# Tool-Augmented VLM Agents for Zero-Shot 3D Visual Grounding on Point Clouds

Cuong Huynh[1], Maxim Popov[1], Denis Gridusov[1] and Sergey Kolyubin[1]

***Abstract*— 3D Visual Grounding (3DVG) is an essential capability for embodied AI, requiring agents to localize objects in 3D scenes based on natural language descriptions. Recent zero-shot methods leverage 2D vision-language models (LVLMs). However, they often rely on existing sets of multi-view images and struggle with the limited semantic and spatial details provided by standard 3D segmentation tools. We present *AgentGrounder*, a zero-shot 3D visual grounding framework that operates directly on colored point clouds without task-specific 3D training. Our approach follows a two-stage design: (1) an offline stage that applies 3D model to build an Object Lookup Table (OLT) with instance IDs, semantic labels, 3D bounding boxes; and (2) an online tool-driven agent that decomposes each query, retrieves only relevant candidates from the OLT, performs geometric scoring, and triggers image rendering on demand when additional visual evidence (e.g., color, material, or viewpoint-sensitive cues) is required. Compared with fixed anchor-target matching pipelines, this design reduces cascading matching errors and avoids prompts filled with irrelevant objects. We evaluate on ScanRefer and Nr3D under a zero-shot setting and observe consistent improvements over SeeGround in our setup, including +2.5% Acc@0.5 on ScanRefer and +6.3% on Nr3D, with a notable +6.3% gain on Nr3D view-independent queries. These results show that combining selective retrieval, geometric reasoning, and adaptive visual inspection yields a practical and robust foundation for open-vocabulary 3D grounding.**

## I. Introduction

3D Visual Grounding (3DVG) is a fundamental task in computer vision and robotics, aimed at localizing a target object within a 3D scene based on a natural language query. This capability is essential for enabling autonomous agents to interact seamlessly with their environment — for instance, following a command to "`pick up the white porcelain sink next to the counter`." Most existing 3DVG methods [1], [2], [3], [4] follow a supervised learning paradigm, requiring large-scale datasets with dense annotations of 3D bounding boxes paired with linguistic descriptions, which is labor-intensive and limits generalization to unseen categories. To address these limitations, zero-shot 3DVG [5], [6], [7], [8], [9], [10], [11] has emerged as a promising direction, leveraging the rich knowledge of Vision-Language Models (VLMs) and Large Language Models (LLMs) without task-specific 3D training. This line of work is further supported by progress in open-vocabulary 3D segmentation [12], [13], [14], [15], [16] and multimodal spatial reasoning [17], [18], [19], [20], which project 2D foundation knowledge into 3D representations. Despite encouraging progress, existing untrained data processing pipelines still face key challenges: (1) logical reasoning is often unreliable due to early anchor-target matching errors, (2) visual analysis is not always selective, leading to unnecessary computations and token overhead, and (3) geometric relationships in 3D scenes are not always exploited in a consistently transparent and deterministic manner.

In this paper, we propose ***AgentGrounder***, a zero-shot 3DVG framework that operates directly on colored 3D point clouds with a tool-driven agent. Our method follows a two-stage design: first, we run a 3D model to obtain instance-level segmentation and build an Object Lookup Table (OLT) containing object IDs, semantic labels, centers, and box sizes. Second, an online LVLM agent performs query decomposition, retrieves only relevant objects from the OLT, applies deterministic geometric scoring, and calls image rendering on demand when a visual prompt (e.g., color, material, or viewpoint-sensitive cues) is required. Compared with fixed matching pipelines, this design reduces cascading matching errors and avoids prompts filled with irrelevant objects. We validated our method on ScanRefer [21] and Nr3D [22], achieving consistent gains over SeeGround in our setting and showing state-of-the-art results.

## II. Methodology

**Overview.** We studied 3D visual grounding where, given a language query $Q$ and a scene $S$, the system predicts the target object and its 3D box. The task is formulated as:

$$bbox = 3DVG(S, Q) \tag{1}$$

where $bbox \in \mathbb{R}^6$ encodes the 3D box center and size, i.e. $bbox = (c, d)$, $c \in \mathbb{R}^3$ is the object center, and $d \in \mathbb{R}^3$ is the box size.

