Text-Only Grid Spatial Understanding for Embodied Agents

Anonymous ACL submission

Abstract

Spatial reasoning abilities have become more important to recent tasks. However, we do not understand how LLMs reason about space or utilize their multi-modal inputs in such tasks. As a starting point, we introduce a dataset of text-only spatial reasoning problems on grids[1] to understand what abilities LLMs have without exposure to visual modalities and compare the performance of a variety of LLMs and VLMs on these tasks. We find that even text-only models have some implicit or mathematical understanding of grids and the 3D space they represent; however, even VLMs and foundation models fall short when asked to reason about space from the perspective of an embodied agent (i.e. with its own frame of spatial reference). We also find that models struggle more when instructing others and when they need to recognize real-world concepts within a grid.

1 Introduction

With the recent ubiquity of LLMs, their applications have widened in range from the original text generation to more varied tasks. Among these applications are those that require the ability to reason about space, such as Vision-and-Language Navigation (VLN), which requires models to understand verbal spatial relations relative to embodied agents to move through an environment (Zhang et al., 2024). Task-oriented multi-modal settings require interacting with objects in complex environments (Padmakumar et al., 2021), while new visual tasks go beyond simple identification and require spatial reasoning for applications such as scene descriptions (Fu et al., 2024a) and reasoning over 2D and 3D images (Cheng et al., 2024; Chen et al., 2024) for task completion and planning.

Many approaches to such tasks incorporate an additional modality, such as vision or a representation of point cloud data, but using these effectively often requires changes to model architecture (to accept a new modality) and/or additional training or fine-tuning. Furthermore, even with such additions, it is unclear if fundamentally non-physical models that have been trained primarily on unpaired text can truly understand 3D space. Despite recent advances on various spatial reasoning tasks, the internal reasoning processes of LLMs, VLMs, and variations thereof are often not interpretable, meaning that merely measuring performance is distinct from evaluating understanding. Even with strong performance, further investigations into how additional modalities, like vision, are used in spatial tasks reveal that multi-modal LLMs receive little additional information from images when provided with rich text representations (Wang et al., 2024a).

In that case, textual descriptions of physical space may serve as a convenient proxy for complex visual inputs when evaluating spatial reasoning. And grids are the natural choice for a textual representation of complex 3D layouts and movement. Thus, we present GSU, a dataset composed of 3 simple text-only grid spatial reasoning tasks geared to applications in embodied domains to explore LLMs inherent ability and potential in spatial reasoning.

- The **Navigation** Task requires instruction generation or following on 2D and 3D grid with two potential spatial frames of reference: *Cardinal* (i.e. fixed), testing recognition of 1D directional differences in coordiantes, and *Egocentric* (i.e. moving with an embodied agent), testing understanding of rotating/updating spatial reference frames.

- The **Object Localization** task asks where targets are relative to a viewer (*Egocentric*), testing 2D and 3D direction differences in coordinates, or relative to a separate spatial reference

[1] Our dataset and code will be freely available at URL.

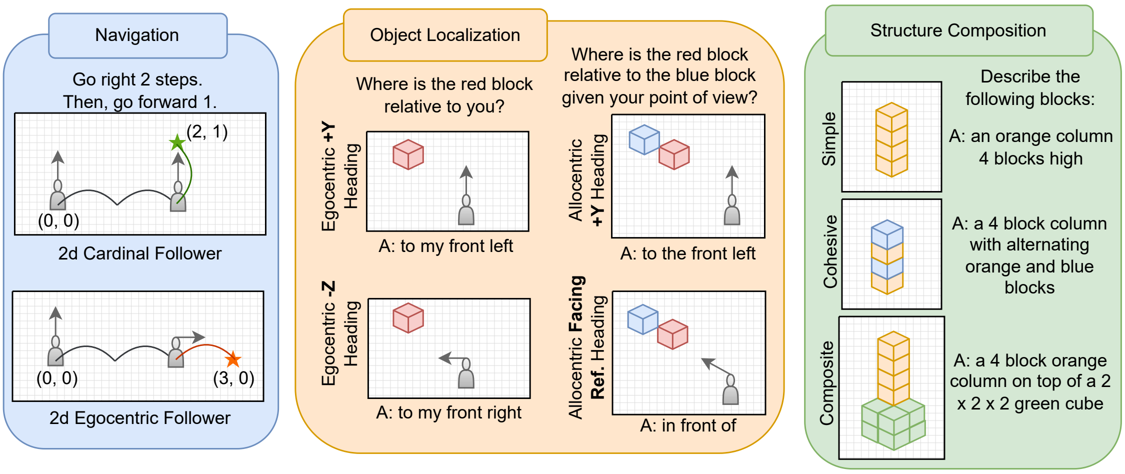

Figure 1: The GSU Dataset tasks and settings visualized for clarity (note that the models do NOT receive these visuals and instead receive a textual version of the environment shown in Appendix E). The grey arrows indicate the heading, i.e. direction that the embodied agent is facing. In the Navigation task, they remain aligned with +Y in the *Cardinal* setting and rotate to reflect the last direction of travel in the *Egocentric* setting, which affects the direction of the second step and the final coordinates. For the Object Localization task, we show how different headings affect the spatial relations between the target and the reference. For the Structure Composition task, we show the 3 structure categories that models may be asked to describe.

(*Allocentric*), testing spatial reference in collaborative settings.

- The **Structure Composition** task needs models to translate sets of coordinates into basic shapes (e.g. rows, columns, cubes), testing ability to link coordinates to real-world shapes and process large numbers of coordinates.

2 The GSU Dataset

We now expand on each of these relatively simple spatial tasks and the specific facets of textual coordinate reasoning they test. For more details on generation, please refer to Appendix A.

2.1 The Navigation Task

The first task measures models' understanding of textual coordinates as a representation of physical space. In this task, models must correlate natural language directions (i.e. left, down, backward) with grid traversals across coordinates differing in 1 dimension at a time, i.e. equating moving from (0, 0) to (2, 0), to (2, 1) with moving "right 2" and "forward 1".

2.1.1 Instructor vs. Follower

VLN tasks are generally a conversation between two counterparts: an instructor and a follower. While, the instructor is often a person and the follower an agent, this is not necessarily always the case (e.g. a collaborative system with multiple agents, Narayan-Chen et al. (2019)'s MDC Architect task, etc). So, models should be proficient in both directions of this mapping between verbal spatial directions and coordinate traversals.

In the *Follower* setting, models receive a set of natural language instructions explaining how to traverse the grid (e.g. "move right 2 steps and then forward 1") and must output a coordinate reflecting their final location (e.g. (2, 1)). Whereas in the *Instructor* setting, models receive a grid traversal expressed as a series of intermediate coordinates (e.g. (0, 0), (2, 0), (2, 1)) and must output the natural language instructions that correlate to that path (e.g. right 2, forward 1).

2.1.2 Spatial Frames of Reference

We consider two separate frames of reference for the spatial directions of the grid. In the *Cardinal* setting, the spatial directions are fixed to the axes of the grid and never change, i.e. to move forward always means to increase the y value, while to move left always means to decrease the x value. While many grid based tasks use this fixed perspective of spatial relations (e.g. Aghzal et al. (2025)), it does not transfer well to practical collaborative settings where an embodied agent or a human it is instructing have their own frame of reference that may not align with a fixed grid.

The *Egocentric* formulation aligns spatial directions with the perspective of an embodied subject navigating the grid, keeping them constant relative to their body, thus changing their meaning relative to the fixed axes of the grid over the course of the task. For example, in the *Egocentric* setting to go right 2 steps and then forward 1 step would first entail turning to the right (90° clockwise to face $+X$) and moving two steps in that direction. Then, moving 1 step forward would no longer mean increasing the y value, but rather continuing forward in our current heading/direction of travel, i.e. increasing our x value. This would be equivalent to moving 2 steps right and then 1 more step right in the *Cardinal* setting. (See Figure 1 for a visualization.)

Additionally, we include a simpler form of the *Egocentric Instructor* setting, *Card2Ego*. Rather than going from coordinates to egocentric spatial directions, *Card2Ego* asks models to turn cardinal spatial directions (North/South/East/West) into egocentric spatial directions (left/right/forward/backward).

2.2 The Object Localization Task

The object localization task focuses on understanding how grid coordinates are situated spatially relative to one another. Specifically, given a viewer at the origin we ask models to determine where a target at a randomly generated coordinate is relative to that viewer (*Egocentric*) or where it is relative to a reference at another randomly generated coordinate (*Allocentric*). In the former, models are expected to reply with something to the effect of the object is "to *my* right" and in the latter with the target is "to *the* right of" the reference.

In this task, we vary the heading of the viewer (i.e. which axis they are facing parallel to), the distance between the target and the reference (are they adjacent to each other or further away), and add distractors (see A for more).

