LLM-Grounder: Open-Vocabulary 3D Visual Grounding with Large Language Model as an Agent

Jianing Yang*,1 Xuweiyi Chen*,1 Shengyi Qian1 Nikhil Madaan2 Madhavan Iyengar1
David F. Fouhey1,3 Joyce Chai1

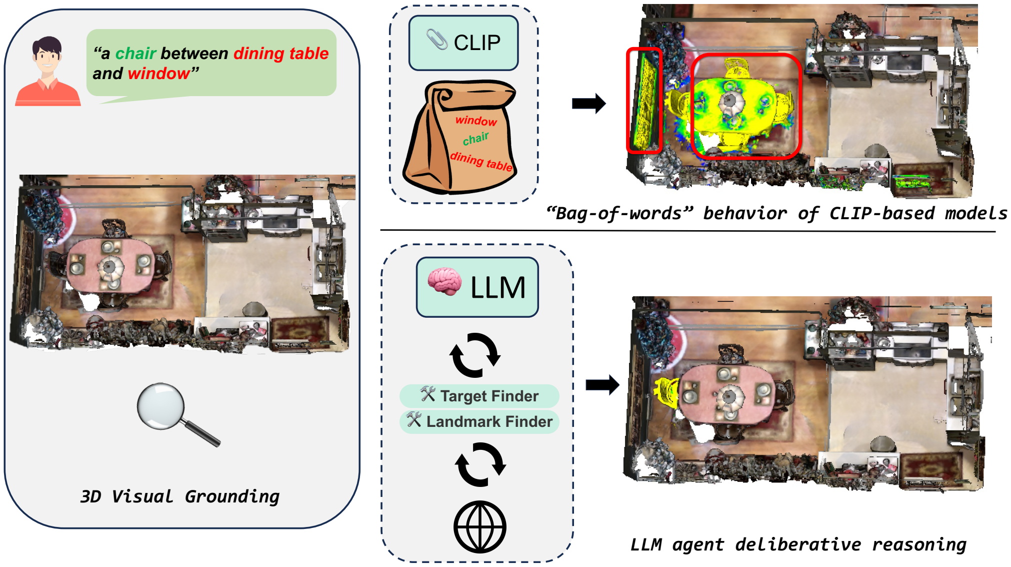

"a chair between dining table and window"
3D Visual Grounding
CLIP
window
chair
dining table
"Bag-of-words" behavior of CLIP-based models
LLM
Target Finder
Landmark Finder
LLM agent deliberative reasoning

Fig. 1: In open-vocabulary 3D visual grounding task, CLIP-based models tend to treat text input as *"bag of words"*, ignoring semantic structures of compositional text input, e.g., consisting of complex spatial relations among objects. On the top-right is a demonstration of such behavior when using OpenScene [1], a CLIP-based 3D grounding method, as a visual grounder. When asked to ground the spatially-informed text query "a chair between the dining table and window", it incorrectly highlights the dining table and window, which are not the target but rather referential landmarks (red bounding boxes). We propose to address this problem by leveraging a large language model (LLM) to 1. Deliberately generate a plan to decompose complex visual grounding queries into sub-tasks; 2. Orchestrate and interact with tools such as target finder and landmark finder to collect information; 3. Leverage spatial and commonsense knowledge to reflect on collected feedback from tools.

***Abstract*— 3D visual grounding is a critical skill for household robots, enabling them to navigate, manipulate objects, and answer questions based on their environment. While existing approaches often rely on extensive labeled data or exhibit limitations in handling complex language queries, we propose LLM-Grounder, a novel zero-shot, open-vocabulary, Large Language Model (LLM)-based 3D visual grounding pipeline. LLM-Grounder utilizes an LLM to decompose complex natural language queries into semantic constituents and employs a visual grounding tool, such as OpenScene or LERF, to identify objects in a 3D scene. The LLM then evaluates the spatial and commonsense relations among the proposed objects to make a final grounding decision. Our method does not require any labeled training data and can generalize to novel 3D scenes and arbitrary text queries. We evaluate LLM-Grounder on the ScanRefer benchmark and demonstrate state-of-the-art zero-shot grounding accuracy. Our findings indicate that LLMs significantly improve the grounding capability, especially for complex language queries, making LLM-Grounder an effective approach for 3D vision-language tasks in robotics.**

I. Introduction

Imagine you are put into a 3D scene and asked to find *"a chair between dining table and window"* (Fig. 1). It is easy for humans to figure out the answer. Such a skill is called 3D visual grounding, and we typically rely on it for daily tasks that range from finding objects to manipulating tools. Mastering such an ability is critical to building any household robots to assist humans, as it serves as a basic skill needed for complex navigation (knowing where to go), manipulation (what/where to grab), and question-answering.

To endow robots with such an ability, researchers have developed a number of approaches. One direction is to train a 3D-and-text end-to-end neural architecture to propose bounding boxes around objects and jointly model text-bounding-box matching [2–11]. However, such models typically need a large amount of 3D-text pairs for training data, which is

*Equal contribution.

[1] Computer Science and Engineering, University of Michigan, Ann Arbor, MI, USA, 48109. Contact: Jianing Yang `jianingy@umich.edu`.

[2] Nikhil Madaan is an independent researcher.

[3] New York University.

This work is generously supported by NSF IIS-1949634, NSF SES-2128623, and Microsoft Academic Program Computing Credit.

