Please adjust group name in tumuser.sty.
Department of Electrical and Computer Engineering
Technical University of Munich

Navigation and Manipulation with Vision-Language Models

TUM Data Innovation Lab Project Report

Beste Aydemir, Yuanchen Sun, Jingkun Feng

Examiner:
Prof. Dr. Daniel Cremers

Supervisor:
Dr. Yan Xia (TUM), Junyu Xie (Oxford)

Submitted:
Munich, 25.07.2025

I hereby declare that this thesis is entirely the result of my own work except where otherwise indicated. I have only used the resources given in the list of references.

Munich, 25.07.2025 Beste Aydemir, Yuanchen Sun, Jingkun Feng

Abstract

This work explores global localization and functional object segmentation problems through the utilization of multimodal models for navigation and manipulation. In the first project, we address global localization in indoor environments, which is an essential task for mobile agents' application. Current map-free methods such as SLAM require large computation and storage costs, while map-based methods are mainly built on depth matching, neglecting the semantic information from the maps. We propose a multimodal pipeline, concatenating map-based depth fusion and semantic localization based on Vision Language Models (VLMs). Our work outperforms the single-step localization accuracy compared to our baseline. In the second project, we focus on segmenting functional parts of objects, such as identifying not just a door but its handle, within 3D point cloud scenes. We construct a pipeline that uses class-agnostic object masks, scene graph parsing for spatially informed querying, and hierarchical language understanding of the input text description by LLMs. This is followed by part segmentation and VLM-guided selection to extract the precise functional components described in natural language. Increasing the localization and functional part segmentation skills of an embodied agent is crucial for any system that requires a deep, grounded understanding of its environment to act effectively and naturally.

Contents

Abstract **v**

1 Introduction **1**
1.1 Indoor localization . . . 1
1.2 Functionality segmentation . . . 1
1.2.1 Outline . . . 1

2 Leveraging VLM for Floorplan-Based Indoor Localization **3**
2.1 Introduction . . . 3
2.2 Related work . . . 3
2.2.1 Map-Based Localization . . . 3
2.2.2 Vision Language Model . . . 4
2.2.3 VLM in Map-Based Localization . . . 4
2.3 Method . . . 4
2.3.1 Problem Definition and Overview . . . 4
2.3.2 Semantic Grounding on Floorplans . . . 5
2.3.3 Masked Depth Fusion Localization . . . 6
2.3.4 VLM-Based Semantic Localization . . . 6
2.4 Experiments . . . 6
2.4.1 Experimental Setting . . . 6
2.4.2 Results . . . 8
2.5 Discussion . . . 9
2.5.1 Limitations . . . 9
2.5.2 Future Work . . . 10

3 Hierarchical Open-vocabulary Functionality Segmentation using Scene Graph for Robotic Manipulation **11**
3.1 Introduction . . . 11
3.2 Related work . . . 12
3.2.1 Open-vocabulary 3D segmentation . . . 12
3.2.2 Scene graph . . . 13
3.2.3 Vision-language models . . . 13
3.3 Method . . . 14
3.3.1 Open-vocabulary 3D instance segmentation . . . 14
3.3.2 Open-vocabulary scene graph . . . 14
3.3.3 Query understanding using LLM . . . 14
3.3.4 Contextual object grounding . . . 15
3.3.5 Functional component segmentation . . . 16
3.4 Experiments . . . 16
3.4.1 Experiment setup . . . 16
3.4.2 Mask3D masks on SceneFun3D dataset . . . 18
3.4.3 Part segmentation on instance masks from SceneFun3D dataset . . . 18
3.4.4 VLM understanding of part segmentation masks . . . 18
3.5 Discussion . . . 20
3.5.1 Conclusion . . . 20

3.5.2 Future Work 20

4 Conclusion **23**

Bibliography **25**

1 Introduction

1.1 Indoor localization

Indoor localization is a critical capability for mobile robots, enabling them to perform tasks in unfamiliar environments. Traditional approaches often rely on 3D reconstruction or training within pre-built 3D models, which are computationally expensive and require significant storage and maintenance. To address these challenges, recent research has explored lightweight representations such as floorplans, which are robust to changes in furniture or equipment while retaining semantic and geometric information. However, current floorplan-based methods suffer from low single-step localization accuracy, requiring multiple iterations to achieve reliable positioning.

Meanwhile, vision-language models (VLMs) have demonstrated strong performance in understanding images and text, and have been applied to map-based localization and navigation. However, existing approaches use VLMs and visual matching separately, resulting in high data requirements for fine-tuning and limited output to high-level information. This work introduces an interactive approach that integrates VLMs with visual matching to enhance single-step localization accuracy and reduce convergence iterations. By combining depth matching between agent view depth estimation and the floorplan's directional Euclidean signed distance field (DESDF) with semantic matching from a VLM, the proposed method achieves nearly double the recall rate of baseline models across all tolerances. Key contributions include an interactive VLM integration strategy requiring minimal training and improved single-step localization performance.

1.2 Functionality segmentation

Open-vocabulary 3D segmentation aims to interpret 3D scenes using natural language, enabling robots to recognize novel objects and object parts without being limited by pre-defined labels. While traditional 3D segmentation methods lack generalization to unseen classes and focus primarily on object-level segmentation, open-vocabulary approaches leverage large language models for more flexible understanding. However, most existing solutions either overlook fine-grained segmentation or perform it across the entire scene, resulting in high computational and memory demands that are impractical for mobile robots. For real-world tasks like identifying a chair arm or a washing machine door, robots require efficient, task-specific fine-grained segmentation systems that balance accuracy with hardware constraints.

To address this, the proposed method builds actionable 3D scene graphs using embedded features instead of fixed labels, enabling the robot to understand inter-object relationships crucial for interactive tasks. Unlike prior hierarchical models that consume extensive memory by processing and storing features for all scene elements, this approach selectively applies fine-grained segmentation only to objects relevant to the input query. This design supports efficient online processing, making it more suitable for real-time robotics applications. Key contributions include a training-free, open-vocabulary functionality segmentation method, and thorough evaluations on the ScanNet and SceneFun3D datasets to validate its performance and design choices.

1.2.1 Outline

The following of this report is divided into two parts, each associated with individual projects. In Chapter 2, we present a method that leverages VLM for indoor localization based on 2D floorplans. In Chapter 3.3, we introduce a pipeline of functionality segmentation tailored for robotic manipulation tasks. We review the related work for each chapter individually (Sec. 3.2 and Sec. 2.2) to provide overviews closely related

to project's objectives. In conclusion, we summarize the current status of the projects and discuss future steps.

2 Leveraging VLM for Floorplan-Based Indoor Localization

2.1 Introduction

Indoor localization has raised practical interest in the mobile robotics community. It is an essential prerequisite for mobile agents to conduct manipulative tasks in a novel environment. Existing indoor localization works mostly requires 3D reconstruction [28] or the training within a pre-built 3D model [14]. However, these methods are costly in terms of computation, storage and maintenance.

Recently, researchers have increasingly leveraged easily accessible representations for indoor localization [21] [6] [9]. Floorplan, in particular, is the most widely used representation for almost all kinds of indoor facilities. This generic representation is light-weight and long-term reservable. It is not influenced by the movement of furniture and equipments, but still preserves high-level semantic and geometric information. However, the current floorplan-based works show poor accuracy in single step localization and requires many iterations to accurately locate mobile agents [6].

Vision-language models (VLMs), in the mean time, have recently shown significant improvements in image and text understanding [1] [9]. Recent works have proved that VLMs can be leveraged for map-based localization and navigation tasks [5] [22] [9] [52]. However, the current approaches implement VLMs semantic analysis and visual matching between agent view and map in a parallel way, meaning that VLMs and visual matching models don't share information during inference. This leads to a high prior-knowledge data demand to fine-tune VLMs.

In this work, we investigate the interactive combination of VLMs and depth fusion to enhance the single step localization accuracy and reduce the number of iterations required for convergence. The depth fusion between agent view depth estimation and floorplan directional euclidean signed distance field (DESDF) provides intuitive localization candidates. Then a VLM analyses the agent view and the pose candidates in the flooplan to do selection based on semantic matching. Despite this straightforward idea, we've seen significant improvement in the single-step localization. The recall rate of our method is approximately twice as high as the baseline at all tolerances. We propose the following contributions in our work:

- We introduce an interactive strategy for implementing VLMs into floorplan-based indoor localization with little or no training.
- We significantly increase the single-step localization accuracy and decrease required iteration number for sequential location filtering to converge.

2.2 Related work

2.2.1 Map-Based Localization

Map-based localization is widely used in indoor and outdoor scenes. In contrast to localization methods without map prior-knowledge such as SLAM, this approach doesn't require the reconstruction of 3D environments and thus saves storage and computation costs [13]. This also avoids the problem of positioning drift caused by 3D reconstruction error accumulation [13]. F^3Loc [6] is a depth matching method for indoor localization. In this work, pre-generated DESDFs of floorplans is match mobile agents' scene geometry on them. The scene geometry is extracted by a single gravity-aligned view or multiple gravity-aligned view, and estimate the horizontal depth. The floorplans used in this work only contains walls and causes ambiguity in the localization. To tackle the ambiguity, a sequential filtering framework was proposed.

[13] introduces switching strategies between block maps to ensure mobile agents can estimate their poses in large-scale environments by loading local map information. Similar to F^{3}Loc, OrienterNet [34] proposed the confidence matching between views and maps. The confidence maps of both views and maps are generated by CNN. FloNa [21] developed a diffusion policy-based framework for indoor navigation. The framework facilitates alignment between the mobile agents' current observation and the floorplan.

