2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

T-FunS3D: Task-Driven Hierarchical Open-Vocabulary 3D Functionality Segmentation

Jingkun Feng[1] and Reza Sabzevari[1]

***Abstract*— Open-vocabulary 3D functionality segmentation enables robots to localize functional object components in 3D scenes. It is a challenging task that requires spatial understanding and task interpretation. Current open-vocabulary 3D segmentation methods primarily focus on object-level recognition, while scene-wide part segmentation methods attempt to segment the entire scene exhaustively, making them highly resource-intensive and time consuming. Balancing segmentation performance in terms of granularity, accuracy, and speed remains a challenge. As one step towards alleviating this, we introduce T-FunS3D, a task-driven hierarchical open-vocabulary 3D functionality segmentation method that provides actionable perception for robotic applications. Our method takes as input the 3D point cloud and posed RGB-D images of an indoor scene. We construct an open-vocabulary scene graph by extracting instances and their visual embeddings in the environment. Given a task description, T-FunS3D identifies the most relevant instances in the scene graph and locates their functional components leveraging a vision-language model. Experiments on the SceneFun3D dataset demonstrate that T-FunS3D is comparable to state-of-the-art in open-vocabulary 3D functionality segmentation, while achieving faster runtime and reduced memory usage.**

—Supplementary materials and code: t-funs3d.github.io.

I. INTRODUCTION

Open-vocabulary 3D functionality segmentation extracts the semantic meaning and location of functional interactive elements in 3D scenes without relying on a fixed set of predefined annotations. This capability provides robots with both concrete objects and their functional parts, enabling them to execute the assigned tasks. Task-driven functional object segmentation, also referred to as task-driven affordance grounding in [1], predicts the masks associated to Gibsonian affordances [2] inferred from telic affordances [3] given as free-form task descriptions. It poses a unique challenge that requires both the physical understanding of the functionalities of objects and the interpretation of complex linguistic expressions. To tackle this challenge, this paper proposes a novel pipeline for hierarchical scene understanding and functionality segmentation driven by tasks using an open-vocabulary scene graph and vision-language models (VLMs).

Conventional 3D semantic understanding methods are primarily trained on labeled datasets to recognize a considerable number of object classes. Using large-scale annotations, these models work well for close-set instance segmentation. However, because they are trained in a supervised or semi-supervised manner, they fall short of generalizing to novel object classes. In contrast, open-vocabulary methods leverage large language models to infer scene semantics, supporting 3D exploration guided by free-form text descriptions.

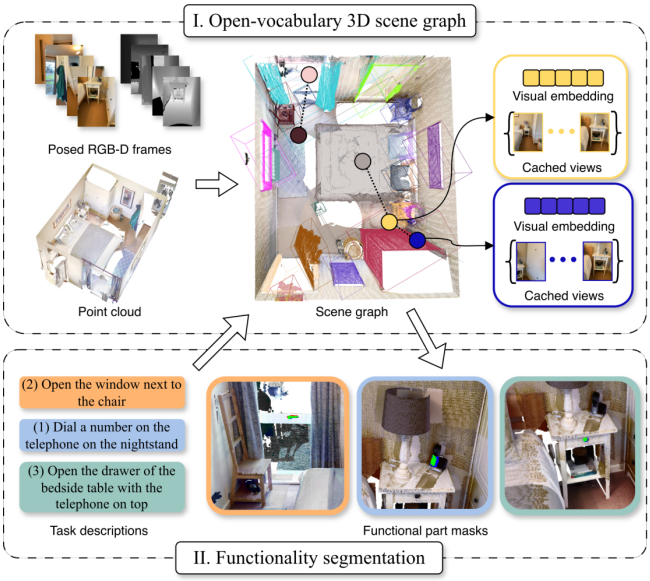

Fig. 1. T-FunS3D: a training-free method for open-vocabulary functionality segmentation in 3D scenes based on free-form text descriptions. Taking as input a point cloud of an indoor environment and the corresponding posed RGB-D images, an open-vocabulary scene graph is constructed at stage I, which contains instance embeddings and inter-object relation embeddings. At stage II, once a task is assigned, T-FunS3D extracts the functional components required to undertake the interaction from the scene.

While most existing methods focus on recognizing object-level instances, only a few particularly address finer-grained segmentation within scenes [4], [5], [6], [7], including functional object parts, such as a door handle. However, such fine-grained entities are crucial for downstream tasks like scene interaction. For example, assistive robots must recognize not only objects in an open-set world but also their fine-grained components, such as the chair arm and door of a washing machine, to execute general interactive tasks. Previous work on this problem often performs fine-grained segmentation throughout the scene without task-specific distinctions [4], [8]. Nonetheless, the methodological design they followed is resource-intensive, imposing high computational overhead and memory costs, which are not accessible in most mobile robotic systems. Therefore, there is a pressing need for more efficient designs that extract fine-grained open-vocabulary segmentation in a task-specific manner.

Hierarchical open-vocabulary scene understanding methods have recently attracted growing interest [9], [10], [4] and demonstrated promising results. The majority of them operates at higher levels of abstraction from region to object.

[1]Jingkun Feng and Reza Sabzevari are with the P4MARS Lab at the Faculty of Aerospace Engineering, Delft University of Technology.

They provide no identification of the fine-grained functional details of scene entities for interactive robot applications. Additionally, methods capable of part-level segmentation such as Search3D [4] often over-segment the entire scene to detect every entity at various level of granularity. However, in practice, only a small subset of objects in the scene is relevant for an assigned task, making scene-wide over-segmentation an unnecessary burden on computation and storage.

In light of these limitations, we advocate for an efficient segmentation approach focusing on task-relevant areas and performing functional part segmentation exclusively for selected objects. Our main contributions are as follows:

- We introduce a training-free method for 3D functionality segmentation based on free-form task descriptions, which leverages an actionable 3D scene graph containing open-vocabulary semantics.
- We present a task-driven hierarchical approach that first decomposes a scene into object instances, then segments fine-grained functional components from task-related entities, thereby facilitating efficient deployment in real-world robotic applications.
- We propose to construct an open-vocabulary scene graph that encodes nodes and inter-object edges with visual embedding features, enabling effective localization of entities with reference to the surrounding environment in an open-world setting.
- Our method reduces the runtime and memory consumption for open-vocabulary 3D segmentation of functional object components on the SceneFun3D dataset [1] while maintaining highly competitive accuracy.

