OpenGraph: Open-Vocabulary Hierarchical 3D Graph Representation in Large-Scale Outdoor Environments

Yinan Deng, Jiahui Wang, Jingyu Zhao, Xinyu Tian, Guangyan Chen, Yi Yang, Yufeng Yue*

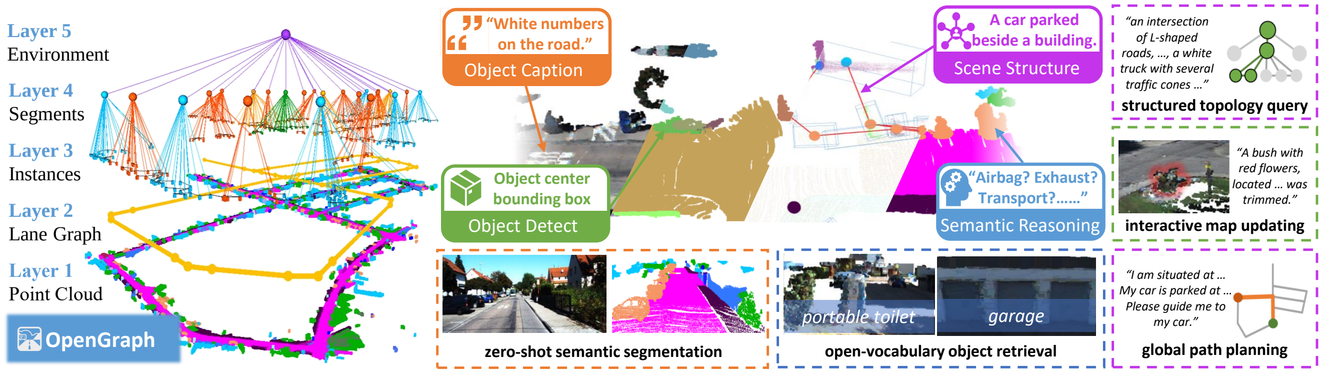

Layer 5
Environment
Layer 4
Segments
Layer 3
Instances
Layer 2
Lane Graph
Layer 1
Point Cloud
OpenGraph
"White numbers on the road."
Object Caption
Object center bounding box
Object Detect
A car parked beside a building.
Scene Structure
"Airbag? Exhaust? Transport?......"
Semantic Reasoning
"an intersection of L-shaped roads, ..., a white truck with several traffic cones ..."
structured topology query
"A bush with red flowers, located ... was trimmed."
interactive map updating
zero-shot semantic segmentation
portable toilet
garage
open-vocabulary object retrieval
"I am situated at ... My car is parked at ... Please guide me to my car."
global path planning

Fig. 1: We introduce **OpenGraph**, a framework of open-vocabulary hierarchical 3D graph representation in large-scale outdoor environments. OpenGraph facilitates various downstream tasks, including zero-shot semantic segmentation, open-vocabulary object retrieval, structured topology query, global path planning, interactive map updating, and so on.

***Abstract*— Environment representations endowed with sophisticated semantics are pivotal for facilitating seamless interaction between robots and humans, enabling them to effectively carry out various tasks. Open-vocabulary maps, powered by Visual-Language models (VLMs), possess inherent advantages, including zero-shot learning and support for open-set classes. However, existing open-vocabulary maps are primarily designed for small-scale environments, such as desktops or rooms, and are typically geared towards limited-area tasks involving robotic indoor navigation or in-place manipulation. They face challenges in direct generalization to outdoor environments characterized by numerous objects and complex tasks, owing to limitations in both understanding level and map structure. In this work, we propose OpenGraph, the first open-vocabulary hierarchical graph representation designed for large-scale outdoor environments. OpenGraph initially extracts instances and their captions from visual images, enhancing textual reasoning by encoding them. Subsequently, it achieves 3D incremental object-centric mapping with feature embedding by projecting images onto LiDAR point clouds. Finally, the environment is segmented based on lane graph connectivity to construct a hierarchical graph. Validation results from public dataset SemanticKITTI demonstrate that OpenGraph achieves the highest segmentation and query accuracy. The source code of OpenGraph is publicly available at `https://github.com/BIT-DYN/OpenGraph`.**

I. Introduction

A comprehensive understanding and representation of the 3D scene are crucial for robots to perform various downstream tasks [1]. Occupancy mapping [2] stands out as the most prevalent technique for building maps, allowing for the retrieval of geometric scene properties. By discerning obstacle positions and shapes, these maps facilitate spatial navigation for autonomous obstacle avoidance. With the advancement of deep learning technology, semantic information is incorporated into classic geometric maps [3]. This integration enables robots to achieve semantic-level intelligent navigation. However, such semantics are confined to predefined labels during the training phase, presenting challenges in heuristic comprehension and effective utilization.

In recent years, the widespread adoption of visual language models (VLMs) [4]–[6] has opened avenues for encapsulating conceptual semantics in maps. VLMs encode images and text into a unified feature space through adversarial learning, enabling seamless interaction between robots and humans. These foundation models, trained on extensive web-based datasets, can uncover novel objects and derive simplistic understandings at inference time. However, the main scope of current open-vocabulary mapping methods is at room-level or desktop-level, primarily used for indoor navigation or in-place manipulation tasks for robots. There are two notable limitations, that restrict their applicability in large-scale outdoor environments:

1) **Weak object-centric comprehension and reasoning capabilities.** Most existing methods directly distil [7] or project [8] 2D VLM features into 3D space as semantic understanding of the constructed open-vocabulary maps. Such primitive VLM features excel at broad recognition but lack robust reasoning capabilities. For instance, when encountering *a patch of grass*, VLM features capture its category (*grass*), color (*green*), and other basic attributes but not encompass common-sense knowledge such as *its function as a soccer field or a primary food for sheep*.

This work is supported by the National Natural Science Foundation of China under Grant 62003039, 61973034, U193203, 62173042. (Corresponding Author: Yufeng Yue, yueyufeng@bit.edu.cn)

All authors are with School of Automation, Beijing Institute of Technology, Beijing, 100081, China.

This limited understanding restricts their applicability when handling various tasks encountered in outdoor environments that demand a certain level of comprehension.

2) **Limited and inefficient map architectures.** Efficiently storing, maintaining, and rapidly retrieving desired objects in non-structured outdoor environments characterized by numerous objects pose a key challenge. The original point-wise open-vocabulary mapping methods [8], [9] are computationally expensive and typically represent retrieval results using feature similarity heatmaps, lacking object boundaries. Although some subsequent methods [10], [11] have achieved indoor instance-level map construction with the assistance of segmentation models like SAM [12], they remain challenging to effectively discriminate the desired one when multiple corresponding objects are present in the scene. This difficulty is compounded by the prevalence of repetitive objects in outdoor settings, making it impractical to directly apply existing open-vocabulary map architectures to such environments.