2.3 The Structure Composition Task

The final task explores a model's ability to understand how blocks/voxels can be composed into larger overall structures. Not only does it require an understanding of how coordinates relate to real space, but also a knowledge of basic shapes/forms (e.g. a series of blocks that differ only in their z-dimension are stacked vertically and thus a column).

2.3.1 Composition Types

In the most basic setting, *Simple*, the selected coordinates form one structure of a single color and the model must interpret what 3D shape their arrangement forms (e.g. a row or a cube). Slightly more complicated is the *Cohesive* setting where multiple colors of blocks form a single structure and the challenge is to understand how those colors interact with the structure (e.g. compare "a row with alternating orange and blue blocks" and "a row with an orange left half and a blue right half"). Finally, the *Composite* setting places 3 single-colored structures adjacent to each other, asking the model to identify the 3 shapes, their colors, and where they are relative to each other.

3 Experiments

To explore grid spatial understanding over a variety of model families, sizes, and types, we tested 5 broad categories of models. First, we tested small text-only models: Mistral-7b (Jiang et al. (2023)'s Mistral-7b-Instruct-v0.3), Llama-8b (Grattafiori et al. (2024)'s Llama3.1-8b-Instruct), and Qwen-7b (Yang et al. (2024); Team (2024)'s Qwen2.5-7b-Instruct). Additionally, we prompted their most similarly sized Vision-Language counterparts to see if exposure to a visual modality (even if it is not being utilized) can induce further spatial understanding: Pixtral-12b (Agrawal et al., 2024), LlamaVL-11b (Grattafiori et al. (2024)'s Llama3.2-11b-Vision-Instruct), and QwenVL-7b (Team (2025a); Bai et al. (2023); Wang et al. (2024b)'s Qwen2.5-VL-7B-Instruct). We also evaluated larger models, as they are known to be more powerful than their smaller counterparts: GPT-OSS20b (OpenAI, 2025), Qwen32b and QwenVL-32b ((Team, 2025b)'s 32b Instruct variants), Olmo-32b (Olmo et al., 2025), Qwen-72b (Yang et al. (2024); Team (2024)'s Qwen2.5-72B-Instruct). Finally, to consider a frontier model, we choose GPT-4o (OpenAI et al., 2024), and to compare overall performance to a much smaller model (<1B params) that can be fully fine-tuned, we fully train FlanT5-large (Chung et al., 2022).

For the purposes of prompting we generate test sets for each task of size 100 and for training we generate another of size 3000 (we did not see marked improvement in training by increasing the dataset size beyond this, see Appendix B for more details). The experiments in 1 and the Object Localization ones in Table 2 are 1-shot prompting with a handwritten example that includes the full reasoning needed to generate an answer. For the Structure Composition task in 2 the experiments are 3-shot prompting with synthetic examples and no reasoning. These setting were chosen based on performance (see Tables 5 and 6 in the Appendix), as the first two tasks rely more on reasoning traces and benefit from one dense examples and the final task requires no reasoning and thus benefits from a larger quantity of examples.

3.1 Metrics

3.1.1 Navigation

For the *Follower* task, we evaluate the accuracy of the predicted final coordinates as an exact match and also compute their distance from the true final coordinates. For the *Instructor* task, we measure the exact match accuracy of the instruction chain. Additionally, we calculate the distance between the true final coordinates (which are the last coordinates given in the *Instructor* prompt) and the final coordinates obtained by executing the predicted instruction chain starting from the beginning coordinates.

3.1.2 Object Localization

To measure model performance on the Object Localization tasks, we compute the spatial overlap, O_S, of the relations in the predicted text, p, and the ground truth text, g.

$$O_s = \frac{E_r(p) \cap E_r(g)}{E_r(p) \cup E_r(g)} * 100$$

where E_r extracts all spatial relations from the text and represents them in a simplified set (i.e. all synonyms map to one relation name, etc). To understand the scale of the overlaps metrics, a 0 indicates no overlapping terms while a 100 is perfect match. If the correct spatial relations are "front" and "left" and the model predicts only one of them, it receives a 50. If it gets one correct and generates an extra, incorrect relation, it gets a 33.33, while if it gets both correct and still adds an extra, it gets a 66.67.

In our prompts (see Appendix E) we specify that all spatial relations should be given as *target* is ⟨*spatial relations*⟩ of *reference* and we found that nearly all generations followed this prescription. Thus, we found no need to account for the order of objects (i.e. in the *Egocentric* setting we assume all generations are relative to the agent, "the target is to the right of me", and ignore the possibility of "I am to the left of the target").

3.1.3 Structure Composition

We continue to use the spatial overlap metric, O_s, described in 3.1.2 and add three additional overlap metrics: O_c, which measures the overlap of the color terms, O_n, which measure numerical overlap, and O_f, which measures the overlap of the structure formation terms. For O_n, we use the same simple overlap as O_s. However, unlike the Object Localization task, we have the opportunity for multiple colors of the same type in the *Composite* setting, so the formulation for O_c is slightly different:

$$O_c = \frac{\sum_{colors} \min(C_c(p, color), C_c(g, color))}{\sum_{colors} \max(C_c(p, color), C_c(g, color))} * 100$$

where $C_c(text, term)$ counts the occurrence of $term$ or its synonyms in the text $text$. Thus, a generation that only generates one "yellow" in a structure with 2 yellow sub-structures is penalized, as is a model that predicts 2 yellow substructures in a structure with just 1.

O_f follows the same base logic as O_c, but with more varied synonyms checked in C_f and additional partial credit afforded for predicting similar but not correct shapes. E.g. rather than getting no credit for predicting "a blue row on the ground" when the structure is a blue column, we award partial credit based on how similar the shapes are. (see Appendix A.1)

4 Results and Discussion

4.1 Navigation

As expected, for the *Navigation Follower* task, we saw much stronger performance in the more predictable *Cardinal* setting (see Table 9 in the Appendix), where even the smaller Vision-Counterpart models were consistently solving the 3D task (receiving scores of 0.67, 0.60, and 0.90, respectively), and were extremely proficient on the 2D task (with the LLama and Qwen models getting every example correct). With all of its rotations relative to an embodied traveller, the *Egocentric* formulation proved to be more difficult, posing a challenge even for GPT-4o and our largest Qwen model. While the 3D *Cardinal* was more challenging than its 2D counterpart, we suspect that the higher scores in the 3D *Egocentric* setting are due to up/down movements not requiring turns by definition, which effectively shorted the number of

		Follower				Instructor					
		Egocentric 2D		Egocentric 3D		Cardinal		Egocentric		Card2Ego	
		acc ↑	dist. ↓	acc ↑	dist. ↓	acc ↑	dist. ↓	acc ↑	dist. ↓	acc ↑	dist. ↓
Small	Mistral-7b	0.30	6.85	0.38	6.14	0.59	2.84	0.15	11.87	0.27	11.23
	Llama-8b	0.33	7.03	0.47	4.85	0.69	1.95	0.06	11.87	0.09	12.30
	Qwen-7b	0.09	22.54	0.19	20.44	0.76	1.57	0.21	8.96	0.36	7.52
Vision Counter-parts	Pixtral	0.33	6.31	0.50	5.60	0.87	0.75	0.26	7.65	0.35	9.01
	LlamaVL-11b	0.32	7.36	0.47	5.78	0.74	1.52	0.14	11.96	0.28	9.59
	QwenVL-7b	0.29	7.9	0.21	12.91	0.44	4.54	0.21	7.46	0.30	7.74
Medium	GPT-oss-20b	0.15	17.67	0.16	13.95	0.26	5.76	0.10	11.35	0.58	4.95
	Olmo-32b	0.34	7.22	0.40	5.94	0.48	4.59	0.14	8.20	0.62	4.24
	Qwen-32b	0.37	6.37	0.32	12.71	0.58	3.14	0.21	8.80	0.60	4.08
	QwenVL-32b	0.38	6.41	0.52	5.16	0.12	6.58	0.17	14.00	0.64	4.44
	Qwen-72b	0.49	5.76	0.75	3.66	**1.0**	**0.0**	0.47	5.67	0.53	5.23
	GPT-4o	0.71	**2.94**	**0.84**	**1.40**	0.89	0.56	0.67	4.04	0.93	0.77
Full FT	FlanT5	**0.80**	13.80	0.82	13.14	0.99	0.03	**0.95**	**0.37**	**1.0**	**0.0**

Table 1: Results of 1-shot prompting on the Navigation task, reported as accuracies out of 1.0 and distances from the correct navigation endpoint. The best results for each experiment are **bolded** while the best within eahc model grouping are underlined.