2.2.2 Vision Language Model

Vision language models (VLMs) are currently capable of querying fine-grained visual representations across multiple images [43] [41]. [49] and SoFA [41] proved the zero-shot generalization capability of LVLMs and provided the model architecture to realize a good performance such as SoFt-Attention [41]. [60] attempted to address abstract visual reasoning (AVR) by leveraging a common LLaVA-NeXT 7B model and surpassed open-sourced and closed-sourced powerful VLMs. LLaVA-ST [20] demonstrates high performance accross spatial-temporal fine-grained understanding tasks. LLaVA-CoT [48] independently engages autonomous multistage reasoning consisting of summarization, visual interpretation, logical reasoning, and conclusion generation. This structured approach overcomes the difficulties in classical chain-of-thought prompting when handling reasoning-intensive tasks. Qwen2.5-7B now allows precise object grounding across formats supporting both box and points output. It also shows strong text recognition capabilities in diverse scenes [1].

2.2.3 VLM in Map-Based Localization

Under the context of map-based localization, VLMs are used to help mobile agents better utilize the semantic information from the floorplan and their current views [9] [52]. Contrastive language-image pre-training (CLIP) [33] is an efficient method of learning from natural language supervision based on a simplified ConVIRT [56]. CLIP introduces a structure that has a modified ResNet image encoder and a transformer text encoder. They are trained jointly to predict the correct pairings of images and texts. During prediction, CLIP is given a set of potential text pairings and the images are matched with one of them. CLIP is widely used in map-based localizations, especially in outdoor scenes and global localization [5] [29] [30] [22]. It is mainly used to interpret semantic features of mobile agents' views [5] [22] [29], but also to analyse the map landmarks [29] [30]. [30] also shows that CLIP can also be implemented to directly match between mobile agents' views and maps, instead of matching visual features with prior-knowledge given in text format.

Apart from that, researchers have also been exploring the potential of intuitively implementing powerful VLMs intuitively into map-based localization tasks [9] [52] [14] [15] [55]. [9] has proven that VLMs are capable of parsing element-enhanced floorplans and generating high-level navigation actions according to text commands. VLMap [14] demonstrated the approach to construct indoor maps with segmentation masks based on top-down view and open-vocabulary labels. The subsequent work IVLMap [15] implemented Segment Anything Model (SAM) to enhance the maps' representation and employed LLM to better parse natural language commands. [52] leveraged VLM to capitalize on visual semantic cues in noval environments and guide the robot to explore the environment and reach the goal more efficiently. MapNav [55] constructed top-down semantic maps and them transform to an annotated semantic maps which contain clear navigation cues for LLM based commands. The maps are refreshed at episodes and updated at timesteps, thus serves as a new memory representation vision-and-language navigation (VLN).

2.3 Method

2.3.1 Problem Definition and Overview

We solve the problem of localizing RGB imgages with respect to a semantic floorplan. Given a single RGB image I_t or a temporal sequence of $k+1$ RGB images $\mathcal{I} = \{\mathbf{I}_\tau \mid \tau \in \{t-k, \ldots, t\}\}$, we aim to find the

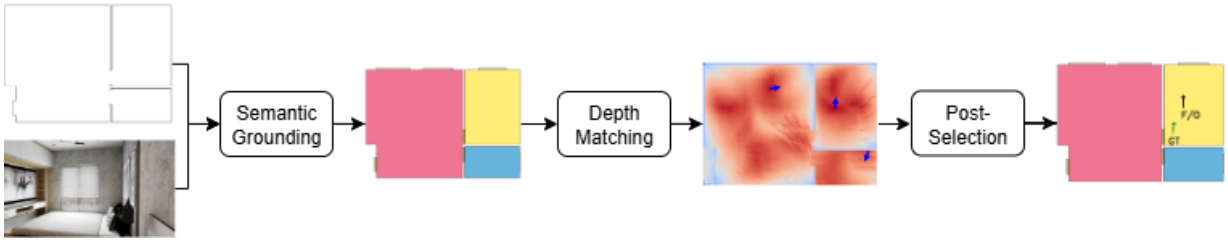


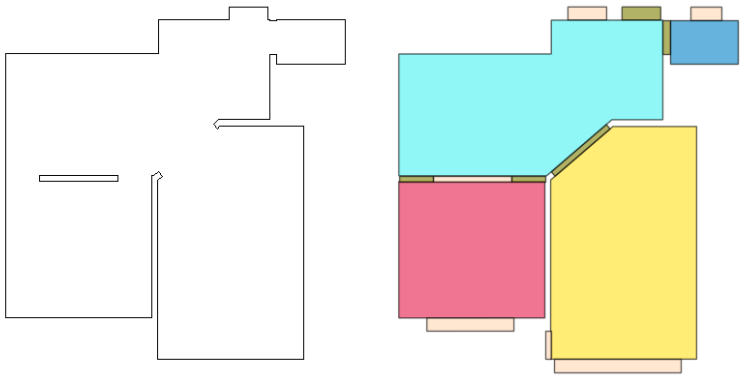
Figure 2.1 Pipeline overview. Our pipeline consists of three parts. Firstly the mobile agent's view and the semantic floorplan are passed to a VLM to generate the semantic groundings of the potential regions. Then a monocular network is adopted to predict the horizontal depth rays of the view image. The resulting depth rays are matched with the floorplan's DESDF and a confidence map is generated. We concatenate the semantic grounding and the confidence map to output the position candidates. Finally a VLM is emplyed to select the best position from the candidates.

current $SE(2)$ camera pose $\mathbf{s_t}$ within a given 2D floorplan, where $\mathbf{s_t} = [s_{x,t}, s_{y,t}, s_{\phi,t}]$ represents the camera x, y coordinate in the floorplan and its orientation.

The methods to estimate the floorplan depth (i.e. DESDF), image horizontal depth and their depth matching are given by F^{3}Loc [6]. Based on that, we firstly enhance the information given by floorplans by extracting semantic landmarks from the dataset **??**. Then the position candidates are generated by depth matching according to the semantic groundings on the floorplan. Finally, we feed the semantic floorplan, the view RGB image to the VLM and prompt it to understand the groundings on the floorplan and select the most likely pose from the candidates.

2.3.2 Semantic Grounding on Floorplans

Currently, there is no dataset explicitly collecting floorplans. Thus the most common floorplan format used in mobile agent localization are generated by extracting map outlines [6] [21]. An example of this floorplan format is given by Fig. 2.2a. This floorplan format causes ambiguity in the localization as it only contains walls and corners. To resolve this issue and make use of the semantic information given by RGB view images, we extract room junctions and lines from Structured3D dataset [57], and project them to top-down view to get the semantic floorplan (Fig. 2.2b).



(a) Normal floorplan. **(b)** Semantic floorplan.

Figure 2.2 Normal and semantic floorplans. (a) Normal floorplans only contain the outlines of indoor spaces, which causes ambiguity in the localization. (b) Our semantic floorplans show the room types as well as doors and windows within an indoor space according to a color-semantic map.

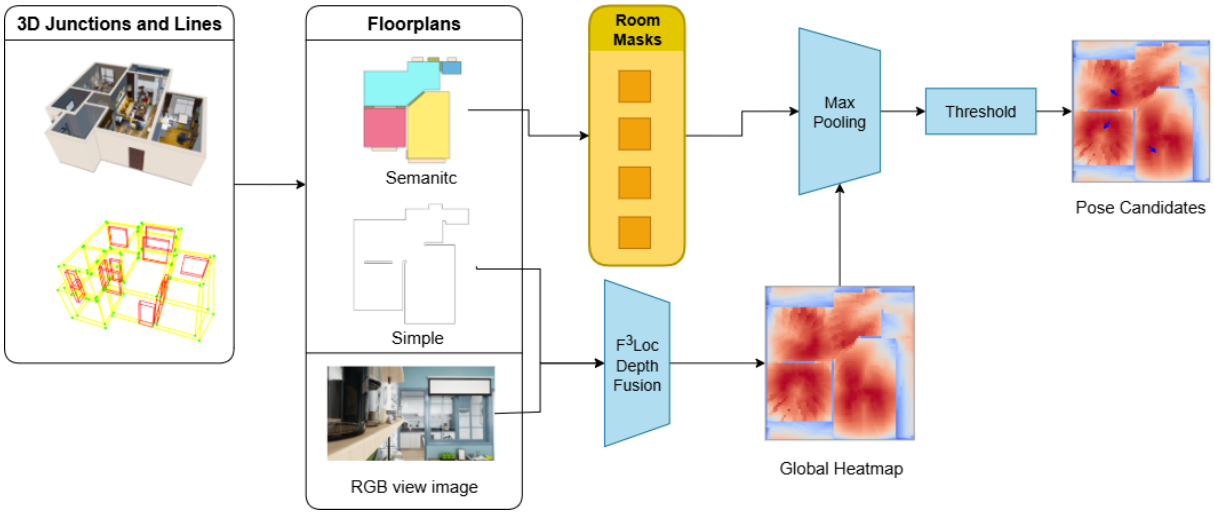


Figure 2.3 Masked Localization. F^{3}Loc performs depth fusion between simple floorplan and RBG view image. It generates a global confidence map. Then a max-pooling followed by a threshold filter is conducted in each room mask generated by semantic floorplan and pose candidates are selected in each room. The room where the maximal confidence is lower than threshold is filtered out (e.g. the blue room).

2.3.3 Masked Depth Fusion Localization

The semantic groundings on the floorplans are used as masks for depth fusion localization. Localization purely based on depth fusion suffers from ambiguity of corners and walls. It could take numerous iterations of sequential filtering until the convergence to convinced pose. To address this issue, we select pose candidates in each the semantic grounding. This is achieved by masking the groundings on the depth fusion confidence map and conduct max-pooling in each mask. During the max-pooling, we also set a threshold to filter unreasonable groundings with respect to the depth feature. This process is shown in Fig. 2.3.

2.3.4 VLM-Based Semantic Localization

Semantic understanding of floorplans based on VLM. We use Qwen2.5-7B as our VLM agent. To input the prior knowledge of the floorplan, we feed the semantics-color map (Tab. 2.1) to it. The map mainly contains the room types shown in the floorplans. *Windows* and *doors* are also labeled for better understanding of spatial connection. They are also ideal long-term features that can be extracted from the view images to assist semantic localization.