II. Related Work

A. Open-Vocabulary 3D Scene Understanding

Recent works on open-vocabulary 3D scene understanding have made significant progress. They typically focus on object-level semantic segmentation following two main streams. One leverage 2D foundation models to decompose the observation and segment objects [11], [12], [13]. The other learns to embed semantics in explicit primitives such as points [14] and Gaussian ellipsoids [15] or in implicit fields [16], [17]. However, they separate objects from scenes but do not understand relationships between entities. Therefore, despite achieving promising instance segmentation, they remain bounded in the scope of open-world querying capabilities, struggle to handle complex reasoning.

To overcome the limitations of isolated instance representations, efforts have been given to understand and represent relationships between entities. For instance, ConceptGrpahs [18], OVSG [19], Open3DSG [20] constructs scene graphs with inter-object relationships. Following another line, Locate3D [21] learns a contextualized point cloud to empower referential grounding. However, they are largely restricted to the instance-level understanding at the finest level, and thus struggle to identify parts. Additionally, they either rely on extensive use of large models or construct exhaustive contextualize point-based representations, imposing high computational and memory costs.

These limitations in prior work highlight a clear absence of a comprehensive method that seamlessly integrates part-level open-vocabulary scene understanding with efficient and actionable representation. To bridge this gap, our method constructs a lightweight open-vocabulary 3D scene graph that encodes both nodes and inter-object edges using visual embeddings. We propose to interpret the scene in a hierarchical task-driven manner, which enables efficient extraction of fine-grained functional parts without expensive computational and memory overhead.

B. 3D Part Segmentation and Functionality Segmentation

3D part segmentation has made some progress in open-vocabulary-based frameworks thanks to recent efforts in [4], [22], [23]. Similar to open-set instance segmentation, a common strategy here is to lift features or predition from 2D foundation models (e.g., [24], [25], [26], [27]) into 3D. For instance, PartSLIP [22] and its successor [23] reprojects language-guided open-vocabulary part detection on rendered images into 3D. Similarly, SAMPart3D [28] distills 2D features into 3D representation. But these methods remain limited to single-object 3D models, restricting their applicability to full-scene understanding.

As an extension of part segmentation, functionality segmentation, recently introduced in SceneFun3D [1], focuses on segmenting functional interactive sub-parts in 3D scenes according to free-form language descriptions of tasks. Given its crucial role in downstream robotic applications like planning and manipulation, a growing body of research [4], [5], [6], [7] aims to address this challenge.

Conceptually, our work is closely related to Search3D [4] and Fun3DU [5]. Search3D segments point clouds and exhaustively embeds all objects and their parts. Fun3DU, in turn, detects objects of interest across all images and lifts 2D segmentation masks of their functional parts into 3D. However, it detects merely objects given in the query. Consequently, it must re-execute the object detection process upon receiving any new query, making it highly inefficient. Unlike these methods, our approach constructs an open-vocabulary scene graph for the entire 3D environment and retrieves functional parts exclusively from the selected objects based on the given task. This eliminates repeated instance detection and enables efficient handling of complex 3D grounding tasks.

C. 3D Scene Graph

3D scene graphs encode objects and spatial concepts (e.g., rooms and floors) as nodes and their relations as edges, offering compact, object-centric representations well suited for robotics [33]. This decomposition of environments facilitates higher-level reasoning and planning in object-centric tasks. Early works like [34] and [35] integrate SLAM with annotated data but remain limited to closed sets of objects. Harnessing multi-modal large language models, methods like HOV-SG [9] and CLIO [10] construct hierarchical open-vocabulary 3D scene graphs focusing on high levels of

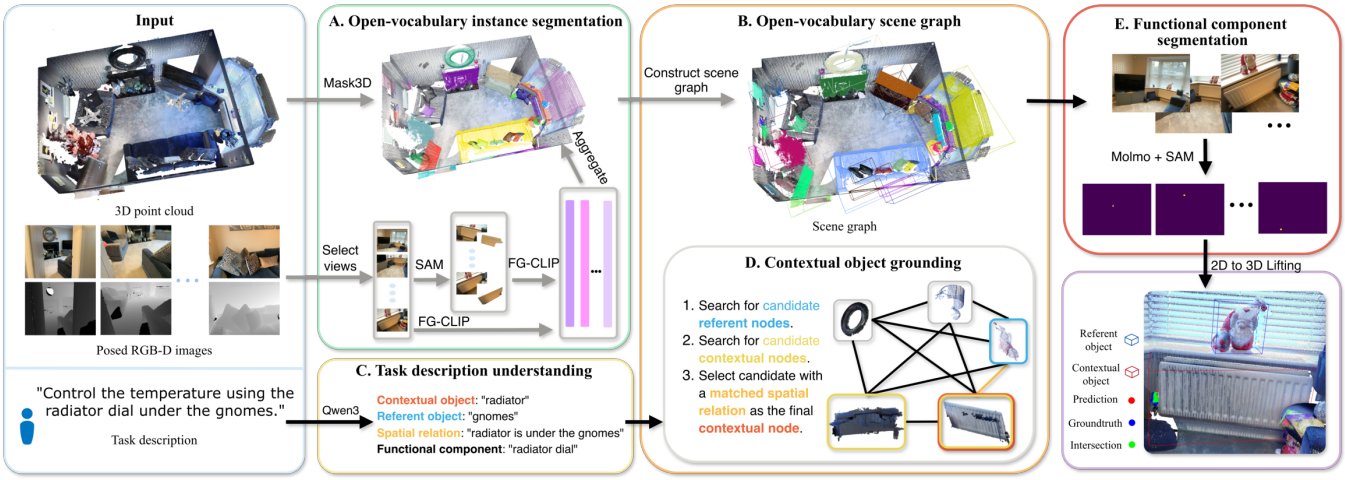

Fig. 2. T-FunS3D overview: The input of T-FunS3D are posed RBG-images and 3D point clouds of an indoor scene. (A) performs open-vocabulary instance segmentation by associating FG-CLIP [29] visual embeddings to class-agnostic instance segmentation from Mask3D [30]. We construct a scene graph of the featurized instances (B). Once a task is assigned, we decompose the description into ontologies using Qwen3 [31] (C) and identify the contextual object in the scene graph (D) by computing the text-visual embedding similarity. Lastly, (E) aggregates 2D masks extracted by combining Molmo [32] with SAM [24] to obtain 3D segmentation of functional parts.