		Object Localization		Structure Composition			
		Egocentric	Allocentric	O_s	O_c	O_f	O_n
Small	Mistral-7b	30.32	46.50	49.75	72.60	30.21	20.01
	Llama-8b	46.17	50.18	62.04	86.69	22.70	37.75
	Qwen-7b	52.81	54.78	54.58	85.43	28.57	42.97
Vision Counterparts	Pixtral	40.83	48.18	66.25	87.68	42.16	55.43
	LlamaVL-11b	48.40	48.08	64.48	86.58	44.44	44.05
	QwenVL-7b	48.67	44.95	54.58	85.43	33.46	43.57
Medium	GPT-oss-20b	59.90	45.46	62.88	57.11	43.60	48.47
	Olmo-32b	85.98	54.17	65.33	91.97	48.60	54.60
	Qwen-32b	33.95	27.25	69.10	87.98	42.76	66.65
	QwenVL-32b	68.03	50.80	67.40	**93.62**	55.09	68.22
	Qwen-72b	79.53	53.83	70.67	88.11	42.81	69.23
	GPT-4o	**96.00**	65.50	69.20	88.63	52.85	**82.28**
Full FT	FlanT5	73.03	**100**	**93.17**	70.93	**81.35**	68.08

Table 2: Left: 1-shot Prompting on the Object Localization Task, measured in spatial term overlap out of 100. Right: 3-shot Prompting on the Structure Composition Task, measured with O_s, spatial overlap, O_c, color overlap, O_f, shape/formation overlap, and O_n, numerical overlap metrics all out of 100. The blocks for the Structure Composition Task are represented as textual lists, for more info on other representations and comparisons see Appendix A The best results for each experiment are **bolded** while the best within each model grouping are underlined.

rotations and led to fewer opportunities to introduce errors.

Looking at the "reasoning" traces generated by the models, we see that compared to the neat, contained logic with the occasional mistake seen in the *Cardinal* setting, models struggle to keep the axes consistent in the *Egocentric* setting. At the start, they seem to understand the standard Cartesian layout (when facing positive y, positive x is to your right), but as they progress through the instruction sequences they fail to keep this consistent with the actions they take. This leads to generations such as *"we are facing negative y, with positive y to our right"* or *"we turned towards positive x, so we're facing negative x"*. Sometimes, these incorrect justifications are followed by correct steps though, as in: *"we are facing negative x with positive x to the right, so that means that turning right means going towards positive x* ... [rest of the step] ... *now, we are facing positive y"* [$+Y$ being the correct axes to the right of $-X$, and thus the correct heading upon finishing the step].

As the models get more powerful, they make these mistakes less often but instead make the correct turns and then forget to adjust their heading in the logic, leading to two steps in the same direction.

Other times they ignore the rotational components and proceed according to a fixed grid.

For the *Instructor* task, we see that performance is weaker than its *Follower* counterpart, but some models are still able to produce strong instructions sequences that often get quite close to the desired endpoint, with Qwen-72b even solving the task completely. But the *Egocentric* setting again breaks what "understanding" these models possess, leading to performances well below a simple model of 875M parameters fully fine-tuned on the problem. Even Qwen-72b and GPT-4o, which show some promise on shorter sequences, get turned around in longer sequences.

To better understand the source of the errors in the *Egocentric* setting, we proposed the *Card2Ego* setting, which just required understanding how cardinal directions would change relative to an embodied traveller, removing the necessity of reasoning about specific coordinates and where they fall relative to each other. We found that models performed better in this setting, with medium sized models seeing notable improvements, but were still far from completely solving the task. This improvement suggests that while reasoning about the coordinates is a difficult aspect of the *Egocentric Instructor* setting for LLMs, models also struggle with the concept of rotating the spatial frame of reference itself.

4.2 Object Localization

For the *Egocentric* Object Localization task, we tested the models on the hardest setting (adjacent target blocks with additional distractors) and in various headings in a 3D grid and saw the expected pattern of improvement from larger models. There was, however, no clear correlation between having exposure to a visual modality and improved reasoning over textual coordinate grids in the smaller models. The significant improvement in the Qwen-32b models once vision was added may suggest that VLMs cannot make use of additional spatial information from exposure to visual modalities until the base model reaches a certain size.

We further looked into accuracy based on the heading (see Table 4 in the Appendix for more), and found that models mostly performed best in the $+Y$ heading that we would consider the "standard" for the Cartesian grid, with the Qwen family and our fine-tuned model showing the strongest preference (perhaps due to exposure to a similar environment or task in pre-training). For the other 3 headings there was no clear preference between all models, with some preferring the $-Y$ heading as second best and others the headings parallel to the X-axis. We see the effects of this axis preference compounded in the *Egocentric* Navigation task, where models struggle to identify spatial directions in non-standard headings.

Especially in the smaller models, we saw a lack of understanding of how axes were situated relative to each other (similar to the Navigation task). We also observed confusion over whether the y or z axis was vertical (despite it being explicitly stated) leading to productions such as, *"the block is below and above me"*. There were common incorrect conclusions such as, *"I am facing negative y so smaller values are in front of me.* [correct] *My y is -4 and the block's is 7, so it is **above** me"* or *"I know that negative x is to my right* [correct] *so smaller x values smaller than me are to my left"*. In general, these mistaken deductions were aligned with a standard $+Y$ heading rather than the correct premise preceding them, showing that despite any intermediate generations to contrary, the models were still operating in that $+Y$ heading.

For the *Allocentric* Object Localization task, we evaluated only on the $+Y$ heading so the slight performance drops compared to the *Egocentric* formulation are actually much more pronounced considering that models perform best on this heading. Most models performed worse on the *Allocentric* variant, suggesting that models struggled to reason about how two grid coordinates were situated relative to each other from the agent perspective (a third coordinate). Those that performed better were likely benefiting from the lack of distractors in the *Allocentric* setting rather than displaying a particular strength at the task itself.

We generally saw similar errors on this task, but we did note that with an allocentric spatial reference models confused the forward and backward directions, often flipping them. That is, if the target was between the viewer and the reference they would describe it as behind the reference rather than in front of it as in Figure 2.

For the Object Localization task, we also explored LORA fine-tuning as an option to improve model performance. (The other tasks did not see improvements from LORA, likely due to the reasoning nature requiring an initial generation before the answer, which is not the most compatible with a supervised training process.) The LORA fine-tuned LLama-8b and Mistral-7b improved performance

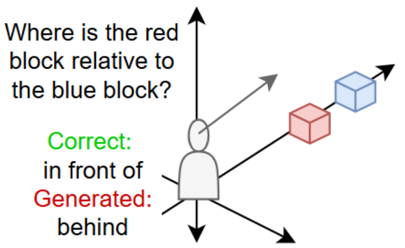

Figure 2: *Allocentric* front/back flipping

significantly–outperforming or matching the Qwen-72b model. (See Table 7 for more) This might indicate that while LLMs are not the strongest textual grid spatial reasoners, they have all the potential to be with some fine-tuning.

4.3 Structure Composition

For the Structure Composition task, we saw spatial results similar to the *Allocentric* ones, with just slight improvements in some models, likely due to the perspective difference in the task definition: in the *Allocentric* setup we treat the reference as having its own spatial reference (i.e. it has a left and right of its own, mirroring those of the viewer), whereas in the Structure Composition setup we defer to a viewer facing along the $+Y$ axis for spatial terms to preserve the simplicity of the task. This simpler spatial referencing is offset by the higher number of blocks and the need to group and then orient them correctly. For the color metric, models predictably perform well, with most mistakes due to forgetting a color rather than hallucinating one. As for numbers, model generations are very often wrong, especially with the larger structures, or the models break up the structure differently, leading to different (and usually incorrect) dimension breakdowns. For example, some models may break a tower 8 blocks high and with a 3 x 3 base into 10 columns each 7 blocks high.

Naturally, the O_f subtask was the most difficult since it requires models to match groupings of coordinates to real-life shapes that exist outside of grid and coordinate settings, making the fully fine-tuned model's performance all the more impressive. The shapes generated by the LLMs seem sensible and comprehensible when viewed independently of the ground truth, but often fail to match it. In the 0-shot setting, models tended to output vague descriptions rather than concrete shapes, like opting to describe a 4 x 7 rectangle as "a continuous layer of blocks". This carried over to the 3-shot setting to an extent, with productions like "structure", "formation", or "platform" as base words and rich adjectives to modify them (e.g. a "large, flat, and expansive structure"). They also had a tendency to flatten 3D shapes, describing cubes as squares or rectangular prisms as rectangles. Reflective of this, we used generous synonyms in the shape metric (see Appendix A.1).

On occasion, the LLMs would name structures incorrectly while still producing the correct dimensions: e.g. calling a 8 x 2 x 2 tower a "cube with height 8 and width 2", which is fundamentally not a cube, but clearly shows some understanding of the formed structure. While such productions are not rewarded by the shape metric, they are reflected in the numerical one. They are also much less common than the LLMs just guessing an incorrect shape. As such, an O_f score around 50, as the best models have, reflects a mastery of simple shapes like columns and rows in their individual and *Composite* settings and some limited success with more complex shapes such as towers, rectangular prisms, and cubes.

