GeoMind: A Benchmark for Spatial Understanding in Robotic Manipulation

Jinghe Wang[1]* **Xinrui Cao**[1]* **Duo Wu**[1]* **Chenghao Gu**[1] **Yong Zhong**[2]
Linjia Kang[1] **Tianyi Xiong**[1] **Zhi Wang**[1]†
[1]Tsinghua Shenzhen International Graduate School, Tsinghua University
[2]Southern University of Science and Technology

Abstract

Recent advances in robotic manipulation have made rapid progress in mapping visual observations and language instructions to actions. However, reliable manipulation in open-world environments requires robust *spatial understanding*, the ability to identify targets from complex layouts (*spatial reasoning*) and adapt to scenes that continuously evolve through interaction (*spatial memory*). Yet existing embodied benchmarks fall short in evaluating spatial understanding: they often introduce strong visual cues that bypass spatial reasoning, and focus on static scenes that offer little insight into an agent's capacity for spatial memory. To address this gap, we introduce GeoMind, a systematic benchmark for structured spatial understanding through tabletop manipulation. GeoMind comprises 54 evaluated task variants derived from 27 non-execution variants organized around spatial reasoning and spatial memory. It also incorporates a scalable pipeline that automates the generation and annotation process of task instances with verified targets and executable interactions. Extensive evaluations across diverse embodied agents reveal their consistent weaknesses in spatial understanding. Our findings highlight that these agents struggle to track identities after spatial changes and convert relational or geometric constraints into executable actions. The proposed GeoMind benchmark will be publicly available.

1 Introduction

Advances in embodied intelligence have enabled robots to translate visual observations and language instructions into manipulation actions [1–14]. However, reliable manipulation often requires more than recognizing individual objects: robots must reason about spatial relationships to identify targets and maintain task-relevant spatial information as scenes change [15–17]. For example, a robot may be asked to pick up a block based on its position relative to other objects, or to pick up the same block after observing several visually similar blocks being rearranged. While the former requires reasoning about the current spatial configuration, the latter additionally requires tracking object identity across scene changes. Such tasks highlight the importance of spatial understanding for reliable manipulation.

Existing embodied benchmarks [18–24] have made substantial progress in evaluating language-guided manipulation and generalization. However, overall task success provides limited insight into the spatial capabilities underlying an agent's behavior. Distinctive visual or semantic cues may allow agents to identify targets without reasoning about spatial relationships. Similarly, tasks involving scene changes may be solved from the current observation without tracking previous spatial states. Conversely, manipulation failures may arise from execution errors even when the intended spatial

*Equal contribution.
†Corresponding author.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

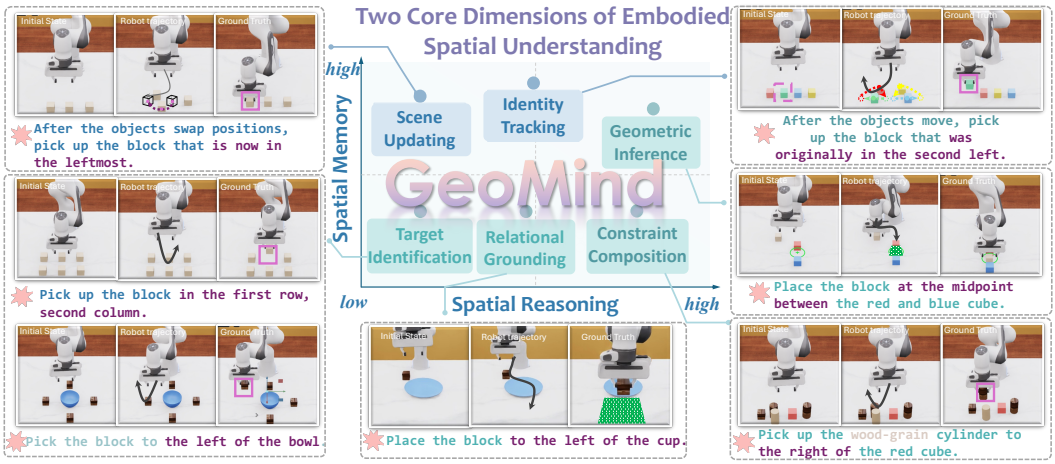


Figure 1: Overview of **GeoMind**. GeoMind organizes structured spatial understanding along two diagnostic dimensions: *spatial memory* and *spatial reasoning*. The six task families cover different combinations of these dimensions. Surrounding examples show representative tasks illustrating how GeoMind couples spatial instructions with structured object layouts and scene changes.

target has been correctly identified. These limitations motivate a diagnostic evaluation that makes spatial requirements explicit and examines agents' spatial capabilities beyond overall task success.

To enable a more targeted assessment of spatial understanding, we focus on two key dimensions, ***spatial reasoning*** and ***spatial memory***, drawing on the cognitive distinction between reasoning about space and maintaining spatial information over time [25]. Spatial reasoning concerns identifying target objects and inferring goal locations or regions from spatial relationships and structured layouts, while spatial memory concerns tracking object identities and updating spatial states as scenes change. These dimensions specify the spatial capabilities that each task is designed to assess.

We introduce **GeoMind**, a benchmark for systematically evaluating spatial understanding through tabletop robotic manipulation. It comprises six task families: (1) *target identification*, (2) *relational grounding*, (3) *constraint composition*, and (4) *geometric inference* for spatial reasoning, and (5) *scene updating* and (6) *identity tracking* for spatial memory. Together, these families span 27 canonical task designs, instantiated as 54 evaluated variants. Tasks are designed around structured object layouts and controlled scene changes, with explicit spatial rules for determining target objects or goal locations. Many base tasks feature visually identical or near-identical objects to reduce reliance on appearance cues. To enable reliable evaluation of these diverse spatial objectives, we develop an agent-assisted pipeline that translates spatial constraints into executable grounding code, computes ground-truth spatial targets, and verifies the resulting task instances through simulation.

We evaluate generalist and spatially enhanced visuomotor policies alongside modular planning methods on GeoMind under a unified evaluation protocol. GeoMind combines a base task suite (GeoMind-Base) with controlled semantic, visual, and execution augmentations (GeoMind-Aug) to probe robustness. Beyond task success rates, we use diagnostic spatial metrics and comparisons with an oracle policy to examine spatial grounding and execution behavior. Our results show that current methods can solve some basic grounding tasks but struggle to reason about structured spatial relationships and track spatial states across scene changes. Performance is sensitive to semantic augmentations, while visual and execution perturbations have more model-dependent effects.

The main contributions of this work are threefold: **(1) A systematic benchmark for structured spatial understanding.** We introduce GeoMind, which organizes spatial understanding into six task families and provides controlled base and augmented evaluations. **(2) An agent-assisted spatial grounding and verification pipeline.** We develop a pipeline that translates spatial constraints into executable grounding programs, derives ground-truth targets, and verifies task instances through simulation. **(3) A comprehensive diagnostic evaluation.** We evaluate generalist and spatially enhanced policies alongside modular planning methods using diagnostic spatial metrics and oracle comparisons, revealing limitations in spatial reasoning, memory, and robustness.

Table 1: Comparison of spatial capabilities and benchmark design. ✓: explicitly evaluated; ✗: no explicit evaluation identified in the reviewed protocol.

Benchmark	Design	Spatial Memory		Spatial Reasoning				Evaluation
	Structure Generation	*Scene Updating*	*Identity Tracking*	*Target Identification*	*Relational Grounding*	*Constraint Composition*	*Geometric Inference*	*Diagnostic Metrics*
VIMA [23]	✓	✓	✓	✓	✓	✓	✗	✗
GENMANIP [31]	✓	✗	✗	✗	✓	✗	✗	✗
RoboMME [33]	✓	✓	✓	✗	✗	✓	✗	✗
MEMOBench [34]	✓	✓	✗	✗	✓	✗	✗	✓
RoboSPA [35]	✓	✗	✓	✓	✓	✓	✗	✗
BEHAVIOR-Spatial [36]	✗	✗	✗	✓	✓	✓	✗	✓
VABench [37]	✓	✓	✗	✗	✓	✗	✗	✓
GeoMind	✓	✓	✓	✓	✓	✓	✓	✓

2 Related Work

General Embodied Manipulation Benchmarks. Prior embodied manipulation benchmarks have progressively advanced standardized evaluation of language-conditioned manipulation, long-horizon control, and generalization across diverse tasks and environments [18, 20, 22, 24, 26–30]. VIMA includes history-dependent rearrangement and visual constraint satisfaction [23], while GENMANIP uses task-oriented scene graphs for constrained scene construction [31]. These benchmarks already cover several spatial requirements considered here; aggregate task success alone, however, does not distinguish spatial goal grounding errors from execution failures. More specialized benchmarks investigate memory-dependent manipulation. RMBench [32] evaluates tasks with varying memory complexity, while RoboMME [33] examines temporal, spatial, object, and procedural memory. MEMOBench [34] further introduces process-level evaluations of memory storage, updating, and compression. Together, these benchmarks examine both memory demands and the processes underlying memory-dependent manipulation. GeoMind complements them by systematically assessing spatial reasoning and spatial memory through capability-specific manipulation tasks defined by explicit spatial constraints and controlled scene changes. Table 1 compares selected task requirements and evaluation designs. A checkmark denotes an explicitly targeted task or evaluation subset with reported results; a cross denotes that no explicit evaluation was identified in the reviewed protocol, not that the capability is absent.

Benchmarks for Spatial Understanding. Spatial understanding benchmarks span static image-based evaluations and interactive embodied settings [25, 38–42]. VSR [43] evaluates spatial relations in image–text pairs, while other benchmarks examine orientation, navigation, mental rotation, localization, and video recall [44–49]. ESI-Bench [50] emphasizes active information acquisition for spatial reasoning, while SpaMEM [51] probes spatial state updating and long-horizon belief maintenance through action-conditioned observation sequences. RoboSPA [35] evaluates spatial reference resolution, compositional spatial selection, and procedural-memory tasks, and reports object-normalized success and progress measures. BEHAVIOR-Spatial [36] evaluates spatial instruction following through question answering, target grounding, and robot execution, while VABench [37] combines visual demonstrations, active perception, metric control, and trajectory-level diagnostics. GeoMind complements these benchmarks by organizing explicit geometric goal construction, current-state re-grounding, and historical identity binding within a shared, controllable tabletop protocol. Structure-preserving sampling, deterministic spatial targets, controlled scene changes, and matched task variants support targeted comparisons across these requirements; scene updating need not imply dependence on unobserved history.

Spatial Capabilities of Embodied Policies. Current embodied agents increasingly rely on vision-language models for language-guided manipulation [3, 4, 52–54], yet complex spatial instructions remain challenging. Recent approaches therefore incorporate more explicit spatial representations and grounding mechanisms. RoboSpatial [55] provides large-scale spatial annotations for training vision-language models, while InternVLA-M1 [16] combines spatial grounding pretraining with spatially guided action learning to connect *where to act* with *how to act*. Other approaches introduce 3D spatial representations to improve the alignment of visual, linguistic, and geometric information for action generation [15, 38, 56, 57]. While these methods aim to improve spatial capabilities, GeoMind provides a complementary benchmark for systematically examining their strengths and limitations.

3 GeoMind

We present the overall design of GeoMind by turning structured spatial understanding into a controllable benchmark construction problem. Sec. 3.1 introduces the GeoMind taxonomy, and Sec. 3.2 describes spatial structure modeling and the benchmark construction pipeline. Sec. 3.3 presents the task suites, showing how executable benchmark episodes are organized into GeoMind-Base and GeoMind-Aug.

3.1 Taxonomy of Embodied Spatial Understanding Tasks

Rather than categorizing tasks solely by manipulation actions, we organize them according to the spatial capabilities required to determine task-relevant targets and goals. Following the dimensions introduced above, the taxonomy organizes structured spatial understanding around *spatial reasoning* and *spatial memory*. The former concerns how agents infer targets or goals from spatial relations and scene structure, while the latter concerns how agents maintain and update task-relevant spatial information as scenes change.

- **Target Identification** focuses on static object selection from structured layouts. The agent resolves a spatial description to the correct target object in a fixed scene.
- **Relational Grounding** requires the agent to realize an explicit spatial relation through action, such as placing one object beside, in front of, or inside another reference object.
- **Constraint Composition** tests whether the agent can jointly satisfy multiple spatial constraints that define a target object or a valid goal region.
- **Geometric Inference** requires the agent to infer implicit target locations or regions from scene geometry, such as alignment, symmetry, or pattern continuation across objects.
- **Scene Updating** evaluates whether the agent can update task-relevant spatial relations and re-ground its target after objects have moved or been manipulated.
- **Identity Tracking** requires the agent to maintain object identity over time by linking a previously identified object to its current state across scene changes, such as object rearrangement or position exchanges.

Fig. 1 qualitatively illustrates the spatial reasoning and memory demands of the six task families and presents representative examples with initial scenes, robot trajectories, and ground-truth spatial targets. The high/low positions in Fig. 1 indicate intended capability demands, rather than an empirically established difficulty ranking. These tasks remain close to common robotic behaviors, such as selecting, placing, and retrieving objects, while exposing the spatial reasoning and memory demands behind seemingly simple actions.

3.2 Spatial Structure Modeling and Benchmark Construction

GeoMind constructs benchmark episodes through spatial structure modeling and an agent-assisted, rule-verified pipeline. Spatial structures allow object configurations to vary while preserving task-relevant constraints. The pipeline combines task template generation, executable spatial grounding, and simulation-based verification to produce structured manipulation tasks with deterministic ground-truth targets (Fig. 3).

