osmAG-LLM: Zero-Shot Open-Vocabulary Object Navigation via Semantic Maps and Large Language Models Reasoning*

Fujing Xie[1], Sören Schwertfeger[1], and Hermann Blum[2]

***Abstract*— Recent open-vocabulary robot mapping methods enrich dense geometric maps with pre-trained visual-language features, achieving a high level of detail and guiding robots to find objects specified by open-vocabulary language queries. While the issue of scalability for such approaches has received some attention, another fundamental problem is that high-detail object mapping quickly becomes outdated, as objects get moved around a lot. In this work, we develop a mapping and navigation system for object-goal navigation that, from the ground up, considers the possibilities that a queried object can have moved, or may not be mapped at all. Instead of striving for high-fidelity mapping detail, we consider that the main purpose of a map is to provide environment grounding and context, which we combine with the semantic priors of LLMs to reason about object locations and deploy an active, online approach to navigate to the objects. Through simulated and real-world experiments we find that our approach tends to have higher retrieval success at shorter path lengths for static objects and by far outperforms prior approaches in cases of dynamic or unmapped object queries. We provide our code and dataset at: https://github.com/xiexiexiaoxiexie/osmAG-LLM.**

I. INTRODUCTION

Navigation typically requires mapping and planning at multiple scales. Applications such as last mile delivery, inner-city logistics, or assistive navigation systems require city-level street maps to get to the right building, but then also need to navigate inside buildings to go to a specific room or find a requested object. Furthermore, these tasks need understanding about how our world is structured, which items are usually kept in which locations, the usual layouts and functions of different rooms or spaces, as well as other environment-related information.

Previous works on utilizing LLMs for object navigation, such as [1], [2], often rely on language-based instructions like "Bring me the bottom picture that is next to the top of stairs on level one" or "find a bean bag in the office on the first floor" to guide robots. These requests already include a lot of map knowledge in the input to the task, which may not always be known to the user or the environment might be too large and complex for such explicit guidance. In this work we also consider less specified queries, where a robot is tasked to retrieve a certain type of object, even if it might have moved, there are multiple possible ones in the map, or it has not been mapped at all.

In this work, we build upon osmAG (Area Graph in OpenStreetMap textual format) [3], a framework that adds room-level mapping to OpenStreetMap, and extends it to object-level mapping and navigation. In particular, we leverage the LabelMaker pipeline [4] as well as VLMs to populate a map with initial object instances and room descriptions. While we prioritize scalability over completeness of the map, we show that through online detection and decision making, we are able to retrieve unmapped objects with close to the same success rate as mapped objects, highlighting the importance of the context a map provides.

In summary, our contributions are:

- We extend osmAG to object-level semantic mapping, creating a textual semantic map that can be parsed by LLMs.
- We develop an online object retrieval algorithm that combines the planning capabilities and semantic priors of LLMs with online open-vocabulary detection to retrieve even rare and unusual objects that are not explicitly mapped.
- We conduct extensive experiments in simulation and in a real-world campus building, extending the common test cases with object queries that have been moved, as well as querying for unmapped objects.

II. APPROACH

This work tackles the challenge of map-guided object navigation by combining semantic maps with large language models. As illustrated in Fig. 1, our approach begins by augmenting the basic osmAG map using RGB-D trajectories to build a semantic-rich map for subsequent tasks (Section II-A). Next, given a human query, an LLM is leveraged to infer probable map nodes corresponding to the target object (Section II-B). Finally, the robot navigates toward the nodes (Section II-C), with navigation further refined through real-time online object detection (Section II-D).

A. Semantic Environment Mapping

1) semantic-osmAG: We build on osmAG [3], a hierarchical topometric semantic map based on the Area Graph concept [5], which has proven effective for LiDAR-based localization [6] and intelligent navigation [7]. We adopt its

*This work was supported by the Shanghai Frontiers Science Center of Human-centered Artificial Intelligence and German Federal Ministry of Education and Research (BMBF) in the project ?Robotics Institute Germany?, grant No. 16ME0999. The work was made possible with the support of a scholarship from the German Academic Exchange Service (DAAD). The experiments of this work were supported by the core facility Platform of Computer Science and Communication, SIST, ShanghaiTech University.