### A. Initial 3D Instance Segmentation

As the first stage, we ran a 3D instance segmentation network on scene $S$ to obtain per-object masks and initial semantic labels:

$$\{(mask_i, sem_i)\}_{i=1}^{N} = Seg(S). \tag{2}$$

For each predicted mask, we computed the corresponding 3D bounding box:

$$bbox_i = Bound(mask_i). \tag{3}$$

This stage was executed offline once per scene and provided the object candidates used in subsequent online grounding.

[1] Biomechatronics and Energy-Efficient Robotics (BE2R) Lab, ITMO University, Saint Petersburg, Russia

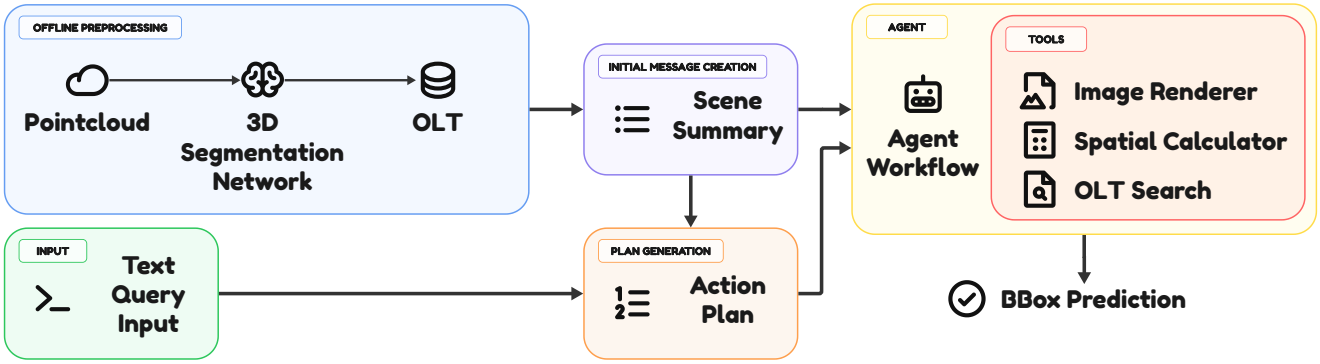


Fig. 1: **Overview of the *AgentGrounder* agent pipeline**. In the offline phase, a 3D segmentation network built an Object Lookup Table. For each scene, the system then generated a textual description that listed all detected objects. In the online phase, the agent received a query and first formulated an explicit plan. Guided by this plan, it retrieved candidate objects based on semantic labels from the scene metadata and computed geometric relationships (such as nearest/farthest, left/right, below) using 3D object centers and bounding box dimensions. For view-dependent queries, the agent invoked a rendering tool to examine selected object IDs and resolved ambiguities. Finally, it output a structured response that included the predicted object ID along with a textual justification.

### *B. Object Lookup Table*

At next stage, each scene was represented as an Object Lookup Table (OLT) constructed from the segmentation outputs:

$$\mathcal{O} = \{(id_i, \ell_i, bbox_i)\}_{i=1}^{N}, \tag{4}$$

where $\ell_i$ is the semantic label. During online inference, the agent queries this table using tool calls (e.g., by label) and obtains pre-calculated 3D segmentation results.

### *C. Agent Planning and Query Decomposition*

Given a language query $Q$, the agent first generated an explicit plan $\mathcal{R}$ that dictated (i) which tools to call and (ii) how to make decisions from the returned metadata. Concretely, it decomposed $Q$ into the target object category (head noun), descriptive attributes (e.g., color/material), spatial expressions (e.g., `left of`, `closest to`), and any anchor objects mentioned for disambiguation:

$$(\mathcal{P}, \mathcal{L}, \mathcal{R}) = PlanExtract(Q), \tag{5}$$

where $\mathcal{L}$ is the set of candidate labels and $\mathcal{R}$ contains spatial predicates (e.g., `next to`, `left of`, `below`, `closest to`). Plan $\mathcal{P}$ guides subsequent tool calls and reasoning steps, allowing the agent to adapt its strategy based on query complexity and content. To prevent unbounded or recursive planning, we limit the maximum plan length. Without explicit planning, the agent must rely on implicit reasoning within a single prompt, which can lead to less structured and more error-prone inference, especially for complex queries with multiple spatial relations or attributes.