semantic meanings from room over region to object. Nevertheless, their graph encodes only parent-child relations(e.g., floor-rooms, room-objects), which is sufficient for navigation but inadequate for interactive tasks. Such interactive applications in practice require a clear reasoning of relations between objects so the manipulator can infer the target object and its reachable grasps in human-centric spaces. ConceptGraphs [18], OVSG [19], and Open3DSG [20] advance this line by encoding open-vocabulary inter-object edges, enabling advance queries including referential expressions. However, they still struggle to identify part-level entities required for task execution.

Contemporary to our work, OpenFunGraph [6] and FunGraph [7] also address the functionality understanding problems using 3D scene graphs. They both establish nodes and edges for interactive object elements. Among them, FunGraph includes both inter-object and intra-object edges with respect to the spatial relationships, while OpenFunGraph only models the part-object relationships, specifying the functionality. Similar to ConceptGraphs [18], these two methods build edges based on multi-view aware natural language descriptions of nodes generated by VLMs and LLMs. In contrast, instead of using LLM-generated text descriptions, we leverage visual embeddings of multi-view observations of nodes to model the edges between objects, reducing computational and time consumption.

III. Problem Formulation

Given an input sequence of posed RGB-D frames $\{\mathcal{I}, \mathcal{D}\}$ and the point cloud P of an indoor environment, the goal is to get a 3D mask that locates the interactive functional component $\mathcal{F}$ in the scene given a manipulation task query $\mathcal{Q}$. We assume that the entities captured in the point cloud are static to the extend that the spatial relationships between instances preserve, while local movements are possible. For example, the coffee table can be moved locally but it is assumed to remain to the right of the sofa and to the left of the radiator.

In this work, we focus on queries that explicitly encompass the information of objects that the robots should interact with, referred as the contextual objects $\mathcal{C}$. For instance, in "*Adjust the room's temperature using the radiator thermostat*", "radiator" is the contextual object. Frequently, referential descriptions of object locations are given in the queries to reduce ambiguity, such as "*Open the nightstand drawer on the right side of the bed*". In such type of expressions, the objects used as a reference coordinate are called the referent objects $\mathcal{R}$ ("bed" in this example). The corresponding spatial relationships between $\mathcal{C}$ and $\mathcal{R}$ are denoted as 2-tuples $\mathcal{S} = (\mathcal{C}, \mathcal{R})$. Thus, a query can be generally assumed to be a collection of **spatial relations** $\mathcal{S}$, **referent object** $\mathcal{R}$, **contextual object** $\mathcal{C}$, and its **functional component** $\mathcal{F}$, i.e.,

$$\mathcal{Q} = \{\mathcal{C}, \mathcal{F}, (\mathcal{S}_1, \mathcal{R}_1, \mathcal{S}_2, \mathcal{R}_2, \dots)\}.$$

Notably, the referent objects and spatial relationships are optional and there can be multiple ones. The same color codes are adopted in block C and D in Fig. 2.

To tackle this task, we introduce an effective and efficient two-step approach. First, we locate the contextual object $\mathcal{C}$ in an open-vocabulary scene graph. After that, we segment the functional element with respect to the query using the images corresponding to the identified instance of interest.

IV. Open-Vocabulary Functionality Segmentation

Our proposed pipeline, T-FunS3D, is illustrated in Fig. 2. Given RGB-D observations, ground truth camera poses, and the 3D point cloud of an indoor environment, T-FunS3D provides task-driven hierarchical open-vocabulary functionality segmentation in 3D scenes. The pipeline consists of four modules organized into two stages. At the first stage, we perform 3D instance-level open-vocabulary segmentation by extracting semantic embeddings for class-agnostic 3D instance segmentation (Sec. IV-A). Then, an open-vocabulary

scene graph is constructed based on the featurized 3D instance segments (Sec. IV-B). Note that this stage only needs to be performed once per scene, provided the assumption that the environmental layout remains static. The second stage begins once a task in free-form text is assigned to the system. At this stage, we analyze the task description with a large language model (LLM) to extract the task ontology (Sec. IV-C). Afterwards, the relevant contextual instance is localized in the scene graph (Sec. IV-D) and we segment its functional components required for task execution (Sec. IV-E).

A. Open-vocabulary 3D instance segmentation

To obtain object-level instance proposals in an open-vocabulary setting, we adapt the approach introduced in OpenMask3D [11]. To achieve this, Mask3D [30] is applied to generate class-agnostic object-level mask proposals and decompose the scene. To bridge the semantic meaning to the proposals, the semantic features of objects are extracted. For this purpose, optimal views of the class-agnostic masks are identified by their visibility on various views, which is computed by projecting the associated 3D points on the input RGB. Once the views with the highest visibility ratio are obtained, semantic features are extracted using the visual encoder of multi-modal language models, inspired by [11], [12], and [9]. Unlike OpenMask3D, which computes visual embedding using only image crops of the objects, we leverage more pixel information to associate richer semantics with each proposal. For every extracted instance P_n, where $n \in \{1, \dots, N\}$, the top $k \in \{1, \dots, K\}$ views are cropped around the instance at multiple scales. The l-th ($l \in \{1, \dots, L\}$) scale crop of the k-th view for the n-th instance is denoted as $I^n_{k,l}$. Subsequently, we compute the open-vocabulary semantics for the instances from both the full-size RGB images and the crops of selected views using FG-CLIP [29], a multimodal vision-language model. However, in practice, the target object in views may be occluded or visible only through transparent materials (e.g., windows). In this case, their crops introduce a foreground-background bias that can reduce the accuracy of visual embeddings dramatically, e.g., the window frame is mistaken for plants outside the window. To mitigate this effect, we additionally generate masked crops, denoted as $M^n_{k,l}$, where irrelevant pixels are masked out. These masked crops are used as complementary cues for the visual encoder. This combination retains contextual information while emphasizing the target instance. The final visual embedding for P_n is obtained by averaging across views and scales:

$$f(P_n) = \frac{1}{2K} \sum_{k}^{K} \left[f(I^n_k) + \frac{1}{2L} \sum_{l}^{L} f(I^n_{k,l}) + f(M^n_{k,l}) \right]$$

where $f(\cdot)$ denotes the visual embedding computation.

B. Open-vocabulary scene graph

A scene graph is defined as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{v_i\}, i \in \{1, \dots, N\}$ denotes the set of vertices and $\mathcal{E} = \{e_{ij} | e_{ij} = (v_i, v_j),$ for $v_i, v_j \in \mathcal{V}$ where $i \neq j\}$ the set of edges. To enable open-vocabulary querying, we store visual embeddings rather than explicit class labels within the nodes, inspired by [9]. Edges between two nodes are established by averaging the visual embedding of the full-size images associated with the two vertices. In this way, we encode the semantic context surrounding the instances, alongside the inter-instance spatial relations captured across multiple views. The edges are computed during querying between all candidate contextual and referent object pairs rather than precomputed and stored in the graph as in previous works like [18] (see Sec. IV-D).

C. Task understanding using LLM

The task descriptions $\mathcal{Q}$ may involve varying levels of complexity. One the one hand, names of the functional entity can be expressed explicitly, like in the query "Control the temperature using the radiator dial", or requires to be inferred as for "Open the door". On the other hand, the descriptions may contain no referent objects, and therefore, no spatial relationships, like in the aforementioned two examples. Thus, basic ontology extraction techniques in natural language processing is insufficient for interpreting free-form task descriptions.

To this end, we leverage a LLM to parse them and extract clues, including spatial relationships $\mathcal{S}$, referent objects $\mathcal{R}$, the contextual object $\mathcal{C}$, and its functional entity $\mathcal{F}$. However, extracting each category individually can lead to ambiguity. Take the query "Open the door next to the window" and the categories, contextual object and functional entity as an example. On the one hand, if the LLM is prompted to extract only the contextual object, it may output a generic label such as "door", which is ambiguous if multiple doors exist in the environment. Thus, identifying the target object category is inadequate for the robot to undertake the task. On the other hand, if the model only extracts the functional object and outputs a "handle", it remains unclear whether the agent should manipulate the handle of a door or a window. Therefore, we prompt a LLM to jointly provide all categories in a single JSON-format output, following the strategy in [5]. Additionally, different from [5], we explicitly prompt the LLM to extract the spatial relationships of different instances from the description if they exist.

D. Contextual object grounding

The contextual object is retrieved by querying the scene graph based on the task description. We denote the visual and textual embedding functions by $f(\cdot)$ and $g(\cdot)$, respectively. Let $t_{\mathcal{R}}$ and $t_{\mathcal{C}}$ indicate the text descriptions for referent and contextual objects. After encoding $t_{\mathcal{C}}$ and $t_{\mathcal{R}}$ into textual embeddings using FG-CLIP [29], we extract candidate nodes for the respective objects based on their embedding similarities. For contextual object $\mathcal{C}$, we retrieve candidate nodes $\{C_i\}$ from the vertices $\mathcal{V}$ whose visual embeddings are most similar to the textual embedding of $\mathcal{C}$:

$$C_i = \arg\max_{v_i \in \mathcal{V}} sim(f(v_i), g(t_{\mathcal{C}}))$$

where $sim(\cdot, \cdot)$ computes the cosine similarity of two embedding vectors. Analogously, if there are referent objects $\mathcal{R}$

specified in $\mathcal{Q}$, we continue to select candidate nodes $\{R_j\}$ associated with $t_{\mathcal{R}}$:

$$R_j = \arg\max_{v_j \in \mathcal{V}} sim(f(v_j), g(t_{\mathcal{R}}))$$

Once we obtain the candidates, we examine the edges between all pairs $\{S_{ij}\} = \{(C_i, R_j)\}$ to identify those that have high embedding similarity scores with the spatial relationships $\mathcal{S}$ described in the query. Here, we denote the text description of spatial relationships by $t_{\mathcal{S}}$. For every extracted spatial relations involving the contextual object, the computation is formulated as:

$$S = \arg\max_{s \in S_{ij}} sim(f(s), g(t_{\mathcal{S}}))$$

The nodes in the corresponding candidate pairs with valid S are anchored as the contextual object and referent objects. If there are no spatial relationships in the descriptions, the best candidate contextual node is fed to the next module.

E. Functional component segmentation

We segment the functional components of the contextual object by combining the VLM Molmo [32] with SAM [24], inspired by Fun3DU [5]. Once the contextual object is identified, we retrieve its corresponding selected views and feed them into Molmo. Specifically, the task description ontology extracted in Sec. IV-C is leveraged as a prompt to guide Molmo in the functional element detection. As a result, Molmo outputs pixel coordinates corresponding to the functional component of interest. These pixel coordinates further serve as prompt for SAM to generate the 2D segmentation masks. By default, multiple masks are extracted for every prompt point in SAM. Each mask is delivered along with a confidence score, i.e., the estimated intersection over union. However, in practice, we notice that the masks with the highest scores tend to segment the entire object instead of the part indicated by the prompts, especially for crops with lower resolution. Therefore, instead of the masks with the highest confidence score, we resort to the smallest masks among the predictions. This design choice is further justified by an ablation study in Sec. V-D. The predicted 2D masks are subsequently back-projected into the 3D space, where multi-view aggregation of these lifted masks yields the final 3D functionality segmentation.