Project website: `https://chat-with-nerf.github.io/`

difficult to obtain [12,13]. As a result, such trained methods often do not obtain good performance on new scenes. More recently, attempts to address open-vocabulary 3D visual grounding have been made [1,14–23], often building on the strength of CLIP [24]. The dependence on CLIP, however, makes them exhibit "bag-of-words" behaviors where orderless content is modeled well, but attributions, relations, and orders are ignored when processing the text and visual information [25]. For example, as illustrated in Fig. 1, if the text query "a chair between dining table and window" is given to OpenScene [1], the model grounds all of the chairs, window, and dining table in the room, ignoring that the window and dining table are just landmarks used to provide spatial relations with the target chair.

At the same time, Large Language Models (LLMs) such as ChatGPT and GPT-4 [26] have demonstrated impressive language understanding capabilities, including planning and tool-using. These abilities enable LLMs to be used as agents to solve complex tasks by breaking the tasks into smaller pieces and knowing when, what, and how to use a tool to complete sub-tasks [27–34]. This is exactly what is needed for 3D visual grounding with complex natural language queries: parsing the compositional language into smaller semantic constituents, interacting with tools and environment to collect feedback, and reasoning with spatial and commonsense knowledge to iteratively ground the language to the target object. Given these observations, we ask the question,

Can we use an LLM-based agent to improve
zero-shot open-vocabulary 3D visual grounding?

In this work, we propose *LLM-Grounder*, a novel open-vocabulary, zero-shot, LLM-agent-based 3D visual grounding pipeline. Our intuition is that an LLM can alleviate the "bag-of-words" weakness of a CLIP-based visual grounder by taking the difficult language decomposition, spatial and commonsense reasoning tasks upon the LLM itself while capitalizing on the strength of a visual grounder to ground simple noun phrases. Described in Section III, LLM-Grounder uses an LLM at its core to orchestrate the grounding process. The LLM first parses compositional natural language queries into semantic concepts such as object category, object attributes (color, shape, and material), landmarks, and spatial relations. These sub-queries are passed into a visual grounder *tool* backed by OpenScene [1] or LERF [35], which are CLIP-based [24] open-vocabulary 3D visual grounding methods, to ground each concept in the scene. The visual grounder proposes a few bounding boxes around the most relevant candidate areas in the scene for a concept. For each of these candidates, the visual grounder tools calculate and provide spatial information such as object volumes and distances to landmarks back to the LLM agent to enable the agent to holistically evaluate the situation, in terms of spatial relation and commonsense and select a candidate that best matches all criteria in the original query. This process is repeated until the LLM agent decides it has reached a conclusion. Notably, our approach extends prior neural-symbolic approaches [36] by giving environment feedback to the agent and making the agent's reasoning process closed-loop.

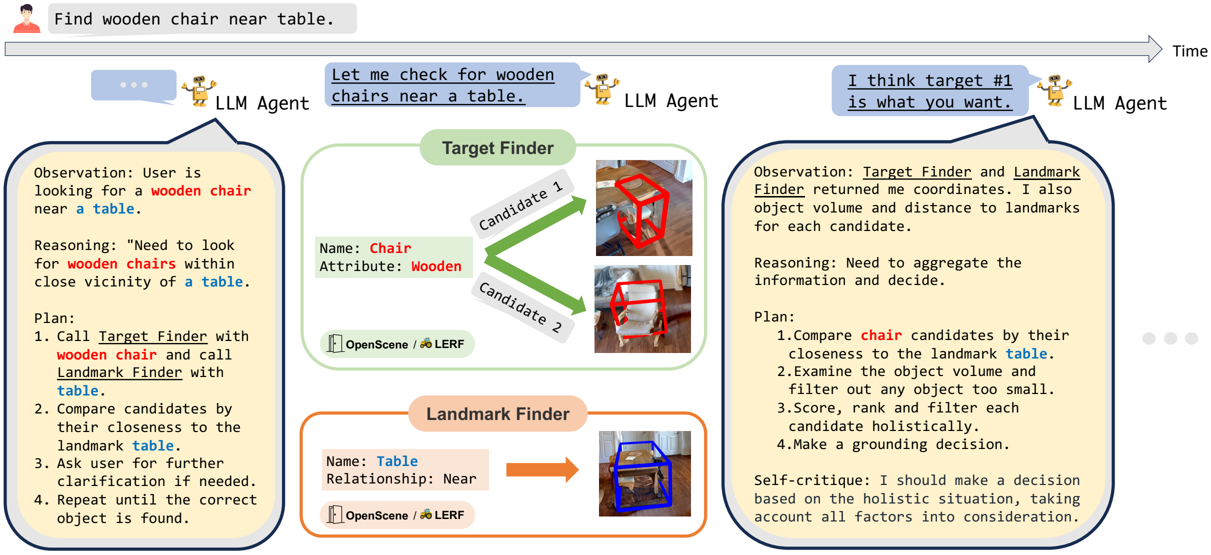
It is important to note that our approach does not need any training on labeled data. It is open-vocabulary and can zero-shot generalize to novel 3D scenes and arbitrary text queries, a desirable property given the semantic diversity of 3D scenes and the limited availability of 3D-text labeled data. In our experiments (Section IV), we evaluate LLM-Grounder on the ScanRefer benchmark [12]. This benchmark primarily evaluates 3D vision-language grounding capability that requires understanding of compositional visual referential expressions. Our approach improves the grounding capability of zero-shot open-vocabulary methods such as OpenScene and LERF, and demonstrates state-of-the-art zero-shot grounding accuracy on ScanRefer with no labeled data used. Our ablation study shows LLM increases grounding capability more as the language query becomes more complex. These findings underscore the potential of LLM-Grounder as an effective approach for 3D vision-language tasks, making it particularly well-suited for robotics applications where understanding complex environments and responding to dynamic queries are essential.