### *D. Candidate Retrieval and Geometric Scoring*

The agent retrieved candidates $\mathcal{C}$ by labels from $\mathcal{O}$:

$$\mathcal{C} = \{o_i \in \mathcal{O} \mid \ell_i \in \mathcal{L}\}. \tag{6}$$

To answer spatial queries, the agent evaluated the predicates in $\mathcal{R}$ using simple geometry derived from the OLT: (i) distances $d(i, j)$ for proximity relations, (ii) coordinate comparisons for directional relations (left/right, above/below) in the selected reference frame, and (iii) box dimensions as proxies for size constraints (e.g., area/volume). For distance-related expressions (e.g., "`closest to the TV`") it selected the candidate with the smallest or largest distance to the referenced object. For between relations (e.g., "`between two tables`"), it picked candidates that were close to both referenced objects:

$$\begin{aligned} d(i, j) &= \|c_i - c_j\|_2, \\ \hat{o} &= Resolve(\mathcal{C}, \mathcal{R}, \mathcal{O}, d(i, j)). \end{aligned} \tag{7}$$

### *E. View-Dependent Disambiguation by Rendering*

If the query contained viewpoint-sensitive language (e.g., `when facing`, `on the right`), the agent called a rendering tool over a subset of candidate IDs and used visual evidence to resolve ambiguity:

$$I = Render(S, \mathcal{I}_{cand}), \qquad \hat{o} = Resolve(I, Q). \tag{8}$$

The camera pose for rendering is computed such that the camera looks at the mean 3D center of the relevant objects from OLT. Object relevancy is defined inside the agent without hard-coded assumptions.

For non-view-dependent queries with clear geometric ranking, this step was skipped.

### *F. Robust Handling of Label Mismatch*

When the query $Q$ referred to an object, the agent first used tool calls to retrieve candidates from the Object Lookup Table (OLT) by semantic label. If the label lookup returned no results (i.e., the mentioned object/label was not present in the OLT; e.g., `board`, `fridge`), the agent mapped that term to the *closest* available object/label in the OLT

based on context (e.g., `tv`, `kitchen cabinet`). After this mapping, the agent continued the same geometric reasoning pipeline on the new candidate set. This simple fallback increased query coverage without any retraining.

### G. Final Prediction

The final output was a structured triplet containing object ID, bounding box, and explanation:

$$\hat{bbox} = BBox(\hat{o}), \qquad \text{answer} = (\hat{o}, \hat{bbox}, \text{rationale}). \quad (9)$$

This formulation captures the agent behavior during inference: metadata retrieval, geometric reasoning, selective image calls for view dependence, and structured final reporting.

## III. Experiments

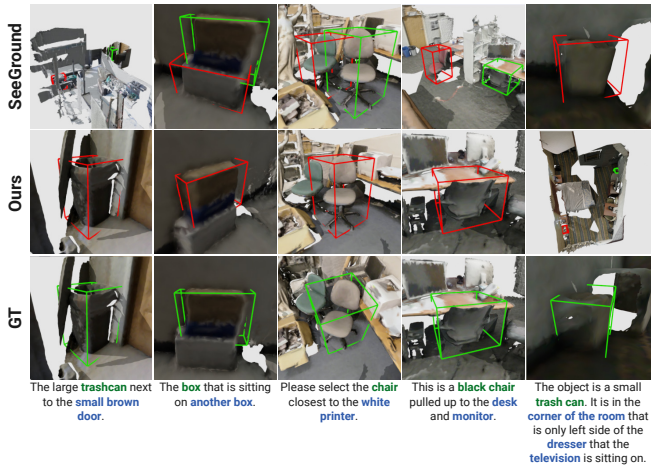


Fig. 2: Results comparison between SeeGround [8] and ***AgentGrounder*** (Ours).

### A. Implementation Details

Our framework followed a two-stage pipeline. In the offline preprocessing stage, we employed Mask3D [28] for 3D instance segmentation to construct an OLT for each scene. Although Mask3D [28] was trained on ScanNet [29], it was not trained for the 3D visual grounding task, which allows us to treat our method as operating in a zero-shot setting.