To address the above limitations, this paper proposes **OpenGraph**, a novel framework for open-vocabulary hierarchical 3D graph representation. **1) Open reasoning:** Unlike approaches that directly employ VLM features for environmental semantic understanding, OpenGraph leverages VLMs as the cognitive front-end. It segments instances from visual images and generates textual captions. Moreover, large language models (LLMs) [13], [14], renowned for their exceptional performance in natural language processing tasks, encode these captions to enrich the open-minded reasoning capabilities. **2) Hierarchical graph:** OpenGraph projects caption features from 2D images onto 3D LiDAR point clouds and incorporates them into the construction of object-centric maps. To facilitate map maintenance and specific object retrieval, OpenGraph introduces a hierarchical graph representation. This representation segments the environment based on the connectivity of the computed lane map and associates it with jurisdictional instances. An illustrative example of OpenGraph's result is presented in Fig. 1. In summary, our contributions are as follows:

- We introduce the first outdoor open-vocabulary object-centric mapping system capable of discovering, building, and comprehending a vast number of instances. We innovatively design the caption feature as the cornerstone for object comprehension, thereby enhancing the cognitive level of the maps.
- We propose a hierarchical 3D graphical representation that supports efficient maintenance and rapid retrieval in large-scale environments.
- Validation on outdoor dataset demonstrates that OpenGraph enables a profound semantic understanding of the environment and facilitates downstream applications.

II. Related Works

A. Closed-vocabulary semantic mapping

While early studies [15]–[17] introduced pretrained deep learning-based segmentation models into basic spatial representations (occupancies, point clouds, etc.) for semantic 3D mapping, their performance was largely constrained by the model capabilities. Recent researches [18] exploit the latest advances in implicit neural representations to achieve geometric and semantic mapping within a unified feature space. Despite these advances, these methods require time-consuming self-supervision and are scenario-tailored, making generalization to other scenarios challenging. Most importantly, all of them either use segmentation models pretrained on a closed set of classes or can only utilize limited classes in the current scene for learning from scratch. This limitation makes them challenging to comprehend unseen object classes in complicated and open scenes.

B. Open-vocabulary 3D mapping

With the impressive progress of web-based pre-trained visual language models and large language models, an increasing number of methods are attempting to extend their 2D open vocabulary understanding capabilities to the 3D world, including the following three mainstream solutions.

1) **Vision-only point-wise mapping**. Originally, OpenScene [19] achieves 3D open-vocabulary scene understanding by projecting point-wise vision features extracted from fine-tuned image segmentation model [20] onto a 3D point cloud, facilitating easy utilization for open-vocabulary queries. However, the fine-tuned models lose their original ability to capture long-tail objects. Even if subsequent works [8], [9] adopt well-designed point-wise feature extraction methods to avoid object forgetting, the inconsistency between point features blurs the boundaries of instances, making object retrieval challenging during downstream robot tasks.

2) **Vision-only instance-level mapping**. The mainstream method for instance-level open-vocabulary 3D mapping involves extracting VLM features at the 2D image mask scale and then fusing point clouds from potentially the same instances in 3D space, considering both spatial and feature similarities. Compared to OVIR-3D [10], which fuses text-aligned 2D region proposals into 3D space using Periodic 3D Instance Filtering and Merging, OpenMask3d [11] leverages predicted class-agnostic 3D instance masks to guide the multi-view fusion of CLIP-based image embeddings. Recently, Open3DIS [21] devises a 2D-guide-3D Instance Proposal Module to further enhance the description of 3D object shapes. However, the above methods still rely on fused CLIP features, which encode limited visual context under multi-view masks, thus lacking high-level natural language reasoning capabilities.

3) **Vision-language mapping**. Although not instance-level, the weakly supervised CLIP-Fields [7] first combines visual features from CLIP and textual features from Sentence-BERT [14] in the architecture of neural implicit representations as robotic semantic memory. Beyond [10], ConceptGraphs [22] further extracts structural captions for each object, feeding them to LLMs for other applications, such as LLM-powered scene graph creation and natural language reasoning. Nonetheless, their reasoning capabilities are still constrained by the utilization of underlying CLIP

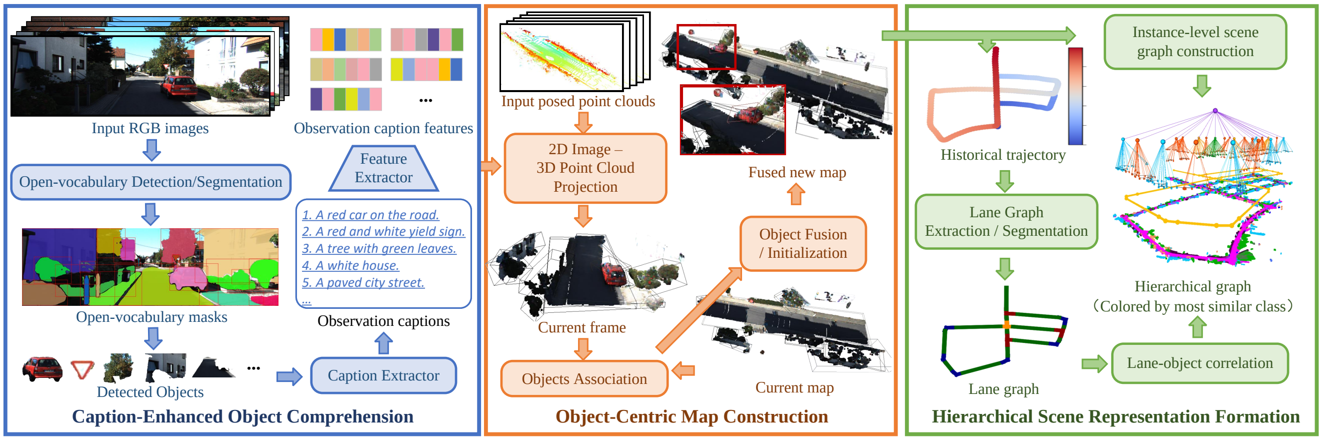

Input RGB images
Observation caption features
...
Open-vocabulary Detection/Segmentation
Feature Extractor
1. A red car on the road.
2. A red and white yield sign.
3. A tree with green leaves.
4. A white house.
5. A paved city street.
...
Open-vocabulary masks
Observation captions
Detected Objects
Caption Extractor
Caption-Enhanced Object Comprehension
Input posed point clouds
2D Image – 3D Point Cloud Projection
Fused new map
Object Fusion / Initialization
Current frame
Objects Association
Current map
Object-Centric Map Construction
Instance-level scene graph construction
Historical trajectory
Lane Graph Extraction / Segmentation
Hierarchical graph (Colored by most similar class)
Lane graph
Lane-object correlation
Hierarchical Scene Representation Formation

Fig. 2: The framework of OpenGraph consists of three main modules: Caption-Enhanced Object Comprehension, Object-Centric Map Construction, and Hierarchical Graph Representation Formation.

features, and their effectiveness in large-scale outdoor scenes is impeded by the reliance on depth cameras.