Pose selection. Our prompt for semantic localization is intuitive (Fig. 2.4. We feed the floorplan with pose candidates generated by depth fusion prediction and the corresponding mobile agent's view. A predefined semantics-color map given by Tab. 2.1 indicates which rooms are contained in the floorplan. The VLM is asked to select the best pose from the candidate.

2.4 Experiments

2.4.1 Experimental Setting

We used Gibson [47] during the debugging of F^{3}Loc [6]. This dataset has already been processed to fit the demands of F^{3}Loc [6], where simple floorplans are generated for all scenes using the outlines of 3D model of the indoor environments. View photos are taken into several sets. Each set contains the views with small interval, simulating the movement of a mobile agents. This fulfills the requirement of F^{3}Loc [6] to perform sequential inference to compensate the randomness in the single step localization.

Semantic Category	Color	Semantic Category	Color
living room		kitchen	
bedroom		bathroom	
balcony		corridor	
dining room		study	
studio		store room	
garden		laundry room	
office		basement	
garage		undefined	
door		window	
outwall			

Table 2.1 Semantic color map for floorplan labeling.

Semantic Pose Selection with Qwen2.5-7B

Text:

You are given two images:

1) A colored floorplan with candidate camera poses marked by arrows and numbered IDs.
2) The view image taken from the camera pose.

The colored floor plan image has rooms, windows and doors filled according to this color-semantic map:

{self.color_map_text}

Each candidate ID corresponds to a possible camera pose on the floorplan. Analyze the visual context and spatial layout to determine which candidate ID best matches the reference view image.

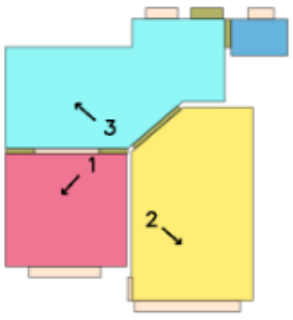
Respond with a JSON object containing only the key "selected_id" with the integer ID of the chosen candidate.

Example:

```json
{"selected_id": 3}
```

Do not include any additional commentary.

Images:



semantic_floorplan

view

Figure 2.4 VLM User Prompt. VLM is fed with semantic floorplan image with depth-fusion-predicted poses, view image and a pre-defined semantics-color map explaining the color representation used in floorplan. The VLM outputss the most-likely pose.

For the experiment of our own pipeline, we use Structured3D [57]. The rooms of different types, windows and doors are labeled in this dataset, enabling it to generate semantic floorplans. Sparse sets of view images with corresponding poses are provided for all scenes.

To implement it into our pipeline, we adjust camera poses from SE(3) to SE(2) with:

$$(x,\, y,\, \theta) = (t_x,\, t_y,\, \arctan 2(r_{21},\, r_{11})) \tag{2.1}$$

where x, y and θ are the x-, y-, orientation coordinates on the floor plane, t_x and t_y are the first two entries of the first column of rotation matrix.

The DESDFs are defined on the rectangular valid area of floorplans (e.g. the white border of the floorplans are deleted). As DESDFs are generated and expressed from the top-left corner, the position of the corner is marked as l and t for x- and y-directional biases from floorplans' top-left corner. To generate DESDFs, we use the normal floorplans (e.g. Fig. 2.2a). We transfer the floorplans to masks where entries with black pixels (wall pixels) are $true$, then calculate the Frobenius norm between the target entry (target point) and the closest true entry (wall point), the directions of detecting rays ϕ is divided into a fixed interval δ. The process is formulated as:

$$\{d_\phi = \min \|\mathbf{p}_\mathsf{t} - \mathbf{p}_\mathsf{w}\|_2 \mid \frac{\mathbf{p}_\mathsf{t} \cdot \mathbf{p}_\mathsf{w}}{\|\mathbf{p}_\mathsf{t}\| \, \|\mathbf{p}_\mathsf{w}\|} = \cos\phi,\ \phi = k\delta,\ k \in \mathbb{N}\}_\phi^{360^\circ} \tag{2.2}$$

where d_ϕ is the distance to the nearest obstacle in the direction of ϕ, $\mathbf{p}_\mathsf{t} \in \mathbb{R}^2$ and $\mathbf{p}_\mathsf{w} \in \mathbb{R}^2$ are the positions of the target point and obstacle point. For simple generalization, DESDFs are expressed with the unit of $0.1m/entry$, while the floorplans from Structured3D [57] has a unit of $0.02m/pixel$. To unify the units and compensate the bias from DESDF and floorplans,

$$x_{desdf} = 0.2 * (x_{occ} - l) \qquad y_{desdf} = 0.2 * (y_{occ} - t) \tag{2.3}$$

where x_{desdf} and y_{desdf} denote x and y coordinates of a point in DESDFs, x_{occ} and y_{occ} denote x and y coordinates of a point in floorplans, l and t denote x- and y-directional bias from DESDFs to floorplans.

2.4.2 Results

Our proposed framework consistently surpasses the baseline F^{3}Loc [6] in the single-step pose–estimation setting. Because the depth model still requires further setup, the values of the evaluation metrics are different from the ones shown in original F^{3}Loc [6] paper. Nevertheless, as summarised in Tab. 2.2, our method roughly doubles the recall across every inlier tolerance threshold, confirming a substantial margin over the baseline.

Metric	Ours	F3Loc
Mean position error (m)	4.85	5.34
Mean yaw error (°)	90.29	88.70
1 m recall	0.070	0.031
0.5 m recall	0.017	0.009
0.1 m recall	0.00047	0.00023
1 m, 30° recall	0.016	0.0082

Table 2.2 Comparison of localization accuracy metrics.

Beyond raw localisation accuracy, the integration of semantic priors yields two additional benefits:

Environment-aware convergence. By explicitly reasoning about scene semantics, the optimiser is biased towards regions whose categorical labels match the query view. Consequently, the estimated heatmap concentrates around semantically plausible areas, as illustrated in Fig. 2.5a. This behaviour is particularly valuable in texture-poor corridors or repetitive office spaces, where purely geometric cues are ambiguous.

Pose plausibility. The baseline occasionally predicts poses that fall outside the physically accessible indoor volume—for example, behind walls or in mid-air—because it relies solely on photometric consistency. In contrast, our method enforces an indoor-only prior during hypothesis sampling, effectively eliminating these outliers (see Fig. 2.5b). As a result, every reported pose lies on a traversable floor surface and obeys scene boundaries, which is critical for downstream robotics applications.

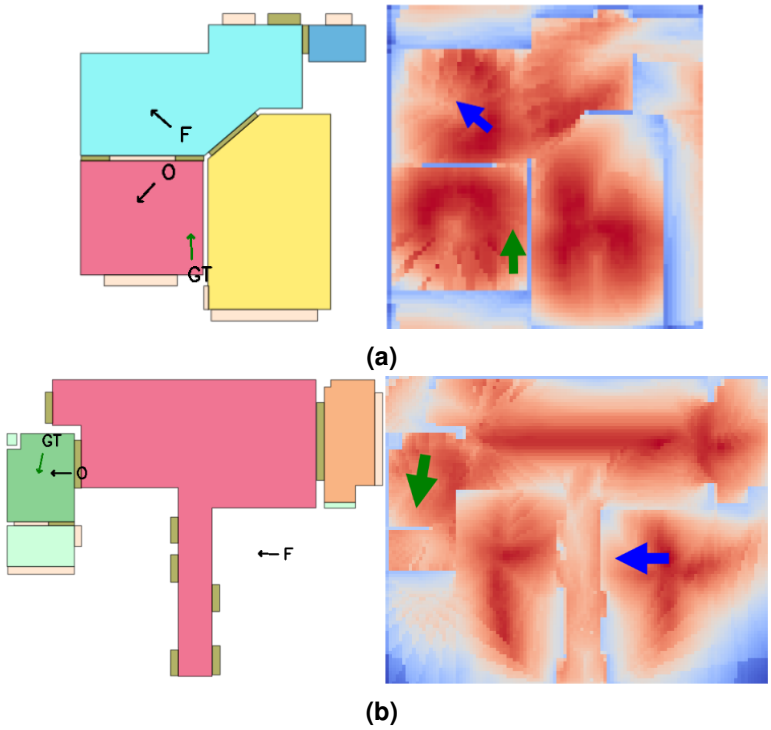


Figure 2.5 Single-step localization. Our prediction is marked as "O", F^{3}Loc prediction is marked as "F" and ground truth is marked as "GT". (a) Our prediction is more likely to locate in a semantically relevant region. (b) Our work resolves the unreasonable localization of F^{3}Loc.

2.5 Discussion

2.5.1 Limitations

Our current pipeline shows several constraints from the intuitive implementation of Qwen2.5-7B [1].

Colour discrimination in semantic floorplans. The semantic floorplans are encoded through a dense colour map in which classes may differ by similar colors. Distinguishing such subtle variations pushes Qwen2.5-7B to the edge of its colour-perception capability, triggering mis-classifications. One obvious solution is to replace colors with textual labels, but text on the floorplan occationally overlays downstream pose annotations.

Ambiguous room-type recognition. When asked to select a category from a closed set of room types, Qwen2.5-7B often resorts to vague descriptors such as “living space” rather than the precise label (“dining room”, “kitchen”, etc.). This behaviour suggests a gap between the model’s high-level scene understanding and the fine-grained semantic resolution demanded by indoor localisation. Additional domain-specific fine-tuning or contrastive supervision is therefore required.

Limited multi-image reasoning. Public benchmarks for Qwen2.5-7B are tested on the single-image single-text setting, whereas our task requires the VLM to reason jointly over a floorplan and a perspective RGB view. The degree to which Qwen2.5-7B retains cross-image correspondences is to be systematically quantified.

2.5.2 Future Work

We outline three directions to mitigate the above issues and further push the framework:

1. **Depth-aware fine-tuning on higher-fidelity data.** The Structured3D dataset [57] provides photorealistic renders and dense depth maps that surpass the visual quality of Gibson [47]. We plan to re-train the depth model on Structured3D.