For the online reasoning stage, we used Qwen3-VL-32B-Instruct [30] as the core Vision-Language Model (VLM). The model was deployed locally using the Ollama framework [31] and interfaced through the LangChain API [32].

### B. Experimental Settings

We evaluated our proposed 3DVG approach on two commonly-used benchmark datasets: ScanRefer [21] and Nr3D [22]. ScanRefer provides natural language descriptions for objects in ScanNet scenes, requiring the model to distinguish target objects based on spatial context and geometric features. Queries are categorized as *Unique* or *Multiple* based on the presence of same-class distractors. Nr3D, part of the ReferIt3D benchmark, contains descriptions collected via a reference game, classified into *Easy* or *Hard* levels and further divided by viewpoint dependency (*View-Dependent* or *View-Independent*). To ensure efficient evaluation while maintaining representativeness, we evaluated on the full validation set of Nr3D and drew a 1,000-query subsample from ScanRefer using stratified random sampling to preserve the *Unique* / *Multiple* split. We followed the evaluation protocol established in ZSVG3D [6], using accuracy with IoU thresholds as the primary metric. The results of the methods were directly taken from original published papers. Our experiments were performed on a workstation equipped with two NVIDIA RTX 4090 GPUs, each providing 24 GB of VRAM. It took approximately one day to evaluate the entire Nr3D dataset, comprising 6,307 queries, with an average inference time of 13.5 seconds per query.

### C. Comparative Study

Table I compares our proposed ***AgentGrounder*** with other 3DVG methods on the ScanRefer validation set. Our method achieved an overall Acc@0.5 of 41.9, outperforming SeeGround by 2.5% in this metric. Breaking down by subset: in the *Unique* category, we achieved Acc@0.5 of 73.7 compared to SeeGround's 68.9 (4.8% gain), demonstrating stronger spatial precision under stricter IoU thresholds. In the *Multiple* subset with same-class distractors, ***AgentGrounder*** obtained 31.7 Acc@0.5 versus SeeGround's 30.0, showing consistent improvements even in cluttered scenes requiring fine-grained reasoning. We also outperform CSVG on Acc@0.5 across both the *Unique* and *Multiple* subsets. The *Overall* metric, however, appears lower for our method because CSVG's evaluation subsample does not preserve the *Unique*/*Multiple* distribution of the full validation set, which inflates their aggregate score.

**Nr3D.** Table II presents results on the Nr3D dataset, where ***AgentGrounder*** achieved an overall accuracy of 52.4, a 6.3% improvement over SeeGround (46.1). Gains are consistent across difficulty and viewpoint splits: on *Hard* queries, we reach 45.4 versus 38.3 (+7.1%), indicating stronger robustness to multiple distractors, while on *Easy* queries we improve to 59.6 over 54.5 (+5.1%), suggesting better precision even with fewer confounders. For viewpoint splits, ***AgentGrounder*** achieves 54.5 on *View-Independent* compared to 48.2 (+6.3%) and 47.9 on *View-Dependent* versus 42.3 (+5.6%), showing improvements in both geometry-driven and appearance-sensitive cases. Several examples of failure cases from SeeGround [8] are also shown in Figure 2. All qualitative failure analyses are conducted on Nr3D.

### D. Ablation Study

We studied how each tool contributes to 3DVG performance on a held-out Nr3D subset ($N = 120$), stratified by *Easy/Hard* and *View-Dependent/View-Independent*, using Acc@0.5.

Table III incrementally enables four tools: (i) *Label-based OLT Retrieval*, (ii) *Geometric Distance Reasoning*, (iii) *Explicit Planning*, and (iv) *On-demand Visual Rendering*. Starting from retrieval-only (40.0%), geometric reasoning

TABLE I: Evaluation results of 3DVG methods on *ScanRefer* [21] validation set. Results are reported for *"Unique"* (scenes with a single target object) and *"Multiple"* (scenes with distractors of the same class) subsets, along with overall performance.