[1]Fujing Xie and Sören Schwertfeger are with the Key Laboratory of Intelligent Perception and Human-Machine Collaboration – ShanghaiTech University, Ministry of Education, China. xiefj, soerensch@shanghaitech.edu.cn

[2]Hermann Blum is with Robot Perception and Learning Lab, University of Bonn and Lamarr Institute for ML and AI, 53115 Bonn, Germany. blumh@uni-bonn.de

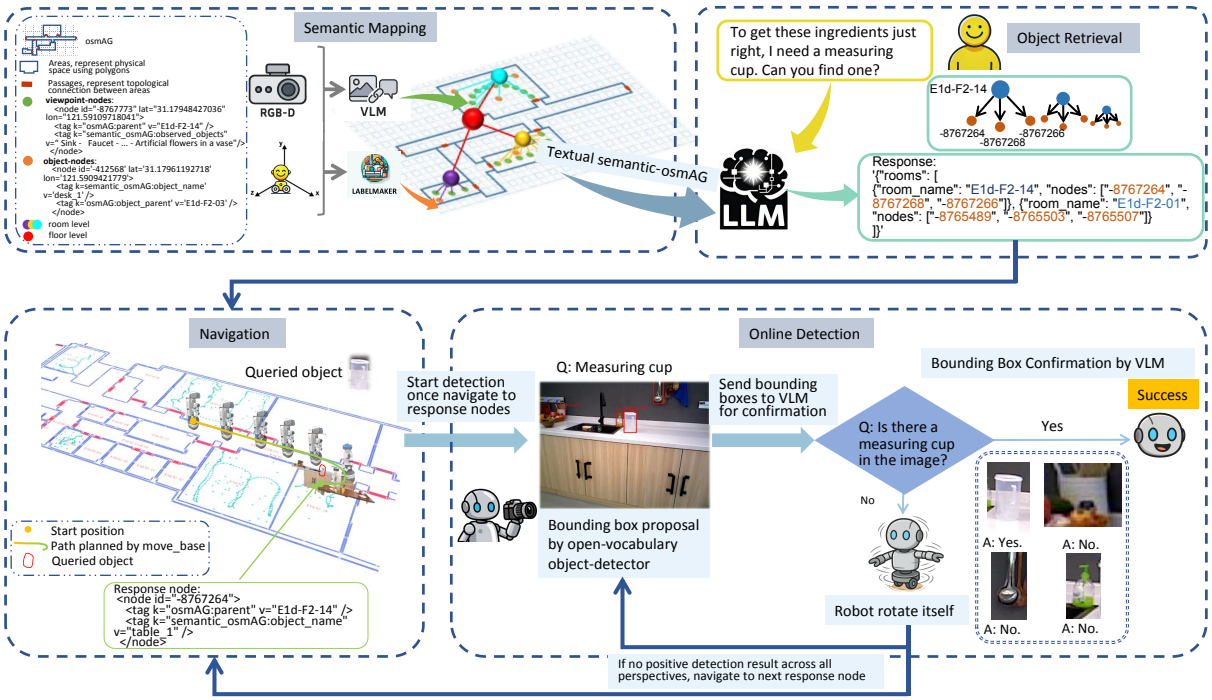

Fig. 1: An overview of our method. We construct a semantic-osmAG offline by augmenting the basic osmAG with two additional keys: object-nodes (extracted via LabelMaker from RGB-D trajectory data) and viewpoint-nodes (processed by a VLM and placed along the trajectory path). When given a human query, the system uses an LLM to generate proposed geometric nodes (response nodes) based on the pre-built textual semantic-osmAG and the query. The robot then navigates to these nodes one by one using ROS move_base. Once at a response node, an open-vocabulary object detector proposes bounding boxes for the queried object, which are then checked by a VLM to verify if the object is actually present. If the object isn't found, the robot turns to capture additional perspectives. If the object still isn't detected after checking all views at that node, the robot moves to the next response node and repeats the detection process. (Cartoon robot in Online Detection module generated by ChatGPT-4V; prompt: 'robot rotating in place, taking pictures'.)

indoor semantic extension, semantic-osmAG, for its compact, shareable, and human- and LLM-interpretable OSM-based representation, as well as its minimal update requirements in dynamic environments. However, the original osmAG lacks sufficient granularity for object-level localization, which we address through the semantic enrichment method in Section II-A.2.