2. **Room-type classifiers specialised for indoor scenes.** Two complementary strategies will be explored: (i) a CLIP-based model [56] trained on image–label pairs drawn from Structured3D; (ii) a Qwen variant further fine-tuned to map a view image directly to one of pre-defined room types. By harvesting room masks from the annotations in Structured3D, we can easily synthesise $\langle$image,mask,label$\rangle$ triplets.

3. **A two-stage VLM pipeline.** Instead of feeding floorplan and view image jointly to one VLM, we propose a two-stage architecture: *Stage 1*: the fine-tuned classifier predicts the most-likely rooms visible in the view image; *Stage 2*: a geometric localizer—constrained to the union of these candidate rooms—generates pose hypotheses, dramatically reducing the search space.

Collectively, these upgrades could enhance both the semantic fidelity and geometric robustness of our indoor localization pipeline.

3 Hierarchical Open-vocabulary Functionality Segmentation using Scene Graph for Robotic Manipulation

3.1 Introduction

Open-vocabulary 3D segmentation extracts semantic meanings from the 3D scenes without relying on a fixed set of pre-defined annotations. Prior methods for 3D segmentation have limited capability to generalize to novel classes as they are primarily trained on labelled datasets. On the contrary, open-vocabulary methods extract semantic meanings using large language models, allowing 3D exploration using free-form text descriptions. Most existing methods are dedicated to segmenting object-level instances, while only a few methods focus on the finer granularity, such as object parts. However, such fine-grained entities are crucial for real-world robotic applications for scene interaction. Assistive robotics systems require capabilities to identify not only objects in an open-set world but also their components of the lower layers, such as the chair arm, door of a washing machine or TV switches to execute general interactive tasks. Nonetheless, existing methods for this purpose perform fine-grained segmentation of nearly the whole scene with no distinction. This leads to high-demand computational expenses and memory footprints, which are not feasible for most mobile robots with limited hardware resources. Therefore, systems for such interactive applications in an open world require a more efficient design that is able to extract fine-grained segmentation in a task-dedicated manner.

Harnessing multi-modal large language models, methods like HOV-SG [45] and CLIO [27] construct hierarchical open-vocabulary 3D scene graphs focusing on high levels of semantic meanings from room over region to object. These graphs consist of merely parent-child relations between rooms and objects, floors and rooms. This tree structure is feasible for navigation but cannot be adapted to interactive tasks such as fetching the user the cup on the table. In order to complete this task, the robot should first identify the cup, for which it has to locate the table. Grounding the correct object to manipulate requires the understanding of inter-object relations in the environment.

To tackle this challenge, our method formulates a scene graph using embedded features instead of traditional labels to denote nodes and represent inter-object spatial relations with edges. Additionally, hierarchical methods preprocess image crops of every instance in the scene to construct 3D part segmentation, resulting in high memory consumption for storing the fine-grained feature representations across the whole scene. However, merely a minor from them is necessary for a specific task. Therefore, our method operates segmentation in finer granularity only on selected objects that match the input queries. This allows for efficient online segmentation without the requirement of high computational power and memory availability, making it well-suited for real-world robotics applications.

In summary, we make the following contributions:

- We present a training-free method for functionality segmentation using actionable 3D scene graphs of objects embedded with open-vocabulary semantics.

- We evaluate the open-vocabulary 3D instance segmentation performance of our method on the ScanNet dataset [8] and evaluate our method's functionality segmentation capability on the SceneFun3D [11] dataset.

3.2 Related work

3.2.1 Open-vocabulary 3D segmentation

3D instance segmentation

Recent works [38][35][11] have explored open-vocabulary 3D scene understanding. The main lines of existing methods can be categorized into two groups based on their architecture, namely, bottom-up and top-down. Bottom-up methods use per-point features distilled from per-pixel image features of the scene and obtain a point-wise scene representation. Methods such as group points with similar features into clusters to form semantic or instance segmentation. Such point-level methods such as OpenScene [32], ConceptFusion [17], LeRF [18], and OpenNeRF [12] excel in detail but lack hierarchical organization. On the other hand, top-down methods like [38][31][37][16] decompose the scene into different levels of aspects and produce a more compact scene representations such as instance-wise or part-wise scene understanding. However, they largely fall short of delivering a unified solution that supports both fine-grained segmentation and explicit hierarchical representations. For instance, object-centric approaches like OpenMask3D [37] and Open3DIS [31] offer object-level segmentation. Several works have attempted to bridge this gap. N2F2 [2] embeds hierarchical features in a 3D Gaussian splitting-based scene representation. However, these methods remain limited in the scope of open-vocabulary querying capabilities. More recent approaches introduce versatile segmentation mechanisms but still do not support an explicit, language-guided hierarchical framework. Another line of methods such as SAM3D [4] integrate 2D masks from multiple views generated by SAM [19], allowing 3D scene segmentation in varying granularity. It also uses a vision-language model to optionally assign semantic labels by using 2D renderings from different directions. However, the 2D-to-3D feature lifting and integration is not trivial and become bottleneck of these methods. Robots are edge-device and limited to the available on-board storage and computational resources. Therefore, we follow the top-down architecture and refine the instance-level scene representation to part-level understanding based on the given tasks.

3D part segmentation

Recent works in 3D part segmentation have focused on class-agnostic and non-supervised approaches that do not rely on predefined part labels or categories, as it is often expensive to obtain labeled 3D data. Additionally, 3D part segmentation of objects requires different framework approaches compared to scene-level segmentation tasks, for example Point Transformer V3 (PTv3) [46] which is designed for scene-point clouds might not be suitable for part segmentation, as smaller individual objects might need less downsampling layers [50].

Commonly, part segmentation models are trained on large scale 3D datasets like Objaverse [10], which contains over 10 million 3D objects. Even though Objaverse does not include segmentation labels on 3D objects, there are smaller scale semantic and instance level part annotations for over 200 objects such as PartObjaverse-Tiny [50].

SAMPart3D [50] is a popular framework that operates by first pre-training a 3D feature extraction backbone on large-scale unlabeled 3D objects. It then distills 2D segmentation masks from SAM to enable zero-shot 3D part segmentation and introduces a scale-conditioned MLP for multi-granularity segmentation. This allows it to segment 3D objects into parts at varying levels of detail, determined by a scale value. Additionally, AGILE3D [53] allows segmentation in varying granularity through user interactions. One other segmentation work called PartField [25] proposes a feedforward approach that learns dense 3D feature fields by using contrastive distillation from 2D segmentations, it then uses the fields for segmentation. Another approach that uses meshes instead of point clouds is Segment Any Mesh [40], which performs zero-shot segmentation directly on 3D meshes by lifting 2D masks from multi-view SAM predictions back to the mesh surface.

Across these models, a trend is the use of multimodal supervision and feature distillation from 2D models. PartSLIP [26] utilizes the image-language model GLIP [23] to perform both semantic and instance

segmentation on 3D object parts, with GLIP predicting multiple bounding boxes for each part. Building on this, PartSLIP++ [58] incorporates SAM, leading to more precise pixel-level part annotations compared to the bounding box approach used in the original PartSLIP.

The field of 3D part segmentation has several open problems, the first being the need for models to generalize effectively to novel, unannotated objects in open-world environments. Further challenges include addressing the inherent ambiguity in defining and controlling the granularity of 3D parts, as well as dealing with occluded geometries. Scan quality issues also pose a problem. These limitations highlight the need for a comprehensive method that seamlessly integrates open-vocabulary understanding with structured, multi-level scene representation.

3.2.2 Scene graph

Scene graphs represent a visual scene in graph structure, contained relations between diverse semantic information including space, object, shapes, and materials. Typically, scene graph is formed as a layered graph. Each layer represents various entities. Hierarchical scene graph has been applied widely in robot navigation and task planning. Methods like HOV-SG [45] and CLIO [27] construct hierarchical open-vocabulary 3D scene graphs focusing on high levels of semantic meanings from room over region to object. These approaches are feasible for navigation but cannot be adapted to interactive tasks such as fetching the user a chair. To tackle this limitation, our method includes fine-grained entities and formulates a scene graph using embedded features instead of traditional labels. Additionally, hierarchical methods preprocess image crops of every instance in the scene to construct 3D part segmentation, resulting in high memory consumption for storing the fine-grained feature representations across the whole scene. However, merely a minor from them is necessary for a specific task. Therefore, our method operates segmentation in finer granularity only on the interesting objects based on the analysis of the input queries. This allows for efficient online segmentation without the requirement of high computational power and memory availability, making it well-suited for real-world robotics applications.

3.2.3 Vision-language models

Vision-language models can be categorized into four groups based on their training strategies, as outlined in [3]: contrastive-based, masking-based, generative, and pretrained backbone-based approaches.

Contrastive-based models aim to align visual and textual representations by learning to bring positive image-text pairs closer in the embedding space while pushing negative pairs apart. An important example in this category is CLIP (Contrastive Language–Image Pretraining) [33], which uses a softmax-based contrastive loss. An alternative is SigLIP (Sigmoid Loss for Language-Image Pretraining) [54], which replaces the softmax loss with a sigmoid-based contrastive loss. This type of loss allows each image-text pair is to be treated as a binary classification problem. This removes the necessity to compute a full similarity matrix over the batch as in CLIP. This allows SigLIP to be trained with smaller batch sizes and less computational resources.

Masking-based models extend the idea of masked language modeling from NLP to the multimodal setting. These models typically learn to predict masked tokens given the unmasked context across both modalities. A prominent example is BEiT-3 [44], which unifies vision and language pretraining by jointly modeling image and text tokens in a single architecture using masked token prediction. FLAVA [36] also follows a similar approach by leveraging a mixture of unimodal and multimodal masking objectives to learn joint representations. These models are particularly strong in understanding fine-grained correspondences between image regions and text.