| Method | Venue | Zero-Shot | Agent | Unique | | Multiple | | Overall | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Acc@0.25 | Acc@0.5 | Acc@0.25 | Acc@0.5 | Acc@0.25 | Acc@0.5 |
| G3-LQ [1] | CVPR'24 | ✗ | - | 88.6 | 73.3 | 50.2 | 39.7 | 56.0 | 44.7 |
| MCLN [2] | ECCV'24 | ✗ | - | 86.9 | 72.7 | 52.0 | 40.8 | 57.2 | 45.7 |
| ConcreteNet [23] | ECCV'24 | ✗ | - | 86.4 | 82.1 | 42.4 | 38.4 | 50.6 | 46.5 |
| Chat-Scene [3] | NIPS'24 | ✗ | Vicuna-7B | 89.6 | 82.5 | 47.8 | 42.9 | 55.5 | 50.2 |
| Video-3D LLM [4] | CVPR'25 | ✗ | LLaVA-Video 7B | 88.0 | 78.3 | 50.9 | 45.3 | 58.1 | 51.7 |
| GPT4Scene [24] | ICLR'26 | ✗ | Qwen2-VL-7B | 90.3 | 83.7 | 56.4 | 50.9 | 62.6 | 57.0 |
| LLM-G [5] | ICRA'24 | ✓ | GPT-3.5 | - | - | - | - | 14.3 | 4.7 |
| LLM-G [5] | ICRA'24 | ✓ | GPT-4 turbo | - | - | - | - | 17.1 | 5.3 |
| ZSVG3D [6] | CVPR'24 | ✓ | GPT-4 turbo | 63.8 | 58.4 | 27.7 | 24.6 | 36.4 | 32.7 |
| SeeGround [8] | CVPR'25 | ✓ | Qwen2-VL-72B | 75.7 | 68.9 | 34.0 | 30.0 | 44.1 | 39.4 |
| CSVG [9] | BMVC'25 | ✓ | Mistral-Large-240B | 68.8 | 61.1 | **38.4** | 27.3 | **49.6** | 39.8 |
| ***AgentGrounder*** (Ours) | - | ✓ | Qwen3-VL-32B | **80.3** | **73.7** | 36.6 | **31.7** | 47.2 | **41.9** |

TABLE II: Performance results on *Nr3D* [22] validation set. Queries are labeled as *"Easy"* (only one distractor) or *"Hard"* (with multiple distractors), and as *"View-Dependent"* or *"View-Independent"* based on viewpoint requirements for grounding.

| Method | Easy | Hard | Dep. | Indep. | Overall |
|---|---|---|---|---|---|
| **Supervision: Fully Supervised** | | | | | |
| MiKASA [25] | 69.7 | 59.4 | 65.4 | 64.0 | 64.4 |
| ViL3DRel [26] | 70.2 | 57.4 | 62.0 | 64.5 | 64.4 |
| SceneVerse [27] | 72.5 | 57.8 | 56.9 | 67.9 | 64.9 |
| **Supervision: Zero-Shot** | | | | | |
| ZSVG3D [6] | 46.5 | 31.7 | 36.8 | 40.0 | 39.0 |
| VLM-Grounder [7] | 55.2 | 39.5 | 45.8 | 49.4 | 48.0 |
| SeeGround [8] | 54.5 | 38.3 | 42.3 | 48.2 | 46.1 |
| ***AgentGrounder*** (Ours) | **59.6** | **45.4** | **47.9** | **54.5** | **52.4** |

TABLE III: Ablation study of agent tools on a held-out Nr3D subset ($N = 120$), measured by $Acc@0.5$.

| # | Retrieval | Distance | Planning | Rendering | Easy | Hard | Dep | Indep | Overall |
|---|---|---|---|---|---|---|---|---|---|
| 1 | ✓ | ✗ | ✗ | ✗ | 51.7 | 29.0 | 35.7 | 42.3 | 40.0 |
| 2 | ✓ | ✓ | ✗ | ✗ | 50.0 | **40.3** | 35.7 | 50.0 | 45.0 |
| 3 | ✓ | ✓ | ✓ | ✗ | 58.6 | 35.5 | 38.1 | **51.3** | 46.7 |
| 4 | ✓ | ✓ | ✗ | ✓ | 62.1 | 33.9 | **50.0** | 46.2 | 47.5 |
| 5 | ✓ | ✓ | ✓ | ✓ | **65.5** | 35.5 | 47.6 | **51.3** | **50.0** |

boosts performance to 45.0%, highlighting the importance of deterministic spatial scoring. Adding planning yields a modest gain (46.7%), while on-demand rendering further improves results (47.5%), showing the value of visual cues when geometry is insufficient. The full toolset achieves the best accuracy (50.0%), indicating that these components are complementary.