Spatial Structure Modeling. To evaluate spatial understanding across different scene configurations, GeoMind represents object layouts using spatial structures that are not tied to a fixed set of coordinates. For example, objects arranged in a line can preserve their left-to-right order while varying in position and spacing.

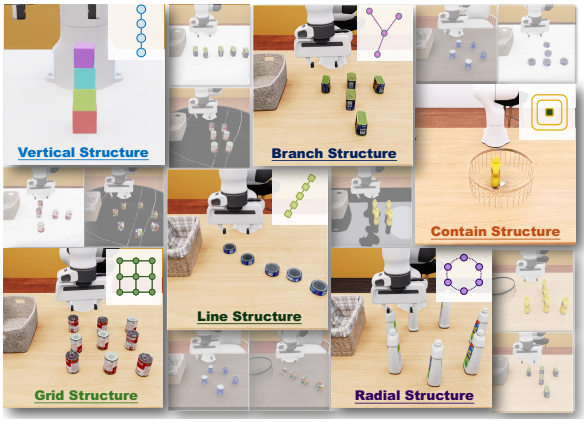


Figure 2: Visualization of the six fundamental structural layouts and their derived variants.

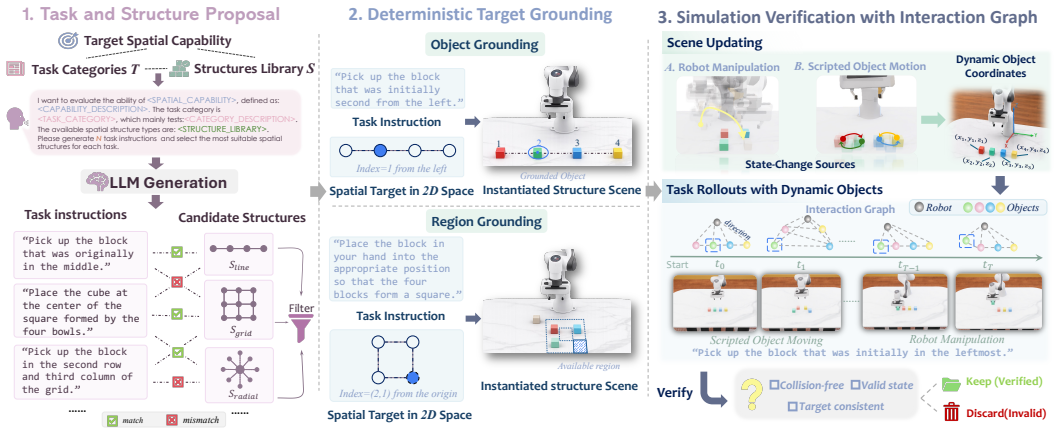


Figure 3: **Overview of the benchmark construction pipeline.** The pipeline first proposes task descriptions and structure templates, then resolves spatial queries over instantiated scenes to derive ground-truth annotations, and finally validates candidate episodes through interaction-graph-based simulation verification.

A task such as selecting the second object from the left therefore follows the same spatial rule across different layouts. We formalize a spatial structure S as a set of relational, topological, and geometric constraints over objects. Given S, GEOMIND generates diverse scene configurations by sampling object poses $\mathbf{X}$ from a structure-preserving distribution:

$$\mathbf{X} \sim \mathcal{P}(\mathbf{X} \mid S, \epsilon),$$

where ϵ controls allowable geometric variations, such as positional jitter and variable spacing, without violating the constraints encoded in S. The structure library $\mathcal{S} = \{S_{\text{line}}, S_{\text{grid}}, \ldots\}$ includes linear, grid, and other canonical layouts (Fig. 2). Sampled configurations must satisfy structural validity and collision-free workspace constraints. This representation supports controlled evaluation across variations in object positions and spacing. Formal definitions are provided in Appendix Appendix C.

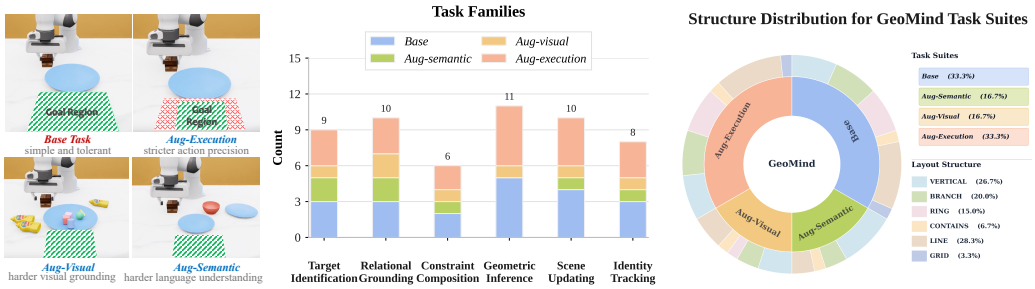
LLM-Assisted Task and Structure Proposal. Given a task category and a candidate set of spatial structure types, the LLM proposes matching instructions and structure templates within a predefined schema. Each template specifies the structure type, object roles, layout constraints, sampling ranges, and optional scene-update rules. Candidate templates are screened to remove ambiguous, logically conflicting, or task-incompatible specifications, with human review during template selection. The retained templates form a fixed library for subsequent programmatic scene instantiation. The benchmark uses a fixed set of tasks selected through human review, with task instances generated programmatically from the corresponding templates.

Spatial Target Grounding. An LLM-based coding agent translates spatial rules into programs that deterministically resolve target objects or goal regions from simulator states. Directional predicates use the canonical front-camera frame, with object coordinates transformed from the robot base frame. Reasoning tasks ground targets in the current configuration. In memory tasks, instructions precede scripted object motion, and evaluated actions begin after it ends: Scene Updating re-grounds targets in the updated scene, while Identity Tracking preserves their initial identities. Each agent can store earlier observations or maintain an internal memory to identify the target after the objects move.

Closed-Loop Verification. Each candidate episode is executed in a physics simulator to check physical feasibility and target consistency. For tasks involving scene changes, objects move through scripted motion or simulated robot manipulation. Throughout execution, we record simulator object states and construct a dynamic interaction graph $G_t = (V, E_t)$, where nodes represent objects and edges encode discretized pairwise directions and distances. The graph sequence is used to check whether scene transitions and resulting spatial states are consistent with the task specifications and ground-truth targets. Episodes failing state-validity or target-consistency checks are discarded.

3.3 Task Suites and Dataset

GEOMIND is organized into two complementary evaluation suites: GEOMIND-BASE and GEOMIND-AUG. Together, they provide **54 evaluated task variants** for structured spatial understanding. A task variant is a specific test setting that pairs a spatial objective with a scene structure, governed by



(a) Task display. (b) Overview of GEOMIND task suites.

Figure 4: Detailed information about GEOMIND task suites. (a) shows how GEOMIND-AUG is constructed from GEOMIND-BASE. (b) shows the distribution of evaluated variants across task families and spatial structures.

specific grounding rules and evaluation conditions. GEOMIND-BASE evaluates the six spatial task families under canonical tabletop settings, while GEOMIND-AUG introduces controlled perturbations to assess robustness under additional semantic, visual, and execution challenges.

GEOMIND-BASE consists of **20 evaluated base variants** with diverse language instructions and multiple spatial structure types. To reduce visual shortcuts, many GEOMIND-BASE variants contain multiple visually identical or near-identical objects. In these settings, target selection or placement depends on spatial cues such as relative position, order, geometric relations, or scene updates, rather than object appearance alone. GEOMIND-BASE serves as the core benchmark suite and evaluates the spatial reasoning and memory capabilities defined in our taxonomy. Fig. 4b presents the distribution of evaluated task variants and their associated spatial structures.

GEOMIND-AUG contains **34 augmented variants**: four semantic variants add grounding anchors while preserving the scene and goal; three visual variants add distractors while preserving the target and success criterion; and 27 execution variants tighten only the success tolerance of the 27 non-execution parents (20 Base, four semantic, and three visual). Each comparison uses the immediate parent as its reference and isolates the designated augmentation factor. Examples and task details appear in Fig. 4a and Appendix D.

4 Experiments

In this section, we systematically evaluate representative embodied agents on the GEOMIND benchmark under a controlled setting. We first describe the experimental setup and the selected embodied agents. Then, we present the evaluation results, analyzing their implications for spatial understanding.

4.1 Experimental Setup

Benchmarked Embodied Agents. We evaluate representative embodied agents across GEOMIND-BASE and GEOMIND-AUG, covering three paradigms: general policies, spatially enhanced policies, and modular agents. These methods differ in whether spatial grounding, planning, and execution are handled implicitly by an end-to-end policy or explicitly by separate modules. Specifically, we evaluate OpenVLA [3], π_0 [4], and $\pi_{0.5}$ [5] as general policies; SpatialVLA [15], InternVLA-M1 [16], and Spatial Forcing [57] as spatially enhanced policies; and COPA [58] and MOKA [59] as modular agents. We additionally include an oracle baseline that uses simulator states and deterministic spatial grounding to compute ground-truth targets, which guide a scripted controller. The oracle provides a reference for assessing task feasibility under correct target information.

Evaluation Protocol. All experiments are conducted in the IsaacLab [60] simulator using a Franka Research 3 (FR3) robotic arm. All methods are evaluated on the same task instances, random seeds, episode horizons, and success criteria. The evaluated VLA policies use fixed checkpoints without GeoMind-specific finetuning, and their outputs are adapted to a common action interface. Privileged scene information, including simulator-resolved object identities and ground-truth target coordinates, is used for evaluation and the oracle baseline but is not provided to the evaluated VLA policies or modular agents during inference. Each episode is limited to 1000 control steps and terminates early upon task success or workspace violation. We evaluate $N = 48$ episodes per task variant, with evaluation seeds saved for reproducibility. The physics simulation runs at 480 Hz, and the policy

control frequency is 24 Hz. Detailed observation and action interfaces, deployment configurations, and evaluation protocols are provided in Appendix B.

Primary Evaluation Metrics. We evaluate task completion and target selection using Task Success Rate (SR) and Target Selection Accuracy (TSA). SR is the proportion of episodes satisfying task-specific success conditions, reported overall, by task family, and by augmentation type. Robustness is measured by $\Delta_{\mathrm{Aug}} = \mathrm{SR}_{\mathrm{Aug}} - \mathrm{SR}_{\mathrm{ref}}$ in percentage points; negative values indicate degradation. The reference is the matched base task for semantic and visual variants, and the corresponding non-execution variant for execution variants. For selected object-grounding tasks, TSA is the fraction of episodes in which the object nearest the end-effector at its closest approach to any manipulable object is the ground-truth target. It serves as a behavioral proxy for target selection. Controlled comparisons with ground-truth targets and scripted controllers further examine grounding and execution errors.

Auxiliary Spatial Metrics. Spatial Goal Error (SGE) measures final spatial deviation, while Target Tracking Error (TTE) measures average end-effector distance to the target:

$$\mathrm{SGE}_i = d_i\left(\mathbf{s}_i^{\mathrm{final}}, \mathbf{s}_i^{\mathrm{goal}}\right), \qquad \mathrm{TTE}_i = \frac{1}{T_i}\sum_{t=1}^{T_i}\left\|\mathbf{p}_{i,t}^{\mathrm{eef}} - \mathbf{p}_{i,t}^{\mathrm{tar}}\right\|_2 .$$

Here, i indexes episodes, d_i is the evaluator's goal-distance function, T_i is the episode length, and $\mathbf{p}^{\mathrm{eef}}$ and $\mathbf{p}^{\mathrm{tar}}$ denote end-effector and target positions. SGE is aggregated only over valid goal-distance values. TTE uses recorded target trajectories when available, otherwise the initial target position as a marked static-layout proxy. Lower values indicate smaller errors. Applicability and aggregation details are provided in Appendix A.

4.2 Main Results

In this section, we conduct a systematic evaluation of representative embodied agents around the two dimensions of spatial understanding and our task taxonomy. We present five core questions (Q1 – Q5) focused on structured spatial understanding, and through empirical analysis, we address the questions that probe the fundamental capabilities of structured spatial understanding.

Q1. How well do embodied agents perform in structured spatial understanding?

We begin with GEOMIND-BASE, which evaluates six spatial capabilities under standard task conditions without the additional perturbations introduced in GEOMIND-AUG. Table 2 reports the success rates of different agents across the six task families. The best-performing learned policy, InternVLA-M1, achieves 26.4% success, followed by Spatial-Forcing at 24.0%, substantially below the oracle's 99.3%. Spatially enhanced policies achieve the highest overall scores, although their improvements over general policies are uneven across models and task families. Modular agents achieve 4.0–16.9% success, indicating that explicit VLM-based grounding and scripted execution also struggle with structured spatial tasks.

To better understand these performance gaps, we examine the contributions of target grounding and execution. The oracle achieves near-perfect success using ground-truth targets and scripted control, demonstrating the feasibility of the benchmark tasks under correct spatial grounding. More directly, in a paired diagnostic evaluation of $\pi_{0.5}$, explicitly specifying the target through its name and a visual cue improves success from 11.3% to 39.2%, while keeping the policy, task instances, action interface, and success criterion fixed. This result provides evidence that target identification contributes substantially to the observed failures, although execution and deployment limitations may also play a role.

***Finding**: Current embodied agents remain far from reliable on structured spatial manipulation tasks. The controlled target-cue intervention suggests that inaccurate spatial target grounding is an important contributor to these failures, beyond execution limitations alone.*

Q2. How robust is structured spatial understanding under semantic, visual, and execution perturbations?