In summary, the contribution of this paper is as follows:

- We find that using LLM as an agent can improve grounding capability for zero-shot, open-vocabulary methods on the 3D visual grounding task.
- We achieve SOTA on ScanRefer in a zero-shot setting, using no labeled data.
- We find LLM is more effective when the grounding query text is more complex.

II. Related Work

3D Visual Grounding with Natural Language. Grounding a natural language query in an unstructured 3D scene is essential for various robotic tasks. Pioneering benchmarks such as ScanRefer [12] and ReferIt3D [13] have advanced this field. As proposed in these benchmarks, the referential tasks in 3D and text necessitate a deep understanding of both the compositional semantics of language and the structures, geometries, and semantics of 3D scenes. Numerous methods that are jointly trained on 3D and language have been proposed [2–11] to advance performance. However, these methods are limited to closed-vocabulary settings due to the specific object classes presented in the original ScanNet [37], upon which these benchmarks are built. Motivated by advances in 2D open-vocabulary segmentation [38–40], researchers have explored 3D open-vocabulary grounding [1,14–21,35,41]. However, these methods mostly rely on CLIP [24] as the underlying vision-language bridge. This works well when the grounding text query is a simple noun phrase (e.g., "a red apple"); however, research has shown CLIP exhibits "bag-of-words" behavior and lacks compositional understanding such as relation, attribution, and order of either text or visual [25], a crucial aspect of the challenges presented in ScanRefer and ReferIt3D. Recognizing this aspect, Semantic Abstraction [22] and 3D-CLR [23] use spatial-informed text-and-3D data to train a dedicated neural



Find wooden chair near table.

Let me check for wooden chairs near a table.

I think target #1 is what you want.

Observation: User is looking for a wooden chair near a table.

Reasoning: "Need to look for wooden chairs within close vicinity of a table.

Plan:
1. Call Target Finder with wooden chair and call Landmark Finder with table.
2. Compare candidates by their closeness to the landmark table.
3. Ask user for further clarification if needed.
4. Repeat until the correct object is found.

Name: Chair
Attribute: Wooden

Name: Table
Relationship: Near

Observation: Target Finder and Landmark Finder returned me coordinates. I also object volume and distance to landmarks for each candidate.

Reasoning: Need to aggregate the information and decide.

Plan:
1.Compare chair candidates by their closeness to the landmark table.
2.Examine the object volume and filter out any object too small.
3.Score, rank and filter each candidate holistically.
4.Make a grounding decision.

Self-critique: I should make a decision based on the holistic situation, taking account all factors into consideration.

Fig. 2: Overview of LLM-Grounder. Given a query to ground an object, our approach, backed by an LLM agent, reasons on the user's request and generates a plan to ground the object by using tools. The agent interacts with tools such as target find and landmark finder to gather information such as object bounding box, object volume, and distances to landmarks from the tools. This information is then returned to the agent to conduct further spatial and commonsense reasoning to rank, filter and select the best matching candidate.

network to parse and ground the compositional semantics of the text query before grounding. In contrast, our method explores the possibilities of using an LLM agent to accomplish the same without training (zero-shot). NS3D [36] uses LLM-based code generation to generate programs to address this problem, which is more similar to our approach, but it also uses ground-truth object segmentation and category to simplify visual grounding and thus lacks open-vocabulary and zero-shot capabilities.

LLM Agents Recent advancements in large language models (LLM) [26,42–46] have demonstrated surprising emerging abilities. Here, we list a few abilities that enable LLM to be used as an agent.

a) Planning: Planning involves breaking complex goals into sub-goals and self-reflecting based on issued actions and environmental feedback. Chain-of-thought [27] shows that LLM demonstrated better planning capabilities when instructed to "think step-by-step" by decomposing complex tasks into smaller tasks. Tree-of-thoughts [28] extends this approach by exploring multiple thoughts per step, turning the chain into a tree. [47–50] demonstrate that LLM, when instructed to self-reflect on its output and environmental feedback, can produce better output.

b) Tool-Using: The ability to use tools is a unique feature of human intelligence. Recognizing that current LLMs are not good at all tasks (math and factual question-answering problems, for example), researchers have explored possibilities of letting LLMs orchestrate tool-using to fulfill a task. At its core, the tool-using problem is to decide which tool to use and when to use them. Socratic Models [29] uses natural language as a medium to engage an LLM agent in a guided discussion with other multimodal language models, such as vision-language models and audio-language models, to complete a task collectively. MRKL [51] and TAML [52] equip an LLM with a calculator and demonstrate its increased ability to solve math problems. Building on these findings, software libraries like LangChain [53] has been developed to streamline LLM tool-using for developers. ToolFormer [30], HuggingGPT [54] and API-Bank [55] push tool-using further by opening up more APIs and machine learning models as tools for LLM to use.

In robotics, SayCan [31], InnerMonologue [32], Code as Policies [33] and LM-Nav [34] use the planning and tool-using capability of LLM to let it serve as a high-level controller of real robots for long-horizon, complex tasks. The success obtained in these tasks motivates us to use LLM as an agent to help solve the compositional language-vision understanding challenges presented in 3D visual grounding.