V. Experiments

A. Experiment setup

1) Implementation: T-FunS3D is a training-free method leveraging several pre-trained models. We build a scene-scale open-vocabulary instance segmentation (A in Sec. 2) based on OpenMask3D [11] with no retraining but several modifications on its pipeline. Their class-agnostic instance proposal model [30] is trained on ScanNet200 [36], whose point clouds are roof-less and have fewer points per scene compared to SceneFun3D [1]. Therefore, we preprocess the point clouds to fit them in the range and form on which the proposal model is trained (see supplementary materials). In addition, we only preserve class-agnostic masks of the highest confidence scores, while OpenMask3D maintains all masks generated by the model. This is based on the observations that more than half of the proposed masks identify no meaningful objects or duplicate each other, resulting in unnecessary consumption in the visual embedding computation. Another deviation from OpenMask3D is that we compute visual features of the proposed instance from richer sources, including the full-size RGB, crops around the instance, and masked crops of the top-k views associated to this instance, described in Sec. IV-A. This combination provides not only more reliable visual information about instances themselves, preventing foreground-background-bias, but also more context information of the surroundings, facilitating referring localization of objects. This design choice is justified in the Sec. V-D. In our standard method, we sample one from every five frames of the complete video as input and set k to 5.

Multi-modal language models, such as CLIP [25], SigLIP [37], and SigLIP2 [38], that offer both visual and textual encoders, are the core component to bridge semantic meaning to 3D scenes. For visual and textual embedding computation, FG-CLIP [29] is applied because of its reliable performance in preserving the fine-grained details.

When a query is given, QWen3-14B [31] is employed to analyze the task descriptions. Specifically, we disable the thinking capability of the model and query in multi-turn conversations to allow faster inference without sacrificing the output accuracy. Eventually, the functional part segmentation is achieved by combining Molmo-7B-D [32] and SAM [24].

2) Dataset: We evaluated our method on the SceneFun3D [1] dataset. It is the only available dataset that supports task-driven functionality segmentation in 3D by providing annotations for functional components together with a diverse set of tasks for indoor scenes. In this paper, we reported the evaluation results on the validation split of this dataset, which contains 30 scenes with 445 task descriptions.

Among the task descriptions in this split, about three quarters possess referring expressions, as shown in Fig. 3. Note that the total number of counted spatial relationships (121 of "None" and 396 of the others) does not match the number of descriptions, as some descriptions attribute multiple relationships. More experiment results can be found on the supplementary materials.

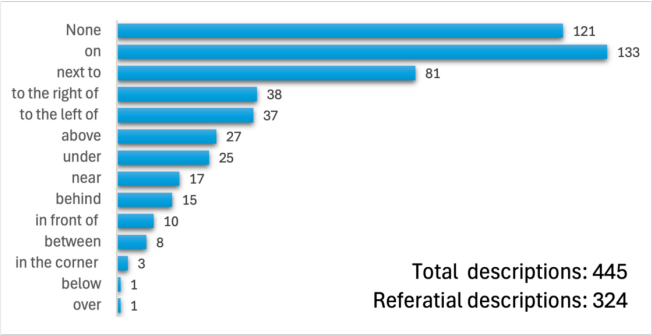

Fig. 3. Distribution of spatial relationships in the task descriptions provided in the validation split of the SceneFun3D dataset [1].

3) Evaluation metrics: As suggested in SceneFun3D [1], we measure different metrics, including the Average Precision (AP) 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, like [5], we also report the mean IoU and the Average Recall (AR), including AR_{25}, AR_{50}, and mAR of the same conditions as AP.

4) Baselines: For comparison, we use Fun3DU [5] as the main baseline method on functionality segmentation tasks. In addition, due to the lack of open-sourced methods dedicated to fine-grained functional component segmentation at the time of writing, T-FunS3D is also compared against state-of-the-art open-vocabulary 3D instance segmentation, including OpenMask3D [11], OpenIns3D [13], and LeRF [16]. To ensure consistency, we report the results of the baseline methods presented in [5]. Additionally, for a fair comparison, we reproduced the results of a primary baseline, Fun3DU [5], using their released code and the same task parsing inputs as T-FunS3D, denoted as Fun3DU† in the tables below.

B. Results

1) Functionality segmentation: TABLE I presents the evaluation on the performance of functional part segmentation. As shown, OpenMask3D scores nearly zero precision and low recall due to the increased complexity of the tested scenes compared to its training data. The other two instance segmentation methods score high recall but zero precision, indicating that they tend to segment the entire object rather than the fine-grained object parts. Fun3DU [5] achieves good figures for all metrics, demonstrating reliable performance. However, it can not be neglected that Fun3DU performs contextual object detection on all 2D views using an additional VLM, which requires repetition for new queries. Despite that, T-FunS3D still surpasses Fun3DU and also the other baselines with the highest precision and IoU, as well as generally high recall. In comparison, T-FunS3D demonstrates strong functionality segmentation performance with at least $+1.2$ AP_{25} and $+0.5$ mIoU improvement over baseline methods. The improvement increases to $+11.1$ AP_{25} and $+3.7$ mIoU compared to the reproduced results of Fun3DU.

TABLE I
PERFORMANCE COMPARISON ON SCENEFUN3D [1] DATASET.