Pretrained-backbone-based models use existing language models or visual encoders rather than training from scratch. These models usually keep the LLM frozen or partially tuned and focus on learning how to connect the visual and language modalities. A common approach is to use a simple linear projection layer that maps image embeddings into the LLM's input space. For example, MiniGPT-4 [59] and LLaVA (Large Language and Vision Assistant) [24] use pretrained vision encoders (like CLIP) and frozen LLMs (like Vicuna or LLaMA) with lightweight adapters that align visual features to the language model's

input space. These models are particularly effective for instruction-following and reasoning tasks, as they leverage the strong prior knowledge in LLMs while grounding it in visual input.

3.3 Method

3.3.1 Open-vocabulary 3D instance segmentation

Open-vocabulary 3D instance segmentation poses a unique challenge as the instance masks are not associated with any explicit class labels (class-agnostic). These masks are queryable with arbitrary text descriptions. To this end, we use the open-vocabulary 3D instance segmentation method OpenMask3D [38]. In OpenMask3D, mask-features are computed using multi-level crops of the selected images that have high visibility scores. However, we noticed that the instances in these crops are often partially occluded by other objects, confusing CLIP. To improve the embedding accuracy, we include both multi-level crops and the 2D segments generated using SAM [19] in the mask-feature computation module. The additional visual information provided by the 2D segments help mitigate confusion foreground and background in the CLIP feature computation.

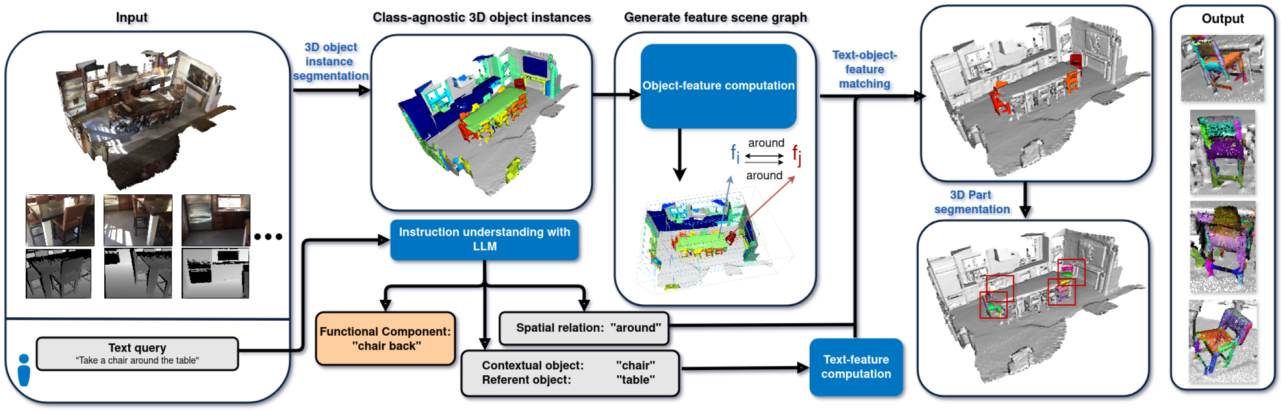


Figure 3.1 Pipeline of hierarchical open-vocabulary functionality segmentation for robotic manipulation.

3.3.2 Open-vocabulary scene graph

To construct the scene graph, we create bounding boxes around the point clouds of the proposed instances. Each node in the graph corresponds to an instance like HOV-SG [45], while the edges denoting the spatial relations are extracted by investigating the coordinates of the bounding boxes. The instances denoted in the nodes are associated with CLIP visual embeddings, which allow open-vocabulary searching. To define relative spatial expression such as "left", "right", "in front of", and "behind", we refer to the inertial coordinate system attached to the whole scene reconstruction.

3.3.3 Query understanding using LLM

The task description given in the query $\mathcal{Q}$ attributes different complexities. For example, in the case of *"Control the temperature using the radiator dial next to the french door"*, there is a spatial expression describing the position of the *radiator*, the contextual object $\mathcal{O}$, and the *french door*, a referent object $\mathcal{R}$. The functional component $\mathcal{F}$, *dial*, is also explicitly mentioned in this example. However, $\mathcal{F}$ are mostly hidden in the query $\mathcal{Q}$. For instance, when assigned the task *"Open the door"*, the agent should be able to infer that *handle* or *knob* on the door are the objects to manipulate.

In this work, we focus on task descriptions that explicitly or implicitly consist of **space contexts**, **spatial relations**, **contextual object**, and **functional component**. To extract this information, we use the LLM to parse the task descriptions. The sole output of individual information categories can be ambiguous. For

example, a *knob* can be part of a *door* or part of a *drawer*. Similarly, a *door* can have either a *handle* or a *knob*. Moreover, if there are multiple instances of the same class in the environment, e.g., a room with two doors, the agent would be confused about which door it should open. In order to mitigate these ambiguities, we query the LLM to provide answers including multiple categories in one JSON output. The query is performed in a multi-turn conversation style using the prompt in Fig 3.2.

Instruction understanding with QWen3-14B in multi-turn conversation

system_prompt:
You are an AI system that generates JSON instructions for a robot to interact with physical objects based on a natural language command. Your goal is to identify the object part to act on and the task-solving sequence. The robot's possible actions are *[rotate, key_press, tip_push, hook_pull, pinch_pull, hook_turn, foot_push, plug_in, unplug]*. Do NOT use rooms (e.g., bedroom, kitchen) as referents or in spatial relations. Only include concrete physical objects or furniture such as drawers, cabinets, TVs, lamps, etc. Keep the objects' adjectives that describe the object's properties [color, material, texture, shape] if there are any, such as blue chair, leather couch, wooden door, etc. Spatial relations and object hierarchies must be **between manipulable or visible objects only**, not rooms or spaces. You must output only valid JSON.

statement:
How do I {query}?
Respond with only the JSON in the following format:
{{

"**prompt**": "original natural language prompt",
"**space**": "room or space (e.g., *bedroom, kitchen, bathroom*) where the object is located, if mentioned",
"**task_solving_sequence**": "a list of strings with the description of what I have to do to accomplish the task described by the prompt, subdivided into subtasks",
"**location_context**": "'true' if any location information is present (but not involving rooms); 'false' otherwise. Output as a string",
"**spatial_relation**": "if 'location_context' is 'true', give a chain of spatial relations ONLY between physical objects (e.g., [*Cabinet is under the TV*]). Otherwise, give an empty list",
"**referent_object_hierarchy**": "list of physical objects used to locate the acted-on part (excluding rooms)",
"**acted_on_object**": "the specific part to act on (e.g., *handle, button, knob*)",
"**acted_on_object_hierarchy**": "list of objects from top-level to the specific part. The low-level object must be a physical part of the high-level object (e.g., [*black cabinet, handle*]). Do NOT include objects that are merely adjacent, nearby, or spatially related."

}}

Figure 3.2 System and user prompts input to the LLM to parse the task instructions.

3.3.4 Contextual object grounding

As illustrated in Figure 3.3, we ground contextual objects by leveraging an open-vocabulary scene graph to resolve object references in complex environments. Given a reference object's textual description and the target contextual object's textual query, we extract their respective CLIP embeddings. We first identify candidate nodes in the scene graph based on semantic and spatial constraints, then refine the selection by computing the similarity between the text embeddings. The most relevant object instance is selected based on the highest similarity score.

For example, when a query refers to "the cabinet next to the desk," and the scene contains two visually similar cabinets, the scene graph which has spatial relationships helps clarify which cabinet is being referenced. By grounding the context ("next to the desk"), we can resolve the correct instance mask that corresponds to the language query.

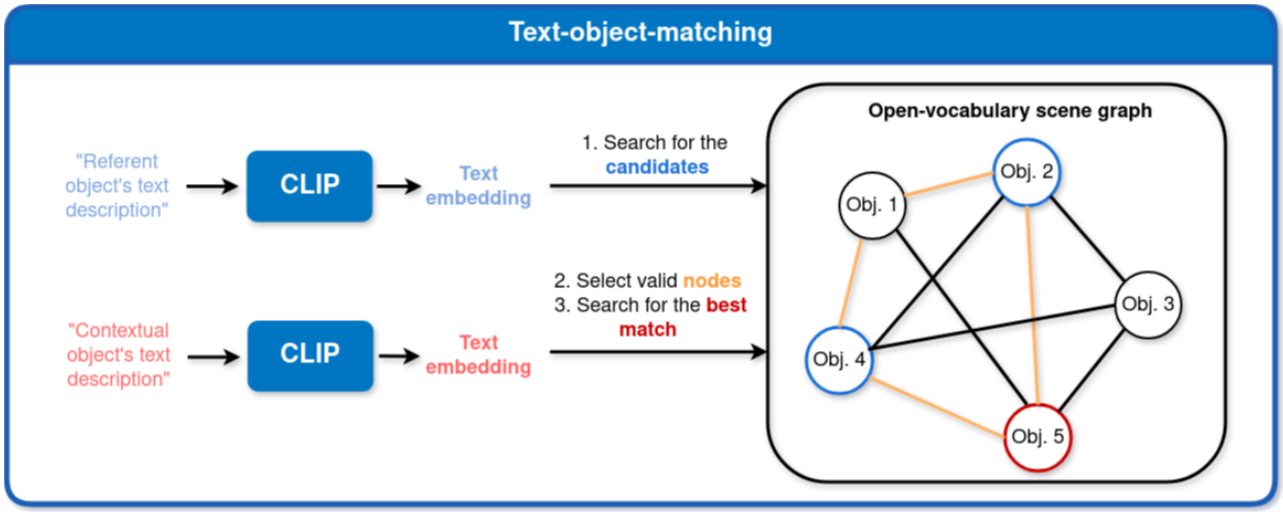


Figure 3.3 Grounding contextual object by searching the valid nodes and computing the similarity of descriptions' text embeddings and objects' visual embeddings.