III. Method

Recently, success stories from Auto-GPT [56], GPT-Engineer [57], and ToolFormer [30] show early signs of success in using LLM as an agent. An agent is different from a traditional model in machine learning in that it has agency: it is an entity that is driven by a goal, reasons about its goal, comes up with plans, examines and uses tools, and interacts with and collects feedback from the environment. In the 3D Visual Grounding setting, an agent can be a promising solution to the "bag-of-words" behavior exhibited by existing models. In LLM-Grounder, we use GPT-4 as the agent and prompt it to complete three tasks: 1. Break down the complex text query into sub-tasks that can be better handled by downstream tools like a CLIP-based 3D visual grounder, such as OpenScene and LERF; 2. Orchestrate and use such tools to solve the sub-tasks it proposes; and 3. Reason on feedback from the environment by incorporating

Training Size	**Open-Vocab**	**Method**	**Visual Grounder**	**+ LLM Agent**	**Acc@0.25 ↑**	**Acc@0.5 ↑**
36k labeled 3D-text data	*closed-vocab*	ScanRefer[12]	–	–	34.4	20.1
		3DVG-Trans[2]	–	–	41.5	28.2
zero-shot	***open-vocab***	LERF[35]	LERF	✗	4.4	0.3
		Ours	LERF	✓ GPT-4	6.9 (+2.5)	1.6 (+1.3)
zero-shot	***open-vocab***	OpenScene[1]	OpenScene	✗	13.0	5.1
		Ours	OpenScene	✓ GPT-3.5	14.3 (+1.3)	4.7 (-0.4)
		Ours	**OpenScene**	**✓ GPT-4**	**17.1 (+4.1)**	**5.3 (+0.2)**

TABLE I: Experiment results on ScanRefer. LLM (GPT-4) agent significantly increases 3D grounding capabilities for zero-shot open-vocabulary 3D grounders such as LERF and OpenScene. We measure grounding capability by Accuracy@0.25 and @0.5, which are accuracies of bounding box predictions whose Intersection-over-Union (IoU) w.r.t. ground-truth box exceeds 0.25 and 0.5, respectively. Numbers in parentheses represent performance gain or loss after adding LLM agent. Results also show that a less powerful LLM, such as GPT-3.5, is not able to achieve strong grounding capability gain. Lastly, although not directly comparable with our method which is ***zero-shot open-vocabulary***, performances are listed for methods that are *trained on ScanRefer and closed-vocabulary* for completeness.

		LERF	OpenScene
Low Visual Difficulty	w/o LLM	10.8	27.6
	w/ LLM	15.1 (+4.3)	33.6 (+6.0)
High Visual Difficulty	w/o LLM	2.5	8.6
	w/ LLM	4.4 (+1.9)	12.1 (+3.5)

TABLE II: Ablation study on visual complexity. LLM agent is more effective for 3D grounding in low visual difficulty settings. Numbers shown are Acc@.25.

spatial understanding and common sense to make grounding decisions.

Planning. The first advantage of LLMs is their ability to plan. Research has shown that chain-of-thought reasoning [27], i.e., explicitly prompting LLM to break complex goals down into smaller sub-tasks ("think step-by-step") can help arithmetic, commonsense, and symbolic reasoning tasks. Inspired by these findings, we design our agent likewise as illustrated in Figure 2. Specifically, we first ask the agent to describe its observation, which gives the agent a chance to summarize the current situation. The context can encompass the human text query and the returned information from tools (described below). The agent then starts a section called reasoning, which serves as a mental scratchpad for the agent to perform high-level planning. Then, in the plan section, the agent must list more specific steps to fulfill the high-level plan, including any tool-using, comparison, or calculation. The agent can reflect on the generated plan in the self-critique section and make any final corrections [50].

Tool-Using. The second advantage of LLMs stems from their ability to use tools. We instruct the LLM agent to use tools to solve the "bag-of-words" behavior (Sec. II). As shown in Fig. 2, we inform LLM of the expected input and output format, i.e., the APIs, of two tools we designed for visual grounding and feedback, and ask the LLM agent to interact with them following the given format. The tools include a Target Finder and a Landmark Finder.

Target Finder and Landmark Finder. The target finder and landmark finder take in a text query input, find bounding boxes of clusters of possible locations for the query, and return a list of candidate bounding boxes in the form of centroids and sizes $(C_x, C_y, C_z, \Delta X, \Delta Y, \Delta Z)$. Target is the main object that a user refers to in a query ("chair" in "a chair between dining table and window"); landmark is the object used to spatially refer to the target ("dining table" and "window"). The target finder additionally computes the volume for each candidate and the landmark finder additionally computes the Euclidean distance from each target candidate's centroid to the landmark's centroid. The volume, distance, and bounding boxes together provide feedback for the LLM agent to conduct spatial and commonsense reasoning. For example, a candidate "chair" with a volume as small as $0.01m^3$ is probably a false positive and should be filtered out; a candidate whose distance to the landmark does not comply with the spatial relation mentioned by the query should be rejected. The target finder and landmark finder are implemented by open-vocabulary CLIP-based 3D visual grounders LERF [35] and OpenScene [1]. These tools alone exhibit "bag-of-words" behaviors (Sec. I) when given complex text queries; however, when given simpler text queries such as a simple noun phrase ("a chair"), such tools can usually work well. The LLM agent capitalizes on this capability of noun-phrase grounding of such 3D visual grounders while compensating for their weaknesses in language understanding and spatial reasoning by decomposing the complex grounding queries, grounding one object at a time, and reasoning about their spatial relation afterward. To use the target finder, we instruct the LLM agent to parse out noun phrases (e.g., "wooden chair") from the original natural language query; to use the landmark finder, we instruct the LLM agent to parse out any landmark objects mentioned in the original query and their spatial relation to the target object.

IV. Experiments

In experiments, we first would like to evaluate how well the LLM-based agent improves zero-shot open-vocabulary 3D visual grounding, compared with CLIP-based 3D visual grounding methods. Then we evaluate our method in the closed-vocabulary setting and compare it with closed-vocabulary and trained approaches. Finally, we show some qualitative examples on in-the-wild scenes, to show the generalization of our approach.

