocabulary object and part detection (GroundingDINO; Liu et al., 2024).
`segment_region`	perc.	v1	`image_path`; [point prompts, box]	mask and overlay artifact	Segments one object or region from point and/or box prompts (SAM 2; Ravi et al., 2025).
`estimate_depth`	perc.	v1	`image_path`	metric depth and point map, intrinsics, depth statistics	Metric monocular depth for one image (UniDepthV2; Piccinelli et al., 2026).
`reconstruct_scene`	perc.	v1	`image_path` (list); [view angles, reference camera, camera view]	point-cloud renders; numeric readout of each camera's position and facing in the reference camera's frame	Multi-view 3D reconstruction (Wang et al., 2026c); the readout is read first for camera-to-camera relations.
`inspect_image`	insp.	v1	`image_path` (list), `operation` (crop, contact sheet, rotate, enhance); [box, scale, angle]	processed image, returned inline	Lightweight 2D inspection; cannot resolve 3D layout, depth order, or camera relations.
`query_points`	geom.	v2	`image_path`, `points`	pairwise 3D distances (also across images), 3D coordinates, z_{cam}, distance to each camera (recon. units)	3D lookup of pixels, e.g., for cross-view correspondence.
`relative_position`	geom.	v2	`image_path`, `observer`, `targets`; [`facing`, `mode`]	per target: signed bearing, front/behind, left/right, above/below labels with deadbands, `UNCERTAIN` flag, distances	The "stand at A, face B: where is C?" transform on the latest reconstruction.
`turn_sequence`	geom.	v2	`image_path`, `waypoints`; [`start_facing`]	per-leg signed turn angle, binned into left, right, straight, back	Turns along a route through ordered waypoints.
`measure_distance`	geom.	v2	`image_path`, `point_a`, `point_b`; [`known_anchors`, `quantity`]	center, ground, and closest-point distances, z_{cam} of each anchor, depth difference (m); an answer-ready or estimate line	Metric distance between two anchors with server-side scale calibration and cross-view registration.
`estimate_room_area`	geom.	v2	`image_path` (all frames)	floor-footprint area (m^2), extent, convex-hull upper bound, coverage warning	Room floor area from a floor plane fitted on the reconstruction.
`count_census`	perc.	v2	`image_path` (all frames), `target`; [`synonyms`, threshold]	instance registry (frames, 3D position, best box), pairwise separations, candidate count, contact sheet	Counting: merges detections across frames into instances by 3D distance; the solver only adjudicates category membership.

Table 11: **The 19 interfaces of the deployed harness (continued):** the eight interfaces added in stages v4 and v5, which return the shared *status* record. Columns and notation as in Table 10.

Interface	Cat.	Stage	Inputs	Returns	Purpose
box_depth	geom.	v4	image_path, box_a; [box_b]	status; depth of each box, depth difference, deeper box	Dominant-mode depth inside the central 60% of a detection box; depth order of two boxes.
depth_order	geom.	v4	image_path, boxes (2–12); [labels]	status; per-box depth, order nearest to farthest, ambiguous pairs	Orders objects by distance from the camera, not by distance between objects.
marker_read	perc.	v4	image_path; [colors]	status; drawn rectangles, rings, dots with color, boxes, label crop	Finds annotations drawn on an input image with color masks (no perception model).
near_touch_guard	diag.	v4	image_path, point_a, point_b (boxes)	status; closest-point and center distances, usable/unusable	Closest-surface distance with a verdict; overlapping or nearly touching boxes return a retry.
answer_check	diag.	v4	proposed_answer; [options, answer_kind, unit]	status; agreements and contradictions with workspace readouts	Checks a proposed final answer against the readouts in the workspace; never changes the answer.
camera_motion	wf.	v5	image_path (reconstructed list), from_camera, to_camera	status; translation bearing and 8-way label, yaw and pitch, motion words, flags (pure rotation, small baseline)	Motion and turn of one camera in another camera's egocentric ground frame, from reconstructed poses.
measure_pair	wf.	v5	image_path, point_a, point_b, question; [known_anchors]	status; selected quantity and matching rule, value in the question's unit, all quantities in m	One distance computation; ordered wording rules select the quantity and unit.
relation_readout	wf.	v5	image_path, scheme; observer, facing, targets (multi-view) or target_box, reference_box (one image)	status; per-target bearing and sector, facing check	Sector of each target for a relocated or camera observer; single-image current-camera variant.

Prompt 2: Excerpts of two model-visible interface descriptions (tool_schemas.json; "[...]" marks omissions).

```
detect_objects: Open-vocabulary object and part detection on one workspace image using
  GroundingDINO. Arguments: image_path (workspace-relative); text_prompt (a compact list of
  noun phrases separated by ' . ', not the full question); optional box_threshold and
  text_threshold. [...] Boxes scoring within 0.10 below box_threshold are returned separately
  in near_miss [...]

measure_pair: one server measure_distance computation on two anchors; all currencies parsed,
  the currency the question wording asks for selected by an ordered rule table, and converted
  to the question's unit. [...] Returns branch RETRY_GROUNDING (decision_stable=false,
  value=None; [...] the measured numbers appear only as 'diagnostic, not an answer' evidence
  lines; the readout ends with an 'instead:' line) when: near_touch_overlap [...];
  closest_point_unavailable [...].
```