Table 3 shows that success in GEOMIND-BASE is fragile under harder variants of the same spatial goals. The clearest drop comes from semantic augmentation: averaged over the 6 embodied policies, success decreases from about 22.6% on the corresponding Base tasks to about 14.2%, and every policy degrades. This indicates that many Base successes still rely on simple instruction grounding rather

Table 2: Success rates of different paradigms of embodied agents on GEOMIND-BASE. Best and second-best overall results are highlighted with **green** shades, while best and second-best family-level results are highlighted with **purple** shades.

Embodied Agents			*Spatial Reasoning*				*Spatial Memory*		AVG
			Target Iden.	Relational Grounding	Constraint Comp.	Geometric Inference	Scene Updating	Identity Tracking	
Oracle			100.0	98.0	100.0	100.0	98.0	100.0	99.3
General Policies		OpenVLA	37.5	0.0	6.2	5.0	41.7	**37.5**	20.7
		π_0	4.2	0.0	0.0	25.0	6.3	20.9	11.3
		$\pi_{0.5}$	4.2	0.0	0.0	**27.5**	6.3	25.0	12.5
Spatial-Enhanced Policies		SpatialVLA	12.5	0.0	0.0	17.5	**45.8**	8.3	18.3
		InternVLA-M1	**42.5**	**50.0**	6.2	12.0	16.7	20.8	**26.4**
		Spatial-Forcing	12.5	0.0	**50.0**	5.0	33.3	**37.5**	24.0
Modular Methods	COPA	GPT-4o	22.9	11.1	16.7	4.2	9.7	13.9	11.1
		Gemini-2.5-Flash	10.4	2.8	6.3	1.7	6.9	0.8	4.6
		Claude-Sonnet-4.6	20.8	13.9	16.7	13.3	16.0	19.4	15.9
	MOKA	GPT-4o	18.8	12.5	18.8	1.7	8.3	13.9	10.1
		Gemini-2.5-Flash	8.3	1.4	4.2	0.8	7.6	1.4	4.0
		Claude-Sonnet-4.6	18.8	13.9	16.7	13.3	18.6	20.8	16.9

than robust spatial-language understanding. The effects of visual and execution augmentation are more uneven. OpenVLA falls to 12.5% under both settings, whereas π_0 remains nearly stable under visual perturbation and only slightly drops under execution. These results suggest that structured spatial understanding degrades under complex instruction semantics and increased visual noise.

***Finding**: Spatial success in embodied agents is condition-dependent rather than goal-invariant: the same spatial objective often fails once its linguistic, visual, or execution support is perturbed.*

Table 3: Performance on GEOMIND-AUG and corresponding GEOMIND-BASE tasks

Policies	Corresponding Base	Semantic	Visual	Execution
OpenVLA	25.0	16.6 (−8.4)	12.5 (−12.5)	12.5 (−12.5)
π_0	16.7	12.5 (−4.2)	16.7 (+0.0)	10.2 (−6.5)
$\pi_{0.5}$	18.5	12.5 (−6.0)	16.7 (−1.8)	10.2 (−6.2)
SpatialVLA	18.8	8.33 (−10.5)	12.5 (−6.3)	12.5 (−6.3)
InternVLA-M1	37.5	27.1 (−10.4)	37.5 (+0.0)	35.4 (−2.1)
Spatial-Forcing	18.8	8.33 (−10.5)	12.5 (−6.3)	16.7 (−2.1)

Q3. Why do embodied agents fail on GEOMIND?

Finally, we summarize representative failure cases in Fig. 5 to connect the quantitative trends with concrete trajectories. For embodied policies in Fig. 5a, the failures mainly fall into three categories: *spatial relation grounding errors*, where policies bind the correct anchor but execute the wrong relation; *spatial relation reasoning errors*, where policies fail to infer structure-defined targets; and *historical state memory errors*, where policies lose the target identity after scene changes. These cases indicate that policy failures are not merely caused by low-level control noise, but often reflect a failure to convert spatial observations into stable action targets. For modular embodied agents in Fig. 5b, the dominant failure comes from upstream object localization errors, which propagate to planning and control. This suggests a different bottleneck: modular agents may expose explicit reasoning steps, but remain vulnerable when intermediate spatial grounding is inaccurate.

***Finding**: The core failure is not the absence of spatial cues, but the absence of an executable spatial state: current agents can often perceive or describe parts of the scene, yet fail to maintain a task-relevant spatial commitment that continuously constrains action throughout the trajectory.*

Q4. What makes spatial memory difficult for embodied agents?

Tables 2 and 4 show that agents struggle with tasks involving scene changes. Scene Updating requires selecting a target from the updated configuration, whereas Identity Tracking requires linking an object

Table 4: **Family-level failure rate and auxiliary spatial error metrics.**

Capability Dimensions	Ability Family	Failure Rate ↑	SGE ↑	TTE ↑
Spatial Memory	Scene Updating	68.89%	0.189	0.330
	Identity Tracking	**85.00%**	**0.192**	**0.486**
Spatial Reasoning	Target Identification	72.08%	0.116	**0.811**
	Relational Grounding	**98.33%**	**0.180**	0.672
	Constraint Composition	87.50%	0.142	0.627
	Geometric Inference	86.67%	0.158	0.727

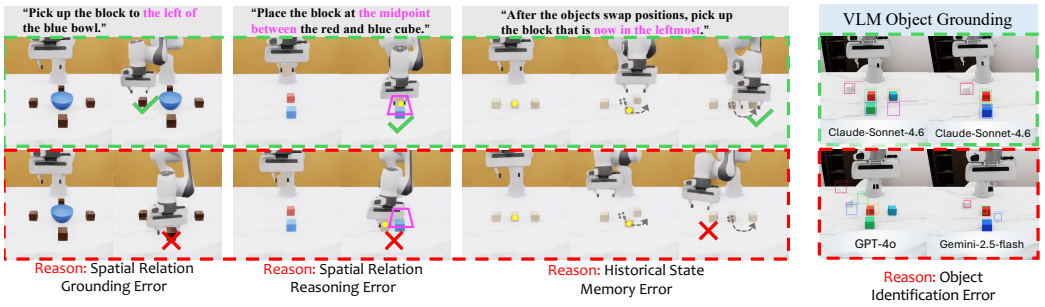
identified before the change to its new position. An agent may therefore fail an identity-tracking task by selecting an object that matches the current spatial relation but is not the original target. This provides a possible explanation for failures beyond imprecise execution. However, aggregate SGE and TTE cannot distinguish these causes: both are affected by scene geometry, initial target distance, and rollout duration. We use them as auxiliary diagnostics rather than direct measures of identity preservation or evidence that one task family is intrinsically harder.

***Finding**: Identity Tracking requires preserving object correspondence across scene changes; the current aggregate results do not establish how much failure is attributable to incorrect target selection versus execution.*

Q5. What makes spatial reasoning difficult for embodied agents?

Table 4 shows that relational grounding is the strongest bottleneck among the reasoning families, with an average failure rate of 98.33%, compared with 72.08% for target identification. Constraint composition and geometric inference are also difficult, with failure rates of 87.50% and 86.67%, respectively. This indicates that current agents do not only struggle with recognizing spatial targets; they also fail to convert explicit spatial relations into correct embodied actions. Moreover, Table 2 shows that geometric inference has the lowest best-agent success rate, with the strongest method reaching only 27.5%, suggesting that implicit structure-based reasoning remains hard even for the best agents. Together, these results suggest that agents remain weak when they must jointly satisfy multiple spatial constraints or infer targets from implicit geometric structure.

***Finding**: Spatial reasoning is difficult not only because agents struggle to identify the correct spatial target, but because they fail to transform spatial constraints into executable manipulation behavior.*



(a) Embodied policies. (b) Modular Agents

Figure 5: Visualization of typical failure cases in GeoMind.

5 Conclusion

We present GeoMind, a systematic benchmark for evaluating spatial understanding in embodied agents. By organizing tasks around spatial memory and spatial reasoning, GeoMind enables a more fine-grained analysis than overall task success alone. We further emphasize how spatial structure underpins spatial understanding, and present an automated generation pipeline that allows scalable benchmark construction. Our experiments show that current embodied agents remain far from

saturated on GEOMIND, with large performance gaps and highly uneven abilities across task families. These findings suggest that spatial understanding remains a core limitation of embodied agents.

Limitations. GEOMIND is limited to simulated, single-arm tabletop manipulation, without real-robot validation. Its controlled layouts do not capture the full diversity of real-world environments. Heavy occlusion, viewpoint changes, sensor noise, and complex contact dynamics are not systematically evaluated, although incidental occlusion may occur during execution. Semantic and visual augmentations cover selected transformations rather than exhaustive robustness conditions. Our experiments evaluate frozen pretrained policies, and their performance may also depend on observation and action-interface compatibility. We do not report benchmark-specific finetuning results; learning spatial skills and generalizing to held-out structures or variants remain untested. The results therefore characterize frozen-policy performance in the evaluated setting and do not establish transfer to real robots.

References

[1] Agrim Gupta, Silvio Savarese, Surya Ganguli, and Li Fei-Fei. Embodied intelligence via learning and evolution. *Nature communications*, 12(1):5721, 2021.

[2] Generalist AI Team. Gen-0: Embodied foundation models that scale with physical interaction. *Generalist AI Blog*, 2025. https://generalistai.com/blog/preview-uqlxvb-bb.html.

[3] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model, 2024.

[4] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. Pi-0: A vision-language-action flow model for general robot control, 2024.

[5] Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Manuel Y. Galliker, Dibya Ghosh, Lachy Groom, Karol Hausman, brian ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Devin LeBlanc, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Allen Z. Ren, Lucy Xiaoyang Shi, Laura Smith, Jost Tobias Springenberg, Kyle Stachowicz, James Tanner, Quan Vuong, Homer Walke, Anna Walling, Haohuan Wang, Lili Yu, and Ury Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization. In *9th Annual Conference on Robot Learning*, 2025.

[6] Qingwen Bu, Yanting Yang, Jisong Cai, Shenyuan Gao, Guanghui Ren, Maoqing Yao, Ping Luo, and Hongyang Li. UniVLA: Learning to act anywhere with task-centric latent actions, 2025.

[7] Songming Liu, Lingxuan Wu, Bangguo Li, Hengkai Tan, Huayu Chen, Zhengyi Wang, Ke Xu, Hang Su, and Jun Zhu. RDT-1b: a diffusion foundation model for bimanual manipulation, 2024.

[8] Brianna Zitkovich, Tianhe Yu, Sichun Xu, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pages 2165–2183. PMLR, 2023.

[9] Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.

[10] Jiaming Zhou, Qihang Zhang, Gangwei Xu, et al. Zero-WAM: In-context world-action modeling from human videos for open-ended task generalization. *arXiv preprint arXiv:2608.26103*, 2026.

[11] Yuran Wang, Siqiao Huang, Mingleyang Li, et al. OpenWAM: An open, modular exploration towards systematic world-action model pretraining. *arXiv preprint arXiv:2609.07398*, 2026.

[12] Heng Yu, David D. Yuan, Juze Zhang, et al. OpenWAM: An open framework for composable world-action models. *arXiv preprint arXiv:2610.07922*, 2026.

[13] Runyu Lu, Yubo Wu, Ethan Kou, et al. ASPIRE: Agentic /skills discovery for robotics. *arXiv preprint arXiv:2607.00272*, 2026.

[14] Wenli Xiao, Jia Xie, Tonghe Zhang, et al. ENPIRE: Agentic robot policy self-improvement in the real world. *arXiv preprint arXiv:2606.19980*, 2026.

[15] Delin Qu, Haoming Song, Qizhi Chen, Yuanqi Yao, Xinyi Ye, Yan Ding, Zhigang Wang, JiaYuan Gu, Bin Zhao, Dong Wang, and Xuelong Li. SpatialVLA: Exploring spatial representations for visual-language-action model, 2025.

[16] Xinyi Chen, Yilun Chen, Yanwei Fu, Ning Gao, Jiaya Jia, Weiyang Jin, Hao Li, Yao Mu, Jiangmiao Pang, Yu Qiao, et al. Internvla-m1: A spatially guided vision-language-action framework for generalist robot policy. *arXiv preprint arXiv:2510.13778*, 2025.

[17] Zhengshen Zhang, Hao Li, Yalun Dai, Zhengbang Zhu, Lei Zhou, Chenchen Liu, Dong Wang, Francis E. H. Tay, Sijin Chen, Ziwei Liu, Yuxiao Liu, Xinghang Li, and Pan Zhou. From spatial to actions: Grounding vision-language-action model in spatial foundation priors, 2025.

[18] Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning, 2023.

[19] Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, Xiaodi Yuan, Pengwei Xie, Zhiao Huang, Rui Chen, and Hao Su. ManiSkill2: A unified benchmark for generalizable manipulation skills, 2023.

[20] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020.

[21] Shiduo Zhang, Zhe Xu, Peiju Liu, Xiaopeng Yu, Yuan Li, Qinghui Gao, Zhaoye Fei, Zhangyue Yin, Zuxuan Wu, Yu-Gang Jiang, and Xipeng Qiu. Vlabench: A large-scale benchmark for language-conditioned robotics manipulation with long-horizon reasoning tasks, 2024.

[22] Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on robot learning*, pages 1094–1100. PMLR, 2020.

[23] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with multimodal prompts. In *Fortieth International Conference on Machine Learning*, 2023.

[24] Tianxing Chen, Zanxin Chen, Baijun Chen, Zijian Cai, Yibin Liu, Zixuan Li, Qiwei Liang, Xianliang Lin, Yiheng Ge, Zhenyu Gu, et al. Robotwin 2.0: A scalable data generator and benchmark with strong domain randomization for robust bimanual robotic manipulation. *arXiv preprint arXiv:2506.18088*, 2025.