A. Dataset

ScanRefer. ScanRefer [12] is a benchmark on 3D object localization in indoor 3D scenes using natural language. It consists of 51,583 human-written descriptions of 11,046 objects of 18 semantic categories from 800 ScanNet [37] 3D scenes, where the train/val/test split contains 36,665, 9,508 and 5,410 descriptions, respectively. We use the first 14 scenes from the validation split for the experiments presented in Table I, which consists of 998 text-and-3D-object pairs. We also report two standard metrics of ScanRefer: Accuracy@0.25 and Accuracy@0.5. 0.25 and 0.5 are different thresholds for IoU of 3D bounding boxes.

B. Baseline Methods

ScanRefer. ScanRefer [12] uses an end-to-end 3D-text neural architecture to localize objects given a natural language input. Specifically, it processes the 3D point cloud into PointNet++ [58] features, then clusters the points and proposes bounding boxes of objects. The language features are then fused together with the clusters and boxes to decide which boxes are the ones referred to by the language. The pipeline uses supervision from the text and b-box pairs and the ground-truth b-boxes and semantic class for all objects in the scene. We include this baseline as a show of the current trained pipeline's performance, serving as a ceiling compared to our zero-shot setting where no supervision is used.

3DVG-Transformer. 3DVG-Transformer [2] builds on ScanRefer's end-to-end neural architecture and proposes a new neural module to aggregate close-by clusters before proposing bounding boxes. Similar to ScanRefer, 3DVG-Transformer also uses supervision of ground-truth object b-boxes, semantic class, and human-annotated descriptions.

OpenScene and LERF. OpenScene [1] and LERF [35] are zero-shot open-vocabulary 3D scene understanding approaches. OpenScene distills 2D CLIP features into a 3D point cloud and allows grounding with a text query by calculating the cosine similarity between the CLIP text embedding of the query and every point in the 3D point cloud. LERF achieves the same by encoding CLIP embeddings into a neural radiance field, These methods, when used alone, exhibit "bag-of-words" behavior as illustrated in 1, a problem we aim to address with LLM agent deliberative reasoning. To produce bounding boxes using OpenScene and LERF for the 3D visual grounding benchmark ScanRefer, we apply DBSCAN clustering [59] on points with high cosine similarity and draw bounding boxes around them.

C. Results

We first show qualitative results of LLM-Grounder in Fig. 3. More results and demonstrations can be found on the project website[1], including in-the-wild scenes.

Compared with baselines, we find *the LLM agent can improve zero-shot, open vocabulary grounding.* As shown in Table I, the addition of an LLM agent can significantly increase the grounding performance of both LERF and OpenScene by achieving 5.0% and 17.1% on Accuracy@0.25, respectively. We attribute the lower increase in performance for LERF to the weaker overall grounding capability of LERF. The lower increase suggests that when the tool provides too noisy of a feedback to an LLM agent, it is hard for the LLM agent to reason with the noisy input and improve performance. We also note the low increase in performance on Accuracy@0.5, which requires the predicted b-box to have more than 50% overlaps with the ground-truth box. We attribute this to the lack of instance segmentation capability of the underlying grounder. We observe that the grounders often predict too large or too small of a bounding for the correctly grounded object. Such prediction is not correctable by an LLM thus causing the difficulty of precise visual grounding and the low performance increase. Additionally, we find that when using GPT-3.5 as the agent for OpenScene, the performance drops compared to without GPT. We attribute this to the weaker tool-using and spatial and commonsense reasoning capability of GPT-3.5.

D. Ablation Study

We then evaluate what the LLM-agent primarily improves on. We test two different settings: (1) does the LLM-agent help more with a more difficult visual context? (2) does the LLM-agent help more for more difficult text queries?

Difficulty of visual context. We categorize the results by vision difficulties in Table II and find that LLM agent is more effective for low vision difficulty queries, evidenced by the higher grounding performance increase. Specifically, we separate the grounding queries into *Low Visual Difficulty* and *High Visual Difficulty* categories. A query has low visual difficulty if the object mentioned in the text query is the sole object of that class in a scene (0 distractor); a query has high visual difficulty if there are more than 1 distractor object of the same class in a scene. Out of the 998 queries we evaluated, 232 queries had low visual difficulty, and 766 queries had high visual difficulty. Results in Table II show that LLM brings more performance increase for the low visual difficulty queries. This behavior can be explained by the different challenges presented in low- and high-visual-difficulty settings. In low visual difficulty settings, the main challenge an open-vocabulary 3D grounder faces is the "bag-of-words" behavior. For example, if the text query is "the sink in the kitchen" and if there is only one sink in the scene, a bag-of-words grounder would highlight the whole kitchen, leading to low grounding precision. An LLM agent is particularly good at solving this problem by