For OpenGraph, we employ the advanced vision language model to extract visual-textural captions directly. These captions are then encoded using LLM, yielding textual features enriched with natural language reasoning abilities.

C. 3D Scene graph

To overcome ambiguity in object retrieval by providing context-aware specifications, 3D scene graphs (3DSGs) are proposed to describe the 3D scene compactly as graph structures, where nodes represent spatial or semantic properties of objects and edges encode inter-object relationships [23]. While original approaches [24] generate real-time closed-set 3D semantic scene graph predictions from image sequences using graph neural networks, recent researchers have explored integrating open-vocabulary foundation models for 3D scene graph generation. ConceptGraphs [22] and OVSG [25] pioneered an open-vocabulary framework for generating indoor robot scene graphs. However, their performance in outdoor environments is compromised by focusing solely on establishing object-level scene graphs. Inspired by closed-vocabulary 3D hierarchical graph generation methods hydra [23] and CURB-SG [26], our OpenGraph proposes the first open-vocabulary hierarchical 3D graph representation in large-scale outdoor environments.

III. OpenGraph

A. Framework Overview

OpenGraph takes a sequence of 2D RGB images $\mathcal{I} = \{I^{(1)}, I^{(2)}, ..., I^{(t)}\}$ and a sequence of 3D LiDAR point clouds $\mathcal{C} = \{C^{(1)}, C^{(2)}, ..., C^{(t)}\}$ with pose $\mathcal{P} = \{P^{(1)}, P^{(2)}, ..., P^{(t)}\}$ as input, and produces a global hierarchical graph $\mathcal{M}_{all}^{(t)}$ of the observed environment as output.

The overall OpenGraph framework, as depicted in Fig. 2, consists of three primary modules. Firstly, the Caption-Enhanced Object Comprehension module focuses on instance segmentation and caption feature extraction from 2D images. Secondly, the Object-Centric Map Construction module is responsible for projecting 2D images and their interpretations into 3D LiDAR point clouds, enabling incremental construction of object-centric maps. Lastly, the Hierarchical Graph Representation Formation module deals with lane graph extraction and segmentation to construct the final hierarchical graph. By sequentially executing the three modules, OpenGraph acquires a profound understanding of the environment. The resulting hierarchical graph consists of the following layers, which can be expanded or collapsed as required by the actual scenario:

1) ***Point Cloud Layer*** represents the metric point cloud $\mathcal{M}_{pc}$, providing the most intuitive representation of the environment.
2) ***Lane Graph Layer*** depicts the lane graph $\mathcal{M}_{lg}$, which contains its own topology and can be utilized for downstream tasks such as path planning.
3) ***Instance Layer*** is a subgraph of instances $\mathcal{M}_{ins} = \langle \mathbf{O}, \mathbf{E} \rangle$, where instances $\mathbf{O}$ are composed of their centers of mass, bounding boxes, captions, and high-dimensional semantic features. Spatial relationships between instances are represented by edges $\mathbf{E}$ connecting them.
4) ***Segment Layer*** represents segments $\mathcal{M}_{seg}$, partitioned based on the connectivity of lane graph $\mathcal{M}_{lg}$, with each one having a center of mass.
5) ***Environment Layer*** encompasses the entire outdoor environment node $\mathcal{M}_{env}$ connected to all segments.

Edges connect nodes within each layer (e.g., to model traversability between segments) or across layers (e.g., to model that point clouds belong to an instance, or that an instance is in a certain segments).

B. Caption-Enhanced Object Comprehension

Vision provides a wealth of information about the shape, size, texture, and location of objects and is an important perception for understanding the environment. The Caption-Enhanced Object Comprehension module sequentially uses three visual language models to realize the caption extraction of objects in each image frame as shown in Fig. 3.

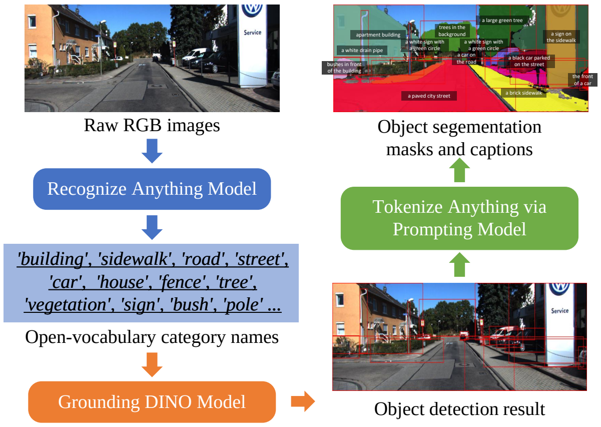

Raw RGB images
Service
Recognize Anything Model
'building', 'sidewalk', 'road', 'street', 'car', 'house', 'fence', 'tree', 'vegetation', 'sign', 'bush', 'pole' ...
Open-vocabulary category names
Grounding DINO Model
Object detection result
Service
Tokenize Anything via Prompting Model
Object segementation masks and captions
a large green tree
trees in the background
apartment building
a sign on the sidewalk
a white sign with a green circle
a white sign with a green circle
a white drain pipe
a car on the road
a black car parked on the street
bushes in front of the building
the front of a car
a brick sidewalk
a paved city street

Fig. 3: We employ three sequential visual language models for image instance segmentation and caption extraction. These models sequentially perform recognition, detection, simultaneous segmentation, and description generation of objects within the input image.