[25] Jihan Yang, Shusheng Yang, Anjali Gupta, Rilyn Han, Li Fei-Fei, and Saining Xie. Thinking in Space: How Multimodal Large Language Models See, Remember and Recall Spaces. *arXiv preprint arXiv:2412.14171*, 2024.

[26] Oier Mees, Lukas Hermann, Erick Rosete-Beas, and Wolfram Burgard. CALVIN: A Benchmark for Language-Conditioned Policy Learning for Long-Horizon Robot Manipulation Tasks, July 2022. arXiv:2112.03227 [cs].

[27] Xuanlin Li, Kyle Hsu, Jiayuan Gu, Karl Pertsch, Oier Mees, Homer Rich Walke, Chuyuan Fu, Ishikaa Lunawat, Isabel Sieh, Sean Kirmani, et al. Evaluating real-world robot manipulation policies in simulation. *arXiv preprint arXiv:2405.05941*, 2024.

[28] Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, Xiaodi Yuan, Pengwei Xie, Zhiao Huang, Rui Chen, and Hao Su. Maniskill2: A unified benchmark for generalizable manipulation skills. In *International Conference on Learning Representations*, 2023.

[29] Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. Robocasa: Large-scale simulation of everyday tasks for generalist robots. In *Robotics: Science and Systems*, 2024.

[30] Stone Tao, Fanbo Xiang, Arth Shukla, Yuzhe Qin, Xander Hinrichsen, Xiaodi Yuan, Chen Bao, Xinsong Lin, Yulin Liu, Tse-kai Chan, et al. Maniskill3: Gpu parallelized robotics simulation and rendering for generalizable embodied ai. *arXiv preprint arXiv:2410.00425*, 2024.

[31] Ning Gao, Yilun Chen, Shuai Yang, Xinyi Chen, Yang Tian, Hao Li, Haifeng Huang, Hanqing Wang, Tai Wang, and Jiangmiao Pang. Genmanip: Llm-driven simulation for generalizable instruction-following manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12187–12198, June 2025.

[32] Tianxing Chen, Yuran Wang, Mingleyang Li, Yan Qin, Hao Shi, Zixuan Li, Yifan Hu, Yingsheng Zhang, Kaixuan Wang, Yue Chen, et al. Rmbench: Memory-dependent robotic manipulation benchmark with insights into policy design. *arXiv preprint arXiv:2603.01229*, 2026.

[33] Yinpei Dai, Hongze Fu, Jayjun Lee, Yuejiang Liu, Haoran Zhang, Jianing Yang, Chelsea Finn, Nima Fazeli, and Joyce Chai. Robomme: Benchmarking and understanding memory for robotic generalist policies. *arXiv preprint arXiv:2603.04639*, 2026.

[34] Haiyang Sun, Haoxiao Wang, Junming Chen, Weicheng Fang, Zihao Su, Jingkun Yi, Wenyou Yi, Hao Chen, and Zhou Zhao. MEMOBench: A process level memory benchmark for robotic manipulation. *arXiv preprint arXiv:2609.07047*, 2026.

[35] Zhenxuan Fan, Bo Zhang, Yutong Lin, Yuqian Yuan, Juekai Lin, Liang Liang, Zhuoyi Huang, Wenqiao Zhang, Juncheng Li, Siliang Tang, Jun Xiao, and Yueting Zhuang. RoboSPA: Can VLA models go beyond simple scenes and short-horizon tasks? In *Proceedings of the 2026 Conference on Empirical Methods in Natural Language Processing*, pages 23457–23492. Association for Computational Linguistics, 2026.

[36] Sirui Chen, Huang Huang, Qineng Wang, Wenlong Huang, Manling Li, Ruohan Zhang, Li Fei-Fei, and Jiajun Wu. BEHAVIOR-Spatial benchmark: Evaluating spatial understanding for robotics. arXiv preprint `https://arxiv.org/abs/2610.34051`, 2026.

[37] Zhongbo Zhang, Jiayi Jin, Yifan Wang, Zaibin Zhang, Haiwen Diao, Lijun Wang, and Huchuan Lu. VABench: Measuring embodied spatial intelligence through visual demonstrations, active perception, and metric control. arXiv preprint `https://arxiv.org/abs/2609.19554`, 2026.

[38] Enshen Zhou, Jingkun An, Cheng Chi, Yi Han, Shanyu Rong, Chi Zhang, Pengwei Wang, Zhongyuan Wang, Tiejun Huang, Lu Sheng, et al. Roborefer: Towards spatial referring with reasoning in vision-language models for robotics. *arXiv preprint arXiv:2506.04308*, 2025.

[39] Yanpeng Zhao, Wentao Ding, Hongtao Li, Baoxiong Jia, and Zilong Zheng. Espire: A diagnostic benchmark for embodied spatial reasoning of vision-language models. *arXiv preprint arXiv:2603.13033*, 2026.

[40] Julius Mayer, Mohamad Ballout, Serwan Jassim, Farbod Nosrat Nezami, and Elia Bruni. ivispar—an interactive visual-spatial reasoning benchmark for vlms. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 26745–26769, 2025.

[41] Zekun Qi, Wenyao Zhang, Yufei Ding, Runpei Dong, Xinqiang Yu, Jingwen Li, Lingyun Xu, Baoyu Li, Xialin He, Guofan Fan, Jiazhao Zhang, Jiawei He, Jiayuan Gu, Xin Jin, Kaisheng Ma, Zhizheng Zhang, He Wang, and Li Yi. SoFar: Language-grounded orientation bridges spatial reasoning and object manipulation, 2025.

[42] Kairun Wen, Yuzhi Huang, Runyu Chen, Hui Zheng, Yunlong Lin, Panwang Pan, Chenxin Li, Wenyan Cong, Jian Zhang, Junbin Lu, et al. Dynamicverse: A physically-aware multimodal framework for 4d world modeling. *arXiv preprint arXiv:2512.03000*, 2025.

[43] Fangyu Liu, Guy Emerson, and Nigel Collier. Visual spatial reasoning. *Transactions of the Association for Computational Linguistics*, 11:635–651, 06 2023.

[44] Yuzhi Huang, Kairun Wen, Rongxin Gao, Dongxuan Liu, Yibin Lou, Jie Wu, Jing Xu, Jian Zhang, Zheng Yang, Yunlong Lin, et al. Thinking in dynamics: How multimodal large language models perceive, track, and reason dynamics in physical 4d world. *arXiv preprint arXiv:2603.12746*, 2026.

[45] Ilias Stogiannidis, Steven McDonagh, and Sotirios A. Tsaftaris. Mind the gap: Benchmarking spatial reasoning in vision-language models, 2025.

[46] Dingming Li, Hongxing Li, Zixuan Wang, Yuchen Yan, Hang Zhang, Siqi Chen, Guiyang Hou, Shengpei Jiang, Wenqi Zhang, Yongliang Shen, Weiming Lu, and Yueting Zhuang. Viewspatial-bench: Evaluating multi-perspective spatial localization in vision-language models, 2025.

[47] Songsong Yu, Yuxin Chen, Hao Ju, Lianjie Jia, Fuxi Zhang, Shaofei Huang, Yuhan Wu, Rundi Cui, Binghao Ran, Zaibin Zhang, Zhedong Zheng, Zhipeng Zhang, Yifan Wang, Lin Song, Lijun Wang, Yanwei Li, Ying Shan, and Huchuan Lu. How far are vlms from visual spatial intelligence? a benchmark-driven perspective. *arXiv preprint arXiv:2509.18905*, 2025.

[48] Yun Li, Yiming Zhang, Tao Lin, XiangRui Liu, Wenxiao Cai, Zheng Liu, and Bo Zhao. Sti-bench: Are mllms ready for precise spatial-temporal world understanding? *arXiv preprint arXiv:2503.23765*, 2025.

[49] 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. *arXiv preprint arXiv:2506.03135*, 2025.

[50] Yining Hong, Jiageng Liu, Han Yin, Manling Li, Leonidas Guibas, Li Fei-Fei, Jiajun Wu, and Yejin Choi. ESI-Bench: Towards embodied spatial intelligence that closes the perception-action loop. arXiv preprint `https://arxiv.org/abs/2605.18746`, 2026.

[51] Chih-Ting Liao, Xi Xiao, Chunlei Meng, Zhangquan Chen, Yitong Qiao, Weilin Zhou, Tianyang Wang, Xu Zheng, and Xin Cao. SpaMEM: Benchmarking dynamic spatial reasoning via perception-memory integration in embodied environments. arXiv preprint `https://arxiv.org/abs/2604.22409`, 2026.

[52] Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, et al. Gr00t n1: An open foundation model for generalist humanoid robots. *arXiv preprint arXiv:2503.14734*, 2025.

[53] Gemini Robotics Team, Saminda Abeyruwan, Joshua Ainslie, Jean-Baptiste Alayrac, Montserrat Gonzalez Arenas, Travis Armstrong, Ashwin Balakrishna, Robert Baruch, Maria Bauza, Michiel Blokzijl, et al. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

[54] Chenghao Gu, Haolan Kang, Junchao Lin, Jinghe Wang, Duo Wu, Shuzhao Xie, Fanding Huang, Junchen Ge, Ziyang Gong, Letian Li, et al. Igen: Scalable data generation for robot learning from open-world images. *arXiv preprint arXiv:2512.01773*, 2025.

[55] Chan Hee Song, Valts Blukis, Jonathan Tremblay, Stephen Tyree, Yu Su, and Stan Birchfield. Robospatial: Teaching spatial understanding to 2d and 3d vision-language models for robotics. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 15768–15780, 2025.

[56] Ji Zhang, Shihan Wu, Xu Luo, Hao Wu, Lianli Gao, Heng Tao Shen, and Jingkuan Song. InSpire: Vision-language-action models with intrinsic spatial reasoning, 2025.

[57] Fuhao Li, Wenxuan Song, Han Zhao, Jingbo Wang, Pengxiang Ding, Donglin Wang, Long Zeng, and Haoang Li. Spatial forcing: Implicit spatial representation alignment for vision-language-action model. *arXiv preprint arXiv:2510.12276*, 2025.

[58] Haoxu Huang, Fanqi Lin, Yingdong Hu, Shengjie Wang, and Yang Gao. Copa: General robotic manipulation through spatial constraints of parts with foundation models. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9488–9495. IEEE, 2024.

[59] Fangchen Liu, Kuan Fang, Pieter Abbeel, and Sergey Levine. Moka: Open-world robotic manipulation through mark-based visual prompting. *arXiv preprint arXiv:2403.03174*, 2024.

[60] Mayank Mittal, Pascal Roth, James Tigue, Antoine Richard, Octi Zhang, Peter Du, Antonio Serrano-Muñoz, Xinjie Yao, René Zurbrügg, Nikita Rudin, et al. Isaac lab: A gpu-accelerated simulation framework for multi-modal robot learning. *arXiv preprint arXiv:2511.04831*, 2025.

Appendix A Metrics Formulation

A.1 Success Rate

We compute the overall and family success rates as

$$\mathrm{SR}_{\mathrm{All}} = \frac{1}{|\mathcal{T}|N} \sum_{t \in \mathcal{T}} \sum_{i=1}^{N} \mathbb{I}[s_{t,i} = 1], \quad \mathrm{SR}_f = \frac{1}{|\mathcal{T}_f|N} \sum_{t \in \mathcal{T}_f} \sum_{i=1}^{N} \mathbb{I}[s_{t,i} = 1],$$

where $s_{t,i}$ indicates whether episode i of task t succeeds,$\mathcal{T}$ denote the set of evaluated tasks and $\mathcal{T}_f$ denote the subset of tasks in family f.

A.2 Spatial Goal Error

Spatial Goal Error (SGE) is an auxiliary diagnostic metric that measures the final distance between the achieved spatial state and the target spatial goal. It is designed to capture final spatial precision, rather than full task completion.

For the i-th episode, we define

$$\mathrm{SGE}_i = d\left(\mathbf{s}_i^{\mathrm{final}}, \mathbf{s}_i^{\mathrm{goal}}\right), \tag{1}$$

where $\mathbf{s}_i^{\mathrm{final}}$ denotes the final achieved state and $\mathbf{s}_i^{\mathrm{goal}}$ denotes the target spatial state. For tasks with an explicit target position, this reduces to the Euclidean distance

$$\mathrm{SGE}_i = \left\| \mathbf{p}_i^{\mathrm{final}} - \mathbf{p}_i^{\mathrm{goal}} \right\|_2. \tag{2}$$

In our implementation, SGE is obtained from the evaluator-provided goal-distance proxy, such as `evaluation.goal_distance` or `distance_m`. If an episode does not provide a valid goal-distance value, its SGE is treated as missing and excluded from SGE aggregation. Therefore, whenever SGE is reported, we also record the number of valid SGE episodes.

For a model m and task family f, we aggregate SGE over valid episodes:

$$\overline{\mathrm{SGE}}_{m,f} = \frac{1}{|\mathcal{I}_{m,f}^{\mathrm{SGE}}|} \sum_{i \in \mathcal{I}_{m,f}^{\mathrm{SGE}}} \mathrm{SGE}_i, \tag{3}$$

where $\mathcal{I}_{m,f}^{\mathrm{SGE}}$ denotes the set of episodes with valid SGE values for model m and family f.

When reporting a family-level average across models, we first compute the model-family average and then average over models:

$$\overline{\mathrm{SGE}}_f = \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} \overline{\mathrm{SGE}}_{m,f}. \tag{4}$$

This avoids biasing the aggregate value toward models or runs with more evaluated episodes.

Interpretation. Lower SGE indicates better final spatial precision. However, SGE does not measure whether the agent followed the correct trajectory, selected the correct object throughout the rollout, or satisfied all spatial relations. It is therefore used only as an auxiliary diagnostic, not as a replacement for official task success.