[1] `https://chat-with-nerf.github.io/`

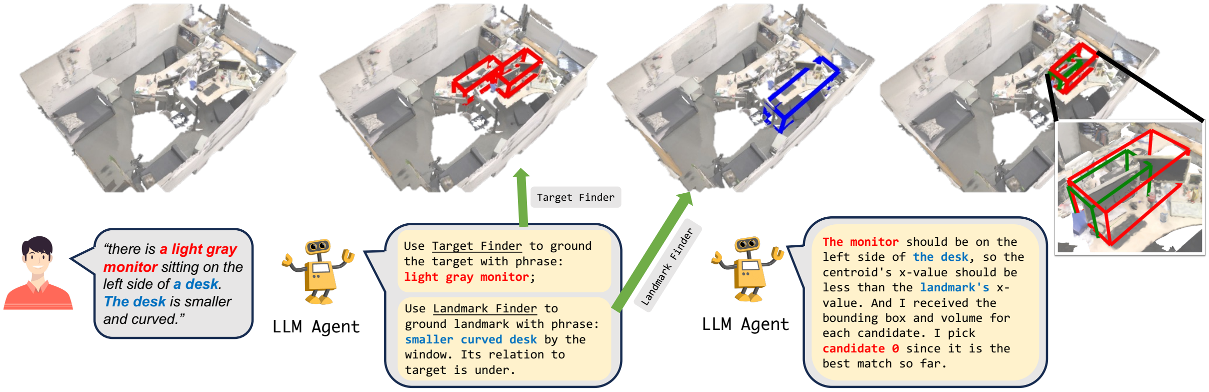

"there is a light gray monitor sitting on the left side of a desk. The desk is smaller and curved."
LLM Agent
Use Target Finder to ground the target with phrase: light gray monitor;
Use Landmark Finder to ground landmark with phrase: smaller curved desk by the window. Its relation to target is under.
Target Finder
Landmark Finder
LLM Agent
The monitor should be on the left side of the desk, so the centroid's x-value should be less than the landmark's x-value. And I received the bounding box and volume for each candidate. I pick candidate 0 since it is the best match so far.

Fig. 3: Qualitative example. LLM agent uses spatial reasoning to successfully disambiguate the correct object instance.

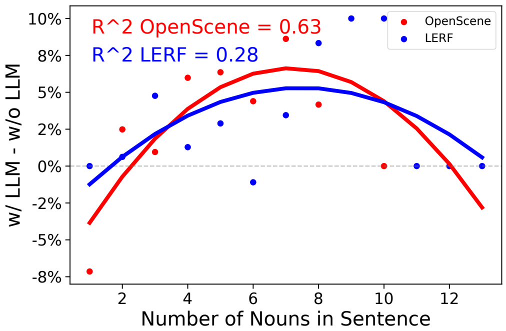


Fig. 4: Performance delta (w/ LLM - w/o LLM) vs. query text complexity. The LLM helps more when the text query is more complex but fails to help significantly at higher complexities.

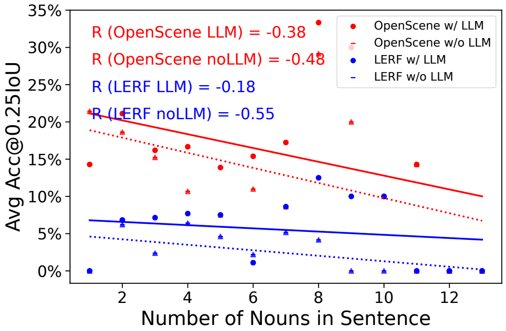


Fig. 5: Performance of various models vs query text complexity. All models struggle with more complex sentences, but models with an LLM agent perform better, especially at these higher complexities.

parsing out the target object "sink" and only issuing this single noun to the grounder, thus circumventing the bag-of-words behavior. For high visual difficulty settings, however, there is one additional challenge: *instance disambiguation*. Because there are multiple instances of the same class in the scene, the visual grounder would return many candidates to the LLM agent. The LLM agent could use its spatial and commonsense reasoning capability to filter out some instances with volume and distance to landmark information, but more complex instance disambiguation usually requires more nuanced visual cues, a privilege an LLM agent doesn't have because it is blind.

Difficulty of text queries. As queries become more complex, the LLM-agent will help performance, but only up to a certain point. We can measure query complexity by counting the number of nouns in the sentence: the more nouns in a description, the more difficult it will be to ground any specific object. We see from Fig. 5 that, both with and without the help of an LLM agent, performance decreases as sentence complexity increases. However, from analyzing the performance difference between using an LLM agent and not using one, we see that there is a quadratic dependence on query complexity (Fig. 4). This suggests that the LLM provides an advantage for grounding when presented with higher-complexity queries, but after reaching some threshold, the performance advantage diminishes. When query complexity is low, models without an LLM can ground objects effectively, so LLMs provide minimal advantage. As complexity increases, baseline models perform worse and LLMs provide a more significant advantage. However, with increased complexity of referential expression, LLM's spatial reasoning capability may not surpass the performance of no-LLM baselines. We may require stronger LLMs to produce advantages in these higher complexity ranges.

V. Conclusion and Limitations

We introduced LLM-Grounder, a novel approach for 3D visual grounding that leverages Large Language Models (LLMs) as the central agent for orchestrating the grounding process. Our empirical evaluations demonstrate that LLM-Grounder excels particularly in handling complex text queries, thereby offering a robust, zero-shot, open-vocabulary solution for 3D visual grounding tasks. However, there are some limitations to consider. Cost: Utilizing GPT-based models as the core reasoning agent can be computationally expensive, which may limit its deployment in resource-constrained environments. Latency: The reasoning process, due to the inherent latency of GPT models, can be slow. This latency could be a significant bottleneck for real-time robotic applications where rapid decision-making is crucial. Despite these limitations, LLM-Grounder sets a new benchmark in 3D visual grounding and opens up avenues for future research in integrating LLMs with robotic systems.

References

[1] S. Peng, K. Genova, C. M. Jiang, A. Tagliasacchi, M. Pollefeys, and T. Funkhouser, "Openscene: 3d scene understanding with open vocabularies," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[2] L. Zhao, D. Cai, L. Sheng, and D. Xu, "3DVG-Transformer: Relation modeling for visual grounding on point clouds," in *ICCV*, 2021, pp. 2928–2937.

[3] J. Roh, K. Desingh, A. Farhadi, and D. Fox, "Languagerefer: Spatial-language model for 3d visual grounding," in *Conference on Robot Learning*. PMLR, 2022, pp. 1046–1056.

[4] D. Cai, L. Zhao, J. Zhang, L. Sheng, and D. Xu, "3djcg: A unified framework for joint dense captioning and visual grounding on 3d point clouds," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 16 464–16 473.