The Structure Composition shape subtask was also the only subtask where we saw a clear and consistent improvement gained by models have some exposure to a visual modality (each of the vision counterparts outperformed their text only base model).

5 Related Works

Many works have explored how best to represent the real world to LLMs. The defaults are external camera image inputs (for general scenes not from the POV of an embodied agent) and egocentric image inputs (from the perspective of a first person viewer) to a VLM, while others consider presenting the visual information from above instead (Li et al., 2024a), but most still find that models lag behind human performance. Also common is forgoing the visual modality altogether and representing visual information with dense pointcloud embeddings that models can be trained to interpret to some degree of success (Hong et al., 2023).

Out of these applications and input formats have come increased efforts to understand VLM spatial reasoning. (Liao et al., 2024; Liu et al., 2023) find that VLMs do not perform as well on spatial VQA tasks as they do on more standard image tasks, struggling particularly with object orientations. OpenEQA (Majumdar et al., 2024) evaluates foundation model performance on episodic VQA, including object localization within scenes and spatial reasoning about object dimensions, and find that they fall well short of human performance. This has led to works like Tang et al. (2025), which proposed pairing 2D image and text tasks (answering questions on the orientation, distance, and location of nodes) as a training dataset to improve spatial reasoning in images to some effect. Wang et al. (2024a)'s SpatialEval, which compares LLM and VLM performance on 2D image and image grid tasks, found that VLMs perform just as well with rich text descriptions instead of image inputs, while Fu et al. (2024b) find the same for VQA.

This lack of understanding into how LLMs reason about space in multiple modalities leads us to text-only spatial reasoning tasks, like Momennejad et al. (2023), who consider textually represented graph traversals and find that LLMs often hallucinate nodes, find suboptimal paths, and fail on denser graphs. However, many text-only spatial reasoning setups describe locations relatively (e.g. A is to the left of B which is below C) instead of using grids to organize space (Li et al., 2024b; Rizvi et al., 2024). Those that do use grids avoid coordinate representations: like Yamada et al. (2024), who explore 2D grid navigation in terms of simple closed loop paths and relative/textual descriptions rather than coordinates, finding that LLMs do not match human performance, or Mayer et al. (2025), who propose a visual puzzle game on a grid and compare visual and chess-style textual formulations. In fact, we found only one work that utilized text-only grid coordinates: Aghzal et al. (2025) explore a grid navigation setup (like our 2d *Cardinal* task), evaluating how LLMs plan paths through targets whilst avoiding obstacles. Given that coordinate grids are the default for representing space and likely plentiful in LLMs pretraining data, they are a clear fit for spatial reasoning tasks.

GSU distinguishes itself from prior tasks by embracing grid coordinates in a fully-textual representation as an efficient way to represent 3D space, following recent advances in solving geometry, challenging math, and complex reasoning problems (Shao et al., 2024; Ahn et al., 2024). Additionally, it incorporates embodied agents (either the model or another collaborator) and their spatial frames of reference into the task, differing from most textual navigation and spatial relation tasks (which assume externally fixed spatial relations irrespective of the rotations/movements of an embodied agent). These alone make GSU unique, but we further propose describing grouping of coordinates as real world shapes and structures as an exploration of both spatial reasoning and mapping coordinate knowledge to other knowledge of the real world.

This has applications in simulated building tasks, such as Narayan-Chen et al. (2019)'s MDC, in which a Builder and Architect collaborate to build a blocks structure on a 3D voxel grid. In fact, each of our tasks can be applied to a different aspect with which models struggle in the MDC. Our *Egocentric Navigation* setups deal with shifting/rotating spatial reference frames that occur as Builders move in their environment, while the distinction between the *Follower* and *Instructor* settings highlights the performance difference between the two roles in the same task. Our Object Localization task addresses the issue of verbally expressing where new blocks must be placed relative to existing ones or the Builder, or helping disambiguate multiple blocks of the same color. Finally, the Structure Description task aligns most closely with the MDC, exploring the potential of LLMs to act as effective Architects by seeing if they can adequately describe block structures. Recent work in the MDC (Jayannavar et al., 2025) has shown improvements on the inverse Builder task: mapping spatial references to correct block coordinates, but progress on the Architect remains largely unexplored.

6 Conclusion

In this paper, we present GSU, a novel reasoning task offering the opportunity to explore models' inherent spatial understanding and supporting extensions such as simulated 3D tasks or interpretability, whereby models could use similar formulations to express their own underlying representations of 3D space and the objects in it. When evaluating a range of models on it, we see that LLMs have some ability to understand 3D space presented as a textual grid, especially in common formats they may have exposure to. They can navigate fixed grids, explain where coordinates are relative to each other, and identify simple shapes from coordinates. However, despite the simplicity of the tasks (a 875M parameter models was able to perform well on all of them), LLMs struggle when tasks are made more complex: serving as the navigation instructor, accounting for shifting spatial reference frames as agents move, and describing more complex shapes and groupings of them all elude current models.

Limitations

We acknowledge that these are simple tasks, and though they are designed as such to best analyse specific aspects of spatial understanding, we think more complex text-based tasks could be an interesting direction of further exploration. Especially when paired with evaluation beyond prompting and simple LORA fine-tuning. In particular, we would be curious to see if RL-style approaches applies to reasoning and math problems could yield stronger Navigation models.

Further, we understand that some of our metrics are quite simple. While they are generally suitable to the complexity of the tasks, we think a stronger metric than shape overlap, O_f, could provide a fuller picture of model abilities by catching edge cases.

Acknowledgments

All models were used within their licensing agreements described on Huggingface. We thank the Mistral team for access to Mistral-7b and Pixtral, the LLama team for access to Llama-8b and Llama-11b-VL, the Qwen teams for access to Qwen-7b, Qwen-7b-VL, and Qwen-72b, AI2 for access to Olmo-32b, and OpenAI for access to GPT-4o and GPT-OSS-20b.

References

Mohamed Aghzal, Erion Plaku, and Ziyu Yao. 2025. Can large language models be good path planners? a benchmark and investigation on spatial-temporal reasoning. *Preprint*, arXiv:2310.03249.

Pravesh Agrawal, Szymon Antoniak, Emma Bou Hanna, Baptiste Bout, Devendra Chaplot, Jessica Chudnovsky, Diogo Costa, Baudouin De Monicault, Saurabh Garg, Theophile Gervet, Soham Ghosh, Amélie Héliou, Paul Jacob, Albert Q. Jiang, Kartik Khandelwal, Timothée Lacroix, Guillaume Lample, Diego Las Casas, Thibaut Lavril, and 23 others. 2024. Pixtral 12b. *Preprint*, arXiv:2410.07073.

Janice Ahn, Rishu Verma, Renze Lou, Di Liu, Rui Zhang, and Wenpeng Yin. 2024. Large language models for mathematical reasoning: Progresses and challenges. *Preprint*, arXiv:2402.00157.

Jinze Bai, Shuai Bai, Shusheng Yang, Shijie Wang, Sinan Tan, Peng Wang, Junyang Lin, Chang Zhou, and Jingren Zhou. 2023. Qwen-vl: A versatile vision-language model for understanding, localization, text reading, and beyond. *arXiv preprint arXiv:2308.12966*.

Boyuan Chen, Zhuo Xu, Sean Kirmani, Brian Ichter, Danny Driess, Pete Florence, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. 2024. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities. *Preprint*, arXiv:2401.12168.

An-Chieh Cheng, Hongxu Yin, Yang Fu, Qiushan Guo, Ruihan Yang, Jan Kautz, Xiaolong Wang, and Sifei Liu. 2024. Spatialrgpt: Grounded spatial reasoning in vision language models. *Preprint*, arXiv:2406.01584.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Eric Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Sharan Narang, Gaurav Mishra, Adams Yu, and 12 others. 2022. Scaling instruction-finetuned language models. *arXiv preprint*.

Rao Fu, Jingyu Liu, Xilun Chen, Yixin Nie, and Wenhan Xiong. 2024a. Scene-llm: Extending language model for 3d visual understanding and reasoning. *Preprint*, arXiv:2403.11401.

Xingyu Fu, Yushi Hu, Bangzheng Li, Yu Feng, Haoyu Wang, Xudong Lin, Dan Roth, Noah A. Smith, Wei-Chiu Ma, and Ranjay Krishna. 2024b. Blink: Multimodal large language models can see but not perceive. *Preprint*, arXiv:2404.12390.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Yining Hong, Haoyu Zhen, Peihao Chen, Shuhong Zheng, Yilun Du, Zhenfang Chen, and Chuang Gan. 2023. 3d-llm: Injecting the 3d world into large language models. *Preprint*, arXiv:2307.12981.