Methods	mAP	AP50	AP25	mAR	AR50	AR25	mIoU
OpenMask3D [11]	0.2	0.2	0.4	20.3	24.5	27	0.2
OpenIns3D [13]	0	0	0	**40.5**	**46.7**	**51.5**	0.1
LERF [16]	0	0	0	34.2	35.1	36	0
Fun3DU [5]	7.6	16.9	33.3	27.4	38.2	46.7	15.2
Fun3DU [5]†	4.4	10.3	23.4	30.9	42.3	49.7	12.0
T-FunS3D (ours)	**8.1**	**17.8**	**34.5**	23.8	35.8	46.9	**15.7**

2) Referring grounding: TABLE II compares T-FunS3D and the baseline method, Fun3DU [5], on a subset of the validation split in SceneFun3D [1] which consists of descriptions that identify the contextual object's location by referring to its surrounding items (e.g., "Turn on the lamp on the side table next to the Christmas tree"), as interpreted in Fig. 3. Results on this subset reflect the capability of localizing objects in 3D scenes from such complex referring expressions. As shown in the TABLE II, T-FunS3D achieves much higher precision and IoU scores in localizing the functional interactive elements than Fun3DU, showing at least $+12.45$ AP_{25} and $+4.51$ mIoU improvement. This better performance is due to two core design choices in T-FunS3D. First, T-FunS3D decomposes the input query and extracts the spatial relations using the LLM, which serves in the subsequent stage as prompts to query on the concrete instances. The extraction of relations makes the prompt more precise and context-aware, increasing the likelihood of accurately locating the target object. Second, the scene graph in T-FunS3D encodes the inter-object spatial relations, which ease the object localization based on referential descriptions. T-FunS3D scores slightly lower recall compared to Fun3DU due to the use of the smallest 2D masks from SAM rather than the ones with the highest confidence, as in Fun3DU. A detailed discussion on the design choice is in Sec. V-D

TABLE II
PERFORMANCE ON TASK DESCRIPTIONS WITH SPATIAL REFERRING EXPRESSIONS IN THE VALIDATION SPLIT OF SCENEFUN3D [1].

Methods	mAP	AP50	AP25	mAR	AR50	AR25	mIoU
Fun3DU [5]†	3.83	9.26	22.53	**30.77**	**41.36**	**48.77**	11.73
T-FunS3D (ours)	**8.11**	**19.20**	**34.98**	25.17	37.46	47.06	**16.24**

3) Qualitative results: Fig. 4 presents qualitative examples of our functionality segmentation, intuitively illustrating both the capabilities and the current limitations of T-FunS3D. We aggregate 2D segmentation masks obtained from selected views based on contextual object's visibility onto the 3D point cloud. However, images captured from large oblique viewing angles can lead to less accurate segmentation, as indicated by the low precision and recall metrics in the second and third columns in Fig. 4. Additionally, our method exhibits limitations when functional elements lack physical attachment to the contextual objects. For instance, it fails to segment the ceiling light switch (yielding zero IoU) in the rightmost column.

C. Runtime comparison

TABLE III compares the runtime performance of T-FunS3D against two baselines for instance ① and functionality segmentation ②. The reported runtimes are experimentally measured on an NVIDIA A40. Considering the full pipeline, T-FunS3D achieves slightly higher accuracy than the state-of-the-art Fun3DU [5] in a much shorter time. Unlike Fun3DU, which segments instances of interest across all images, T-FunS3D and OpenMask3D generate class-agnostic instance proposals from point clouds (ⓐ) and compute embeddings only from views where each instance is most visible (ⓑ). T-FunS3D is faster than OpenMask3D at ① because T-FunS3D retains only high-confidence proposals. A comparison with OpenMask3D at ② is not applicable, as OpenMask3D does not perform functionality segmentation.

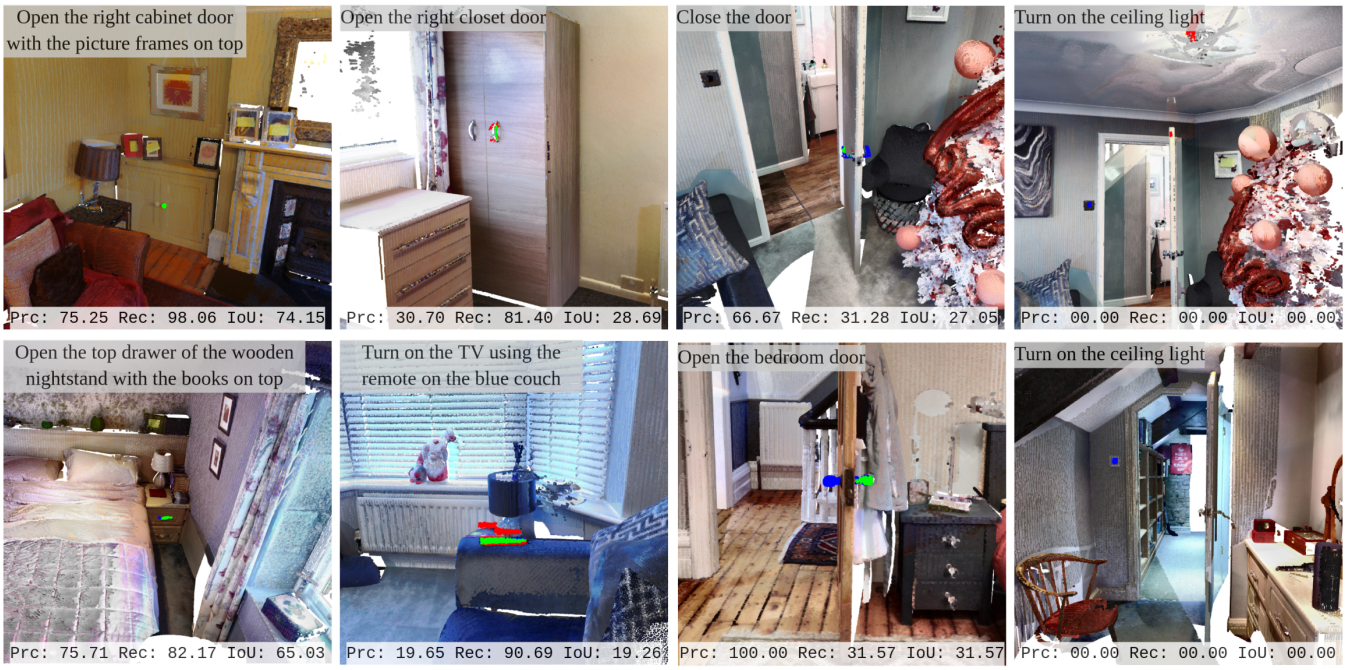

Fig. 4. Qualitative examples of T-FunS3D. We visualize point clouds around the functional components: red points indicate predictions, blue points denote ground truth, and green points represent overlaps.