[5] J. Chen, W. Luo, R. Song, X. Wei, L. Ma, and W. Zhang, "Learning point-language hierarchical alignment for 3d visual grounding," 2022.

[6] D. Z. Chen, Q. Wu, M. Nießner, and A. X. Chang, "D3net: A speaker-listener architecture for semi-supervised dense captioning and visual grounding in rgb-d scans," 2021.

[7] Z. Yuan, X. Yan, Y. Liao, R. Zhang, S. Wang, Z. Li, and S. Cui, "Instancerefer: Cooperative holistic understanding for visual grounding on point clouds through instance multi-level contextual referring," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 1791–1800.

[8] E. Bakr, Y. Alsaedy, and M. Elhoseiny, "Look around and refer: 2d synthetic semantics knowledge distillation for 3d visual grounding," *Advances in Neural Information Processing Systems*, vol. 35, pp. 37 146–37 158, 2022.

[9] H. Liu, A. Lin, X. Han, L. Yang, Y. Yu, and S. Cui, "Refer-it-in-rgbd: A bottom-up approach for 3d visual grounding in rgbd images," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 6032–6041.

[10] A. Jain, N. Gkanatsios, I. Mediratta, and K. Fragkiadaki, "Bottom up top down detection transformers for language grounding in images and point clouds," in *European Conference on Computer Vision*. Springer, 2022, pp. 417–433.

[11] S. Huang, Y. Chen, J. Jia, and L. Wang, "Multi-view transformer for 3d visual grounding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 15 524–15 533.

[12] D. Z. Chen, A. X. Chang, and M. Nießner, "Scanrefer: 3d object localization in rgb-d scans using natural language," *16th European Conference on Computer Vision (ECCV)*, 2020.

[13] P. Achlioptas, A. Abdelreheem, F. Xia, M. Elhoseiny, and L. J. Guibas, "ReferIt3D: Neural listeners for fine-grained 3d object identification in real-world scenes," in *16th European Conference on Computer Vision (ECCV)*, 2020.

[14] B. Chen, F. Xia, B. Ichter, K. Rao, K. Gopalakrishnan, M. S. Ryoo, A. Stone, and D. Kappler, "Open-vocabulary queryable scene representations for real world planning," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 11 509–11 522.

[15] R. Ding, J. Yang, C. Xue, W. Zhang, S. Bai, and X. Qi, "Pla: Language-driven open-vocabulary 3d scene understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 7010–7019.

[16] S. Y. Gadre, M. Wortsman, G. Ilharco, L. Schmidt, and S. Song, "Cows on pasture: Baselines and benchmarks for language-driven zero-shot object navigation," *CVPR*, 2023.

[17] C. Huang, O. Mees, A. Zeng, and W. Burgard, "Visual language maps for robot navigation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 10 608–10 615.

[18] K. M. Jatavallabhula, A. Kuwajerwala, Q. Gu, M. Omama, T. Chen, A. Maalouf, S. Li, G. S. Iyer, S. Saryazdi, N. V. Keetha *et al.*, "Conceptfusion: Open-set multimodal 3d mapping," in *ICRA2023 Workshop on Pretraining for Robotics (PT4R)*, 2023.

[19] K. Mazur, E. Sucar, and A. J. Davison, "Feature-realistic neural fusion for real-time, open set scene understanding," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 8201–8207.

[20] N. M. M. Shafiullah, C. Paxton, L. Pinto, S. Chintala, and A. Szlam, "Clip-fields: Weakly supervised semantic fields for robotic memory," in *ICRA2023 Workshop on Pretraining for Robotics (PT4R)*, 2023.

[21] A. Takmaz, E. Fedele, R. W. Sumner, M. Pollefeys, F. Tombari, and F. Engelmann, "OpenMask3D: Open-Vocabulary 3D Instance Segmentation," *arXiv preprint arXiv:2306.13631*, 2023.

[22] H. Ha and S. Song, "Semantic abstraction: Open-world 3d scene understanding from 2d vision-language models," in *6th Annual Conference on Robot Learning*, 2022. [Online]. Available: https://openreview.net/forum?id=lV-rNbXVSaO

[23] Y. Hong, C. Lin, Y. Du, Z. Chen, J. B. Tenenbaum, and C. Gan, "3d concept learning and reasoning from multi-view images," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023.

[24] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.

[25] M. Yuksekgonul, F. Bianchi, P. Kalluri, D. Jurafsky, and J. Zou, "When and why vision-language models behave like bags-of-words, and what to do about it?" in *The Eleventh International Conference on Learning Representations*, 2022.

[26] OpenAI, "Gpt-4 technical report," 2023.

[27] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in Neural Information Processing Systems*, vol. 35, pp. 24 824–24 837, 2022.

[28] S. Yao, D. Yu, J. Zhao, I. Shafran, T. L. Griffiths, Y. Cao, and K. Narasimhan, "Tree of thoughts: Deliberate problem solving with large language models," *arXiv preprint arXiv:2305.10601*, 2023.

[29] A. Zeng, M. Attarian, K. M. Choromanski, A. Wong, S. Welker, F. Tombari, A. Purohit, M. S. Ryoo, V. Sindhwani, J. Lee *et al.*, "Socratic models: Composing zero-shot multimodal reasoning with language," in *The Eleventh International Conference on Learning Representations*, 2022.

[30] T. Schick, J. Dwivedi-Yu, R. Dessì, R. Raileanu, M. Lomeli, L. Zettlemoyer, N. Cancedda, and T. Scialom, "Toolformer: Language models can teach themselves to use tools," 2023.