Prashant Jayannavar, Liliang Ren, Marisa Hudspeth, Risham Sidhu, Charlotte Lambert, Ariel Cordes, Elizabeth Kaplan, Anjali Narayan-Chen, and Julia Hockenmaier. 2025. Bap v2: An enhanced task framework for instruction following in minecraft dialogues. *Preprint*, arXiv:2501.10836.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. *Preprint*, arXiv:2310.06825.

Chengzu Li, Caiqi Zhang, Han Zhou, Nigel Collier, Anna Korhonen, and Ivan Vulić. 2024a. Topviewrs: Vision-language models as top-view spatial reasoners. *Preprint*, arXiv:2406.02537.

Fangjun Li, David C. Hogg, and Anthony G. Cohn. 2024b. Advancing spatial reasoning in large language models: An in-depth evaluation and enhancement using the stepgame benchmark. *Preprint*, arXiv:2401.03991.

Yuan-Hong Liao, Rafid Mahmood, Sanja Fidler, and David Acuna. 2024. Reasoning paths with reference objects elicit quantitative spatial reasoning in large vision-language models. *Preprint*, arXiv:2409.09788.

Fangyu Liu, Guy Emerson, and Nigel Collier. 2023. Visual spatial reasoning. *Preprint*, arXiv:2205.00363.

Arjun Majumdar, Anurag Ajay, Xiaohan Zhang, Pranav Putta, Sriram Yenamandra, Mikael Henaff, Sneha Silwal, Paul Mcvay, Oleksandr Maksymets, Sergio Arnaud, Karmesh Yadav, Qiyang Li, Ben Newman, Mohit Sharma, Vincent Berges, Shiqi Zhang, Pulkit Agrawal, Yonatan Bisk, Dhruv Batra, and 5 others. 2024. Openeqa: Embodied question answering in the era of foundation models. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16488–16498.

Julius Mayer, Mohamad Ballout, Serwan Jassim, Farbod Nosrat Nezami, and Elia Bruni. 2025. ivispar – an interactive visual-spatial reasoning benchmark for vlms. *Preprint*, arXiv:2502.03214.

Ida Momennejad, Hosein Hasanbeig, Felipe Vieira, Hiteshi Sharma, Robert Osazuwa Ness, Nebojsa Jojic, Hamid Palangi, and Jonathan Larson. 2023. Evaluating cognitive maps and planning in large language models with cogeval. *Preprint*, arXiv:2309.15129.

Anjali Narayan-Chen, Prashant Jayannavar, and Julia Hockenmaier. 2019. Collaborative dialogue in Minecraft. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5405–5415, Florence, Italy. Association for Computational Linguistics.

Team Olmo, :, Allyson Ettinger, Amanda Bertsch, Bailey Kuehl, David Graham, David Heineman, Dirk Groeneveld, Faeze Brahman, Finbarr Timbers, Hamish Ivison, Jacob Morrison, Jake Poznanski, Kyle Lo, Luca Soldaini, Matt Jordan, Mayee Chen, Michael Noukhovitch, Nathan Lambert, and 50 others. 2025. Olmo 3. *Preprint*, arXiv:2512.13961.

OpenAI. 2025. gpt-oss-120b gpt-oss-20b model card. *Preprint*, arXiv:2508.10925.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Aishwarya Padmakumar, Jesse Thomason, Ayush Shrivastava, Patrick Lange, Anjali Narayan-Chen, Spandana Gella, Robinson Piramuthu, Gokhan Tur, and Dilek Hakkani-Tur. 2021. Teach: Task-driven embodied agents that chat. *Preprint*, arXiv:2110.00534.

Md Imbesat Rizvi, Xiaodan Zhu, and Iryna Gurevych. 2024. SpaRC and SpaRP: Spatial reasoning characterization and path generation for understanding spatial reasoning capability of large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4750–4767, Bangkok, Thailand. Association for Computational Linguistics.

Zhihong Shao, Peiyi Wang, Qihao Zhu, Runxin Xu, Junxiao Song, Xiao Bi, Haowei Zhang, Mingchuan Zhang, Y. K. Li, Y. Wu, and Daya Guo. 2024. Deepseekmath: Pushing the limits of mathematical reasoning in open language models. *Preprint*, arXiv:2402.03300.

Yihong Tang, Ao Qu, Zhaokai Wang, Dingyi Zhuang, Zhaofeng Wu, Wei Ma, Shenhao Wang, Yunhan Zheng, Zhan Zhao, and Jinhua Zhao. 2025. Sparkle: Mastering basic spatial capabilities in vision language models elicits generalization to spatial reasoning. *Preprint*, arXiv:2410.16162.

Qwen Team. 2024. Qwen2.5: A party of foundation models.

Qwen Team. 2025a. Qwen2.5-vl.

Qwen Team. 2025b. Qwen3 technical report. *Preprint*, arXiv:2505.09388.

Jiayu Wang, Yifei Ming, Zhenmei Shi, Vibhav Vineet, Xin Wang, Yixuan Li, and Neel Joshi. 2024a. Is a picture worth a thousand words? delving into spatial reasoning for vision language models. *Preprint*, arXiv:2406.14852.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024b. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Yutaro Yamada, Yihan Bao, Andrew K. Lampinen, Jungo Kasai, and Ilker Yildirim. 2024. Evaluating spatial understanding of large language models. *Preprint*, arXiv:2310.14540.

An Yang, Baosong Yang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Zhou, Chengpeng Li, Chengyuan Li, Dayiheng Liu, Fei Huang, Guanting Dong, Haoran Wei, Huan Lin, Jialong Tang, Jialin Wang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Ma, and 40 others. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Yue Zhang, Ziqiao Ma, Jialu Li, Yanyuan Qiao, Zun Wang, Joyce Chai, Qi Wu, Mohit Bansal, and Parisa Kordjamshidi. 2024. Vision-and-language navigation today and tomorrow: A survey in the era of foundation models. *Preprint*, arXiv:2407.07035.

A Dataset Generation

A.0.1 Navigation

To generate the Navigation task, we simulate a grid traversal by randomly generating steps lengths (e.g. how many spaces will be traveled) and randomly choosing a direction from the available options (i.e. in 2d we pick from left/right/forward/back and in 3d we also consider up/down) and pair them together. Then a function determines the final coordinates reached given the list of $(direction, length)$ pairs and the mode (e.g. *Egocentric* or *Cardinal*). When evaluating the *Follower* role, we provide the $(direction, length)$ pairs in natural language and expect a final coordinate as in Figure 1. In the *Instructor* role, we provide intermediate coordinates along the path as shown in Appendix C. As the base dataset construction is simply $(direction, length)$ pairs, the generated datasets are as similar as possible to compare the effects of various settings. That is, the *Instructor* and *Follower* tasks explore the exact same grid traversals but with switched inputs and outputs.

A.0.2 Object Localization

For the *Egocentric* setting, we randomly sample 2 locations within the grid, which extends $(-20, 20)$ along all axes, designating one the target and one the reference/agent, and a heading along one of the horizontal axes ($+X$, $+Y$, $-X$, $-Y$. For the adjacent case, we also sample one of the coordinates adjacent to the agent location (differing in each dimension by at most 1). In the *Allocentric* setting, we assume that the agent is at the origin and use the two sampled locations as the target and reference blocks. We set the heading either along the $+Y$ axis or towards the reference block.

For both settings, we select distractor blocks that do not overlap with the target block and programmatically derive the spatial relations.

Heading: In the Object Localization task, we can also vary the direction the embodied agent is facing. For the *Egocentric* setting, facing down a different axis greatly affects the spatial relations (e.g. a block that is in front of an agent facing the $+Y$ direction is to the right of an agent facing $-X$ as in 1). For the *Allocentric* setting, the agent may look in a fixed direction down the $+Y$ axis or directly at the reference object.

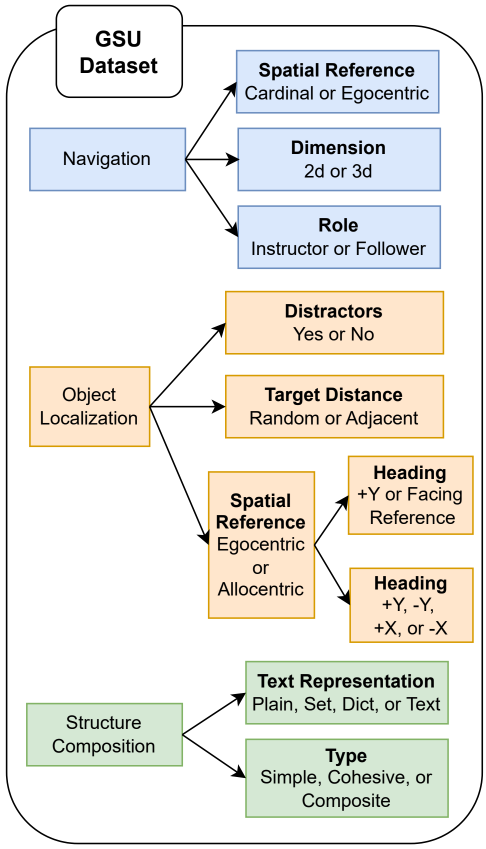

Figure 3: GSU Dataset Tasks and Settings

Target Distance: We choose if the target structures are placed adjacent to the spatial reference (i.e. adjacent to the agent in the *Egocentric* setting and adjacent to the reference block in the *Allocentric* setting) or anywhere on the grid, which is effectively further away due to how few coordinates are adjacent to a single coordinate. Unexpectedly, blocks that are directly adjacent to the reference location are harder for the model to describe spatially (7). This trend was observed for the *Allocentric* setting as well, see 8.