### *E. Discussion*

The performance superiority of ***AgentGrounder*** can be attributed to two key factors working in concert.

*(1) Tool-driven reasoning with deterministic geometry:* Unlike fixed anchor-target decomposition and learned reranking, our agent adaptively selects tools and ranks candidates using explicit geometric relations from OLT metadata (centers, box sizes). This enables stable spatial reasoning on ScanRefer *Unique* and Nr3D *View-Independent*, where viewpoint-agnostic cues dominate.

*(2) Selective rendering and efficient context use:* The agent renders images only when visual cues are needed and otherwise keeps prompts compact by retrieving only relevant objects. In contrast, SeeGround [8] uses broader context, increasing token overhead and matching errors. This targeted prompting improves performance on *Hard* and *Multiple* subsets.

A potential confound is that ***AgentGrounder*** uses Qwen3-VL-32B, while SeeGround [8] reports results with Qwen2-VL-72B. However, ***AgentGrounder*** also outperforms zero-shot pipelines built on much larger proprietary models (GPT-4 Turbo in LLM-Grounder [5] and ZSVG3D [6], Mistral-Large-240B in CSVG [9]), despite using a base model roughly 2× smaller than SeeGround's and far smaller than these alternatives. This suggests that the gains mainly stem from the agent design rather than model capacity. A fully controlled comparison is left for future work.

Despite these advances, our method is limited by runtime latency (rendering and LVLM inference) and by failures in the initial 3D instance segmentation, which can propagate to downstream reasoning. Future work might explore human-in-the-loop clarification and adaptive tools (e.g., fallback detectors or re-segmentation heuristics) to improve robustness for real-world robotic deployment.

## IV. Conclusion

In this paper, we introduced ***AgentGrounder***, a zero-shot framework for 3D visual grounding on colored point clouds. Our method couples 3D segmentation network with a tool-driven VLM agent. We demonstrated that tools enabling geometric reasoning, scene image generation, and querying an object lookup table can enhance the performance of 3D visual grounding approaches. As evidence, ***AgentGrounder*** achieved state-of-the-art results on ScanRefer and Nr3D, with clear gains on view-independent queries and cluttered scenes.

## References


[1] Y. Wang, Y. Li, and S. Wang, "Gˆ 3-lq: Marrying hyperbolic alignment with explicit semantic-geometric modeling for 3d visual grounding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 13 917–13 926.

[2] Z. Qian et al., "Multi-branch collaborative learning network for 3d visual grounding," in *European Conference on Computer Vision*, Springer, 2024, pp. 381–398.

[3] H. Huang et al., "Chat-scene: Bridging 3d scene and large language models with object identifiers," *Advances in Neural Information Processing Systems*, vol. 37, pp. 113 991–114 017, 2024.

[4] D. Zheng, S. Huang, and L. Wang, "Video-3d llm: Learning position-aware video representation for 3d scene understanding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025, pp. 8995–9006.

[5] J. Yang et al., "Llm-grounder: Open-vocabulary 3d visual grounding with large language model as an agent," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2024, pp. 7694–7701.

[6] Z. Yuan, J. Ren, C.-M. Feng, H. Zhao, S. Cui, and Z. Li, "Visual programming for zero-shot open-vocabulary 3d visual grounding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 20 623–20 633.

[7] R. Xu, Z. Huang, T. Wang, Y. Chen, J. Pang, and D. Lin, "Vlm-grounder: A vlm agent for zero-shot 3d visual grounding," in *Conference on Robot Learning*, PMLR, 2025, pp. 3961–3985.

[8] R. Li, S. Li, L. Kong, X. Yang, and J. Liang, "Seeground: See and ground for zero-shot open-vocabulary 3d visual grounding," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 3707–3717.