[31] M. Ahn, A. Brohan, N. Brown, Y. Chebotar, O. Cortes, B. David, C. Finn, C. Fu, K. Gopalakrishnan, K. Hausman *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[32] W. Huang, F. Xia, T. Xiao, H. Chan, J. Liang, P. Florence, A. Zeng, J. Tompson, I. Mordatch, Y. Chebotar *et al.*, "Inner monologue: Embodied reasoning through planning with language models," in *Conference on Robot Learning*. PMLR, 2023, pp. 1769–1782.

[33] J. Liang, W. Huang, F. Xia, P. Xu, K. Hausman, B. Ichter, P. Florence, and A. Zeng, "Code as policies: Language model programs for embodied control," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 9493–9500.

[34] D. Shah, B. Osinski, B. Ichter, and S. Levine, "LM-nav: Robotic navigation with large pre-trained models of language, vision, and action," in *6th Annual Conference on Robot Learning*, 2022. [Online]. Available: https://openreview.net/forum?id=UW5A3SweAH

[35] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik, "Lerf: Language embedded radiance fields," in *International Conference on Computer Vision (ICCV)*, 2023.

[36] J. Hsu, J. Mao, and J. Wu, "Ns3d: Neuro-symbolic grounding of 3d objects and relations," *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2614–2623, 2023. [Online]. Available: https://api.semanticscholar.org/CorpusID:257687234

[37] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[38] B. Li, K. Q. Weinberger, S. Belongie, V. Koltun, and R. Ranftl, "Language-driven semantic segmentation," in *International Conference on Learning Representations*, 2022. [Online]. Available: https://openreview.net/forum?id=RriDjddCLN

[39] G. Ghiasi, X. Gu, Y. Cui, and T.-Y. Lin, "Scaling open-vocabulary image segmentation with image-level labels," in *European Conference on Computer Vision*. Springer, 2022, pp. 540–557.

[40] F. Liang, B. Wu, X. Dai, K. Li, Y. Zhao, H. Zhang, P. Zhang, P. Vajda, and D. Marculescu, "Open-vocabulary semantic segmentation with mask-adapted clip," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 7061–7070.

[41] S. Kobayashi, E. Matsumoto, and V. Sitzmann, “Decomposing nerf for editing via feature field distillation,” in *Advances in Neural Information Processing Systems*, vol. 35, 2022. [Online]. Available: https://arxiv.org/pdf/2205.15585.pdf

[42] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, “Language models are few-shot learners,” *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[43] J. Wei, M. Bosma, V. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le, “Finetuned language models are zero-shot learners,” in *International Conference on Learning Representations*, 2021.

[44] L. Ouyang, J. Wu, X. Jiang, D. Almeida, C. Wainwright, P. Mishkin, C. Zhang, S. Agarwal, K. Slama, A. Ray *et al.*, “Training language models to follow instructions with human feedback,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 27 730–27 744, 2022.

[45] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, “Exploring the limits of transfer learning with a unified text-to-text transformer,” *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 5485–5551, 2020.

[46] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar *et al.*, “Llama: Open and efficient foundation language models,” *arXiv preprint arXiv:2302.13971*, 2023.

[47] S. Yao, J. Zhao, D. Yu, N. Du, I. Shafran, K. Narasimhan, and Y. Cao, “React: Synergizing reasoning and acting in language models,” *arXiv preprint arXiv:2210.03629*, 2022.

[48] N. Shinn, F. Cassano, B. Labash, A. Gopinath, K. Narasimhan, and S. Yao, “Reflexion: Language agents with verbal reinforcement learning,” *arXiv preprint arXiv:2303.11366*, 2023.

[49] H. Liu, C. Sferrazza, and P. Abbeel, “Chain of hindsight aligns language models with feedback,” *arXiv preprint arXiv:2302.02676*, vol. 3, 2023.

[50] E. Jang, “Can llms critique and iterate on their own outputs?” *evjang.com*, Mar 2023. [Online]. Available: https://evjang.com/2023/03/26/self-reflection.html

[51] E. Karpas, O. Abend, Y. Belinkov, B. Lenz, O. Lieber, N. Ratner, Y. Shoham, H. Bata, Y. Levine, K. Leyton-Brown *et al.*, “Mrkl systems: A modular, neuro-symbolic architecture that combines large language models, external knowledge sources and discrete reasoning,” *arXiv preprint arXiv:2205.00445*, 2022.

[52] A. Parisi, Y. Zhao, and N. Fiedel, “Talm: Tool augmented language models,” *arXiv preprint arXiv:2205.12255*, 2022.

[53] H. Chase, “LangChain,” Oct. 2022. [Online]. Available: https://github.com/hwchase17/langchain

[54] Y. Shen, K. Song, X. Tan, D. Li, W. Lu, and Y. Zhuang, “Hugginggpt: Solving ai tasks with chatgpt and its friends in huggingface,” *arXiv preprint arXiv:2303.17580*, 2023.

[55] M. Li, F. Song, B. Yu, H. Yu, Z. Li, F. Huang, and Y. Li, “Api-bank: A benchmark for tool-augmented llms,” *arXiv preprint arXiv:2304.08244*, 2023.

[56] “Auto-gpt,” https://github.com/Significant-Gravitas/Auto-GPT, 2013.

[57] “Gpt-engineer,” https://github.com/AntonOsika/gpt-engineer, 2013.

[58] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, “Pointnet++: Deep hierarchical feature learning on point sets in a metric space,” *Advances in neural information processing systems*, vol. 30, 2017.

[59] M. Ester, H.-P. Kriegel, J. Sander, X. Xu *et al.*, “A density-based algorithm for discovering clusters in large spatial databases with noise,” in *kdd*, vol. 96, no. 34, 1996, pp. 226–231.