At the current time t, for the input RGB image $I^{(t)}$, we first utilize Recognize Anything Model (RAM) [27] $\text{Rec}(\cdot)$ to recognize the categories present in it. With its open-set capability, RAM is feasible to recognize any common category. Subsequently, we feed the generated open-vocabulary category names along with the original image into the Grounding DINO model [28] $\text{Det}(\cdot, \cdot)$ for open-set object detection. This yields the object detection bounding boxes, which are precisely what is required for the TAP (Tokenize Anything via Prompting) model [29] $\text{SegCap}(\cdot, \cdot)$. Prompted by the object detection bounding boxes, the TAP model segments and describes the main objects within them, producing a set of masks $\{m_i^{(t)}\}_{i=1,,,m}$ and a set of captions $\{c_i^{(t)}\}_{i=1,,,m}$ for the current frame $I^{(t)}$. The entire process can be represented as (1), and the specific model can be substituted with others possessing similar functionality.

$$\{m_i^{(t)}, c_i^{(t)}\} = \text{SegCap}\left(I^{(t)}, \text{Det}\left(I^{(t)}, \text{Rec}(I^{(t)})\right)\right) \quad (1)$$

VLMs aid in expressing the visual semantic understanding of objects through caption text, thereby facilitating the conversion from visual to language modality. However, the captions $\{c_i^{(t)}\}$ lack inherent common sense reasoning abilities. Conversely, LLMs find widespread application across various natural language processing tasks, having been pre-trained on extensive text datasets. Hence, as depicted in (2), we leverage LLMs to encode object captions, thereby generating high-dimensional embedded features to enhance comprehension and reasoning. Consequently, we employ the SBERT model [14] in our experiments.

$$\mathbf{f}_i^{(t)} = \text{Embed}(c_i^{(t)}) \quad (2)$$

After processing with the foundation model described above, we derive the 2D masks $\{m_i^{(t)}\}$ and captions $\{c_i^{(t)}\}$ for candidate objects, along with their corresponding caption features $\{\mathbf{f}_i^{(t)}\}$, based on the input RGB images $I^{(t)}$ observed at the current time.

C. Object-Centric Map Construction

To achieve 3D mapping, it's crucial to incorporate metric information regarding detected objects. We employ a multi-sensor calibration and fusion method to project the 3D point cloud acquired from LiDAR onto a 2D image plane. Before projection, we employ 4DMOS [30] to detect and filter dynamic points from the point cloud $C^{(t)}$ to eliminate trailing shadows. The projection process generates 3D point clouds $\mathbf{p}_i^{(t)}$ of the object instances that are aligned with the corresponding masks $m_i^{(t)}$:

$$\mathbf{p}_i^{(t)} = \{l_k | KTl_k \in m_i^{(t)}, l_k \in C^{(t)}\} \quad (3)$$

where l_k is the LiDAR point, K is the internal camera parameter, and T is the external lidar-to-camera parameter. Considering the existence of bias in the calibration parameters and the noise of the sensors themselves, we use the DBSCAN clustering algorithm to reduce the noise of the point clouds $\mathbf{p}_i^{(t)}$ for each object and transform them to the map frame according to the pose $P^{(t)}$.

For objects $\mathbf{o}_i^{(t)} = \langle \mathbf{p}_i^{(t)}, c_i^{(t)}, \mathbf{f}_i^{(t)} \rangle$ detected in the current frame, we need to fuse them into the existing map $\mathbf{M}^{(t-1)} = \{\mathbf{m}_j^{(t-1)}\} = \{\langle \mathbf{p}_j^{(t-1)}, c_j^{(t-1)}, \mathbf{f}_j^{(t-1)} \rangle\}$. We compute the geometric similarity $\phi_{geo}(i,j)$, caption similarity $\phi_{cap}(i,j)$ and feature similarity $\phi_{fea}(i,j)$ of each newly detected object with all objects in the map. Geometric similarity $\phi_{geo}(i,j)$ represents the 3D bounding box Intersection over Union (IoU) of point clouds $\mathbf{p}_i^{(t)}$ and $\mathbf{p}_j^{(t-1)}$. Caption similarity $\phi_{cap}(i,j)$ is calculated using cosine similarity after vectorizing the caption text $c_i^{(t)}$ and $c_j^{(t-1)}$ with TF-IDF (Term Frequency-Inverse Document Frequency). Feature similarity $\phi_{fea}(i,j)$ is the cosine similarity between the two features $\mathbf{f}_i^{(t)}$ and $\mathbf{f}_j^{(t-1)}$. The overall similarity measure $\phi(i,j)$ is obtained as the weighted sum of the three similarities:

$$\phi(i,j) = \omega_{geo}\phi_{geo} + \omega_{cap}\phi_{cap} + \omega_{fea}\phi_{fea} \quad (4)$$

We perform object association using a greedy assignment strategy, where each newly detected object $\mathbf{o}_i^{(t)}$ is paired with the existing object $\mathbf{m}_j^{(t-1)}$ possessing the highest similarity score. If an object fails to find a match with a similarity score exceeding the threshold δ_{sim}, it is initialized as a new object in the map $\mathbf{M}^{(t)}$. For associated objects, we conduct object fusion. This involves merging the point clouds as $\mathbf{p}_j^{(t)} = \mathbf{p}_i^{(t)} \cup \mathbf{p}_j^{(t-1)}$ and updating the features as

$$\mathbf{f}_j^{(t)} = (\mathbf{f}_i^{(t)} + n_{\mathbf{m}_j}\mathbf{f}_j^{(t)})/(n_{\mathbf{o}_j} + 1) \quad (5)$$

where $n_{\mathbf{m}_j}$ is the number of detections associated with $\mathbf{m}_j$ so far. Additionally, to merge textual captions $c_i^{(t)}$ and $c_j^{(t-1)}$, we utilize the open-source LLM LLaMA [13] to ensure a comprehensive, conflict-free description through prompts.

After incorporating all the objects $\mathbf{o}_i^{(t)}$ of the current frame into the map, we get the incrementally updated map $\mathbf{M}^{(t)} = \{\mathbf{m}_j^{(t)}\} = \{\langle \mathbf{p}_j^{(t)}, c_j^{(t)}, \mathbf{f}_j^{(t)} \rangle\}$.

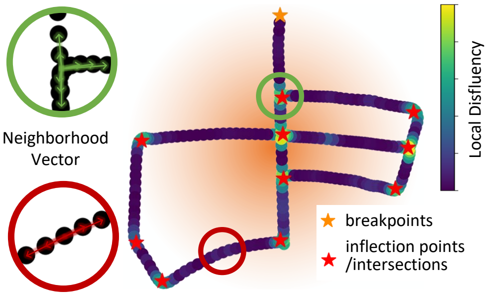


Fig. 4: We extract the lane graph $\mathcal{M}_{lg}$ from historical trajectories $P^{(1:t)}$, whose nodes are derived from vector pinch angle $\Theta^{(n)}$ (breakpoints) and local disfluency $\lambda^{(n)}$ (inflection points or intersections).

D. Hierarchical Graph Representation Formation

On the basis of the map $\mathbf{M}^{(t)}$, we generate the hierarchical graph representation $\mathcal{M}_{all}^{(t)}$.