2) Semantic Enrichment: To integrate semantic information into osmAG, we developed a pipeline for generating semantic nodes from RGB-D data. We generate the object-nodes by first utilizing LabelMaker[4] to process RGB-D trajectories through ensemble voting across state-of-the-art models to produce point clouds for each object category. We calculate each instance's centroid and embed it as a 'node' (with longitude/latitude coordinates) within the osmAG structure with semantic_osmAG:object_name as tag's key as illustrated in Fig. 1. However, LabelMaker inherits the categorical limitations of its underlying models, failing to recognize objects outside standard datasets like ScanNet200 [8], which limits its ability to process unrestricted natural language queries about objects (e.g., robot dog). To address this, we augment the system with ChatGPT's [9] visual query understanding capability. By prompting with 'What objects are in this image?' at capture locations, we generate viewpoint-nodes at the image's captured location with tag key as semantic_osmAG:observed_object and value as the response of VLM as illustrated in Fig. 1. This hybrid approach generates part of the nodes with exact geometric location and gives part of the nodes open-vocabulary capability from a more intelligent model, overcoming categorical rigidity inherent in LabelMaker alone. Beyond embedding geometric object nodes, we generate semantic room descriptions by prompting a VLM with 'Describe the image.' for all images captured in a room. The resulting descriptions are then summarized by an LLM and attached directly to the corresponding area elements in the semantic-osmAG map.

B. Object Retrieval

When processing a human query, our method generates a structured LLM prompt that: (1) explains semantic-osmAG's representation, (2) defines the object retrieval task objectives, and (3) provides a simplified semantic-osmAG containing only semantic nodes and hierarchical relationships.

To improve search efficiency and match practical navigation, we instruct the LLM to return JSON-formatted response nodes (viewpoint or object nodes) grouped by room (up to 3 nodes per room and 3 rooms total) and ranked by likelihood (Fig. 1), mimicking human room-by-room search rather than probability-only node hopping.

C. Navigation

Unlike methods like HOV-SG [2] using pre-built navigation graphs, we dynamically navigate via online perception from rendered osmAG. We first convert osmAG to an occupancy grid (area polygons as occupied, passages as free). The system then sequentially visits response nodes from LLM: (1) setting navigation goals from sorted nodes, (2) planning paths via A* on the occupancy grid, and (3) continuously updating the rendered occupancy grid map using real-time depth data and triggering path replanning when collisions are detected in the updated map. By building the grid only from permanent structures and adapting online, we avoid needing map updates when environments change.

TABLE I: Object Retrieval & Goal Navigation from Language Queries on HM3DSEM

Query Type	Method	R-RSR	O-RSR$_{top5}$			O-RSR$_{top1}$			AMD [m]	DIR	APL [m]
			@1m	@2m	@3m	@1m	@2m	@3m			
(o,r,f)	HOV-SG	0.27	**0.31**	0.39	0.44	**0.20**	0.28	0.33	8.85	0.05	20.15
	our method	**0.66**	0.25	**0.59**	**0.70**	0.16	**0.37**	**0.54**	**4.65**	**0.27**	**12.32**
(o,r)	HOV-SG	0.32	**0.35**	0.44	0.49	**0.20**	0.28	0.33	7.87	0.06	16.81
	our method	**0.67**	0.28	**0.58**	**0.68**	0.18	**0.37**	**0.55**	**5.27**	**0.25**	**11.54**
(o)	HOV-SG	0.65	**0.76**	0.82	0.88	**0.52**	**0.64**	**0.70**	**3.10**	0.05	22.44
	our method	**0.83**	0.47	**0.83**	**0.90**	0.28	0.50	0.69	3.37	**0.31**	**11.36**

D. On-line Object Detection

To ensure robust adaptation to environmental changes and verify object presence at response nodes, our system performs real-time detection upon arrival. As shown in Fig. 1, when the robot arrives at a response node, it captures images and executes a two-stage verification process. We run open-vocabulary object-detectors (Yolo-world [10] or DINO-X [11]) on the captured images to generate a list of bounding boxes ranked by confidence. Each box is cropped and validated by a VLM (StepFun [12]) using the prompt "Is there a [object] here?", with the highest-confidence positive detection returned as the final result. We find that this two-step approach is more reliable with long-tail object queries where the detectors would otherwise return too many false positives. If no object is detected, the robot rotates its body or head to inspect additional viewpoints until the target is found or all viewpoints at the current node are exhausted, then moves to the next response node.

III. Experiments

We evaluated our approach on eight scenes from the HM3D-SEM dataset [13] (consistent with our baseline) as well as on a real-world dataset collected in our campus building. In Section III-A details our evaluation metrics, followed by Habitat simulation experiments using the HM3D-SEM dataset in Section III-B, and real-world experiments in Section III-C.