[9] Q. Yuan, K. Li, and J. Zhang, "Solving zero-shot 3d visual grounding as constraint satisfaction problems," *arXiv preprint arXiv:2411.14594*, 2024.

[10] N. Zantout et al., "Sort3d: Spatial object-centric reasoning toolbox for zero-shot 3d grounding using large language models," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2025, pp. 2201–2208.

[11] J. Fang et al., "Transcrib3d: 3d referring expression resolution through large language models," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2024, pp. 9737–9744.

[12] 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.

[13] T. D. Ngo, B.-S. Hua, and K. Nguyen, "Isbnet: A 3d point cloud instance segmentation network with instance-aware sampling and box-aware dynamic convolution," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 13 550–13 559.

[14] Y. Yang, X. Wu, T. He, H. Zhao, and X. Liu, "Sam3d: Segment anything in 3d scenes," *arXiv preprint arXiv:2306.03908*, 2023.

[15] P. Nguyen et al., "Open3dis: Open-vocabulary 3d instance segmentation with 2d mask guidance," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2024, pp. 4018–4028.

[16] P. Nguyen, M. Luu, A. Tran, C. Pham, and K. Nguyen, "Any3dis: Class-agnostic 3d instance segmentation by 2d mask tracking," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 3636–3645.

[17] K. M. Jatavallabhula et al., "Conceptfusion: Open-set multimodal 3d mapping," *arXiv preprint arXiv:2302.07241*, 2023.

[18] Q. Gu et al., "Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2024, pp. 5021–5028.

[19] D. Guo et al., "Phygrasp: Generalizing robotic grasping with physics-informed large multimodal models," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2025, pp. 14 915–14 922.

[20] C. Wen et al., "Zero-shot object navigation with vision-language models reasoning," in *International Conference on Pattern Recognition*, Springer, 2025, pp. 389–404.

[21] D. Z. Chen, A. X. Chang, and M. Nießner, "Scanrefer: 3d object localization in rgb-d scans using natural language," in *European conference on computer vision*, Springer, 2020, pp. 202–221.

[22] P. Achlioptas, A. Abdelreheem, F. Xia, M. Elhoseiny, and L. Guibas, "Referit3d: Neural listeners for fine-grained 3d object identification in real-world scenes," in *European conference on computer vision*, Springer, 2020, pp. 422–440.

[23] O. Unal, C. Sakaridis, S. Saha, and L. Van Gool, "Four ways to improve verbo-visual fusion for dense 3d visual grounding," in *European Conference on Computer Vision*, Springer, 2024, pp. 196–213.

[24] Z. Qi, Z. Zhang, Y. Fang, J. Wang, and H. Zhao, "Gpt4scene: Understand 3d scenes from videos with vision-language models," *arXiv preprint arXiv:2501.01428*, 2025.

[25] C.-P. Chang, S. Wang, A. Pagani, and D. Stricker, "Mikasa: Multi-key-anchor & scene-aware transformer for 3d visual grounding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 14 131–14 140.

[26] S. Chen, P.-L. Guhur, M. Tapaswi, C. Schmid, and I. Laptev, "Language conditioned spatial relation reasoning for 3d object grounding," *Advances in neural information processing systems*, vol. 35, pp. 20 522–20 535, 2022.

[27] B. Jia et al., "Sceneverse: Scaling 3d vision-language learning for grounded scene understanding," in *European Conference on Computer Vision*, Springer, 2024, pp. 289–310.

[28] J. Schult, F. Engelmann, A. Hermans, O. Litany, S. Tang, and B. Leibe, "Mask3d: Mask transformer for 3d semantic instance segmentation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2023, pp. 8216–8223.

[29] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5828–5839.

[30] A. Yang et al., "Qwen3 technical report," *arXiv preprint arXiv:2505.09388*, 2025.

[31] F. S. Marcondes, A. Gala, R. Magalhães, F. Perez de Britto, D. Durães, and P. Novais, "Using ollama," in *Natural Language Analytics with Generative Large-Language Models: A Practical Approach with Ollama and Open-Source LLMs*, Springer, 2025, pp. 23–35.

[32] H. Chase, *Langchain*, `https://github.com/langchain-ai/langchain`, Version accessed via GitHub, Oct. 2022.