3.3.5 Functional component segmentation

For part segmentation on the instance masks coming from the open-vocabulary 3D instance segmentation section, we use PartField [25], which operates by learning a continuous 3D feature field from an input 3D shape in a feedforward manner. This feature field encodes the underlying structure of parts and their hierarchy. The idea is that points belonging to the same part will have similar features in this latent space, while points from different parts will have distinct features. This allows for a flexible definition of parts without relying on predefined templates.

The training of PartField involves distilling part proposals from both 2D and 3D data. For 2D proposals, the model leverages large-scale unlabeled datasets by rendering multi-view images of 3D shapes and applying 2D foundation models like SAM to predict class-agnostic 2D masks. These masks are then projected back onto the 3D shape. For 3D proposals, existing 3D datasets with part annotations are used. These 2D and 3D proposals provide hints about which points should be grouped together.

A key aspect of PartField's training is its use of a triplet-based contrastive learning objective. This objective encourages positive point pairs (from the same part) to have closer features than negative point pairs (from different parts). This approach helps to overcome challenges related to varying part granularity and inconsistent part labels across different datasets. The model also employs a hard negative sampling strategy to improve training efficiency and the sharpness of part boundaries.

Once trained, PartField can be applied to various 3D modalities, including meshes, point clouds, and 3D Gaussian splats, in our case, we use point clouds. At inference, the model generates the feature field, which can then be clustered using k-means to yield a part decomposition of the 3D shape. This process is significantly faster than prior methods, as it only requires a single feedforward pass, which makes it more suitable for our pipeline with time concerns. Additionaly, the fact that PartField has been trained on Objaverse [10] makes it promising for our pipeline which contains mostly furniture, which Objaverse already contains.

3.4 Experiments

3.4.1 Experiment setup

Implementation

T-Search3D is a training-free method leveraging several pre-trained models. For class-agnostic instance segmentaion in Sec 3.3.1, we employ Mask3D [35] and compute image features using CLIP [33], following

OpenMask3D [38]. Different from OpenMask3D, we additionally compute visual features of the 2D masks of the instances that is extracted by SAM [19]. These masks provide only visual information of instances themselves. Taking their CLIP features into account can help prevent foreground biases.

We apply Qwen3-14B to analyze the task descriptions. Specifically, we disable the thinking capability of Qwen and query in multi-turn conversations to allow faster inference without sacrificing the output accuracy. Even though, the LLM inference takes around 4.5 s for arbitrary tasks descriptions in one sentence, which becomes the efficiency bottleneck of our method. The scene graph searching and contextual object querying have real-time performance ($\sim 1-2$ ms) for the extracted task ontology.

After obtaining the instance masks (e.g., for objects like a door, cabinet, or chair), we proceed to the part segmentation module. Our goal in this stage is to generate clean and semantically meaningful part segmentation masks. To achieve this, we utilize PartField [25], a model capable of producing masks at multiple levels of granularity. This allows us to specify the number of parts we want per object. PartField is also efficient in practice, with a reported inference time of around 10 seconds for objects containing $100k$ points, which aligns well with the size of the instance masks we extract. These instance masks are subsets of the full scene, which typically consists of several million points.

Once we obtain the part segmentation masks, we render the 3D parts into 2D images and use a vision-language model (VLM) to determine which part is relevant to the user's query. Instead of relying on the VLM to directly ground the desired part in raw image pixels—as done in methods like Fun3DU [11], which projects part predictions from multi-view images into 3D, we follow a different approach. We first use the segmentation model to extract all parts, and then let the VLM identify the part of interest based on the query. This decouples segmentation from grounding. PartField is used for high-quality geometric decomposition, and the VLM is used for semantic understanding.

To interface with the VLM, we use simple visual markers, such as assigning distinct colors or numeric labels to each segmented part in the 2D rendering. These references make it easy for the VLM to distinguish between parts and to identify the one most relevant to the query.

Dataset

Following other exiting works, we evaluate our method on SceneFun3D [11]. This is the only existing dataset that allows evaluation of functionality segmentation in 3D scenes. It provides annotations for functional components along with a comprehensive collection of various tasks in home environments. It provides over 14,800 annotated functional interactive elements across 710 high-resolution point clouds obtained via Faro laser scans. These annotations are complemented by nine Gibsonian affordance categories (e.g., rotate, hook pull), natural language task descriptions (e.g., open the fridge), and motion parameters defining how to physically interact with each element (e.g., push, pull, rotate).

The dataset supports three key tasks: (1) Functionality Segmentation for detecting interactive elements and their affordances in 3D scenes; (2) Task-Driven Affordance Grounding for locating the correct interaction point based on natural language instructions; and (3) 3D Motion Estimation for predicting how to move or manipulate an element. To support these tasks, the dataset introduces detailed annotations and a benchmarking suite with baseline models. These include adaptations of Mask3D [35], SoftGroup [42], and open-vocabulary methods like LERF [18] and OpenMask3D [38]. Results show that existing models struggle with detecting small, functionally-relevant regions which highlights the difficulty of these tasks and the potential for future improvement through affordance-aware and language-guided 3D perception.

Evaluation metrics

As suggested in SceneFun3D [11], we measure different metrics, including the Average Precision at Intersection over Union (IoU) thresholds of 0.25 (AP_{25}) and 0.5 (AP_{50}), and the mean Average Precision (mAP) averaging over IoU thresholds from 0.5 to 0.95 with a step of 0.05. In addition, we also report the Average Recall, including AR_{25}, AR_{50}, and mAR of the same conditions as of the Average Precision.

Figure 3.4 Example scene from SceneFun3D [11] dataset.

Baselines

We use Fun3DU [7] as the main baseline for comparison on the functionality segmentation tasks. Additionally, we report performance of selected open-vocabulary 3D segmentation methods, including OpenMask3D [38] and OpenIns3D [16]. We test OpenMask3D on SceneFun3D [11] without retraining, with the original implementation and the released checkpoint trained on ScanNet200 [8]. Instead of running OpenIns3D [16] on SceneFun3D by ourselves, we reference the results reported in Fun3DU [7], as they adapt OpenIns3D's implementation specifically for evaluation on SceneFun3D. Similar to Fun3DU [7], we use the original task descriptions as input.

3.4.2 Mask3D masks on SceneFun3D dataset

As the first step of our pipeline, we use a 3D mask proposal network from Mask3D [35], similar to the pipeline in OpenMask3D. Figure 3.5 shows two masks from the scene in Figure 3.4. As a result of this initial processing, the model typically gives over 300 instance masks per SceneFun3D scene, which typically has 5 million points. Visual inspection, exemplified by figures showing masks with the largest point counts (e.g., a door and a sofa), shows the framework's ability to identify prominent objects. However, despite the generation of numerous instance masks, we observed limitations such as unclear edges, fragmented object parts (e.g., lossy cushion areas on a sofa), and a significant number of extraneous masks that do not correspond to meaningful objects.

3.4.3 Part segmentation on instance masks from SceneFun3D dataset

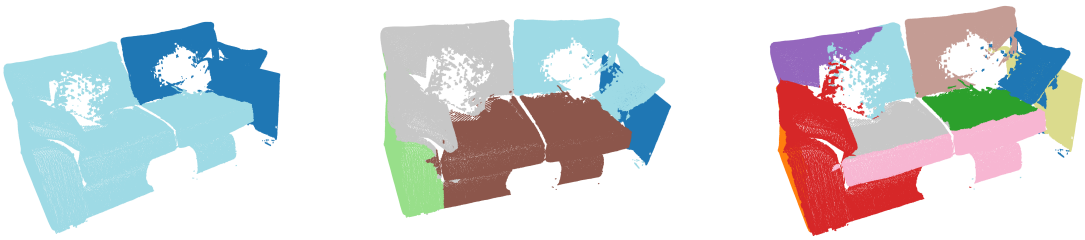
As the next step in the pipeline, the instance masks coming from the mask proposal network have to be segmented. In order to obtain part segmentation masks for the objects, we utilize PartField [25], as explained in 3.3.5.

In Figures 3.7 and 3.6 are the part segmentation results for the sofa and the door from Figure 3.5 for different number of segmentation masks. In the case for the sofa, we see that the parts are meaningfully segmented and are only obstructed by the cushion. For the case of the door, the segmentation mask is less successful and non-meaningful masks are returned. We leave the automatic selection of the number of mask and increasing the accuracy for the meaningful parts for future work.

3.4.4 VLM understanding of part segmentation masks

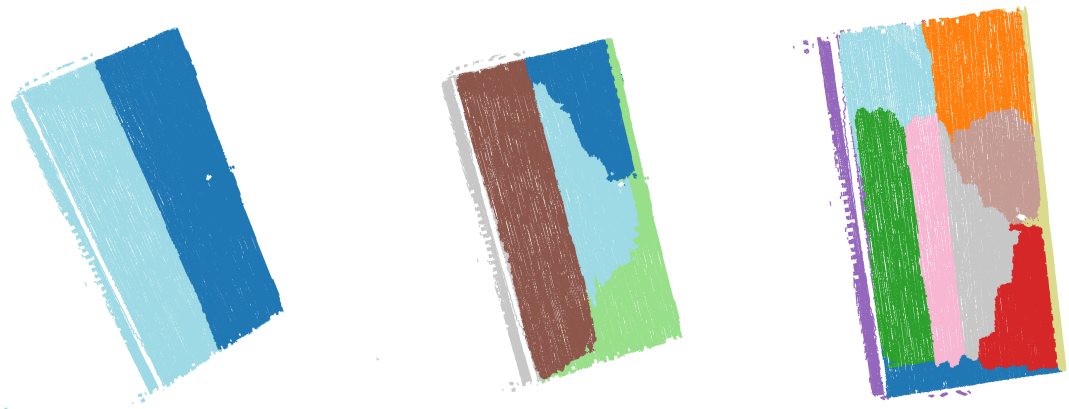
Figure 3.8 illustrates the ability of Qwen2.5-VL-3B to interpret rendered 3D segmentation masks of objects. The input images, adapted from the SAMPart3D [51] dataset, visually resemble our part segmentation outputs. Each example pairs an image with a natural language query and the model's response. The

Figure 3.5 Two segmentation masks corresponding to a sofa and the door in one of the scenes in SceneFun3D.