T-FunS3D further reduces runtime at stage ② by reusing the cached views, which are less than the required images in Fun3DU. As a reference, Fun3DU processes one task at ② averagely in 118.4 seconds on an NVIDIA A100 as reported in [5]. Moreover, for every new query, T-FunS3D requires shorter process time by caching object visual embeddings with top-ranked views obtained at ①, whereas Fun3DU must repeat the complete segmentation pipeline.

TABLE III
RUNTIME COMPARISON AT INSTANCE AND FUNCTIONALITY SEGMENTATION

Methods	① OV Inst. Segm. (per-scene average)	② Func. Segm. (per-query average)
OpenMask3D [11]	ⓐ 30s + ⓑ 720s	-
Fun3DU [5]	1920s	167s
T-FunS3D (ours)	ⓐ **12s** + ⓑ **580s**	**78s**

D. Ablation study

We assess the impact of two methodological design choices introduced in Sec. IV, i.e., visual embedding computation with richer information and aggregation of the smallest SAM masks into 3D, on the final performance. Experiments in this ablation study are conducted on the first 10 visits in the validation split of the SceneFun3D [1] dataset. The results in TABLE IV emphasize the importance of these designs.

As the primary comparison baseline, we report the reproduced results of Fun3DU [5] under the same task understanding condition as ours in row 1 (†). In rows 2 and 3, we replace the collection of images, including RGB images in full size, crops, and masked crops, with varying combinations. Row 2 (w/o FI) ignores full-size images in visual embedding computation, while row 3 (w/o MC) combines full-size images and crops, but discards the masked crops. In row 4 (w/o SM), we explore the use of 2D masks produced by SAM [24] with the highest score for every prompt point instead of the smallest ones. All configurations exhibit general performance declines compared to our standard method, except the method in row 1 achieves higher recalls. The abundant pixel information fed into the embedding computation contributes around $+0.8$ mAP and $+1.5$ mIoU, while leveraging the smallest masks generated by SAM leads to about $+2.0$ mAP and nearly $+3.0$ mIoU improvement. The drop of recall from row 4 to row 5, implying more false negatives in the prediction, indicates, in turn, a higher concentration on the functional entities by lifting the smallest 2D masks to 3D.

TABLE IV
ABLATION STUDY ON T-FUNS3D DESIGN CHOICES FOR FUNCTIONALITY SEGMENTATION ON A SUBSET OF THE VALIDATION SPLIT IN SCENEFUN3D [1]

Methods	mAP	AP50	AP25	mAR	AR50	AR25	mIoU
Fun3DU [5]†	4.41	10.49	22.38	30.28	41.26	49.65	11.07
Ours w/o FI	5.73	13.99	26.57	22.80	33.57	41.26	13.40
Ours w/o MC	5.53	17.02	29.08	21.84	28.37	36.17	13.37
Ours w/o SM	4.46	10.36	23.42	30.95	42.34	49.77	12.03
Ours standard	6.41	15.49	31.69	23.03	32.39	40.14	14.98

VI. CONCLUSION

We presented T-FunS3D, a novel training-free method for task-driven functionality segmentation in 3D scenes leveraging open-vocabulary scene graphs and VLMs. As its core, the proposed open-vocabulary scene graph represents nodes and edges with semantic embeddings, enabling T-FunS3D to efficiently localize task-relevant objects and achieve accurate functionality segmentation in a hierarchical manner. In

future work, fault-handling mechanisms could be integrated between modules to enhance the overall robustness of the pipeline. We hope this work inspires continued advancement in open-vocabulary functionality segmentation, facilitating the flexible and efficient grounding of task-relevant scene entities for interactive robotic applications in real-time.

REFERENCES

[1] A. Delitzas, A. Takmaz, F. Tombari, R. Sumner, M. Pollefeys, and F. Engelmann, "Scenefun3d: Fine-grained functionality and affordance understanding in 3d scenes," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[2] J. J. Gibson, *The ecological approach to visual perception: classic edition*. Psychology press, 2014.

[3] J. Pustejovsky, "The generative lexicon," *Computational linguistics*, 1991.

[4] A. Takmaz, A. Delitzas, R. W. Sumner, F. Engelmann, J. Wald, and F. Tombari, "Search3D: Hierarchical Open-Vocabulary 3D Segmentation," *IEEE Robotics and Automation Letters*, 2024.

[5] J. Corsetti, F. Giuliari, A. Fasoli, D. Boscaini, and F. Poiesi, "Functionality understanding and segmentation in 3d scenes," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[6] C. Zhang, A. Delitzas, F. Wang, R. Zhang, X. Ji, M. Pollefeys, and F. Engelmann, "Open-vocabulary functional 3d scene graphs for real-world indoor spaces," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[7] D. Rotondi, F. Scaparro, H. Blum, and K. O. Arras, "Fungraph: Functionality aware 3d scene graphs for language-prompted scene interaction," in *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2025.

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

[9] 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.

[10] D. Maggio, Y. Chang, N. Hughes, M. Trang, D. Griffith, C. Dougherty, E. Cristofalo, L. Schmid, and L. Carlone, "Clio: Real-time task-driven open-set 3d scene graphs," *IEEE Robotics and Automation Letters*, 2024.

[11] A. Takmaz, E. Fedele, R. W. Sumner, M. Pollefeys, F. Tombari, and F. Engelmann, "Openmask3d: open-vocabulary 3d instance segmentation," in *37th International Conference on Neural Information Processing Systems*, 2023.

[12] P. D. A. Nguyen, T. D. Ngo, C. Gan, E. Kalogerakis, A. Tran, C. Pham, and K. Nguyen, "Open3DIS: Open-Vocabulary 3D Instance Segmentation with 2D Mask Guidance," *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[13] Z. Huang, X. Wu, X. Chen, H. Zhao, L. Zhu, and J. Lasenby, "Openins3d: Snap and lookup for 3d open-vocabulary instance segmentation," in *European Conference on Computer Vision*, 2024.