A.0.3 Structure Composition

To generate the Structure Composition task, we first generate 3 shapes and associated verbal descriptions using a template with minor variations for synonyms and grammatical structure. In the *Simple* setting we select the first shape and associated description, whereas in the *Cohesive* setting we then adjust half the blocks in the structure to be a different color and update the verbal description accordingly. Finally, for the *Composite* setting we translate the 3 shapes to be adjacent and join their textual descriptions with the appropriate spatial terms.

We initially explored a variety of textual representations for coordinates. In our *Plain* representation, coordinates are input into LLMs as sequences resembling pseudo-code:

$$COLOR_1\ X_1\ Y_1\ Z_1$$
$$\ldots$$
$$COLOR_N\ X_N\ Y_N\ Z_N$$

We also explore alternate representations of coordinates, which this task is particularly suited for since it requires processing far more coordinates than previous tasks. The *Set* representation closely resembles *Plain*, but utilizes structures that the model might have encountered in code:

"($COLOR_1$, X_1, Y_1, Z_1), ..., ($COLOR_N$, X_N, Y_N, Z_N)"

Dict resembles a dictionary where each value is explicitly labeled:

"(color = $COLOR_1$, x = X_1, y = Y_1, z = Z_1), ..., (color = $COLOR_N$, x = X_N, y = Y_N, z = Z_N)"

Text is a more verbal representation and the one used in the previous tasks:

"a $COLOR_1$ block at (X_1, Y_1, Z_1), ..., and a $COLOR_N$ block at (X_N, Y_N, Z_N)".

However, for brevity, we chose the overall best performing representation for the main paper, and include the supplementary results here (see 3. We found it interesting that, in general, the *Set* and *Text* representations were the best performing across all models (with notable exceptions, such as Qwen family models highly preferring the *Dict* representation), despite the convention in prior works to use the *Plain* formulation. We also anticipated the *Set* representation performing most similarly to the *Plain* as they were the most similar and were surprised to see that performance was quite different across these two settings. While the *Text* representation is not the most efficient, it follows that textual models would be able to best parse information presented in a form closer to natural language than to a set or dictionary.

A.1 Shape Metric

For the shape metric, we check for synonyms in the generation and map them to their corresponding shape. The synonyms are built around the base vocabulary and then appended to with common outputs in actual generations. For columns, we accept "column", as well any production with "vertical" or "upright" and "line" or "row". For rows, we accept any generation with "row" or "line" that does not mention verticality. For towers (multiple columns grouped together), we accept "tower", "rectangular prism", and "pillar". For planes (rectangular prisms only one block thick), we accept "plane", "platform", "rectangle", "wall", "square", "ring", and "O". For cubes, we accept "cube".

Furthermore, since shapes can be similar to others, if a generation does not match at least one of the ground truth shapes, we offer it partial credit based on how similar the shape it predicted was. Confusing rows and columns leads 0.6 of the full 1.0 received for a total match, as does mixing up a column and a tower. Since towers and cubes are both rectangular prisms, we offer half credit for such mistakes, and 0.1 for towers and planes or planes and cubes, and 0.2 for rows and towers.

B Training Details

For the LORA fine-tuning runs, we experimented wit various ranks, learning rates, and training set sizes. Overall, we found that the model performance plateaued at its highest level with rank 64, a learning rate of 1e-4, and a training set of size 3000. We used alpha=8 as recommended in cite and an AdamW optimizer. For the FlanT5-large runs, we allow the model to train for 15 epochs or until the loss converges using an AdamW optimizer with a learning rate of 5e-5.

This project was run on up to 4 gpuA100x4's using approximately 1000 GPU hours of compute.

C Model Selection

For the newer models used in the prompting experiments, we considered using the Thinking versions as they might be better suited to reasoning over grid logic, but found that the instruct models performed better. We also explored the Deepseek model distilled on Qwen32b, which received a 0.15 and a 0.26 on the *Egocentric Navigation Follower* task in 2D and 3D (compared to the 0.37 and 0.32 in the standard 32b model shown in 1). It also only received a 0.15 on the *Egocentric Navigation Instructor* task in 3D.

D Additional Results

		Plain			Dict			Set		
		$O_s \uparrow$	$O_c \uparrow$	$O_f \uparrow$	$O_s \uparrow$	$O_c \uparrow$	$O_f \uparrow$	$O_s \uparrow$	$O_c \uparrow$	$O_f \uparrow$
0-shot	Mistral-7b	15.92	42.64	12.45	13.72	47.93	10.87	15.88	46.70	13.83
	Pixtral	37.67	74.36	14.64	44.67	70.13	13.62	32.08	72.91	7.25
	Llama-8b	21.29	63.90	13.00	22.78	63.91	15.56	25.47	68.83	18.65
	LlamaVL-11b	35.83	69.07	11.83	24.57	60.48	12.09	20.94	62.14	13.37
	Qwen-7b	32.17	68.72	11.48	40.03	68.32	16.73	28.52	62.76	16.65
	QwenVL-7b	18.21	60.37	8.08	33.47	52.78	12.81	30.32	45.83	11.62
1-shot	Mistral-7b	27.56	58.54	13.41	32.00	55.90	13.06	38.44	71.95	11.75
	Pixtral	64.58	82.49	24.20	63.00	81.98	20.22	64.20	89.59	31.18
	Llama-8b	59.90	63.37	17.81	55.83	66.74	20.83	59.70	85.83	30.19
	Llama-11b	63.83	64.75	22.54	59.50	79.86	16.40	60.33	88.96	27.35
	Qwen-7b	50.75	76.25	16.00	63.65	81.87	23.08	54.00	88.80	29.67
	Qwen-7b	52.52	65.72	8.24	55.67	62.57	13.38	66.90	85.27	23.37
Full FT	FlanT5	92.92	70.32	81.59	84.25	67.27	80.88	90.08	68.07	79.68

Table 3: Structure Composition results for Plain, Dict, and Set Representations

	$+Y$	$+X$	$-Y$	$-X$
Mistral-7b	36.90	34.98	29.13	32.26
Llama-8b	51.33	40.23	43.80	37.84
Qwen-7b	77.27	52.10	38.26	44.32

Table 4: Average Spatial Overlaps by Heading

		Cardinal 3D		Egocentric 3D	
		acc	dist	acc	dist
1-shot	Mistral-7b	0.48	4.14	0.38	6.14
	Llama-8b	0.62	2.65	0.47	4.85
	Qwen-7b	0.63	7.79	0.19	20.44
Few-shot	Mistral-7b	0.19	6.23	0.17	8.32
	Llama-8b	0.32	4.84	0.20	7.69
	Qwen-7b	0.50	4.31	0.29	7.07

Table 5: 1-shot vs Few-shot Navigation: Comparison over 1-shot performance with reasoning vs Few-Shot without reasoning examples

		Set			Text		
		$O_s \uparrow$	$O_c \uparrow$	$O_f \uparrow$	$O_s \uparrow$	$O_c \uparrow$	$O_f \uparrow$
1-shot	Mistral-7b	26.14	54.12	15.18	34.73	57.10	24.22
	Llama-8b	59.98	63.58	18.69	60.26	67.05	22.16
	Qwen-7b	54.70	74.86	20.85	57.18	71.78	19.14
Few-shot	Mistral-7b	38.44	71.95	11.75	49.75	72.60	26.33
	Llama-8b	59.70	85.83	30.19	62.04	86.69	26.10
	Qwen-7b	54.00	88.80	29.69	54.58	85.43	25.67

Table 6: 1-shot vs Few-shot Structure Composition: Comparison over 1-shot performance with reasoning vs Few-Shot without