For ***Point Cloud Layer***, we stack the point clouds of objects as $\mathcal{M}_{pc} = \bigcup_j \mathbf{p}_j^{(t)}$ and downsample them. To enhance the visualization, we can render the point cloud using the commonly used colors of the 19 outdoor classes. Specifically, we assign a class to the point cloud $\mathbf{p}_j^{(t)}$ that has the highest similarity to the caption $\mathbf{c}_j^{(t)}$ among all classes.

For ***Lane Graph Layer***, lane graph $\mathcal{M}_{lg}$ is derived from historical trajectories $P^{(1:t)}$. Initially, trajectories $P^{(1:t)}$ are projected onto a 2D plane as $\widetilde{P}^{(1:t)}$ by eliminating vertical displacements. Nodes in the lane graph $\mathcal{M}_{lg}$ represent inflection points, intersections, and breakpoints. To detect them, we assess local disfluency $\lambda^{(n)}$ at each trajectory point $\widetilde{P}^{(n)}$. Specifically, we consider a neighborhood within a radius R, forming sets of neighborhood vectors as

$$\mathbf{V}^{(n)} = \mathcal{N}\left(\widetilde{P}^{(n)}\right) - \widetilde{P}^{(n)} \tag{6a}$$

$$\mathcal{N}\left(\widetilde{P}^{(n)}\right) = \left\{\widetilde{P}^{(m)} \middle| \left|\widetilde{P}^{(m)} - \widetilde{P}^{(n)}\right| < R\right\} \tag{6b}$$

The angles are computed pairwise within the set of vectors, and the difference between each angle and 0 or π is calculated. Then, the average of the smaller values is determined and considered as the local disfluency measure:

$$\Theta^{(n)} = \left\{\arccos\left(\frac{v_i \cdot v_j}{\|v_i\| \cdot \|v_j\|}\right)\right\}, \forall v_i, v_j \in \mathbf{V}^{(n)} \tag{7a}$$

$$\lambda^{(n)} = \text{mean}\left(\min\left(\left|\theta_i\right|, \left|\theta_i - \pi\right|\right)\right), \forall \theta_i \in \Theta^{(n)} \tag{7b}$$

We employed DBSCAN to cluster trajectory points exhibiting disfluency exceeding a threshold δ_{dis}, thereby identifying inflection points and intersections. To discern breakpoints, we assess whether the mean of $\Theta^{(n)}$ is proximate to 0. Fig. 4 exemplifies the detection of lane graph nodes. The edges of the lane graph $\mathcal{M}_{lg}$ are subsequently derived from the inter-node links and are trimmed based on trajectory alignment.

For ***Instance Layer***, we calculate the bounding box and center of mass for each object $\mathbf{m}_j^{(t)}$ within the object-centric map $\mathbf{M}^{(t)}$, treating them as nodes $\mathbf{O}$. Subsequently, we compute the bounding box IoUs between pairs of objects, resulting in a dense graph which we refine by estimating a Minimum Spanning Tree (MST). Additionally, to infer relations among nodes, we feed the captions, poses and bounding boxes of object pairs into LLaMA, yielding an open-vocabulary instance-layer scene graph $\mathcal{M}_{ins} = \langle \mathbf{O}, \mathbf{E} \rangle$. The nodes $\mathbf{O}$ are connected to the corresponding point cloud in the Point Cloud Layer.

For ***Segment Layer***, connectivity of lane graph $\mathcal{M}_{lg}$ forms the foundation for distinguishing segments as $\mathcal{M}_{seg}$. We designate a small section of the path as a subordinate area near inflection and intersection points on the lane graph. Additionally, intersections have been further categorized into ”intersections” and ”T-intersections”. Inflection points are defined as ”L-intersections”. The stretch of roadway that doesn't intersect with any inflection or intersection points is termed the ”straight roadway”. The nodes $\mathbf{O}$ in the Instance Layer are linked to the closest road segments.

For ***Environment Layer***, we construct a global environment node $\mathcal{M}_{env}$ that connects all road segments $\mathcal{M}_{seg}$ in Segment Layer.

Collectively, the aforementioned layers constitute a comprehensive OpenGraph $\mathcal{M}_{all}^{(t)}$ that encompasses a semantic comprehension and a hierarchical topology of the outdoor environment.

IV. Experimental Results

In this section, the performance of the proposed OpenGraph is validated through experiments performed on the public ourdoor dataset SemanticKITTI [34]. We aim to use experiments to validate our framework, through the following specific questions:

1) Without fine-tuning any models, can OpenGraph accomplish zero-shot 3D semantic understanding?
2) Does Caption-Enhanced Object Comprehension provide more comprehensible object retrieval?
3) What are the potential applications of open-vocabulary hierarchical graph representation?

A. 3D Semantic Segmentation

We first validate OpenGraph's ability for zero-shot semantic understanding in outdoor environments. As detailed in subsection III-D, we conduct 3D point cloud semantic segmentation at the Point Cloud Layer using 19 predefined classes from the SemanticKITTI dataset. Due to the absence of a direct zero-shot outdoor semantic segmentation comparison baseline, we initially consider two classes of fully supervised methods. The first class comprises a point cloud segmentation technique trained and fine-tuned on the SemanticKITTI dataset, namely **RangeNet++** [31]. We generate a global semantic map using its predicted labels. The second class consists of 2D image segmentation methods trained on outdoor image datasets, including **DeepLab v3** [32] and **DecoupleSegNets** [33]. We derive 3D semantic maps by projecting their image segmentation onto point

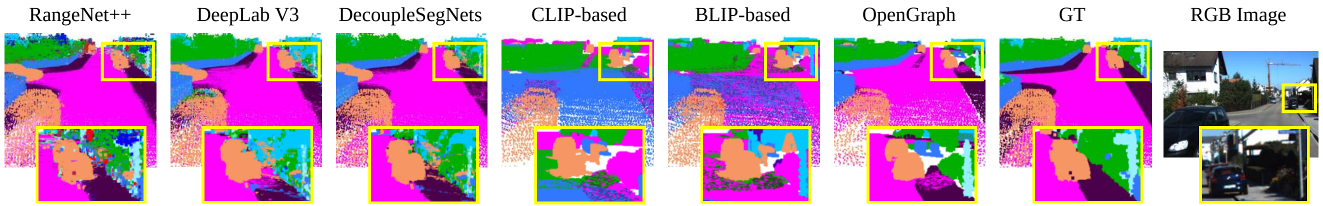


Fig. 5: Semantic segmentation results on the SemanticKITTI dataset, utilizing 19 classes, indicate that despite not undergoing fine-tuning, OpenGraph demonstrates higher segmentation accuracy and reduced noise levels.