(a) Mask number = 2. **(b)** Mask number = 5. **(c)** Mask number = 10.

Figure 3.6 Part segmentation masks for different granularities of the sofa in Figure 3.5.



(a) Mask number = 2. **(b)** Mask number = 5. **(c)** Mask number = 10.

Figure 3.7 Part segmentation masks for different granularities of the door in Figure 3.5.

textual prompts reference either colors, numbers, or both—reflecting different levels of ambiguity in spatial and semantic grounding. The model successfully identifies parts based on either visual color cues or numeric labels embedded in the segmentation mask. However, we found number-based references to be more reliable, as they reduce ambiguity compared to color-based descriptions alone.

In the first example in Figure 3.8, the model correctly identifies the "blue part" as the region that would contact a person's back when seated. In the second case, it enhances this by grounding the same region to both its color and its part number ("part 1"). Finally, the third example shows a query requiring multiple region identifications ("the area of the wheels"), where the model accurately associates the wheel components with their corresponding part numbers and colors (numbers 6, 7, and 8 in yellow), demonstrating both spatial reasoning and mask-level understanding.

3.5 Discussion

3.5.1 Conclusion

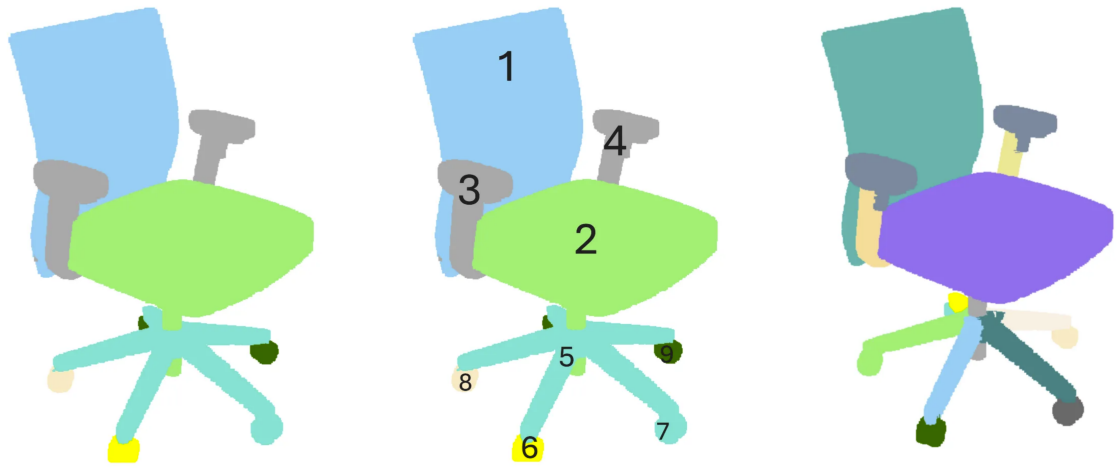
The modular design of T-Search3D's pipeline offers flexibility to replace modules to further improve the system's performance. However, the pervious modules' output becomes the bottleneck for the following modules. For instance, the number of nodes in the open-vocabulary scene graph depends on the number of instances we obtain in the instance segmentation. Moreover, the quality of instance masks places a limit on the accuracy and even the success rate of part segmentation. Therefore, it is worth revisiting the instance segmentation module, finetuning OpenMask3D [39], or replacing it with other methods.

Similar to existing works, in T-Search3D, we construct scene graphs by investigating the positional relations of instances' bounding boxes. Our scene graph is attached to a predefined world coordinate system. The spatial relations encoded in graph edges are also extracted referent to the world frame. To query the scene graph, we have to align the robot's frame with the world frame, which is a difficult problem itself. Utilizing VLMs to create open-vocabulary scene graphs from either RGB images or 3D reconstruction of the scene is a potential direction to explore in the future, posing unique contribution to the field of computer vision and robotics.

3.5.2 Future Work

T-Search3D is intended to work as the perception module for robot manipulation in the real world. Grounding the contextual object and its functional component in the environment allows the robot navigate to the target and manipulate the functional element to execute the given tasks. Deploying experiments using real robots in the realistic settings is one of the important work left for the future.

VLM querying with 2D renderings of the part segmentation masks



input: "You are seeing the rendering of 3D segmentation masks on an item.
Which part of the item does my back touch if I sit on the chair?
Indicate the color of the part."
answer: 'If you sit on the chair, your back would touch the blue part of the chair.'

input: "You are seeing the rendering of 3D segmentation masks on an item.
Which part of the item does my back touch if I sit on the chair?
Indicate the color and the number of the part."},
answer: 'Your back touches the blue part of the chair, which is labeled as part 1.

input: "You are seeing the rendering of 3D segmentation masks on an item.
Which numbers are written on the area of the wheels?
Indicate the color and the numbers of the parts."},
answer: 'The numbers written on the area of the wheels are:\n\n-
Number 8: Yellow\n- Number 6: Yellow\n- Number 7: Yellow\n\n
These numbers are located on the bottom part of the chair,
which is typically where the wheels are attached.'

Figure 3.8 Early experiments on querying Qwen2.5-VL-3B with the 2D renderings of part segmentation masks on point clouds in order to return the final part segmentation mask according to the text description.

4 Conclusion

This report details a research project in the scope of TUM Data Innovation Lab, focusing on exploring the usage of LLM and VLM for navigation and manipulation. The project separates into two sub-projects. One of them focuses on the localization for indoor navigation, while the other investigates functionality segmentation for robot manipulation. They are both integral building blocks for robot application in the real world, while each project poses unique challenges. In this report, we elaborate on the methods to tackle localization and functionality segmentation followed by experiment results to justify our method design choice. These intermediate results indicate that we are on the right track and have a good progress. As next step, we will complete the implementation of our methods and conduct comprehensive experiments to evaluate them qualitatively and quantitatively.

Bibliography

[1] Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, Binyuan Hui, Luo Ji, Mei Li, Junyang Lin, Runji Lin, Dayiheng Liu, Gao Liu, Chengqiang Lu, Keming Lu, Jianxin Ma, Rui Men, Xingzhang Ren, Xuancheng Ren, Chuanqi Tan, Sinan Tan, Jianhong Tu, Peng Wang, Shijie Wang, Wei Wang, Shengguang Wu, Benfeng Xu, Jin Xu, An Yang, Hao Yang, Jian Yang, Shusheng Yang, Yang Yao, Bowen Yu, Hongyi Yuan, Zheng Yuan, Jianwei Zhang, Xingxuan Zhang, Yichang Zhang, Zhenru Zhang, Chang Zhou, Jingren Zhou, Xiaohuan Zhou, and Tianhang Zhu. Qwen technical report.

[2] Yash Bhalgat, Iro Laina, João F. Henriques, Andrew Zisserman, and Andrea Vedaldi. N2f2: Hierarchical scene understanding with nested neural feature fields, 2024.

[3] Florian Bordes, Richard Yuanzhe Pang, Anurag Ajay, Alexander C. Li, Adrien Bardes, Suzanne Petryk, Oscar Mañas, Zhiqiu Lin, Anas Mahmoud, Bargav Jayaraman, Mark Ibrahim, Melissa Hall, Yunyang Xiong, Jonathan Lebensold, Candace Ross, Srihari Jayakumar, Chuan Guo, Diane Bouchacourt, Haider Al-Tahan, Karthik Padthe, Vasu Sharma, Hu Xu, Xiaoqing Ellen Tan, Megan Richards, Samuel Lavoie, Pietro Astolfi, Reyhane Askari Hemmat, Jun Chen, Kushal Tirumala, Rim Assouel, Mazda Moayeri, Arjang Talattof, Kamalika Chaudhuri, Zechun Liu, Xilun Chen, Quentin Garrido, Karen Ullrich, Aishwarya Agrawal, Kate Saenko, Asli Celikyilmaz, and Vikas Chandra. An introduction to vision-language modeling, 2024.

[4] Nhat-Tan Bui, Dinh-Hieu Hoang, Minh-Triet Tran, Gianfranco Doretto, Donald Adjeroh, Brijesh Patel, Arabinda Choudhary, and Ngan Le. Sam3d: Segment anything model in volumetric medical images, 2024.

[5] Vicente Vivanco Cepeda, Gaurav Kumar Nayak, and Mubarak Shah. GeoCLIP: Clip-inspired alignment between locations and images for effective worldwide geo-localization.

[6] Changan Chen, Rui Wang, Christoph Vogel, and Marc Pollefeys. F^{3}loc: Fusion and filtering for floorplan localization.

[7] Jaime Corsetti, Francesco Giuliari, Alice Fasoli, Davide Boscaini, and Fabio Poiesi. Functionality understanding and segmentation in 3D scenes. *ArXiv*, abs/2411.16310:null, 2024.

[8] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017.

[9] David DeFazio, Hrudayangam Mehta, Jeremy Blackburn, and Shiqi Zhang. Vision language models can parse floor plan maps.

[10] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects, 2022.

[11] Alexandros Delitzas, Ayca Takmaz, Federico Tombari, Robert Sumner, Marc Pollefeys, and Francis Engelmann. Scenefun3d: Fine-grained functionality and affordance understanding in 3d scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14531–14542, 2024.

[12] Francis Engelmann, Fabian Manhardt, Michael Niemeyer, Keisuke Tateno, Marc Pollefeys, and Federico Tombari. Opennerf: open set 3d neural scene segmentation with pixel-wise features and rendered novel views. *arXiv preprint arXiv:2404.03650*, 2024.

[13] Yixiao Feng, Zhou Jiang, Yongliang Shi, Yunlong Feng, Xiangyu Chen, Hao Zhao, and Guyue Zhou. Block-map-based localization in large-scale environment.

[14] Chenguang Huang, Oier Mees, Andy Zeng, and Wolfram Burgard. Visual language maps for robot navigation.