A.3 Target Tracking Error

Target Tracking Error (TTE) is an auxiliary trajectory-level diagnostic metric. While SGE measures final spatial error, TTE measures whether the end-effector remains close to the intended target throughout the rollout. It is especially useful for analyzing spatial memory and target-tracking behavior.

For the i-th episode with trajectory length T_i, we define

$$\mathrm{TTE}_i = \frac{1}{T_i} \sum_{t=1}^{T_i} \left\| \mathbf{p}_{i,t}^{\mathrm{eef}} - \mathbf{p}_{i,t}^{\mathrm{tar}} \right\|_2, \tag{5}$$

where $\mathbf{p}_{i,t}^{\mathrm{eef}}$ is the end-effector position at timestep t, and $\mathbf{p}_{i,t}^{\mathrm{tar}}$ is the ground-truth target position at the same timestep.

When dynamic object poses are available in the trajectory trace, the target position is taken from the target object's dynamic pose:

$$\mathbf{p}_{i,t}^{\mathrm{tar}} = \mathbf{p}_{i,t}^{\mathrm{obj}}. \tag{6}$$

When dynamic object poses are unavailable, we use the static target position from the sampled layout as a proxy:

$$\mathbf{p}_{i,t}^{\mathrm{tar}} = \mathbf{p}_{i,0}^{\mathrm{tar}}. \tag{7}$$

In this case, the metric is marked as a static-layout proxy.

For a model m and task family f, TTE is first computed at the episode level and then averaged over valid episodes:

$$\overline{\mathrm{TTE}}_{m,f} = \frac{1}{|\mathcal{I}_{m,f}^{\mathrm{TTE}}|} \sum_{i \in \mathcal{I}_{m,f}^{\mathrm{TTE}}} \mathrm{TTE}_i, \tag{8}$$

where $\mathcal{I}_{m,f}^{\mathrm{TTE}}$ denotes the set of episodes with valid trajectory traces for model m and family f.

When reporting a family-level average across models, we compute

$$\overline{\mathrm{TTE}}_{f} = \frac{1}{|\mathcal{M}|} \sum_{m \in \mathcal{M}} \overline{\mathrm{TTE}}_{m,f}. \tag{9}$$

This model-wise aggregation prevents a run with more episodes from dominating the reported value. We avoid pooling all timesteps across episodes directly, because such pooling would give longer trajectories larger weight.

Interpretation. Lower TTE means that the end-effector stays closer to the intended target over the trajectory. Unlike SGE, which only evaluates the final state, TTE captures trajectory-level target following. However, TTE still does not directly verify successful grasping, placement, or spatial-relation satisfaction, so it is used as an auxiliary diagnostic rather than an official success metric.

Appendix B Detailed Experimental Setup

This section provides the implementation details needed to reproduce the GeoMind evaluation. We focus on the experimental variables that directly affect validity and reproducibility: simulator and robot setup, control frequency, policy inputs, action conversion, rollout protocol, baseline deployment, and information access. Unless otherwise stated, all methods are evaluated on the same task instances, random seeds, workspace constraints, episode horizon, and success criteria.

B.1 Simulation, Robot, and Control

All experiments are conducted in GeoMind Lab, an IsaacLab-based benchmark package built on Isaac Sim. We use a single Franka Research 3 (FR3) arm equipped with the FR3 parallel-jaw gripper. The robot receives relative end-effector commands and is controlled by a differential inverse-kinematics controller. The gripper is controlled through finger joint-position commands and is exposed to policies as a binary open/close command.

The physics simulation runs at 480 Hz, while policies are queried at 24 Hz. Each policy action is therefore executed through the low-level controller over 20 physics substeps. An episode terminates when the task succeeds, the robot violates the workspace guard, or the maximum horizon is reached. Table 5 summarizes the main simulator and control settings.

B.2 Observation and Action Interface

All end-to-end VLA policies are evaluated through a common observation and action interface. The policy input contains the language instruction, two RGB camera observations, and robot proprioception. The two cameras consist of a static table-view camera and a wrist-mounted camera, both rendered at 224×224 resolution.

Table 5: Simulation and control settings used for all evaluations.

Item	Setting
Simulator	Isaac Sim / IsaacLab
Benchmark package	GEOMIND LAB
Robot	Franka Research 3 (FR3)
Gripper	FR3 parallel-jaw gripper
Low-level controller	Differential inverse kinematics
Physics frequency	480 Hz
Policy control frequency	24 Hz
Episode horizon	1000 control steps
Termination	Success / workspace violation / horizon
Workspace guard (x, y, z)	$[0.15, 0.75] \times [-0.45, 0.45] \times [0.70, 1.20]$ m

Table 6: Observation and action interface. Privileged simulator states are excluded from VLA policy inputs.

Type	Field	Description
Policy input	Instruction	Natural-language task instruction
	Table-view RGB	Static external camera observation, 224×224
	Wrist RGB	Wrist-mounted camera observation, 224×224
	Joint state	Robot joint positions and velocities
	End-effector / gripper state	Current end-effector pose and gripper state
Policy action	Translation	Relative Cartesian displacement
	Rotation	Relative rotation-vector command
	Gripper	Binary open/close command
Evaluation only	Object poses	Used for success checking and diagnostics
	Object identities	Used for ground-truth target resolution
	Segmentation masks	Not exposed to VLA policies
	Task metadata	Used for evaluation and modular references
	Target regions / coordinates	Not exposed to VLA policies

Importantly, privileged simulator information is not exposed to VLA policies. Object poses, object identities, segmentation masks, ground-truth task variables, oracle target coordinates, and Task metadata are used only for evaluation and for the state-based modular reference agents described in Section B.5. Table 6 summarizes the policy interface and the evaluation-only states.

The common action interface is a 7-DoF relative end-effector command,

$$\mathbf{a}_t = [\Delta x, \Delta y, \Delta z, \Delta r_x, \Delta r_y, \Delta r_z, g],$$

where the first three dimensions represent Cartesian displacement, the next three dimensions represent a relative rotation-vector command, and the final dimension controls the gripper. For policies whose native output format differs from this interface, we convert the output to the same 7-DoF command before execution.

We use a unified normalized action protocol. Each policy output is first converted to a 7-DoF action vector and clipped to $[-1, 1]$. The translational and rotational components are then scaled by a model-specific multiplier s_m:

$$u_{xyz} = \mathrm{clip}(s_m a_{0:3}, -1, 1),$$

$$u_{rot} = \begin{cases} 0, & \text{if rotation is disabled for the model,} \\ \mathrm{clip}(2 s_m a_{3:6}, -1, 1), & \text{otherwise.} \end{cases}$$

The IsaacLab controller applies an additional controller scale of normalized gain, yielding the executed relative command

$$\Delta p = 0.5 u_{xyz} \text{ m}, \qquad \Delta r = 0.5 u_{rot} \text{ rad}.$$

Table 7: Evaluation protocol for GEOMIND-BASE and GEOMIND-AUG.

Item	Setting
GEOMIND-BASE	20 variants, 48 episodes per variant, 960 episodes per method
GEOMIND-AUG	34 augmented variants, 48 episodes per variant, 1632 episodes per method
Task instances	Shared across methods
Random seeds	Shared across methods
Episode horizon	1000 control steps
Policy control frequency	24 Hz
Physics frequency	480 Hz
Camera resolution	224×224
Primary metric	Success rate
VLA adaptation	No GeoMind-specific finetuning

The gripper command is thresholded at zero: positive values are mapped to open and negative values are mapped to close. For baselines without reliable gripper outputs, we keep the gripper fixed according to their default deployment protocol and report this setting explicitly.

B.3 Evaluation Protocol

We evaluate each method on the same task instances and random seeds. Each evaluated task variant contains 48 evaluation episodes. GEOMIND-BASE contains 20 variants, resulting in 960 episodes per method. GEOMIND-AUG contains 34 augmented variants, resulting in 1632 episodes per method, for 2592 episodes across both suites. Success rate is the primary metric and is computed over all episodes in the corresponding task suite or task family.

B.4 VLA Baseline Deployment

All VLA baselines are evaluated as frozen policies without GeoMind-specific finetuning. Each model is deployed as a remote policy server and queried online during rollout. For each model, we use a fixed checkpoint and server configuration across all evaluated episodes.

The policy server receives only the allowed policy inputs described in Table 6. No VLA baseline receives privileged simulator object poses, segmentation masks, ground-truth task variables, oracle target coordinates, or task metadata through the policy interface.

Some VLA models return action chunks rather than a single action. In our deployment, the client executes only the first valid action from the returned chunk and then queries the policy again at the next control step. This keeps all policies closed-loop at the shared 24 Hz control frequency while preserving each model's native action-generation interface.

Different VLA models use different native action heads and deployment conventions. We therefore report the model-specific action multiplier, checkpoint setting, rotation setting, and gripper setting in Table 8. These settings are fixed for a given model across the full evaluation.Simulations were conducted on a workstation equipped with an NVIDIA RTX 3090 GPU, while model inference was performed on an NVIDIA RTX 4090 (48GB). A single evaluation round takes approximately 3 hours.

B.5 Modular Baselines and Diagnostic Use

We additionally evaluate modular agents based on COPA [58] and MOKA [59]. These methods represent a different embodied-agent paradigm from end-to-end VLA policies: instead of directly predicting low-level actions from visual observations and language instructions, they decompose manipulation into explicit visual-language grounding, target selection, and scripted execution. We include them as modular baselines to compare different design paradigms for structured spatial manipulation, and also use their intermediate predictions to diagnose where failures arise.

Each modular agent receives the task instruction and RGB observations, and uses a VLM backbone to identify the task-relevant object or spatial target from the scene. We instantiate the modular agents

Table 8: Deployment settings of VLA baselines. s_m denotes the normalized action gain used before controller-level command limiting. All models are evaluated with fixed checkpoints and without GeoMind-specific finetuning.

Model	**GeoMind finetune**	**Checkpoint**	s_m	**Rotation / gripper**
OpenVLA	No	Fixed	2.0	Rotation enabled; binary gripper
π_0	No	Fixed	0.5	Rotation disabled; fixed gripper
$\pi_{0.5}$	No	Fixed	0.5	Rotation disabled; fixed gripper
SpatialVLA	No	Fixed	5.0	Rotation enabled; binary gripper
InternVLA-M1	No	Fixed	0.5	Rotation enabled; binary gripper
Spatial-Forcing	No	Fixed	0.2	Rotation enabled; binary gripper

Table 9: Information access for end-to-end VLA policies and modular baselines. Both method families are evaluated without privileged simulator information during inference. Modular baselines use explicit VLM-based grounding and scripted execution, so their results are interpreted as cross-paradigm comparisons with additional diagnostic value.

Information	**VLA policies**	**Modular baselines**
Language instruction	Yes	Yes
RGB observations	Yes	Yes
Robot proprioception	Yes	Yes
Object poses	No	No
Object identities	No	No
Segmentation masks	No	No
Task metadata	No	No
Oracle target coordinates	No	No
Ground-truth task variables	No	No
GeoMind-specific finetuning	No	No
Role in evaluation	End-to-end policy	Modular baseline
Inference structure	Direct action prediction	VLM grounding + scripted execution
Diagnostic signal	Trajectory/action behavior	Grounding and target-selection errors

with three VLM backbones: GPT-4o, Gemini-2.5-Flash, and Claude-Sonnet-4.6. The predicted target is then converted into Cartesian waypoints or a target region and executed with the same Franka differential-IK controller used in the VLA evaluation. For pick-and-place tasks, the scripted routine follows a top-down sequence of approach, grasp, lift, place, release, and retreat. For reaching tasks, the controller moves the end effector toward the predicted target region and evaluates success using the same task evaluator as other methods.

All modular agents are evaluated on the same task instances, random seeds, workspace constraints, episode horizon, and success criteria as the end-to-end policies. They do not receive privileged simulator information during inference: oracle target coordinates, simulator-resolved object identities, ground-truth task variables, segmentation masks, and task metadata are not exposed to the VLM or the planner. These privileged states are used only by the evaluator for success checking and diagnostic logging.

Because modular agents use a different inference and execution structure from VLA policies, their results should be interpreted as a comparison across embodied-agent paradigms rather than as a fully controlled comparison of identical policy interfaces. This distinction is important: end-to-end policies test whether a learned visuomotor model can directly map observations and language to actions, whereas modular agents test whether explicit VLM-based grounding and scripted execution can solve the same spatial objectives. Their low success rates therefore provide complementary evidence that structured spatial understanding is difficult not only for end-to-end action generation, but also for explicit target grounding and target-to-action conversion pipelines.