		Egocentric				Allocentric	
		Base		Distractors		+Y heading	Ref. heading
		adj. ↑	rand. ↑	adj. ↑	rand. ↑	adj. ↑	adj. ↑
0-shot	Mistral-7b	30.87	34.67	30.20	31.90	32.90	29.67
	Pixtral-12b	35.22	42.54	31.42	46.73	40.73	34.43
	Llama-8b	36.52	38.78	37.08	33.10	37.11	35.52
	LlamaVL-11b	35.33	41.50	33.08	36.55	41.08	39.72
	Qwen-7b	<u>53.55</u>	<u>48.20</u>	<u>53.07</u>	<u>49.02</u>	<u>51.75</u>	<u>46.82</u>
	QwenVL-7b	26.18	37.83	31.98	36.15	46.75	43.25
1-shot	Mistral-7b	30.30	40.22	30.32	39.61	46.50	32.32
	Pixtral-12b	<u>57.55</u>	**65.87**	40.83	54.30	48.18	42.25
	Llama-8b	48.48	52.82	46.17	50.28	50.18	42.37
	LlamaVL-11b	53.56	58.27	48.40	51.55	48.08	<u>48.20</u>
	Qwen-7b	53.80	59.70	52.81	<u>55.42</u>	<u>54.78</u>	47.93
	QwenVL-7b	51.78	52.42	48.67	52.42	44.95	41.08
	QwenVL-72b	-	-	<u>79.53</u>	**79.45**	53.83	47.62
LORA	Mistral-7b	-	-	36.72	62.62	59.03	47.89
	Llama-8b	-	-	**80.05**	<u>72.60</u>	<u>61.58</u>	<u>84.03</u>
Full FT	FlanT5	**98.08**	65.75	73.03	62.50	**100**	**96.1**

Table 7: Object Localization spatial overlap, O_s, scores across the *Egocentric* and *Allocentric* settings, as well various headings and distractor settings. VLM counterparts are in light blue backgrounds, best results overall are **bolded** and best results in generation setting are <u>underlined</u>.

	$+Y$ Heading		Facing Reference	
	adj	rand	adj	rand
0-shot Mistral-7b	32.90	38.67	29.67	38.12
1-shot Mistral-7b	46.50	39.83	32.32	41.75

Table 8: Adjacent vs Random Block placements in the Allocentric Object Localization Task

		Follower								Instructor			
		Cardinal				Egocentric				Cardinal		Egocentric	
		2d		3d		2d		3d		2d	3d	2d	3d
		acc.↑	dist↓	acc↑	dist↓	acc↑	dist↓	acc↑	dist↓	acc↑	acc↑	acc↑	acc↑
0-shot	Mistral-7b	0.35	6.15	0.24	14.56	0.22	9.92	0.09	17.49	0.01	0.04	0.0	0.01
	Pixtral-12b	0.85	2.39	0.54	5.34	0.29	7.34	0.34	7.05	0.21	0.10	0.09	0.07
	Llama-8b	0.81	2.34	0.35	8.52	0.33	6.86	0.36	6.55	0.05	0.11	0.02	0.02
	LlamaVL-11b	0.80	2.28	0.37	9.20	0.28	7.55	0.38	6.13	0.06	0.10	0.03	0.04
	Qwen-7b	0.96	0.49	0.86	0.85	**0.56**	**4.45**	**0.53**	**5.41**	0.20	0.19	0.16	0.14
	QwenVL-7b	0.82	2.92	0.78	3.05	0.35	9.13	0.48	6.75	0.11	0.05	0.09	0.06
	Qwen-72b	-	-	0.67	2.40	-	-	0.36	6.21	-	**0.62**	-	0.37
1-shot	Mistral-7b	0.88	0.92	0.48	4.14	0.30	6.85	0.38	6.14	0.59	0.38	0.15	0.11
	Pixtral-12b	0.97	0.37	0.67	4.20	0.33	6.31	0.50	5.60	**0.87**	0.80	**0.26**	0.28
	Llama-8b	0.99	0.04	0.62	2.65	0.33	7.03	0.47	4.85	0.69	0.54	0.06	0.23
	LlamaVL-11b	**1.00**	**0.0**	0.60	2.94	0.32	7.36	0.47	5.78	0.74	0.59	0.14	0.21
	Qwen-7b	0.89	3.11	0.63	7.79	0.09	22.54	0.19	20.44	0.76	0.83	0.21	0.19
	QwenVL-7b	**1.0**	**0.0**	0.90	1.02	0.29	7.9	0.21	12.91	0.44	0.66	0.21	0.23
	Qwen-72b	-	-	**1.00**	**0.00**	-	-	0.75	**3.66**	-	**0.99**	-	0.54
Full FT	FlanT5	-	-	0.94	0.09	-	-	**0.82**	13.14	-	0.90	-	**0.85**

Table 9: Navigation task results. We compare 0- and 1-shot prompting on base text-only LLMs, similarly sized VLMs (represented with light blue backgrounds), and a larger LLM variant of the overall best performing model, as well as a fully fine-tuned LM. We report the accuracies out of 1 for the final coordinates and their distance from the true coordinates for the *Follower* setting and the accuracy out of 1 of the generated instruction for the *Instructor* setting. The best results for each experiment are **bolded** while the best in each section are underlined.

E Prompt and Generation Examples

E.1 Instructor Navigation

2d Egocentric Navigation Instructor Prompt w/ 1-shot Example

You are in a 2D environment with (x, y) coordinates set up like a standard horizontal Cartesian plane. You will start at the origin (0, 0), which is at the center of this grid. You are currently facing the positive y direction, with the positive x direction to your right. However, when you move in a direction you must turn to face that direction, rotating your frame of reference. For example, if you move left, you will rotate 90 degrees and be facing the negative x direction with positive y to your right.
First, let me give you an example! You start at (0, 0), go to (3, 0), go to (3, 1), and end at (3, -1). Describe the path that traverses the provided coordinates.
I must remember that each time I move left, right, or back, I will be turning to face a new direction.
1. To get from (0, 0) to (3, 0), I must move 3 steps to the right. I will now be facing the positive x direction, with negative y to my right.
2. To get from (3, 0) to (3, 1), I must move 1 step left. I will now be facing positive y, with positive x to my right.
3. To get from (3, 1) to (3, -1), I must move 2 steps back. So, my path is [ANS] right 3, left 1, back 2 [/ANS]. Great, now let's try a real problem!
Start at (0, 0, 0). Go to (0, 7, 0). Go to (0, -1, 0). End at (-4, -1, 0). Describe the path that you will take to traverse the provided coordinates. Format your answer as a series of directions and distances, e.g. [ANS] forward 2, right 3, back 1 [/ANS]. Let's go!

Mistral-7b Generation

CORRECT ANSWER : ['forward 7', ' backward 8', ' right 4']
GENERATED ANSWER : ['forward 7', 'backward 8', 'backward 4', 'forward 1']

2d Card2Ego Navigation Instructor Prompt w/ 1-shot Example

You are on a 2D grid and will be given a path using cardinal directions (North, East, South, West) that you need to convert into egocentric directions (left, right, forward, backward). Keep in mind that to move in a cardinal direction, you must turn to face it. This means that the egocentric instructions will not map directly to cardinal ones, but change depending on the direction you last moved. E.g. if you just moved East and then want to move South, then the egocentric instruction is to move right, since South is to your right if you're facing East.
Let's begin with an example. You start by facing North and the path is: West 2, North 3, East 1.
The first step is easy. I am facing North and must move West which is to my left. So the first step is: left 2.
Now, for the second step I am facing the direction I just moved in, West, and I want to move North. If I am facing West, North is to my right. So the instruction is right 3.
Then, for the third step I am facing North and want to go East 1. East is to my right and so I need to move right 1.
Finally, I can put that all together for: [ANS] left 2, right 3, right 1 [/ANS] Now, you give it a try! You start facing North and the path is: West 3, East 8, South 1, South 10. What is the path expressed with egocentric directions?

GPT-4o Generation

CORRECT ANSWER : [left 3, backward 8, right 1, forward 10]
GENERATED ANSWER : Let's break it down step by step.
1. Start facing North. To go West means turning left. - Instruction: left 3
2. Now facing West. To go East means turning around (backward). - Instruction: backward 8
3. Now facing East. To go South means turning right. - Instruction: right 1
4. Still facing South. To go South again means going forward. - Instruction: forward 10
Putting it all together: [ANS] left 3, backward 8, right 1, forward 10 [/ANS]