[14] K. Jatavallabhula, A. Kuwajerwala, Q. Gu, M. Omama, T. Chen, S. Li, G. Iyer, S. Saryazdi, N. Keetha, A. Tewari, J. Tenenbaum, C. de Melo, M. Krishna, L. Paull, F. Shkurti, and A. Torralba, "Conceptfusion: Open-set multimodal 3d mapping," *Robotics: Science and Systems (RSS)*, 2023.

[15] Y. Li, Q. Ma, R. Yang, H. Li, M. Ma, B. Ren, N. Popovic, N. Sebe, E. Konukoglu, T. Gevers *et al.*, "Scenesplat: Gaussian splatting-based scene understanding with vision-language pretraining," in *IEEE/CVF International Conference on Computer Vision*, 2025.

[16] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik, "Lerf: Language embedded radiance fields," in *IEEE/CVF international conference on computer vision*, 2023.

[17] F. Engelmann, F. Manhardt, M. Niemeyer, K. Tateno, and F. Tombari, "Opennerf: Open set 3d neural scene segmentation with pixel-wise features and rendered novel views," in *The Twelfth International Conference on Learning Representations*, 2024.

[18] Q. Gu, A. Kuwajerwala, S. Morin, K. M. Jatavallabhula, B. Sen, A. Agarwal, C. Rivera, W. Paul, K. Ellis, R. Chellappa *et al.*, "Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[19] H. Chang, K. Boyalakuntla, S. Lu, S. Cai, E. P. Jing, S. Keskar, S. Geng, A. Abbas, L. Zhou, K. Bekris *et al.*, "Context-aware entity grounding with open-vocabulary 3d scene graphs," in *7th Annual Conference on Robot Learning*, 2023.

[20] S. Koch, N. Vaskevicius, M. Colosi, P. Hermosilla, and T. Ropinski, "Open3dsg: Open-vocabulary 3d scene graphs from point clouds with queryable objects and open-set relationships," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[21] P. McVay, S. Arnaud, A. Martin, A. Majumdar, K. M. Jatavallabhula, P. Thomas, R. Partsey, D. Dugas, A. Gejji, A. Sax *et al.*, "Locate 3d: Real-world object localization via self-supervised learning in 3d," in *Forty-second International Conference on Machine Learning*, 2025.

[22] M. Liu, Y. Zhu, H. Cai, S. Han, Z. Ling, F. Porikli, and H. Su, "Partslip: Low-shot part segmentation for 3d point clouds via pretrained image-language models," in *IEEE/CVF conference on computer vision and pattern recognition*, 2023.

[23] Y. Zhou, J. Gu, X. Li, M. Liu, and H. Su, "Partslip++: Enhancing low-shot 3d part segmentation via multi-view instance segmentation and maximum likelihood estimation," in *ICCV 2025 Workshop on Wild 3D: 3D Modeling, Reconstruction, and Generation in the Wild*, 2025.

[24] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo *et al.*, "Segment anything," in *IEEE/CVF international conference on computer vision*, 2023.

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

[26] C. J. Balibar and C. T. Walsh, "Glip, a multimodular nonribosomal peptide synthetase in aspergillus fumigatus, makes the diketopiperazine scaffold of gliotoxin," *Biochemistry*, 2006.

[27] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin, "Emerging properties in self-supervised vision transformers," in *IEEE/CVF international conference on computer vision*, 2021.

[28] Y. Yang, Y. Huang, Y.-C. Guo, L. Lu, X. Wu, E. Y. Lam, Y.-P. Cao, and X. Liu, "Sampart3d: Segment any part in 3d objects," *arXiv preprint arXiv:2411.07184*, 2024.

[29] C. Xie, B. Wang, F. Kong, J. Li, D. Liang, G. Zhang, D. Leng, and Y. Yin, "Fg-clip: Fine-grained visual and textual alignment," in *International Conference on Machine Learning*, 2025.

[30] J. Schult, F. Engelmann, A. Hermans, O. Litany, S. Tang, and B. Leibe, "Mask3D: Mask Transformer for 3D Semantic Instance Segmentation," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[31] A. Yang, A. Li, B. Yang, B. Zhang, B. Hui, B. Zheng, B. Yu, C. Gao, C. Huang, C. Lv *et al.*, "Qwen3 technical report," *arXiv preprint arXiv:2505.09388*, 2025.

[32] M. Deitke, C. Clark, S. Lee, R. Tripathi, Y. Yang, J. S. Park, M. Salehi, N. Muennighoff, K. Lo, L. Soldaini *et al.*, "Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025.

[33] I. Armeni, Z.-Y. He, J. Gwak, A. R. Zamir, M. Fischer, J. Malik, and S. Savarese, "3d scene graph: A structure for unified semantics, 3d space, and camera," in *IEEE/CVF international conference on computer vision*, 2019.

[34] A. Rosinol, M. Abate, Y. Chang, and L. Carlone, "Kimera: an open-source library for real-time metric-semantic localization and mapping," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

[35] A. Rosinol, A. Gupta, M. Abate, J. Shi, and L. Carlone, "3d dynamic scene graphs: Actionable spatial perception with places, objects, and humans," *Robotics: Science and Systems XVI*, 2020.

[36] D. Rozenberszki, O. Litany, and A. Dai, "Language-grounded indoor 3d semantic segmentation in the wild," in *European conference on computer vision*, 2022.

[37] X. Zhai, B. Mustafa, A. Kolesnikov, and L. Beyer, "Sigmoid loss for language image pre-training," in *IEEE/CVF international conference on computer vision*, 2023.

[38] M. Tschannen, A. Gritsenko, X. Wang, M. F. Naeem, I. Alabdulmohsin, N. Parthasarathy, T. Evans, L. Beyer, Y. Xia, B. Mustafa *et al.*, "Siglip 2: Multilingual vision-language encoders with improved semantic understanding, localization, and dense features," *arXiv preprint arXiv:2502.14786*, 2025.