TABLE I: Quantitative Results (IoU and F1 Score) of Semantic Segmentation on SemanticKITTI Dataset

Seq.	**Method**	Car	Road	Sidewalk	Building	Fence	Veget.	Pole	T.-sign	**Average**	**F1 Score**
03	RangeNet++ [31]	0.4191	0.8096	0.4656	0.3962	0.3483	0.5410	0.1245	0.1302	0.4780	0.6115
	DeepLab V3 [32]	0.2535	0.6784	0.3392	0.3262	0.0632	0.5445	0.1270	0.1521	0.4626	0.6025
	DecoupleSegNets [33]	0.3075	0.8283	0.5314	0.3912	0.0646	0.6624	0.1412	0.2457	0.6025	0.7277
	CLIP-based [4]	0.5851	0.4349	0.0000	0.1654	0.0823	0.3497	0.0000	0.0138	0.3972	0.5797
	BLIP-based [6]	0.3215	0.4478	0.0000	0.3240	0.0227	0.3789	0.0122	0.1202	0.4017	0.5673
	OpenGraph	0.4191	0.7299	0.2552	0.1737	0.6599	0.0083	0.2960	0.0000	**0.6051**	**0.7302**
05	RangeNet++ [31]	0.5634	0.8575	0.5591	0.5933	0.3605	0.5368	0.1285	0.0950	0.4990	0.6200
	DeepLab V3 [32]	0.3627	0.7731	0.3749	0.5094	0.2855	0.6066	0.0986	0.0388	0.5203	0.6579
	DecoupleSegNets [33]	0.4489	0.8501	0.5850	0.5228	0.3022	0.6293	0.1260	0.0845	0.5895	0.7212
	CLIP-based [4]	0.5618	0.0356	0.0000	0.1769	0.3427	0.0028	0.0312	0.0000	0.2964	0.4518
	BLIP-based [6]	0.5997	0.4334	0.0000	0.3318	0.1237	0.1926	0.0000	0.0342	0.3320	0.4781
	OpenGraph	0.7000	0.7963	0.4823	0.7961	0.3957	0.6312	0.1552	0.1056	**0.6598**	**0.7749**
08	RangeNet++ [31]	0.6339	0.8214	0.4786	0.5500	0.1086	0.6083	0.1607	0.1504	0.5111	0.6295
	DeepLab V3 [32]	0.4167	0.7162	0.3087	0.4923	0.0736	0.6758	0.1074	0.0691	0.5425	0.6733
	DecoupleSegNets [33]	0.4445	0.8370	0.4870	0.5414	0.0845	0.7152	0.1026	0.0631	0.6376	0.7576
	CLIP-based [4]	0.3942	0.3456	0.0000	0.2439	0.0952	0.2166	0.0033	0.0693	0.2753	0.4130
	BLIP-based [6]	0.4025	0.4318	0.0000	0.3909	0.0633	0.2840	0.0008	0.0422	0.3677	0.5250
	OpenGraph	0.6667	0.7436	0.3594	0.7581	0.1447	0.6384	0.1892	0.1635	**0.6463**	**0.7633**

clouds. Additionally, we compare methods that directly utilize VLM features to replace caption features in OpenGraph, specifically **CLIP-based** [4] and **BLIP-based** [6].

We selected three sequences from SemanticKITTI: short (03), medium (05), and long (08). As shown in Fig. 5, due to point-wise segmentation in 2D or 3D, the fully supervised approaches generate many scattered points. Additionally, small objects like bicycles suffer from lower segmentation accuracy due to limited training samples. While CLIP-based and BLIP-based, two methods employing VLM features for scene understanding, fail to achieve accurate object classification, OpenGraph excels in this aspect with the assistance of LLM features. Tab. I presents the results of semantic segmentation quantification, including IoU for common classes and overall F1 scores. Notably, OpenGraph outperforms even comparable fully supervised methods across these sequences. These results demonstrate that OpenGraph achieves accurate zero-shot semantic understanding in outdoor environments.

B. Open-vocabulary Object Retrieval

To illustrate the advantages of OpenGraph for complex semantic queries, we conducted object retrieval experiments that center around three distinct text types:

Ontology: A direct description of the object itself. For instance, *A tall tree*.

Proximity: The description of the object relies on objects related to it. For example, *I want to find some green leaves* (a tree or a bush).

TABLE II: Quantitative Results (top-1,2,3 recall) of Object Retrieval

Seq.	**Query-Type**	**Methods**	**R@1**	**R@2**	**R@3**	**#Query**
03/05/08	Onotology	CLIP-based [4]	0.60	0.70	0.73	30
		BLIP-based [6]	0.50	0.60	0.70	
		OpenGraph	**0.90**	**0.90**	**0.90**	
		OpenGraph-LLM	0.70	0.80	0.87	
	Proximity	CLIP-based [4]	0.37	0.47	0.53	30
		BLIP-based [6]	0.30	0.40	0.47	
		OpenGraph	**0.80**	**0.87**	**0.87**	
		OpenGraph-LLM	0.70	0.73	0.80	
	Functionality	CLIP-based [4]	0.47	0.53	0.57	30
		BLIP-based [6]	0.37	0.37	0.47	
		OpenGraph	0.63	**0.80**	0.80	
		OpenGraph-LLM	**0.67**	**0.80**	**0.87**	

Functionality: The description focuses on the inferred functions of the object in question. For instance, *Something that can be used for driving* (a car).

The experiments involve small segments extracted from the three sequences, each containing a diverse array of objects. For each query type on each sequence, we generated 10 distinct descriptions, with relevant objects manually chosen as ground truth values. The Fig. 6 illustrates some of the comparison results of OpenGraph with other zero-shot methods, where the colors are rendered according to the degree of similarity of the features (VLM features for CLIP-based and BLIP-based, LLM features for OpenGraph). Since OpenGraph maintains an explicit caption for each object,