E.2 Follower Navigation

3d Egocentric Navigation Follower Prompt w/ 1-shot Example

You are in a 3D environment with (x, y, z) coordinates set up like a standard Cartesian plane. The x and y dimensions are horizontal, while the z dimension is the vertical component. You will start at the origin (0, 0, 0), which is at the center of this grid. You are currently facing the positive y direction, with the positive x direction to your right. However, when you move in a direction you must turn to face that direction, rotating your frame of reference. For example, if you move left, you will rotate 90 degrees and be facing the negative x direction with positive y to your right. Explain your final coordinates after travelling. Please format your final coordinates as: [ANS] (x, y, z) [/ANS]. Let's go!
Let's start with an example: First, you move 3 steps to your right. Next, you move 2 steps down. You move 4 steps backwards. Finally, you move 2 steps left. Where are you now?
To solve this, we should break down the steps we take.
1. We start at (0, 0, 0) facing the positive y direction, with positive x to our right.
2. Moving 3 steps right means moving along positive x, i.e. increasing the x value by 3. So, our new position is (3, 0, 0). Also, we turned to face positive x, so now negative y is to our right.
3. Moving 2 steps down means moving along negative z, i.e. decreasing the z value by 2. So, our new position is (3, 0, -2). Remember that moving up and down doesn't change our heading, so we are still facing positive x with negative y to our right.
4. Moving 4 steps back means moving opposite the direction we are facing. So we will move along negative x, i.e. decrease the x value by 4. So, our new position is (-1, 0, -2). Since we turned to move, we are now facing negative x and positive y is to our right.
5. Moving 2 steps left means moving away from the right. So we will move along negative y, i.e. decrease y by 2. So, our new position is (-1, -2, -2). Our final position is [ANS] (-1, -2, -2) [/ANS]
Now, let's try a real problem! First, you move 7 steps to your right. You then move 5 steps forward. You move 10 steps up. Lastly, you move 5 steps backward. Explain your final coordinates:

Mistral-7b Generation

Correct Answer is : 'x': 7, 'y': 0, 'z': 10
Generated Coordinates are : (7, 0, 10)

E.3 Egocentric Object Localization

3d Egocentric Object Localization w/ Distractors, 1-shot Example

You are in a 3D environment with (x, y, z) coordinates set up like a standard Cartesian plane. The x and y dimensions are horizontal, while the z dimension is the vertical component. All axes range from (-10, 10). Your task is to describe where objects are relative to you, without using coordinates. Instead use relative descriptions like 'directly behind me' or 'to my left'. Explain your thought process and please format your final answer with [ANS] tags like so: the green block is [ANS] in front of me and to my left [/ANS]. Let's get started!
Here is an example: You are at (2, 2, 0), facing the negative y direction, so negative x is to your right. The positive z axis is always up. There is a blue block at (-3, -1, 0). Where is the blue block relative to you?
For the x dimension, I know that negative x is to my right. That means x values smaller than mine are to my right and ones bigger than mine are to my left. My x coordinate is 2 and the blue block's is -3. So it is to my right.
Next, for the y dimension, I know that I am facing negative y, so it is in front of me. This means that y values smaller than mine are in front of me and those bigger than mine are behind me. My y coordinate is 2 and the blue is at -1. So, it is in front of me.
Now, for the z dimension, my orientation does not matter. Higher z values are above me and lower z values are below me. However, we are both at z = 0, so we are level. Putting that together, the blue block is [ANS] to my right and slightly to the front [/ANS]
Now let's try a real problem! You are at (-1, 5, -9), facing the negative y direction, so negative x is to your right. The positive z axis is always up. There is a yellow block at (-2, -1, -3), a orange block at (3, 2, -3), a green block at (-1, 4, 10), a red block at (-7, 1, 7), and a blue block at (2, 9, -9). Where is the red block relative to you?

Generation by Qwen-7b

Correct Answer is : The red block is [ANS] in front of me, to my right, and above me [/ANS]
Generated Description is : The red block is [ANS] to my left and behind me [/ANS]
BLEU : 0.0
BERT : 0.8915
SPATIAL OVERLAP : 0.0

E.4 Allocentric Object Localization

3d Allocentric Object Localization w/ +Y heading, 1-shot Example

You are in a 3D environment with (x, y, z) coordinates set up like a standard Cartesian plane. The x and y dimensions are horizontal, while the z dimension is the vertical component.All axes range from (-20, 20). Your task is to describe where objects are relative to you and other objects, without using coordinates. Instead use relative descriptions like 'directly in front of the blue cylinder'. Explain your thought process and please format your final answer with [ANS] tags like so: the yellow block is [ANS] below and to the back right of [/ANS] the purple block. Let's get started!
You are at the origin. There is a red block at (0, 8, -7) and a blue block at (0, 7, -8). You may assume that you are facing the positive y direction. The positive z axis is always up. Where is the blue block relative to the red block given your point of view?

Llama-8b Generation

Correct Answer is : The red block is [ANS] below and in front of [/ANS] the blue block.
Generated Description is : The red block is [ANS] below and to the back of [/ANS] the blue block.
BLEU : 0.0
BERT : 0.826
SPATIAL OVERLAP : 0.33

E.5 Composite Description

Composite Structure Composition Task

You are in a 3D grid environment with (x, y, z) coordinates set up like a standard Cartesian plane. The x and y dimensions are horizontal, while the z dimension is the vertical component. Your task is to describe a set of blocks to someone without mentioning coordinates or axes, instead describe the structures as a whole. Format your answer with [ANS] tags like so: there are [ANS] 6 orange blocks in a column [/ANS].
Blocks: The blocks placed on the grid are: (color : purple, x : 0, y : 0, z : 0), (color : purple, x : 0, y : 0, z : 1), (color : purple, x : 0, y : 0, z : 2), (color : purple, x : 0, y : 0, z : 3), (color : purple, x : 0, y : 0, z : 4), (color : purple, x : 0, y : 0, z : 5), (color : purple, x : 0, y : 0, z : 6), (color : purple, x : 0, y : 0, z : 7), (color : purple, x : 0, y : 1, z : 0), (color : purple, x : 0, y : 1, z : 1), (color : purple, x : 0, y : 1, z : 2), (color : purple, x : 0, y : 1, z : 3), (color : purple, x : 0, y : 1, z : 4), (color : purple, x : 0, y : 1, z : 5), (color : purple, x : 0, y : 1, z : 6), (color : purple, x : 0, y : 1, z : 7), (color : purple, x : 1, y : 0, z : 0), (color : purple, x : 1, y : 0, z : 1), (color : purple, x : 1, y : 0, z : 2), (color : purple, x : 1, y : 0, z : 3), (color : purple, x : 1, y : 0, z : 4), (color : purple, x : 1, y : 0, z : 5), (color : purple, x : 1, y : 0, z : 6), (color : purple, x : 1, y : 0, z : 7), (color : purple, x : 1, y : 1, z : 0), (color : purple, x : 1, y : 1, z : 1), (color : purple, x : 1, y : 1, z : 2), (color : purple, x : 1, y : 1, z : 3), (color : purple, x : 1, y : 1, z : 4), (color : purple, x : 1, y : 1, z : 5), (color : purple, x : 1, y : 1, z : 6), (color : purple, x : 1, y : 1, z : 7), (color : yellow, x : 0, y : -2, z : 0), (color : yellow, x : 0, y : -2, z : 1), (color : yellow, x : 0, y : -2, z : 2), (color : yellow, x : 0, y : -2, z : 3), (color : yellow, x : 0, y : -2, z : 4), (color : yellow, x : 0, y : -1, z : 0), (color : yellow, x : 0, y : -1, z : 1), (color : yellow, x : 0, y : -1, z : 2), (color : yellow, x : 0, y : -1, z : 3), (color : yellow, x : 0, y : -1, z : 4), (color : yellow, x : 1, y : -2, z : 0), (color : yellow, x : 1, y : -2, z : 1), (color : yellow, x : 1, y : -2, z : 2), (color : yellow, x : 1, y : -2, z : 3), (color : yellow, x : 1, y : -2, z : 4), (color : yellow, x : 1, y : -1, z : 0), (color : yellow, x : 1, y : -1, z : 1), (color : yellow, x : 1, y : -1, z : 2), (color : yellow, x : 1, y : -1, z : 3), (color : yellow, x : 1, y : -1, z : 4), (color : red, x : 0, y : 0, z : 8), (color : red, x : 0, y : 1, z : 8), (color : red, x : 0, y : 2, z : 8), (color : red, x : 0, y : 3, z : 8), (color : red, x : 0, y : 4, z : 8), (color : red, x : 1, y : 0, z : 8), (color : red, x : 1, y : 4, z : 8), (color : red, x : 2, y : 0, z : 8), (color : red, x : 2, y : 4, z : 8), (color : red, x : 3, y : 0, z : 8), (color : red, x : 3, y : 1, z : 8), (color : red, x : 3, y : 2, z : 8), (color : red, x : 3, y : 3, z : 8), (color : red, x : 3, y : 4, z : 8) Now describe the structure they made.

Qwen-7b Generation

Correct Answer is : 8 x 2 x 2 purple tower with a tower of yellow blocks 5 high and 2 wide in front of it and a empty 4 x 5 red wall above it
Generated Description is : two large purple platforms stacked on top of each other, both extending from the bottom to the top of the visible area, and a smaller yellow platform that extends from slightly below the bottom to just above the middle of the visible area. Additionally, there is a red block formation at the very top, creating a cross-like pattern centered around the central vertical axis
BLEU : 0.0
BERT : 0.8357
SPATIAL OVERLAP : 0.2
COLOR OVERLAP : 1.0
SHAPE OVERLAP : 0.4
NUMERICAL OVERLAP : 0.0