A. Metrics

We employ the following metrics to evaluate performance across both simulated and real-world environments:

- Room Retrieval Success Rate (R-RSR): Percentage of queries where the top prediction's room contains at least one ground truth instance of the target object.
- Object Retrieval Success Rate (O-RSR$_{top-n}$@k): Percentage of queries where at least one top-n prediction ($n \in 1, 5$) is within k meters of ground truth.
- Average Minimum Distance (AMD): Computes the mean distance error of the closest prediction among the top-5 results to evaluate object localization precision.
- Average Path Length (APL): Mean navigation path length that the robot drove (in meters) for queries where both our method and baseline successfully retrieve the target (within 1m). Measures navigation efficiency.
- Detection Improvement Rate (DIR): Percentage of failed retrievals recovered by online detection at response nodes.
- Map size: Compares the representation size of our method and HOV-SG.

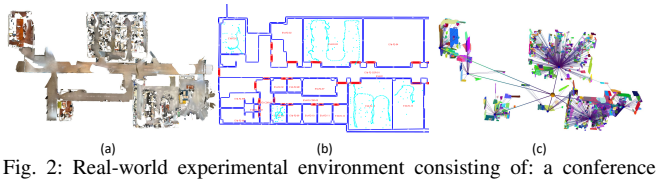

Fig. 2: Real-world experimental environment consisting of: a conference room (upper left), a student office (upper right), a professor's office (lower left), a robotics lab (lower middle), and a relaxation lounge (lower right). (a) Data collected using an Apple scanner with the "3D Scanner App"; (b) the semantic-osmAG map; (c) the HOV-SG generated from the collected data.

B. HM3D-SEM Dateset Experiment

1) Setup: We evaluate our approach on 8 diverse HM3D-SEM scenes (00824, 00829, 00843, 00861, 00862, 00873, 00877, 00890) which span multiple rooms and floors. We generate queries using the same approach as HOV-SG with three granularity levels for 20 common objects: 1. object (o): "pillow"; 2. object + room (o,r): "pillow in the living room"; 3. object + room + floor (o,r,f): "pillow in the living room on floor 0".

2) Results: Table I presents comprehensive evaluation results across three query granularities, our method demonstrates significant improvements in key metrics over the HOV-SG baseline:

- Superior room and object retrieval: Notably, we achieve 144% higher Room Retrieval Success Rate (R-RSR) for complex (o,r,f) queries (0.66 vs. 0.27) and maintain better performance at practical operating distances (O-RSR@2m/3m). The AMD metric confirms our predicted nodes are closer to ground truth positions in (o,r) and (o,r,f) query types.
- Effective online detection compensation: Online detection effectively compensates for retrieval errors, as shown by higher DIR.
- Efficient navigation: The Average Path Length (APL) metric shows that our method reduces traversal distance by 31-49% across all conditions.
- Significantly smaller storage requirements: As shown in Table II, our method requires substantially less storage space than the baseline.

TABLE II: Representation Size Comparison (MB)

	HM3D-SEM (8 scenes)	Real Dataset
HOV-SG	1493	129
Our method	3.2	0.62

C. Real-World Environment Experiment

1) Setup: We conducted experiments in a large and cluttered campus building spanning multiple functional areas: a conference room (46 m^2), a robotics lab (120 m^2), a student office (147 m^2), a professor's office (15 m^2), and a relaxation lounge (82 m^2) (Fig. 2). To scale our approach to larger environments, we further propose a two-stage query

TABLE III: Object Retrieval & Goal Navigation from Language Queries on Real Data

Object Type	Method	R-RSR	O-RSR$_{top5}$			O-RSR$_{top1}$			AMD [m]	DIR	APL [m]
			@1m	@2m	@3m	@1m	@2m	@3m			
static (SO)	ConceptGraph	0.50	**0.50**	0.50	0.60	**0.20**	0.30	0.40	3.52 (3.52)	/	/
	HOV-SG	0.40	0.20	**0.70**	**0.80**	0.00	0.30	0.30	1.84 (1.84)	0.40 (0.40)	47.00
	our method	**1.00**	0.30	0.50	**0.80**	**0.20**	**0.40**	**0.80**	**1.78 (1.78)**	0.40 (0.40)	**18.13**
relocated (RO)	ConceptGraph	0.50	0.20	0.30	0.50	0.00	0.10	0.20	5.51 (5.86)	/	/
	HOV-SG	0.40	**0.30**	0.50	0.50	0.00	0.20	0.20	5.36 (5.79)	0.00 (0.00)	22.05
	our method	**1.00**	**0.30**	**0.70**	**0.90**	**0.20**	**0.60**	**0.80**	**1.63 (1.72)**	**0.70 (0.78)**	**20.43**
unmapped (UO)	ConceptGraph	0.40	**0.10**	0.20	0.30	0.00	**0.10**	0.20	8.35 (8.83)	/	/
	HOV-SG	0.20	0.00	**0.30**	0.50	0.00	0.00	0.20	7.42 (7.63)	0.30 (0.33)	52.03
	our method	**1.00**	0.00	**0.30**	**0.80**	0.00	**0.10**	**0.60**	**2.25 (2.33)**	**0.80 (0.78)**	**30.68**