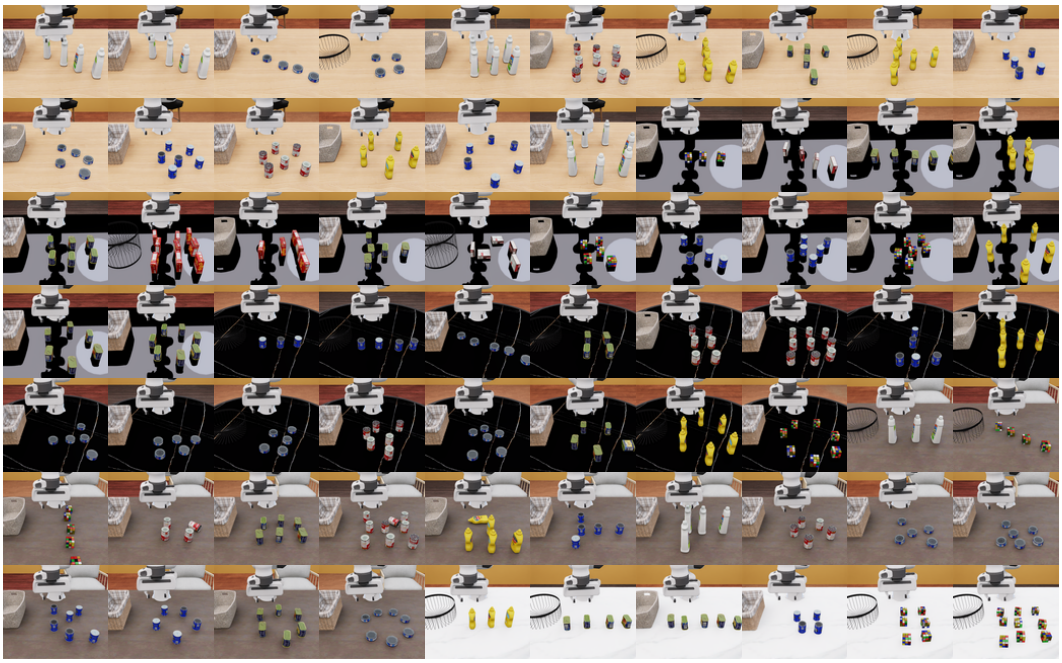


Figure 6: Structure gallery from our Structure Modeling and Constructing Pipeline

.

B.6 Fairness and Information Access

The main comparison in GEOMIND covers multiple embodied-agent paradigms under the same task instances, random seeds, workspace constraints, episode horizon, and success criteria. For end-to-end VLA policies, all models receive the same observation modalities and are evaluated as frozen policies without GeoMind-specific finetuning. For modular baselines, the VLM grounding stage receives the same task instruction and RGB observations, while execution is performed by a scripted controller after target selection.

No evaluated method receives privileged simulator information during inference. Object poses, simulator-resolved object identities, oracle target coordinates, ground-truth task variables, segmentation masks, and task metadata are excluded from both VLA policies and modular baselines. These states are used only for evaluation, success checking, and diagnostic logging.

Since VLA policies and modular baselines differ in inference structure, their results should be read as a cross-paradigm comparison rather than a comparison of identical policy interfaces. This is intentional: GEOMIND aims to evaluate whether different embodied-agent designs can solve the same structured spatial objectives under the same task conditions. The modular results complement the VLA results by revealing whether explicit visual grounding and scripted execution can overcome the spatial failures observed in end-to-end policies.

Appendix C Spatial Structure Formalization

This appendix gives a compact description of the spatial structures used in GEOMIND. Fig. 6 shows representative structure layouts. Our goal is not to bind each task to one fixed object coordinate template. Instead, each task is associated with a spatial structure, and GEOMIND samples multiple physical realizations that preserve the same task-relevant relation.

C.1 Structure-Preserving Sampling

Let a scene contain objects $\mathcal{O} = \{o_i\}_{i=1}^{N}$. The pose of object o_i is

$$\mathbf{x}_i = (\mathbf{p}_i, \mathbf{q}_i), \tag{10}$$

where $\mathbf{p}_i$ is the object position and $\mathbf{q}_i$ is the object orientation. For tabletop relations, we use the planar position $\bar{\mathbf{p}}_i = (x_i, y_i)$. The full scene state is

$$\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^{N}. \tag{11}$$

We denote a spatial structure as S. It specifies what should remain invariant during sampling, such as an object order, a grid layout, an anchor-based relation, or a geometric target rule. Given S, GEOMIND samples object poses as

$$\mathbf{X} \sim \mathcal{P}(\mathbf{X} \mid S, \epsilon), \tag{12}$$

where ϵ controls small layout variations such as jitter or spacing changes. A sampled scene is accepted only if

$$\mathbf{X} \in \Omega_{\text{struct}}(S, \epsilon) \cap \Omega_{\text{phys}}. \tag{13}$$

Here, Ω_{struct} means the intended spatial relation is still correct, and Ω_{phys} means the scene is physically valid in simulation.

In short, S describes the stable spatial meaning of the task, while $\mathbf{X}$ is one sampled scene that realizes this meaning.

C.2 Representative Spatial Structures

As illustrated in Fig. 6, GEOMIND uses several common spatial structures. We summarize them below with minimal constraints.

Ordered structure. This structure is used for tasks such as "pick the leftmost object" or "pick the second object from the right." Let $\mathbf{u}$ be the ordering direction and define the projected coordinate

$$s_i = \mathbf{u}^\top \bar{\mathbf{p}}_i. \tag{14}$$

If the intended order is $o_{\pi(1)} \prec \cdots \prec o_{\pi(N)}$, then a valid sample must satisfy

$$s_{\pi(k+1)} - s_{\pi(k)} \geq \delta_{\min}. \tag{15}$$

This keeps the object order unchanged while allowing the exact coordinates to vary.

Grid structure. This structure is used for row-column layouts, such as "the object in the top-left cell" or "the object in the same row as the bowl." Each object has a grid index $g_i = (r_i, c_i)$. A sampled position is generated around the corresponding grid cell:

$$\bar{\mathbf{p}}_i = \mathbf{c} + (c_i - 1)d_x\mathbf{u} + (r_i - 1)d_y\mathbf{v} + \boldsymbol{\eta}_i, \qquad \|\boldsymbol{\eta}_i\|_\infty \leq \epsilon. \tag{16}$$

Thus, the visual layout can vary, but the row-column relation is preserved.

Anchor-based relation. This structure is used for tasks such as "the object left of the cup", "the object near the plate", or "the object between two bowls." For an object o_i and an anchor o_a, define

$$\Delta_{ia} = \bar{\mathbf{p}}_i - \bar{\mathbf{p}}_a. \tag{17}$$

A directional relation is valid when

$$\mathbf{u}_r^\top \Delta_{ia} \geq \tau_r, \tag{18}$$

where $\mathbf{u}_r$ is the relation direction and τ_r avoids ambiguous cases. The target is accepted only when the relation selects one unique object:

$$o^* = \arg\operatorname*{unique}_{o_i \in \mathcal{O}} R(o_i, o_a). \tag{19}$$

Geometric target. This structure is used when the target is an implicit position rather than a visible object. For example, the target may be the midpoint between two anchors:

$$\mathbf{g}_{\text{mid}} = \frac{1}{2}(\bar{\mathbf{p}}_a + \bar{\mathbf{p}}_b), \tag{20}$$

or the center of multiple anchors:

$$\mathbf{g}_{\text{center}} = \frac{1}{|\mathcal{A}|} \sum_{a \in \mathcal{A}} \bar{\mathbf{p}}_a. \tag{21}$$

These tasks require the agent to infer a target from scene geometry, instead of directly selecting an existing object.

C.3 Dynamic Relations

For tasks with scene changes, GEOMIND distinguishes two cases. In scene-updating tasks, the target is determined by the current relation:

$$o_t^* = \arg \underset{o_i \in \mathcal{O}}{\text{unique}}\, R_t(o_i, o_a). \tag{22}$$

For example, the correct object is the one currently left of the cup.

In identity-tracking tasks, the target is determined by the initial relation and then kept fixed:

$$o^* = \arg \underset{o_i \in \mathcal{O}}{\text{unique}}\, R_0(o_i, o_a), \qquad \text{target}(t) = o^*. \tag{23}$$

For example, the correct object is the one that was originally left of the cup, even if it later moves away.

C.4 Physical Validation

Finally, each sampled scene is checked in the simulator. We only keep instances where objects do not overlap, objects are inside the workspace, and the target is reachable by the robot:

$$\|\bar{\mathbf{p}}_i - \bar{\mathbf{p}}_j\|_2 \geq \rho_i + \rho_j + d_{\text{safe}}, \qquad \bar{\mathbf{p}}_i \in \Omega_{\text{table}}, \qquad \mathbf{g} \in \Omega_{\text{reach}}. \tag{24}$$

These checks ensure that the generated task is not only spatially well-defined, but also physically executable.

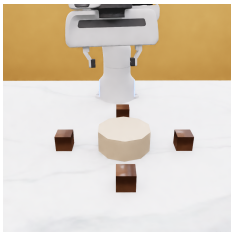
C.5 Summary

The structure gallery in Fig. 6 provides visual examples of the layouts used in GEOMIND. The formulas above specify only the minimal constraints needed to preserve these structures during sampling. This allows GEOMIND to generate diverse scenes with the same task semantics, so evaluation focuses on spatial understanding rather than memorization of fixed coordinates.

Appendix D Task Details

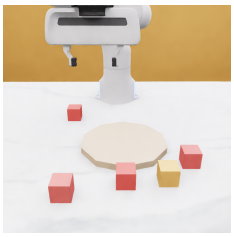
We present 26 task cards covering a subset of the 27 non-execution variants. Execution-augmented variants impose stricter action precision constraints on these non-execution variants. The full 54 evaluated variants consist of 20 base variants and 34 augmented variants: 4 semantic, 3 visual, and 27 execution variants.

Each card gives a compact specification of one task variant, including the task instruction, the corresponding suite and ability family, the spatial structure used for scene generation, the ground-truth target, the numerical ground-truth pose, and the evaluation criterion. The scene image on the left illustrates one sampled realization of the task, while the fields on the right define the task in a reproducible form. All target poses are reported in the simulator world frame, and distances are measured in meters unless otherwise stated.



BindingBase_AnchorDirection_v0

Task suite and family. GEOMIND-BASE / Target Identification
Instruction. Move the end effector near the cube to the right of the blue bowl.
Structure. anchor_direction_injected structure.
GT Target. Protocol target object block_right proximity region.
GT Pose. $\mathbf{p}^* = (0.40, -0.15, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.



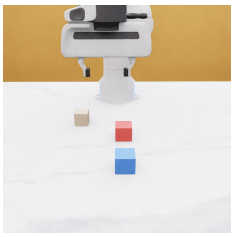
ConstraintBase_SelectExclusionUnique_v0

Task suite and family. GEOMIND-BASE / Constraint Composition
Instruction. Move the end effector near the red cube closest to the blue plate that is not the leftmost.
Structure. exclusion_distance_constraint_injected structure.
GT Target. Protocol target object red_block_target proximity region.
GT Pose. $\mathbf{p}^* = (0.56, 0.02, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.



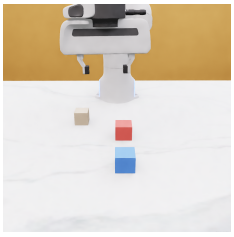
ConstraintBase_SelectVisualSpatial_v0

Task suite and family. GEOMIND-BASE / Constraint Composition
Instruction. Move the end effector near the wood-grain cylinder to the right of the red block.
Structure. visual_spatial_constraint_injected structure.
GT Target. Protocol target object wood_cylinder_target proximity region.
GT Pose. $\mathbf{p}^* = (0.40, -0.08, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.



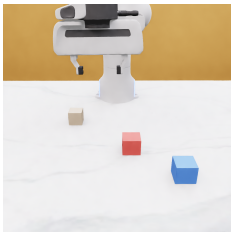
GeometricBase_IsometricExtension_v0

Task suite and family. GEOMIND-BASE / Geometric Inference
Instruction. You already hold the cube. Place it on the red–blue line extension at equal spacing.
Structure. 2-anchor line_blocks_runtime_injected structure.
GT Target. Implicit equal-spacing extension point beyond the second anchor.
GT Pose. $\mathbf{p}^* = (0.66, 0.02, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final object center distance < 0.05 m and remains stable for 1.0 s.



GeometricBase_PerfectMidpoint_v0

Task suite and family. GEOMIND-BASE / Geometric Inference
Instruction. You already hold the cube. Place it midway between the red and blue reference cubes.
Structure. 2-anchor line_blocks_runtime_injected structure.
GT Target. Implicit midpoint region between the two colored anchor blocks.
GT Pose. $\mathbf{p}^* = (0.42, 0.02, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final object center distance < 0.05 m and remains stable for 1.0 s.



GeometricBase_PointSymmetry_v0

Task suite and family. GEOMIND-BASE / Geometric Inference
Instruction. You already hold the cube. Place it symmetrically to the blue cube about the red cube.
Structure. 2-anchor diagonal_blocks_runtime_injected structure.
GT Target. Implicit point-symmetric reflection of the reference anchor across the center anchor.
GT Pose. $\mathbf{p}^* = (0.30, -0.08, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final object center distance < 0.05 m and remains stable for 1.0 s.



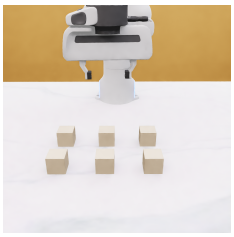
GeometricBase_SquareCompletion_v0

Task suite and family. GEOMIND-BASE / Geometric Inference
Instruction. You already hold the fourth cube. Place it to complete a square with the three anchor cubes.
Structure. 3-anchor square_corner_runtime_injected structure.
GT Target. Implicit missing square corner inferred from three anchors.
GT Pose. $\mathbf{p}^* = (0.48, 0.12, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final object center distance < 0.05 m and remains stable for 1.0 s.



GeometricBase_TrendContinuation_v0

Task suite and family. GEOMIND-BASE / Geometric Inference
Instruction. You already hold the next cube. Place it to continue the visible row/column pattern.
Structure. 3-anchor diagonal_trend_blocks_runtime_injected structure.
GT Target. Implicit next point in the anchor trend.
GT Pose. $\mathbf{p}^* = (0.61, 0.15, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final object center distance < 0.05 m and remains stable for 1.0 s.



BindingBase_GridRowcolGrounding_v0

Task suite and family. GEOMIND-BASE / Target Identification
Instruction. Move the end effector near the cube at row 1, column 2.
Structure. runtime_grid_2x3_static structure.
GT Target. Protocol target object grid_cell_1 proximity region.
GT Pose. $\mathbf{p}^* = (0.37, -0.03, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.