[15] Jiacui Huang, Hongtao Zhang, Mingbo Zhao, and Zhou Wu. IVLMap: Instance-aware visual language grounding for consumer robot navigation.

[16] Zhening Huang, Xiaoyang Wu, Xi Chen, Hengshuang Zhao, Lei Zhu, and Joan Lasenby. OpenIns3D: Snap and Lookup for 3D Open-vocabulary Instance Segmentation. *ArXiv*, abs/2309.00616:null, 2023.

[17] Krishna Murthy Jatavallabhula, Alihusein Kuwajerwala, Qiao Gu, Mohd Omama, Tao Chen, Alaa Maalouf, Shuang Li, Ganesh Iyer, Soroush Saryazdi, Nikhil Keetha, et al. Conceptfusion: Open-set multimodal 3d mapping. *arXiv preprint arXiv:2302.07241*, 2023.

[18] Justin Kerr, Chung Min Kim, Ken Goldberg, Angjoo Kanazawa, and Matthew Tancik. Lerf: Language embedded radiance fields. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 19729–19739, 2023.

[19] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4015–4026, 2023.

[20] Hongyu Li, Jinyu Chen, Ziyu Wei, Shaofei Huang, Tianrui Hui, Jialin Gao, Xiaoming Wei, and Si Liu. LLaVA-ST: A multimodal large language model for fine-grained spatial-temporal understanding.

[21] Jiaxin Li, Weiqi Huang, Zan Wang, Wei Liang, Huijun Di, and Feng Liu. FloNa: Floor plan guided embodied visual navigation.

[22] Ling Li, Yu Ye, Bingchuan Jiang, and Wei Zeng. GeoReasoner: Geo-localization with reasoning in street views using a large vision-language model.

[23] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, Kai-Wei Chang, and Jianfeng Gao. Grounded language-image pre-training, 2022.

[24] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023.

[25] Minghua Liu, M. Uy, Donglai Xiang, Hao Su, Sanja Fidler, Nicholas Sharp, and Jun Gao. PARTFIELD: Learning 3D Feature Fields for Part Segmentation and Beyond. 2025.

[26] Minghua Liu, Yinhao Zhu, Hong Cai, Shizhong Han, Zhan Ling, Fatih Porikli, and Hao Su. Partslip: Low-shot part segmentation for 3d point clouds via pretrained image-language models, 2023.

[27] Dominic Maggio, Yun Chang, Nathan Hughes, Matthew Trang, Dan Griffith, Carlyn Dougherty, Eric Cristofalo, Lukas Schmid, and Luca Carlone. Clio: Real-time task-driven open-set 3d scene graphs. *IEEE Robotics and Automation Letters*, 2024.

[28] Hidenobu Matsuki, Riku Murai, Paul H. J. Kelly, and Andrew J. Davison. Gaussian splatting SLAM.

[29] Shigemichi Matsuzaki, Takuma Sugino, Kazuhito Tanaka, Zijun Sha, Shintaro Nakaoka, Shintaro Yoshizawa, and Kazuhiro Shintani. CLIP-loc: Multi-modal landmark association for global localization in object-based maps.

[30] Shigemichi Matsuzaki, Kazuhito Tanaka, and Kazuhiro Shintani. CLIP-clique: Graph-based correspondence matching augmented by vision language models for object-based global localization. 9(11):10399–10406.

[31] Phuc D. A. Nguyen, Tuan Duc Ngo, Evangelos Kalogerakis, Chuang Gan, Anh Tran, Cuong Pham, and Khoi Nguyen. Open3dis: Open-vocabulary 3d instance segmentation with 2d mask guidance, 2024.

[32] Songyou Peng, Kyle Genova, Chiyu Jiang, Andrea Tagliasacchi, Marc Pollefeys, Thomas Funkhouser, et al. Openscene: 3d scene understanding with open vocabularies. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 815–824, 2023.

[33] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021.

[34] Paul-Edouard Sarlin, Daniel DeTone, Tsun-Yi Yang, Armen Avetisyan, Julian Straub, Tomasz Malisiewicz, Samuel Rota Bulo, Richard Newcombe, Peter Kontschieder, and Vasileios Balntas. OrienterNet: Visual localization in 2d public maps with neural matching.

[35] Jonas Schult, Francis Engelmann, Alexander Hermans, Or Litany, Siyu Tang, and Bastian Leibe. Mask3d: Mask transformer for 3d semantic instance segmentation. *arXiv preprint arXiv:2210.03105*, 2022.

[36] Amanpreet Singh, Ronghang Hu, Vedanuj Goswami, Guillaume Couairon, Wojciech Galuba, Marcus Rohrbach, and Douwe Kiela. Flava: A foundational language and vision alignment model, 2022.

[37] Ayca Takmaz, Alexandros Delitzas, Robert W. Sumner, Francis Engelmann, Johanna Wald, and Federico Tombari. Search3D: Hierarchical Open-Vocabulary 3D Segmentation. *IEEE Robotics and Automation Letters*, 10:2558–2565, 2024.

[38] Ayça Takmaz, Elisabetta Fedele, Robert W Sumner, Marc Pollefeys, Federico Tombari, and Francis Engelmann. Openmask3d: Open-vocabulary 3d instance segmentation. *arXiv preprint arXiv:2306.13631*, 2023.

[39] Ayça Takmaz, Elisabetta Fedele, Robert W. Sumner, Marc Pollefeys, Federico Tombari, and Francis Engelmann. OpenMask3D: Open-Vocabulary 3D Instance Segmentation, October 2023. arXiv:2306.13631 [cs].

[40] George Tang, William Zhao, Logan Ford, David Benhaim, and Paul Zhang. Segment any mesh, 2025.

[41] Xinyu Tian, Shu Zou, Zhaoyuan Yang, and Jing Zhang. Identifying and mitigating position bias of multi-image vision-language models.

[42] Thang Vu, Kookhoi Kim, Tung M. Luu, Xuan Thanh Nguyen, and Chang D. Yoo. Softgroup for 3d instance segmentation on point clouds, 2022.

[43] Muntasir Wahed, Kiet A. Nguyen, Adheesh Sunil Juvekar, Xinzhuo Li, Xiaona Zhou, Vedant Shah, Tianjiao Yu, Pinar Yanardag, and Ismini Lourentzou. PRIMA: Multi-image vision-language models for reasoning segmentation.

[44] Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, and Furu Wei. Image as a foreign language: Beit pretraining for all vision and vision-language tasks, 2022.

[45] Abdelrhman Werby, Chenguang Huang, Martin Büchner, Abhinav Valada, and Wolfram 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.

[46] Xiaoyang Wu, Li Jiang, Peng-Shuai Wang, Zhijian Liu, Xihui Liu, Yu Qiao, Wanli Ouyang, Tong He, and Hengshuang Zhao. Point transformer v3: Simpler, faster, stronger, 2024.

[47] Fei Xia, Amir Zamir, Zhi-Yang He, Alexander Sax, Jitendra Malik, and Silvio Savarese. Gibson env: Real-world perception for embodied agents.

[48] Guowei Xu, Peng Jin, Ziang Wu, Hao Li, Yibing Song, Lichao Sun, and Li Yuan. LLaVA-CoT: Let vision language models reason step-by-step.

[49] Ziang Yan, Zhilin Li, Yinan He, Chenting Wang, Kunchang Li, Xinhao Li, Xiangyu Zeng, Zilei Wang, Yali Wang, Yu Qiao, Limin Wang, and Yi Wang. Task preference optimization: Improving multimodal large language models with vision task alignment.

[50] Yunhan Yang, Yukun Huang, Yuan-Chen Guo, Liangjun Lu, Xiaoyang Wu, Edmund Y. Lam, Yan-Pei Cao, and Xihui Liu. Sampart3d: Segment any part in 3d objects, 2024.

[51] Yunhan Yang, Yukun Huang, Yuan-Chen Guo, Liangjun Lu, Xiaoyang Wu, Edmund Y. Lam, Yan-Pei Cao, and Xihui Liu. SAMPart3D: Segment Any Part in 3D Objects, November 2024. arXiv:2411.07184 [cs].

[52] Naoki Yokoyama, Sehoon Ha, Dhruv Batra, Jiuguang Wang, and Bernadette Bucher. VLFM: Vision-language frontier maps for zero-shot semantic navigation.

[53] Yuanwen Yue, Sabarinath Mahadevan, Jonas Schult, Francis Engelmann, Bastian Leibe, Konrad Schindler, and Theodora Kontogianni. Agile3d: Attention guided interactive multi-object 3d segmentation. *arXiv preprint arXiv:2306.00977*, 2023.

[54] Xiaohua Zhai, Basil Mustafa, Alexander Kolesnikov, and Lucas Beyer. Sigmoid loss for language image pre-training, 2023.

[55] Lingfeng Zhang, Xiaoshuai Hao, Qinwen Xu, Qiang Zhang, Xinyao Zhang, Pengwei Wang, Jing Zhang, Zhongyuan Wang, Shanghang Zhang, and Renjing Xu. MapNav: A novel memory representation via annotated semantic maps for vision-and-language navigation.

[56] Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D. Manning, and Curtis P. Langlotz. Contrastive learning of medical visual representations from paired images and text.

[57] Jia Zheng, Junfei Zhang, Jing Li, Rui Tang, Shenghua Gao, and Zihan Zhou. Structured3d: A large photo-realistic dataset for structured 3d modeling.

[58] Yuchen Zhou, Jiayuan Gu, Xuanlin Li, Minghua Liu, Yunhao Fang, and Hao Su. Partslip++: Enhancing low-shot 3d part segmentation via multi-view instance segmentation and maximum likelihood estimation, 2023.

[59] Deyao Zhu, Jun Chen, Xiaoqian Shen, Xiang Li, and Mohamed Elhoseiny. Minigpt-4: Enhancing vision-language understanding with advanced large language models, 2023.

[60] Ke Zhu, Yu Wang, Jiangjiang Liu, Qunyi Xie, Shanshan Liu, and Gang Zhang. On data synthesis and post-training for visual abstract reasoning.