Values in parentheses indicate: results after excluding inaccessible rooms (Section III-C.1), maintaining the same trends as overall results.

variant. The first stage uses only the room-level descriptions for coarse room selection, while the second stage provides the detailed map of the chosen room to select specific target nodes. To analyze the trade-off in map complexity, we also investigated the impact of viewpoint node density–examining how performance varies when too few nodes may provide insufficient context versus when too many may confuse the LLM. We used an Apple scanner with the '3D Scanner App' for scanning. The robot platform was a Fetch robot with a base-mounted 2D LiDAR, a 180° rotatable head, and a head-mounted RealSense RGB-D camera.

2) Queries: To evaluate both methods under realistic conditions, we conducted 30 experiments divided into three distinct categories (shown in Fig. 3): Static Objects (SO, e.g., sink/TV) remaining approximately at mapped positions; Relocated Objects (RO, e.g., screwdriver/robot dog) moved from initial locations; and Unmapped Objects (UO, e.g., presentation remote/measuring cup) absent during mapping. Each category (SO, RO, UO) contained six unique single-location objects and four multi-instance objects, totaling 11, 12, and 13 objects per category respectively.

3) Results: Our real-world experiments demonstrate consistent advantages over baseline (Table III):

- Superior room and object retrieval: Our method achieves 100% room retrieval, surpasses baselines in O-RSR metrics as distance increases, and maintains lower AMD (nodes closer to ground truth).
- Effective online detection compensation: Our higher DIR provides strong evidence that our online detection approach effectively compensates for sparse mapping requirements while eliminating the need for precise 3D point cloud data saved in the map.
- Efficient navigation: The APL metric shows our method reduces path length across all conditions.
- Environmental robustness: Our method outperforms baselines on relocated and unmapped objects, showing robustness to environmental changes.
- Two-stage query performance: Table IV shows that the two-stage approach works well for static objects and remains competitive for relocated or unmapped objects, indicating promising scalability to large environments.
- Viewpoint node density: Fig. 4 shows a non-linear effect of viewpoint node density: a few nodes bring large gains, while too many yield diminishing returns or slight degradation, suggesting that only a modest number of nodes per room is sufficient.

Fig. 3: Experimental objects used for real-world evaluation: The first row shows static objects, the second row contains relocated objects, and the final row displays unmapped objects absent from the map. Successful detections are shown with automatically generated bounding boxes during experiments, while text overlays indicate object names and instance counts in our environment (blue = success, red = failure).

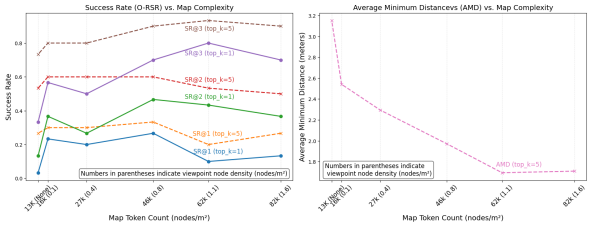

Fig. 4: Object retrieval performance versus map complexity, represented by map token count and its corresponding viewpoint node density (parenthetical). (Left) Object Retrieval Success Rate (O-RSR) and (Right) Average Minimum Distance (AMD).

4) Ablation Studies: To evaluate the contributions of the two key components in our map–object nodes and viewpoint nodes–we conduct an ablation study using real-world data (Table IV). First, we remove all object nodes generated by LabelMaker. This degrades RSR and AMD performance, since the system loses precise object location information. Next, we eliminate all viewpoint nodes generated by the VLM, which leads to rapid performance degradation, since our language queries contain many objects outside our detection model's fixed category set. Our results demonstrate that both node types are essential, as removing either component leads to substantially worse performance compared to the complete method.