BindingBase_OrdinalInsertStatic_v0

Task suite and family. GEOMIND-BASE / Target Identification
Instruction. Move the end effector near the second cube from the left.
Structure. runtime_block_line_static_ordinal structure.
GT Target. Protocol target object ordinal_static_1 proximity region.
GT Pose. $\mathbf{p}^* = (0.43, -0.06, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.



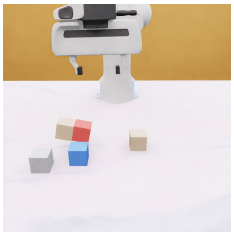
ReindexBase_ReindexExtreme_v0

Task suite and family. GEOMIND-BASE / Scene Updating
Instruction. Move the end effector near the leftmost cube after reordering.
Structure. runtime_block_line structure.
GT Target. Protocol target object reindex_block_1 proximity region.
GT Pose. $\mathbf{p}^* = (0.42, -0.05, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.



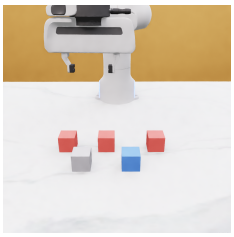
ReindexBase_ReindexInsert_v0

Task suite and family. GEOMIND-BASE / Scene Updating
Instruction. Move the end effector near the second-from-left cube.
Structure. runtime_block_line_insert structure.
GT Target. Protocol target object insert_block_3 proximity region.
GT Pose. $\mathbf{p}^* = (0.43, 0.18, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.



ReindexBase_ReindexNeighbor_v0

Task suite and family. GEOMIND-BASE / Scene Updating
Instruction. Move the end effector near the cube immediately left of the red cube.
Structure. runtime_red_anchor_neighbor structure.
GT Target. Protocol target object right_neighbor_block proximity region.
GT Pose. $\mathbf{p}^* = (0.40, 0.06, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.



ReindexBase_ReindexSubset_v0

Task suite and family. GEOMIND-BASE / Scene Updating
Instruction. Move the end effector near the middle cube among the red cubes only.
Structure. runtime_red_subset_with_distractors structure.
GT Target. Protocol target object red_subset_block_2 proximity region.
GT Pose. $\mathbf{p}^* = (0.40, 0.11, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.



ReindexAugSem_ReindexExtreme_v0

Task suite and family. GEOMIND-AUG / Scene Updating
Instruction. Move the end effector near the leftmost cube after reordering.
Structure. runtime_block_line structure.
GT Target. Protocol target object reindex_block_1 proximity region.
GT Pose. $\mathbf{p}^* = (0.42, -0.05, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.



ReindexAugVisual_ReindexExtreme_v0

Task suite and family. GEOMIND-AUG / Scene Updating
Instruction. Move the end effector near the leftmost cube after reordering.
Structure. runtime_block_line structure.
GT Target. Protocol target object reindex_block_1 proximity region.
GT Pose. $\mathbf{p}^* = (0.42, -0.05, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.



RelationAugSem_PlaceDirectional_v0

Task suite and family. GEOMIND-AUG / Relational Grounding
Instruction. You already hold the wooden cube. Place it in the legal region in front of the blue plate.
Structure. directional_region_semantic_distractors structure.
GT Target. Directional target zone relative to blue_plate.
GT Pose. $\mathbf{p}^* = (0.57, 0.02, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final object center distance < 0.09 m and remains stable for 1.0 s.



RelationAugVisual_PlaceDirectional_v0

Task suite and family. GEOMIND-AUG / Relational Grounding
Instruction. You already hold the wooden cube. Place it in the legal region in front of the blue plate.
Structure. directional_region_visual_clutter structure.
GT Target. Directional target zone relative to blue_plate.
GT Pose. $\mathbf{p}^* = (0.57, 0.02, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final object center distance < 0.09 m and remains stable for 1.0 s.



RelationBase_PlaceDirectional_v0

Task suite and family. GEOMIND-BASE / Relational Grounding
Instruction. You already hold the wooden cube. Place it in the legal region in front of the blue plate.
Structure. directional_region_injected structure.
GT Target. Directional target zone relative to blue_plate.
GT Pose. $\mathbf{p}^* = (0.65, 0.02, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final object center distance < 0.09 m and remains stable for 1.0 s.



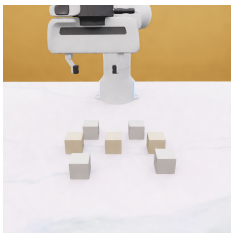
RelationBase_PlaceInside_v0

Task suite and family. GEOMIND-BASE / Relational Grounding
Instruction. You already hold the cube. Put it fully inside the blue box.
Structure. containment_standard_injected structure.
GT Target. Stable containment inside blue_box_floor.
GT Pose. $\mathbf{p}^* = (0.43, 0.04, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final object containment stability and center error < 0.06 m and remains stable for 1.0 s.



RelationBase_PlaceOnTop_v0

Task suite and family. GEOMIND-BASE / Relational Grounding
Instruction. You already hold the cube. Release it onto the top surface of the red wooden block so it rests stably.
Structure. support_top_injected structure.
GT Target. Stable top placement on red_support_block.
GT Pose. $\mathbf{p}^* = (0.41, 0.03, 0.07)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final object support-top stability and placement error < 0.01 m and remains stable for 1.0 s.



TrackBase_TrackClutter_v0

Task suite and family. GEOMIND-BASE / Identity Tracking
Instruction. Move the end effector near the cube that started in the middle after clutter settles.
Structure. clutter_robust_runtime_identity_line structure.
GT Target. Protocol target object identity_block_1 proximity region.
GT Pose. $\mathbf{p}^* = (0.41, -0.01, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.



TrackBase_TrackMulti_v0

Task suite and family. GEOMIND-BASE / Identity Tracking
Instruction. Move the end effector near the cube that started at the far left.
Structure. runtime_identity_multistep_line structure.
GT Target. Protocol target object identity_block_0 proximity region.
GT Pose. $\mathbf{p}^* = (0.40, -0.20, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.



TrackBase_TrackSingle_v0

Task suite and family. GEOMIND-BASE / Identity Tracking
Instruction. Move the end effector near the cube that started in the middle.
Structure. runtime_identity_line structure.
GT Target. Protocol target object identity_block_1 proximity region.
GT Pose. $\mathbf{p}^* = (0.41, -0.01, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.



TrackAugSem_TrackSingle_v0

Task suite and family. GEOMIND-AUG / Identity Tracking
Instruction. Move the end effector near the cube that started in the middle.
Structure. runtime_identity_line structure.
GT Target. Protocol target object identity_block_1 proximity region.
GT Pose. $\mathbf{p}^* = (0.41, -0.01, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.



TrackAugVisual_TrackSingle_v0

Task suite and family. GEOMIND-AUG / Identity Tracking
Instruction. Move the end effector near the cube that started in the middle.
Structure. runtime_identity_line structure.
GT Target. Protocol target object identity_block_1 proximity region.
GT Pose. $\mathbf{p}^* = (0.41, -0.01, 0.03)$ m; $\theta^* = 0.00$ rad.
Eval. Success if final end-effector distance to target object center < 0.06 m and remains stable for 1.0 s.

Appendix E Overall Experimental Results

This section reports the complete experimental results for all evaluated agents on GEOMIND.

While the main paper focuses on the major trends, the appendix provides the full tables for detailed comparison and reproducibility.

Table 10: Detailed per-task success rates on the 54-variant VLA evaluation set.

Task ID	*Open VLA*	π_0	$\pi_{0.5}$	*Spatial VLA*	*Intern VLA*	*Spatial Forcing*
BindingBase_AnchorDirection_v0	75.0	0.0	0.0	0.0	0.0	12.5
BindingAugExcu_AnchorDirection_v0	0.0	0.0	0.0	0.0	0.0	0.0
BindingAugBase_AnchorDistance_v0	37.5	18.8	18.8	62.5	62.5	12.5
BindingAugExcu_AnchorDistance_v0	50.0	0.0	0.0	75.0	75.0	0.0
ConstraintBase_SelectExclusionUnique_v0	0.0	0.0	0.0	0.0	12.5	0.0
ConstraintAugExcu_SelectExclusionUnique_v0	0.0	0.0	0.0	0.0	0.0	0.0
ConstraintBase_SelectVisualSpatial_v0	37.5	0.0	0.0	0.0	0.0	100.0
ConstraintAugExcu_SelectVisualSpatial_v0	0.0	0.0	0.0	0.0	0.0	75.0
GeometricBase_IsometricExtension_v0	0.0	0.0	0.0	12.5	0.0	0.0
GeometricAugExcu_IsometricExtension_v0	0.0	0.0	0.0	0.0	0.0	0.0
GeometricBase_PerfectMidpoint_v0	87.5	0.0	0.0	62.5	87.5	25.0
GeometricAugExcu_PerfectMidpoint_v0	75.0	0.0	0.0	75.0	100.0	25.0
GeometricBase_PointSymmetry_v0	0.0	0.0	0.0	0.0	0.0	0.0
GeometricAugExcu_PointSymmetry_v0	0.0	0.0	0.0	0.0	0.0	0.0
TrackBase_TrackClutter_v0	25.0	6.2	6.2	12.5	25.0	100.0
TrackAugExcu_TrackClutter_v0	25.0	12.5	12.5	0.0	0.0	100.0
TrackBase_TrackMulti_v0	0.0	37.5	37.5	0.0	0.0	0.0
TrackAugExcu_TrackMulti_v0	0.0	25.0	25.0	0.0	0.0	0.0
TrackBase_TrackSingle_v0	50.0	12.5	12.5	12.5	37.5	12.5
TrackAugExcu_TrackSingle_v0	25.0	0.0	0.0	50.0	0.0	50.0
TrackBaseSem_TrackSingle_v0	50.0	0.0	0.0	37.5	12.5	37.5
TrackAugExcuSem_TrackSingle_v0	25.0	0.0	0.0	25.0	0.0	25.0
TrackBaseVisual_TrackSingle_v0	25.0	0.0	0.0	50.0	25.0	50.0
TrackAugExcuVisual_TrackSingle_v0	25.0	0.0	0.0	50.0	25.0	50.0
GeometricBase_TrendContinuation_v0	0.0	0.0	0.0	0.0	0.0	0.0
GeometricAugExcu_TrendContinuation_v0	0.0	0.0	0.0	0.0	0.0	0.0
GeometricBase_SquareCompletion_v0	50.0	0.0	0.0	12.5	0.0	0.0

Continued on next page

Table 10 continued from previous page

Task ID	*Open VLA*	π_0	$\pi_{0.5}$	*Spatial VLA*	*Intern VLA*	*Spatial Forcing*
GeometricAugExcu_SquareCompletion_v0	50.0	0.0	0.0	0.0	0.0	0.0
RelationAugSem_PlaceDirectional_v0	0.0	0.0	0.0	0.0	0.0	0.0
RelationAugExcuSem_PlaceDirectional_v0	0.0	0.0	0.0	0.0	0.0	0.0
RelationAugVisual_PlaceDirectional_v0	0.0	0.0	0.0	0.0	0.0	0.0
RelationAugExcuVisual_PlaceDirectional_v0	0.0	0.0	0.0	0.0	0.0	0.0
RelationBase_PlaceDirectional_v0	0.0	0.0	0.0	0.0	0.0	0.0
RelationAugExcu_PlaceDirectional_v0	0.0	0.0	0.0	0.0	0.0	0.0
RelationBase_PlaceInside_v0	12.5	0.0	0.0	12.5	0.0	0.0
RelationAugExcu_PlaceInside_v0	0.0	0.0	0.0	25.0	0.0	0.0
RelationBase_PlaceOnTop_v0	0.0	0.0	0.0	0.0	0.0	0.0
RelationAugExcu_PlaceOnTop_v0	50.0	0.0	0.0	0.0	0.0	0.0
BindingBase_GridRowcolGrounding_v0	0.0	0.0	0.0	0.0	75.0	12.5
BindingAugExcu_GridRowcolGrounding_v0	0.0	0.0	0.0	0.0	25.0	0.0
BindingBase_OrdinalInsertStatic_v0	75.0	25.0	25.0	12.5	25.0	100.0
BindingAugExcu_OrdinalInsertStatic_v0	75.0	12.5	12.5	75.0	25.0	100.0
ReindexBase_ReindexExtreme_v0	62.5	93.8	93.8	37.5	50.0	100.0
ReindexAugExcu_ReindexExtreme_v0	50.0	50.0	50.0	0.0	0.0	75.0
ReindexBaseSem_ReindexExtreme_v0	62.5	81.2	81.2	50.0	62.5	87.5
ReindexAugExcuSem_ReindexExtreme_v0	50.0	75.0	75.0	25.0	25.0	75.0
ReindexBaseVisual_ReindexExtreme_v0	50.0	75.0	75.0	37.5	75.0	75.0
ReindexAugExcuVisual_ReindexExtreme_v0	25.0	50.0	50.0	0.0	0.0	75.0
ReindexBase_ReindexInsert_v0	0.0	0.0	0.0	0.0	0.0	0.0
ReindexAugExcu_ReindexInsert_v0	0.0	0.0	0.0	0.0	0.0	0.0
ReindexBase_ReindexNeighbor_v0	25.0	12.5	12.5	100.0	0.0	0.0
ReindexAugExcu_ReindexNeighbor_v0	25.0	0.0	0.0	50.0	0.0	0.0
ReindexBase_ReindexSubset_v0	37.5	0.0	0.0	0.0	0.0	0.0
ReindexAugExcu_ReindexSubset_v0	50.0	0.0	0.0	0.0	0.0	0.0

Appendix F Prompt for Tasks Generating

Coordinate convention. GEOMIND uses two coordinate conventions for different purposes.