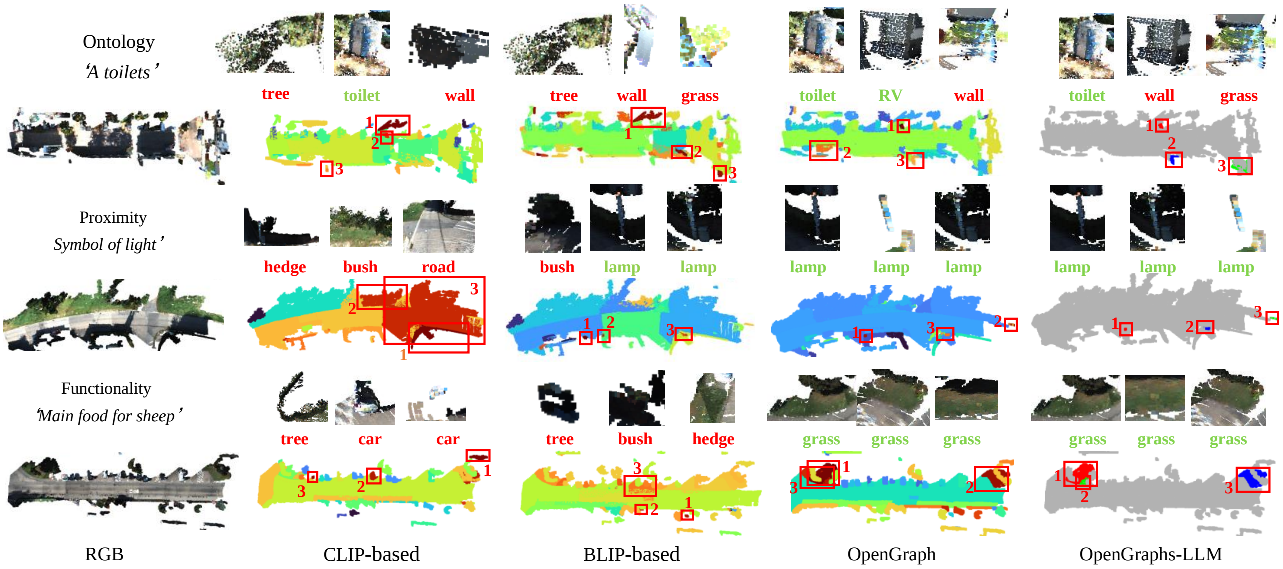
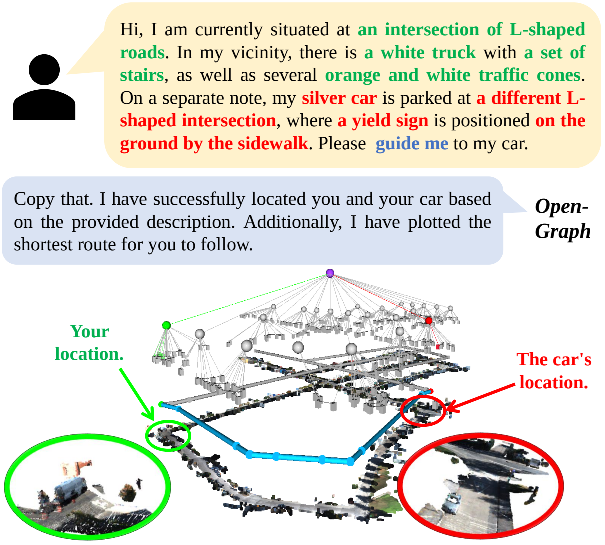


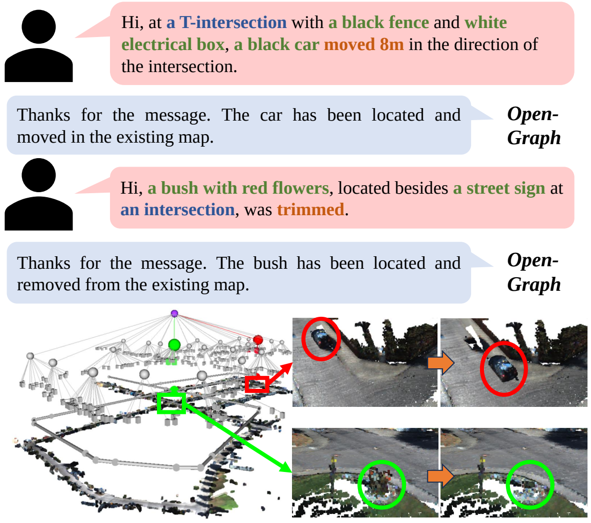
Fig. 6: The outcomes from various open-vocabulary text queries (displaying the Top-3 objects). In the visual representation, OpenGraph-LLM highlights the Top-3 objects in red, green, and blue, while the other methods render all objects based on relevance. The text beneath each retrieved object in the figure corresponds to its actual category, where green signifies successful retrieval and red indicates failure.



Hi, I am currently situated at an intersection of L-shaped roads. In my vicinity, there is a white truck with a set of stairs, as well as several orange and white traffic cones. On a separate note, my silver car is parked at a different L-shaped intersection, where a yield sign is positioned on the ground by the sidewalk. Please guide me to my car.

Copy that. I have successfully located you and your car based on the provided description. Additionally, I have plotted the shortest route for you to follow.

Fig. 7: OpenGraph efficiently identifies the start (green) and end (red) points by utilizing user descriptions and performs path planning to determine the optimal route (blue) between them.



Hi, at a T-intersection with a black fence and white electrical box, a black car moved 8m in the direction of the intersection.

Thanks for the message. The car has been located and moved in the existing map.

Hi, a bush with red flowers, located besides a street sign at an intersection, was trimmed.

Thanks for the message. The bush has been located and removed from the existing map.

Fig. 8: OpenGraph's open-vocabulary hierarchical map representation facilitates human-interactive map updating. Here shows the updated point cloud in the Point Cloud Layer.

it facilitates effortless access to LLM for object retrieval, denoted as *OpenGraph-LLM*. To elaborate, we feed both the map object captions and query text into GPT4 [35], tasking it with identifying the top three most pertinent objects, labeled red, green, and blue, respectively. Tab. II presents the overall top-1, top-2, and top-3 recall measurements.

First, in ontology queries, OpenGraph demonstrates the highest recall rate, mitigating the positional bias when LLM is applied, such as recognizing a toilet in a previously observed RV. Secondly, in proximity queries, the performance of VLM-based methods sharply drops, due to limited natural language comprehension like 'symbol of light' versus 'lamp'. Lastly, OpenGraph excels in challenging functional queries with LLM boosting its natural language reasoning. In summary, OpenGraph and its LLM variants demonstrate superior natural language reasoning in a variety of outdoor object retrieval tasks.

C. Hierarchical Graph Structured Query

Hierarchical graphs offer the benefit of structured top-down queries. By integrating with LLMs on the front end, OpenGraph empowers users to accomplish a wide range of tasks more effectively and efficiently. Imagine the queriers situated within a scene, capable of observing only a limited portion of their surroundings. By indicating the type of road

(Segment Layer) they are on and the relationships between objects nearby (Instance Layer), they can swiftly localize themselves. Moreover, if they can provide hints about their destination, OpenGraph can facilitate global path planning on the lane graph to guide them effectively. A related case is shown in Fig. 7.