TABLE IV: Comparison of Method Variants on Real Data

Object Type	Method	R-RSR	O-RSR$_{top5}$			O-RSR$_{top1}$			AMD [m]
			@1m	@2m	@3m	@1m	@2m	@3m	
SO	Ours	**1.0**	0.3	0.5	0.8	0.2	0.4	**0.8**	1.78
	Ours-2stages	**1.0**	**0.4**	**0.8**	**1.0**	**0.3**	**0.5**	**0.8**	**1.30**
	w/o ON	**1.0**	0.3	0.5	0.7	**0.3**	0.4	0.6	2.14
	w/o VN	0.9	0.2	0.5	0.6	0.1	0.2	0.5	3.77
RO	Ours	**1.0**	**0.3**	**0.7**	**0.9**	0.2	**0.6**	**0.8**	**1.63**
	Ours-2stages	0.7	**0.3**	0.5	0.7	0.2	0.4	0.6	7.44
	w/o ON	0.9	**0.3**	**0.7**	0.8	**0.3**	0.5	0.6	2.00
	w/o VN	0.7	0.2	0.2	0.5	0.0	0.1	0.2	4.18
UO	Ours	**1.0**	0.0	**0.3**	**0.8**	0.0	**0.1**	**0.6**	**2.25**
	Ours-2stages	0.9	0.0	0.2	0.7	0.0	**0.1**	**0.6**	4.37
	w/o ON	0.9	0.0	0.1	0.4	0.0	**0.1**	0.2	4.73
	w/o VN	0.7	0.0	0.2	0.5	0.0	0.0	0.1	6.24

"w/o ON" = without object nodes, "w/o VN" = without viewpoint nodes.

REFERENCES

[1] Y. Qi, Q. Wu, P. Anderson, X. Wang, W. Y. Wang, C. Shen, and A. v. d. Hengel, "Reverie: Remote embodied visual referring expression in real indoor environments," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9982–9991.

[2] A. Werby, C. Huang, M. Büchner, A. Valada, and W. Burgard, "Hierarchical open-vocabulary 3d scene graphs for language-grounded robot navigation," in *First Workshop on Vision-Language Models for Navigation and Manipulation at ICRA 2024*, 2024.

[3] D. Feng, C. Li, Y. Zhang, C. Yu, and S. Schwertfeger, "Osmag: Hierarchical semantic topometric area graph maps in the osm format for mobile robotics," in *2023 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 2023, pp. 1–7.

[4] S. Weder, H. Blum, F. Engelmann, and M. Pollefeys, "Labelmaker: automatic semantic label generation from rgb-d trajectories," in *2024 International Conference on 3D Vision (3DV)*. IEEE, 2024, pp. 334–343.

[5] J. Hou, Y. Yuan, and S. Schwertfeger, "Area graph: Generation of topological maps using the voronoi diagram," in *2019 19th International Conference on Advanced Robotics (ICAR)*. IEEE, 2019, pp. 509–515.

[6] F. Xie and S. Schwertfeger, "Robust lifelong indoor lidar localization using the area graph," *IEEE Robotics and Automation Letters*, vol. 9, no. 1, pp. 531–538, 2023.

[7] F. Xie, J. Zhang, and S. Schwertfeger, "Intelligent lidar navigation: Leveraging external information and semantic maps with llm as copilot," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2025.

[8] D. Rozenberszki, O. Litany, and A. Dai, "Language-grounded indoor 3d semantic segmentation in the wild," in *European Conference on Computer Vision*. Springer, 2022, pp. 125–141.

[9] J. Achiam, S. Adler, S. Agarwal, L. Ahmad, I. Akkaya, F. L. Aleman, D. Almeida, J. Altenschmidt, S. Altman, S. Anadkat *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.

[10] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, and Y. Shan, "Yolo-world: Real-time open-vocabulary object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 16 901–16 911.

[11] T. Ren, Y. Chen, Q. Jiang, Z. Zeng, Y. Xiong, W. Liu, Z. Ma, J. Shen, Y. Gao, X. Jiang *et al.*, "Dino-x: A unified vision model for open-world object detection and understanding," *arXiv preprint arXiv:2411.14347*, 2024.

[12] StepFun, "step-1o-turbo-vision," StepFun API documentation, https://platform.stepfun.com/docs/llm/vision, 2025, accessed 2025-06-13; visual understanding model for text, image, and video input.

[13] K. Yadav, R. Ramrakhya, S. K. Ramakrishnan, T. Gervet, J. Turner, A. Gokaslan, N. Maestre, A. X. Chang, D. Batra, M. Savva *et al.*, "Habitat-matterport 3d semantics dataset," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 4927–4936.