The coordinates provided in the task-generation prompt are expressed in the robot/simulator coordinate frame and are used only by the generation script to instantiate object layouts and check physical feasibility.

They are not exposed to evaluated agents.

In contrast, natural-language spatial relations used in the benchmark, such as *left*, *right*, *front*, and *behind*, are defined from the canonical table-view camera frame.

This is the coordinate convention used in the main benchmark tasks, because agents observe the scene through camera images and should interpret spatial language according to the visual observation.

During task instantiation, object poses are maintained in the simulator frame for stable physics and control, and are transformed to the camera-view convention when resolving view-dependent spatial predicates.

Prompt

```
You are a embodied intelligent task designer. You need to design a series of
    tasks for the scenario to enhance the agent's perception of spatial
    relationships.
```

We have the following basic task types:

Topological Relationship: Refers to the topological relationships between objects. For example, whether there are objects in contact with each other, whether there is an object inside a basket or other container. The tasks you can generate for this type of task are similar to:

- Put the canned food in the basket

- Take the bottle out of the basket

Projective Relation: Refers to the directional relationship between objects. For example, whether the object is on the left or right side of the reference object. The tasks you can generate for this type of task are similar to:

- Put the canned food on the left side

- Pick up the box on your right

Ordinary Relation: Refers to the sequential relationship between objects, such as the order in which objects are counted from top to bottom. The tasks you can generate for this type of task are similar to:

- Pick up the first can from top to bottom

- Pick up the can in the middle of this row

Structural Relation: Refers to the structure of object placement. For example, objects are placed in a square, circular, or T-shaped pattern. The tasks you can generate for this type of task are similar to:

- Pick up the bottle at the intersection of the T-shaped shape

- Pick up the can located in the 1 o'clock direction

- Arrange all the boxes in a straight line

We also have some composite task types, listed below are some:

Topological →Projective: a combination of the two types of tasks mentioned earlier. The tasks you can generate for this type of task are similar to:

- Put the leftmost can in the basket

- Put the box on the shelf on the right

Projective →Structural: a combination of the two types of tasks mentioned earlier. The tasks you can generate for this type of task are similar to:

- Pick up the bottles from the left column of the formation and place them on the right side of the formation

- Pick up the can located at 3 o'clock on the ring and place it in the center of the ring

Ordinal →Projective: a combination of the two types of tasks mentioned earlier. The tasks you can generate for this type of task are similar to:

- Pick up the second bottle from left to right and place it in the bottle of the object on the far right

- Pick up the top can and place it on the left

Structural →Topological: a combination of the two types of tasks mentioned earlier. The tasks you can generate for this type of task are similar to:

- Pick up the can in the center of the formation and put it in the basket

- Take off the boxes that are pressed on top of the other boxes, and then arrange all the boxes in a row

Structural →Ordinal: a combination of the two types of tasks mentioned earlier. The tasks you can generate for this type of task are similar to:

- Pick up the cans in the second column from the left and the center row of the square

- Pick up the can in the upper left corner of the square and place it next to the can in the lower right corner

Mixed Composite: A combination of two or more task types mentioned earlier. The tasks you can generate for this type of task are similar to:

- Pick up the can in the upper left corner of the square and stack it on top of the can in the lower right corner

- Arrange all the boxes in a row, and then put the third box from right to left into the basket

- Put the top can in the box, and then stack the rightmost can in the second row on top of the leftmost can

We also have some more complex types of tasks, in which the spatial relationships of objects undergo more changes. Below is a list of some:

Sequential Reconfigure: Operate the object multiple times and place it in various directions of the reference object. The tasks you generate for this type of task are similar to:

- Put the cans on the left side of the basket, then on the right side of the basket, and finally on the left side of the other cans

- Take down the top can, place it on the left side of the other cans, and then place it on the right side of the other cans

Dynamic Reference Switching: involves changing reference objects and requires multi-step operations. The tasks you generate for this type of task are similar to:

- Put the can on the far left into the basket, and then put the can closest to the basket into the basket

- Put the cans in the top left corner of the square into the basket, then place the cans in the center row of the column closest to the basket in the original position of the cans that were just taken, and then place the cans closest to you into the basket

Goal Replanning: The goal changes midway and requires multiple steps of operation. The tasks you generate for this type of task are similar to:

- Arrange all the bottles in a line... no, arrange all the bottles in a square matrix of 2 rows and 2 columns

- Stack all the cans up... no, now line them up in a line... no, let's just stack them up

```
**Interactive Dialogue**: It involves answering and revising tasks, and the
    tasks you generate are similar to:

- Pick up the object on the far side. (Not that one?) The other one nearby.

- Place cube left.(Is this correct) Move a bit right.

**Sequential Spatial Reordering:** It involves changing the order of objects and
     requires multi-step operations. The tasks you generate for this type of
    task are similar to:

- Place the leftmost bottle on the rightmost side and keep all bottles in a
    straight line. Pick up the bottle currently located in the center and place
     it on the leftmost side while keeping all bottles in a straight line. Pick
     up the currently third bottle from left to right and place it in the
    basket

- Place the cans located in the two o'clock direction on the ring at the center
    of the ring, then place the cans located in the six o'clock direction in
    the two o'clock direction, then place the cans located in the center of the
     ring in the six o'clock direction, and stack the cans currently located in
     the two o'clock direction above the cans located in the eight o'clock
    direction.



I will provide you with the location of all objects and their placement (
    including Line/Grid/Shape/Ring/Random). The robotic arm is located at the
    coordinate (0,0) on the plane, facing the positive x-axis direction, and
    the negative y-axis direction is on the left side of the robotic arm. You
    need to randomly select five to eight types of tasks mentioned earlier and
    generate task descriptions for them, which can be repeated. Output your
    generated task in the following JSON format:

[
       {
    "task_category": "Goal Replanning",
    "instruction": "Move them closer to the center... no, spread them out
        further apart."
       },
       {
    "task_category": "Structural Relation",
    "instruction": "Arrange all the scattered objects into a straight horizontal
         line."
   }
]

That is, a tasks array containing several dictionaries, each containing two keys
    . task_category is the task type, and instructions are the task
    descriptions generated by you.DO NOT output any other content besides this
    JSON.

Here are some examples for you to learn from:

========INPUT========

Object placement: Grid
Objects: 9 can are located at:
[0.257, -0.050]
[0.275, 0.101]
[0.269, 0.253]
[0.413, -0.030]
```

```
[0.391, 0.093]
[0.391, 0.227]
[0.530, -0.041]
[0.534, 0.112]
[0.545, 0.244]
Basket: 1 basket at:
[0.3, -0.35]

========OUTPUT(You should generated it========

[
    {
        "task_category": "Sequential Spatial Reordering",
        "instruction": "Move the can in the middle of the first column from the
            left to the far right of the row it is in. Then move the cans in the
             upper left corner to the next row in the same column. Next, move
            the cans in the middle of the line closest to you to the upper left
            corner of the original grid. Now, put the can currently located in
            the middle row, second from left, into the basket."
    },
    {
        "task_category": "Ordinal →Projective",
        "instruction": "Put the cans in the third row from near to far and the
            second column from left to right into the basket."
    }
]

========INPUT========

Object placement: Tshaped
Objects: 5 box are located at:
[0.383, 0.248]
[0.517, 0.226]
[0.249, 0.216]
[0.406, 0.110]
[0.391, -0.033]
Basket: 1 basket at:
[0.3, -0.35]

========OUTPUT(You should generated it========

[
    {
        "task_category": "Sequential Spatial Reordering",
        "instruction": "Keep the three horizontal boxes on the same line, move
            the rightmost box to the leftmost, then move the box currently
            located at the intersection point to the rightmost, and finally,
            place the box currently located at the leftmost in the basket."
    },
    {
        "task_category": "Structural Relation",
        "instruction": "Move only one box and place all boxes in a symmetrical
            cross shape."
    },
    {
        "task_category": "Structural Relation",
        "instruction": "Arrange all the boxes in a straight line."
    },
    {
        "task_category": "Dynamic Reference Switching",
        "instruction": "Pick up the box to the right of the intersection and
            place it in the basket. Then put the box farthest from you into the
```

```
            basket, and finally put the box closest to the basket into the
            basket as well."
    },
    {
        "task_category": "Projective →Structural",
        "instruction": "Pick up the lowest box from the vertical column of the T-
            shape and place it on the left side of the horizontal row."
    },
    {
        "task_category": "Ordinal →Projective",
        "instruction": "Pick up the second box from the left and put it in the
            basket."
    }
]

========INPUT========

Object placement: Random
Objects: 6 can are located at:
[0.287, 0.059]
[0.339, 0.211]
[0.423, 0.004]
[0.456, 0.116]
[0.468, 0.249]
[0.546, 0.070]
Basket: 1 basket at:
[0.3, -0.35]

========OUTPUT(You should generated it========

[
    {
        "task_category": "Topological Relationship",
        "instruction": "Put the can that is closest to the basket into the
            basket."
    },
    {
        "task_category": "Structural Relation",
        "instruction": "Arrange all cans into a grid with 2 rows and 3 columns."
    },
    {
        "task_category": "Goal Replanning",
        "instruction": "Arrange all the cans into a grid of 2 rows and 3 columns
            ... No, there's no need to organize anymore. Put the leftmost cans
            in the basket."
    }
]

========INPUT========

Object placement: Ring
Objects: 6 bottle are located at:
[0.581, 0.114]
[0.499, 0.266]
[0.300, 0.249]
[0.202, 0.098]
[0.325, -0.039]
[0.473, -0.043]
Basket: 1 basket at:
[0.3, -0.35]

========OUTPUT(You should generated it)========
```

```
[
    {
        "task_category": "Structural Relation",
        "instruction": "Arrange all cans into a grid with 3 rows and 2 columns."
    },
    {
        "task_category": "Sequential Spatial Reordering",
        "instruction": "Take the leftmost can as the first one, count clockwise
            downwards, and place the second can in the basket. Then, following
            the same counting method, put the current fourth can in the basket."
    },
    {
        "task_category": "Topological →Projective",
        "instruction": "Pick up the can closest to the four o'clock direction
            with respect to the center of the ring and place it in the basket.
            Then put the can closest to the ten o'clock direction into the
            basket."
    }
]

Pay attention to the following points:

1. All objects have volume, and the robotic arm may not accurately determine
    their position. You can assume that only when the distance between two
    coordinates is not less than 0.02 can they be barely distinguished by the
    robotic arm, and it will be more pronounced at 0.05.
2. Do not include low-level information such as x-axis, y-axis, and specific
    coordinates in the description you generate. The robotic arm should not know
     anything other than visual information.
3. The robotic arm is located at (0,0), facing the positive x-axis direction,
    the negative y-axis direction is on the right, and the positive y-axis
    direction is on the left.
4. Bottles cannot be stacked.

5. The description of the location can be combined with the placement structure
    of the object (such as Tshape, Grid, etc.), using your imagination and
    various description methods.
6. DO NOT output any other content besides the JSON
7. The task you generated should require the robotic arm to move some objects to
     complete

Here is the information for all objects:
```

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes], [No], or [N/A].
- [N/A] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for [N/A]).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will also be asked to include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While [Yes] is generally preferable to [No], it is perfectly acceptable to answer [No] provided a proper justification is given (e.g., error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering [No] or [N/A] is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS Paper Checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the benchmark scope, task taxonomy, construction pipeline, and empirical findings, and the experimental sections report results that directly support these claims.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: The Limitations paragraph in Section 5 discusses the simulated tabletop setting, limited perturbation coverage, deployment-interface effects, and the absence of finetuning and real-robot validation.

Guidelines:

- The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A]

   Justification: The paper does not present formal theoretical results, theorems, or proofs; the mathematical notation is used only to describe the benchmark construction process.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

Justification: The paper provides the simulator setup, task suites, evaluation protocol, baseline deployment details, metrics, and implementation details needed to reproduce or verify the main experimental results.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The paper provides open access to the benchmark assets, code, configuration files, and reproduction instructions in the supplemental material.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.

- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper specifies the benchmark suites, simulator environment, robot setup, observation and action interfaces, evaluation protocol, baseline settings, and metric definitions in the main text and appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: The paper specifies the benchmark suites, simulator environment, robot setup, observation and action interfaces, evaluation protocol, baseline settings, and metric definitions in the main text and appendix.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
   - If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

Answer: [Yes]

Justification: The paper reports the simulator backend, model deployment settings, hardware resources, and approximate compute requirements needed to reproduce the reported evaluations.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

Answer: [Yes]

Justification: The research follows the NeurIPS Code of Ethics, uses controlled simulated environments, does not involve human subjects, and does not release assets with high-risk personal or sensitive content.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [N/A]

Justification: The work is a simulated benchmark for diagnostic evaluation of robot spatial understanding and does not directly deploy systems in real-world social, safety-critical, or user-facing settings.

Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.

- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer:[N/A]

    Justification: Existing simulators, robot assets, baseline models, and software packages are cited, and their licenses and terms of use are respected where applicable.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: Although the paper introduces new benchmark assets, the current submission does not yet provide a complete standardized asset documentation sheet covering all metadata, limitations, and release details.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [No]

    Justification: Although the paper introduces new benchmark assets, the current submission does not yet provide a complete standardized asset documentation sheet covering all metadata, limitations, and release details.

Guidelines:

- The answer [N/A] means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing, user studies, or experiments with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or human-subject research, so IRB approval or equivalent review is not applicable.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [Yes]

Justification: The paper describes the use of LLMs in the benchmark construction pipeline, including their role in task and structure proposal before deterministic grounding and simulation verification.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.