Furthermore, OpenGraph facilitates human-interactive map updating. Individuals within the environment can contribute the latest map patches to OpenGraph by detecting changes in their surroundings, ensuring the continual upkeep of the environment map. Fig. 8 visually demonstrates this process, where a person actively observes the removal of a bush beneath a street sign at an intersection. OpenGraph promptly identifies the object based on the description and promptly updates the map accordingly.

V. Conclusion

This paper introduces OpenGraph, an open vocabulary hierarchical 3D graph representation framework for large-scale outdoor environments. Initially, the extraction of instance masks, captions, and features occurs in the 2D images, facilitated by VLMs and LLMs. Subsequently, an object-centric map is incrementally constructed and fused by projecting onto 3D point clouds. Finally, the extraction of the lane graph and subsequent scene segmentation culminates in the derivation of a hierarchical graph. The results from evaluations on public datasets reveal that OpenGraph, operating as a zero-shot method, even exhibits superior performance in 3D semantic segmentation compared to fully supervised methods, while also demonstrating advantages in open-vocabulary retrieval over alternative types. Moreover, the hierarchical graph representation facilitates rapid structured queries. Moving forward, we need to inject richer semantics at the edges between instances and provide component-level understanding.

References

[1] Y. Deng, *et al.*, "Macim: Multi-agent collaborative implicit mapping," *IEEE Robotics and Automation Letters*, DOI 10.1109/LRA.2024.3379839, pp. 1–8, 2024.

[2] A. Hornung, *et al.*, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous robots*, vol. 34, no. 3, pp. 189–206, 2013.

[3] Y. Deng, *et al.*, "See-csom: Sharp-edged and efficient continuous semantic occupancy mapping for mobile robots," *IEEE Transactions on Industrial Electronics*, 2023.

[4] A. Radford, *et al.*, "Learning transferable visual models from natural language supervision," in *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

[5] C. Jia, *et al.*, "Scaling up visual and vision-language representation learning with noisy text supervision," in *International conference on machine learning*, pp. 4904–4916. PMLR, 2021.

[6] J. Li, *et al.*, "Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation," in *International Conference on Machine Learning*, pp. 12 888–12 900. PMLR, 2022.

[7] N. M. M. Shafiullah, *et al.*, "Clip-fields: Weakly supervised semantic fields for robotic memory," *arXiv preprint arXiv:2210.05663*, 2022.

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

[9] J. Kerr, *et al.*, "Lerf: Language embedded radiance fields," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 19 729–19 739, 2023.

[10] S. Lu, *et al.*, "Ovir-3d: Open-vocabulary 3d instance retrieval without training on 3d data," in *Conference on Robot Learning*, pp. 1610–1620. PMLR, 2023.

[11] A. Takmaz, *et al.*, "Openmask3d: Open-vocabulary 3d instance segmentation," *arXiv preprint arXiv:2306.13631*, 2023.

[12] A. Kirillov, *et al.*, "Segment anything," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4015–4026, 2023.

[13] H. Touvron, *et al.*, "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[14] N. Reimers, *et al.*, "Sentence-bert: Sentence embeddings using siamese bert-networks," *arXiv preprint arXiv:1908.10084*, 2019.

[15] S. Yang, *et al.*, "Semantic 3d occupancy mapping through efficient high order crfs," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 590–597. IEEE, 2017.

[16] Y. Deng, *et al.*, "S-mki: Incremental dense semantic occupancy reconstruction through multi-entropy kernel inference," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 3824–3829. IEEE, 2022.

[17] Y. Deng, *et al.*, "Hd-ccsom: Hierarchical and dense collaborative continuous semantic occupancy mapping through label diffusion," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2417–2422. IEEE, 2022.

[18] Y. Shi, *et al.*, "City-scale continual neural semantic mapping with three-layer sampling and panoptic representation," *Knowledge-Based Systems*, vol. 284, p. 111145, 2024.

[19] S. Peng, *et al.*, "Openscene: 3d scene understanding with open vocabularies," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 815–824, 2023.

[20] B. Li, *et al.*, "Language-driven semantic segmentation," *CoRR*, vol. abs/2201.03546, 2022. [Online]. Available: https://arxiv.org/abs/2201.03546

[21] P. D. Nguyen, *et al.*, "Open3dis: Open-vocabulary 3d instance segmentation with 2d mask guidance," *arXiv preprint arXiv:2312.10671*, 2023.

[22] Q. Gu, *et al.*, "Conceptgraphs: Open-vocabulary 3d scene graphs for perception and planning," *arXiv preprint arXiv:2309.16650*, 2023.

[23] N. Hughes, *et al.*, "Hydra: A real-time spatial perception system for 3d scene graph construction and optimization," *arXiv preprint arXiv:2201.13360*, 2022.

[24] S.-C. Wu, *et al.*, "Scenegraphfusion: Incremental 3d scene graph prediction from rgb-d sequences," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7515–7525, 2021.

[25] H. Chang, *et al.*, "Context-aware entity grounding with open-vocabulary 3d scene graphs," *arXiv preprint arXiv:2309.15940*, 2023.

[26] E. Greve, *et al.*, "Collaborative dynamic 3d scene graphs for automated driving," *arXiv preprint arXiv:2309.06635*, 2023.

[27] Y. Zhang, *et al.*, "Recognize anything: A strong image tagging model," *arXiv preprint arXiv:2306.03514*, 2023.

[28] S. Liu, *et al.*, "Grounding dino: Marrying dino with grounded pre-training for open-set object detection," *arXiv preprint arXiv:2303.05499*, 2023.

[29] T. Pan, *et al.*, "Tokenize anything via prompting," *arXiv preprint arXiv:2312.09128*, 2023.

[30] B. Mersch, *et al.*, "Receding Moving Object Segmentation in 3D LiDAR Data Using Sparse 4D Convolutions," *IEEE Robotics and Automation Letters (RA-L)*, vol. 7, no. 3, pp. 7503–7510, 2022.

[31] A. Milioto, *et al.*, "Rangenet++: Fast and accurate lidar semantic segmentation," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 4213–4220. IEEE, 2019.

[32] L.-C. Chen, *et al.*, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proceedings of the European conference on computer vision (ECCV)*, pp. 801–818, 2018.

[33] X. Li, *et al.*, "Improving semantic segmentation via decoupled body and edge supervision," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16*, pp. 435–452. Springer, 2020.

[34] J. Behley, *et al.*, "Towards 3D LiDAR-based semantic scene understanding of 3D point cloud sequences: The SemanticKITTI Dataset," *The International Journal on Robotics Research*, vol. 40, DOI 10.1177/02783649211006735, no. 8-9, pp. 959–967, 2021.

[35] J. Achiam, *et al.*, "Gpt-4 technical report," *arXiv preprint arXiv:2303.08774*, 2023.